NASA CR 185680 1 of 3

CSDL-T-1106

ROBUST INTELLIGENT FLIGHT CONTROL FOR HYPERSONIC VEHICLES

by
Gregory Errol Chamitoff

February 1992

Doctor of Philosophy Thesis
Massachusetts Institute of Technology

DRAPER
LABORATORY

The Charles Stark Draper Laboratory, Inc.
555 Technology Square, Cambridge, Massachusetts 02139-3563

Robust Intelligent Flight Control For Hypersonic Vehicles

by
Gregory Errol Chamitoff

B.S., California Polytechnic State University (1984)
M.S., California Institute of Technology (1985)

Submitted to the Department of
Aeronautics & Astronautics
in Partial Fulfillment of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
February 1992

Signature of Author ______________________________
Department of Aeronautics and Astronautics
Feb 20, 1992

Certified by Wallace E Vander Velde
Professor Wallace E. Vander Velde
Thesis Supervisor and Committee Chairman

Certified by ______________________________
Dr. Philip D. Hattis
CSDL Technical Supervisor

Certified by Lena Valavani
Professor Lena S. Valavani
Thesis Committee Member

Certified by M. Martinez-Sanchez
Professor Manuel Martinez-Sanchez
Thesis Committee Member

Certified by Edward V. Bergmann
Edward V. Bergmann
CSDL Thesis Committee Member

Accepted by ______________________________
Professor Harold Y. Wachman
Chairman, Department Graduate Committee

Robust Intelligent Flight Control for Hypersonic Vehicles

by

Gregory Errol Chamitoff

Submitted to the Department of Aeronautics and Astronautics
on February 20, 1992 in partial fulfillment of the requirements for
the degree of Doctor of Philosophy

Abstract

Intelligent optimization methods are applied to the problem of real–time flight control for a class of airbreathing hypersonic vehicles (AHSV). The extreme flight conditions that will be encountered by single–stage–to–orbit vehicles, such as the National Aerospace Plane, present a tremendous challenge to the entire spectrum of aerospace technologies. Flight control for these vehicles is particularly difficult due the combination of nonlinear dynamics, complex constraints, and parametric uncertainty.

This thesis presents an approach that utilizes all available a *priori* and in–flight information to perform robust, real–time, short–term trajectory planning. Stable tracking of a desired trajectory is achieved through the repetitive solution of a receding–horizon nonlinear optimal control problem, which includes all constraints and uncertainties. A viable correction trajectory is generated, followed for a short interval of time, and then recomputed. The flight control approach consists of an enhanced A–Star optimization technique that incorporates a Lyapunov stability criterion in a highly parallelizable algorithm. The efficiency of the A–Star search, and the theoretical guarantees of a Lyapunov approach, are both achieved. Conditions are derived in order to assure controllability of the vehicle, convergence of the optimization algorithm, and stability of the correction trajectory. Robustness of the solution to interval bounded parametric uncertainty is achieved through a minimax optimization in which the worst–case cumulative tracking error is minimized by the solution trajectory. Finally, some enhancements to the algorithm address practical implementation issues such as memory and time limitations. The resulting Robust Intelligent Flight Controller (RIFC) provides guaranteed tracking performance in the presence of uncertainty while observing all physical constraints.

The effectiveness of this approach is demonstrated through a series of flight tests using a realistic hypersonic vehicle simulation that includes detailed models of the airframe, aerodynamics, and scramjet propulsion system in order to provide representative AHSV characteristics. The results are compared to those for a single–step optimal controller.

Thesis Committee:

Dr. Wallace E. Vander Velde,	Professor of Aeronautics and Astronautics
Dr. Philip D. Hattis,	Technical Staff, C.S. Draper Laboratory, Inc.
Dr. Lena S. Valavani,	Associate Professor of Aeronautics and Astronautics
Dr. Manuel Martinez-Sanchez,	Associate Professor of Aeronautics and Astronautics
Edward V. Bergmann,	Technical Staff, C.S. Draper Laboratory, Inc.

Acknowledgements

Somewhere along the way from the first ideas to the finished product, I learned how important others can be in the process of achieving ones goals. Countless discussions, covering the full range from technical to moral support, have been the backbone of this thesis.

This research was made possible through the financial support of the Charles Stark Draper Laboratory. The opportunity to pursue a graduate education at MIT, and the luxury of a work environment that provides an exposure to real–world applications, has been greatly appreciated.

I am very grateful for the efforts and interest of my thesis committee. They have been a major contributing factor to the quality of my education, and I feel that they have also become good friends. I could not have asked for a better supervisor than Phil Hattis. His concern for the welfare of the student, his attention to detail, and his advice and friendship have made my time at Draper Labs an extremely rewarding experience. I sincerely appreciate all he has done for me, and I only hope that I can do the same for another student someday. I owe special thanks to my committee chairman, Professor Vander Velde, who has devoted so much of his time to assure the quality of this thesis. His experience, wisdom, and guidance have been instrumental in keeping this research on track. Professor Martinez–Sanchez has been very helpful as an advisor on the aerodynamic and propulsion issues. I've also appreciated his amazing insight on all aspects of my research. It has been a pleasure working with him, and I will miss our many discussions. Professor Valavani has been a teacher and friend for most of my residency at MIT. Her enthusiasm about estimation and control theory has been inspirational, and her participation on my committee has served to strengthen many of the theoretical aspects. I would also like to thank Ed Bergmann for being one of my technical advisors. His excitement, interest, and ideas are not only appreciated, they have motivated some of the main concepts in this thesis.

Several other colleagues at Draper Labs have made significant contributions to this research. In particular, I would like to thank Joe Paradiso for many enlightening discussions, and for his generous help with details of the approach, theory, and manuscript. His assistance and friendship have been very important to me. Brent Appleby and Nazareth Bedrossian have been my classmates, colleagues, and friends throughout my graduate education. Our many discussions and study sessions have been invaluable to my success at MIT. Special thanks also goes to Rami Mangoubi and Bryan Kang, who, along with Brent and Nazareth, have given much of their time to the discussion of my thesis topic. I would also like to mention Neil Adams, Dino DeAngelis, Achille Messac, Mark Koenig, and Jay Farrell for their technical support and fellowship.

I have been very fortunate to have many supportive friends and roommates over the years. Special thanks to Roger Alexander for being there to listen to all my

ideas, and for his insightful skepticism. I would like to acknowledge Perry Bonanni, for his close friendship through many years of graduate school, and for his technical assistance with some of the figures. Also deserving special mention is Julie Welsh, whose friendship and moral support have helped me through many of life's challenges ever since our high school days.

The confidence and support of many of my relatives has been a continual source of strength that I have tapped into time and time again. Special mention goes to my grandparents Murtle and Jack Wexler, whose love and encouragement have always been there for me through all my endeavors and accomplishments. And to the memory of my grandparents Clara and Morris Chamitoff, who can still make me laugh and smile.

I am especially thankful for Chantal Caviness, whose love and support have kept my spirits high through the hardest times. Her understanding and confidence have made it possible to complete this project.

Finally, I am eternally grateful for the love and encouragement of my parents, Al and Shari Chamitoff, and my brother Ken. Without my family to be proud of me, my accomplishments would truly seem meaningless.

This thesis was prepared at The Charles Stark Draper Laboratory, Inc., under contract NAS9–18426 for the Johnson Space Center of the National Aeronautics and Space Administration.

Publication of this thesis does not constitute approval by Draper or the sponsoring agency of the findings or conclusions contained herein. It is published for the exchange and stimulation of ideas.

I hereby assign my copyright of this thesis to The Charles Stark Draper Laboratory, Inc., Cambridge, Massachusetts.

(author's signature)

Permission is hereby granted by The Charles Stark Draper Laboratory, Inc., to the Massachusetts Institute of Technology to reproduce any or all of this thesis.

Table Of Contents

List Of Figures

List Of Tables

List Of Symbols & Acronyms

AHSV	Airbreathing Hypersonic Vehicle
AI	Artificial Intelligence
A^*, A^*_ϵ	A-star optimization algorithm, suboptimal search
α, α_d	Actual and desired angle of attack
$\mathcal{AR}_B$	Body aspect ratio
BD	Best/depth search control parameter
BF	Best-First (Search Algorithm)
b	Vehicle wing span
B_V, B_γ, B_ω, B_θ	Maximum tolerances on the state tracking errors
β	Scramjet nozzle angle, also yaw angle
CFD	Computational Fluid Dynamics
$\bar{c}$	Mean chord length for wing
$c(n_i, n_j)$	Cost of a trajectory from node n_i to node n_j
$c^*(n_i, n_j)$	Cost of optimal trajectory from node n_i to node n_j
$\underline{C}$	Coordinate transformation matrices
C, C^*	Cost of a trajectory, cost of an optimal trajectory
C_D	Aerodynamic drag coefficient
C_f	Skin friction coefficient
C_L	Aerodynamic lift coefficient
C_{Ma}	Aerodynamic moment coefficient
C_{Mp}	Propulsion moment coefficient
C_{Tn}	Thrust coefficient - normal component ($\perp$ to V)
C_{Tv}	Thrust coefficient - tangential component ($\|$ to V)
$C_{M_{\delta e}}$, $C_{M_{\delta_T}}$, $C_{M_{\delta v}}$	Control moment coefficients - wind axes (pitch up)
$C_{X_{\delta e}}$, $C_{X_{\delta_T}}$, $C_{X_{\delta v}}$	Control coefficients in the X-direction - wind axes (along V)
$C_{Z_{\delta e}}$, $C_{Z_{\delta_T}}$, $C_{Z_{\delta v}}$	Control coefficients in the Z-direction - wind axes (down)
CM_b	Center of mass bias
C_p	Pressure coefficient
C_v, C_p	Specific heat at constant volume and constant pressure
DOF	Degrees Of Freedom
$\underline{d}(t)$	Disturbance vector
δ	Wing sweep angle
δe, δ_T, δv	Elevon, throttle, and thrust vector control inputs
δ_F, δ_M	Equivalent force and moment control inputs

$\delta_V, \delta_\gamma, \delta_\omega, \delta_\theta$	Bounds of controllability envelope
Δt_a	Time required to achieve desired attitude
Δt_i	Time required to achieve desired inputs
Δt_L	Time required for decrease in Lyapunov function
Δt_c	$\Delta t_c = \Delta t_L - \Delta t_a - \Delta t_i$
Δt_s	Control cycle time-step
Δu_i	Quantization interval for the ith control
Δx_i	Quantization interval for the ith state
$\underline{e}, \underline{\Omega}_e$	State tracking error vector, set of admissible errors
$\dot{e}^{+/-}_{V_{marg}}, \dot{e}^{+/-}_{\gamma_{marg}}, \dot{e}^{+/-}_{\omega_{marg}}$	Error rate margins in velocity, flight path, and pitch rate
$e_V, e_\gamma, e_\omega, e_\theta$	State tracking errors = actual - desired
$f(n)$	Cost estimate for an entire path, $f(n) = g(n) + h(n)$
f^*	Value of $f(n)$ for a node along an optimal path
$\underline{F}_p, \underline{F}_s$	Net pressure forces, net shear forces
$g(n)$	Cost from the start node to the node n (from S to n)
g^*	Value of $g(n)$ for a node along an optimal path
γ, γ_d	Actual and desired flight path angle
$\gamma^h(t)$	Heuristic flight path trajectory (optimistic tracking rates)
γ_a	Ratio of the specific heats for air ($\gamma_a = C_p/C_v$)
γ_1, γ_2	Forebody ramp angles
$\dot{\gamma}^{+/-}_{marg}$	Achievable acceleration margins for flight path
Γ	Set of terminal nodes that are goal nodes
$\underline{\Gamma}$	Vehicle geometry - panels, normals, centroids, and areas
$\underline{\Gamma}_B, \underline{\Gamma}_W$	Vehicle geometry in body and wind coordinates
HEOC	Heuristically Enhanced Optimal Control
HOT	Higher Order Terms
HSV	Hypersonic Vehicle
HTOL	Horizontal Take-Off and Landing
$h(n)$	Terminating cost estimate (from n to T)
h^*	Value of $h(n)$ for a node along an optimal path
h_c, h_i	Scramjet capture height and inlet height
IC	Intelligent Control
$\underline{I}, I_x, I_y, I_z, I_{xz}$	Vehicle inertia matrix and individual moments of inertia
J	Cost function for tracking performance of trajectory
κ_V, κ_γ	$\kappa_V = (\dot{e}_{V_{marg}}) \cdot \Delta t_L$, $\kappa_\gamma = (\dot{e}_{\gamma_{marg}}) \cdot \Delta t_c$ (see section 7.4)
LEO	Low Earth Orbit
LHP	Left Half Plane
LTI	Linear Time Invariant
$L, \Delta L(\Delta t)$	Lyapunov function and its change in a time Δt
l_c	Scramjet combustor length
l_e	Elevon length
l_{f1}, l_{f2}	Forebody ramp lengths
l_N, l_l	Scramjet nozzle length and nozzle lip length
L_T	Value of Lyapunov function at target envelope
l_v	Vehicle length

λ_V, λ_γ, λ_ω, λ_θ	Target range for tracking errors
MIMO	Multi-Input-Multi-Output
MIT	Massachusetts Institute of Technology
MRAC	Model Reference Adaptive Control
M	Mass of the vehicle
Ma	Mach number
M_E, ω_E	Mass of the Earth, rotation rate of the Earth
$\underline{M}_P$, $\underline{M}_S$	Net moment due to pressure forces, net shear moment
$\underline{\mu}$	Mass properties - mass, inertia matrix, fuel flow rate
NASP	National Aerospace Plane
NASA	National Aeronautics and Space Administration
NDV	NASP Derived Vehicle
n	A node, used in the search to represent a state along a path
$\hat{n}$	Outward pointing normal vector for a surface panel
n^*, n'	A node on the optimal path, a successor node
N_B	Branching degree of a search graph - nodes per expansion
OR	Operations Research
OTV	Orbital Transfer Vehicle
$\underline{P}$	Design parameters - aerodynamic & propulsion design value
P, $\mathbf{P}$, P^*	A path between two nodes, a set of paths, an optimal path
$\underline{\phi}$, $\underline{\Omega}_\phi$	Vector of parameters, the set of possible parameter values
$\underline{\phi}$	Flow variables - temperature, density, pressure, etc.
ϕ_l	Latitude
ψ	Heading angle
ψ_i	Flow incidence angle for ith surface
$Q = \frac{1}{2}\rho V^2$	Dynamic pressure
$\underline{Q}$, Q_i	Weighting matrix for Lyapunov function, individual weights
RHP	Right Half Plane
RIFC	Robust Intelligent Flight Control
r	Distance from the center of the Earth
R	Gas constant for air
R_e, R_{ex}	Reynolds number, local Reynolds number
ρ	Flow density
SISO	Single-Input-Single-Output
SMC	Sliding Mode Control
SSOC	Single Step Optimal Control
SSTO	Single Stage To Orbit
S	Start node in a search tree representing the initial state
S_i	Surface area of the ith panel
S_{ref}	Wing reference area
σ	Bank angle
$\underline{\sigma}$	Forces & moments - aerodynamic, propulsion, and control
T	Terminal node in a search tree, also temperature
$\underline{T}$	Tabulated data - scramjet tables or coefficient database
θ, θ_d	Actual and desired pitch attitude
$\underline{\theta}$	Attitude states
$\underline{\theta}^h(t)$	Heuristic attitude trajectory (optimistic tracking rates)
θ_l	Longitude

$\underline{u}$, $\underline{\Omega}_{\mathrm{u}}$	Control input vector, the set of admissible control inputs
V, V_{d}	Actual and desired flight velocity
$\underline{V}$	Velocity vector
$V^{\mathrm{h}}(t)$	Heuristic velocity trajectory (optimistic tracking rates)
$\dot{V}^{+/-}_{\mathrm{marg}}$	Achievable acceleration margins for velocity
ω, ω_{d}	Actual and desired pitch rate
$\underline{\omega}$	Attitude rates
$\omega^{\mathrm{h}}(t)$	Heuristic pitch rate trajectory (optimistic tracking rates)
ω_{c}	Scramjet combustor width
$\dot{\omega}^{+/-}_{\mathrm{marg}}$	Achievable acceleration margins for pitch rate
$\underline{x}$, $\underline{x}_{\mathrm{d}}$, $\underline{\Omega}_{\mathrm{x}}$	Actual and desired state vector, the set of admissible states
$\underline{x}^{\mathrm{h}}(t)$	Heuristic state trajectory (optimistic tracking rates)

Chapter 1

Introduction

1.1 Motivation

As human exploration and development of space continues to expand toward orbiting space stations, a permanent lunar base, and missions to Mars, it becomes evident that a new fleet of advanced spacecraft will be required to support these future ventures. Despite many problems, the National Aeronautics and Space Administration (NASA) has demonstrated the capability to transport cargo and crew to and from low–earth–orbit (LEO) in a reusable winged spacecraft. Since the maiden voyage of *Columbia* on April 12th, 1981, the Space Shuttle has been launched into space and returned to Earth on numerous successful missions. The shuttle's versatility as a research and multipurpose space vehicle has also been well established on many flights, including several related to satellite recovery and repair. Unfortunately, however, the Space Shuttle program suffers from several serious drawbacks. The shuttle can only be prepared and launched at specially equipped (and expensive) facilities, only one of which is operational at this time. Furthermore, the hardware required for each shuttle flight is not all reusable, (such as the external tank), and many of the systems are very complex, requiring extensive refurbishing prior to each mission.[1] These factors, as well as the limited size of the fleet (currently four orbiters), contribute to the severely restricted flight schedule and the tremendous costs associated with delivering payload to orbit on the shuttle. For the specific purpose of launching satellites, the shuttle is really quite inefficient since it is burdened by the immense additional overhead of requiring crew life support. This makes it important for NASA to combine several mission objectives into every flight. These and other disadvantages suggest the need for a more diversified fleet of future spacecraft that can handle all space related missions more effectively. [Mcc1, Cha1, Fre1, Bak1]

Research related to potential future spacecraft designs has included a variety of new vehicle concepts ranging from inexpensive unmanned cargo boosters[2], to futuristic interstellar spacecraft powered by fusion reactors. Other proposed spacecraft include the following: orbital transfer vehicles (OTV's), for positioning payloads in desired orbits; reentry or aero–capture vehicles, for exploration of the planets or return to Earth; and hypersonic vehicles (HSV's), for maneuvering within the atmosphere of the Earth or other planets at (or near) orbital speeds. [Dem1, Cra1] Many applications are possible for vehicles that are capable of operating in the hypersonic flight regime. One such concept is a fully reusable single–stage–to–orbit (SSTO) winged spacecraft that is capable of flight from subsonic through hypersonic speeds powered by an air–breathing propulsion system. An experimental prototype of these air–breathing hypersonic vehicles (AHSV's), called the National Aerospace Plane (NASP), is currently under development to assess the applicable technology. If successful, other NASP derived vehicles could eventually replace the shuttle in the task of transporting personnel, instrumentation, and other sensitive cargo to and from low–earth–orbit. As fully reusable vehicles, with the promised ability to takeoff and land at any conventional airport, AHSV's have the potential to make transportation from Earth to orbit an inexpensive and almost routine procedure. [Klu1, Dem2, Bek1]

[1] More than 8000 people and almost one million manhours are required to perform more than 760, 000 separate operations in order to prepare a Shuttle for its next launch.

[2] Shuttle-C and the Advanced Launch System (ALS) are examples of launch vehicles intended for lifting heavy payloads to orbit.

A successful flight of the National Aerospace Plane would represent a remarkable achievement in the integration of today's state–of–the–art technology in a wide range of engineering disciplines. The most prominent characteristic of the NASP vehicle (or an AHSV) is the desired capability for sustained hypersonic flight powered by an air–breathing supersonic combustion ramjet (scramjet) engine. This propulsion system is the key element that would make it possible to achieve single–stage–to–orbit operation. It is also one of the main complications for the NASP, since its performance is highly sensitive to vehicle geometry and orientation. As such, the airframe and propulsion design have to be integrated to a degree never before encountered for other flight vehicles. The planned Earth–to–orbit trajectory of the NASP would also include some of the most extreme flight conditions ever experienced by a manned aircraft. Chemically reacting high speed gases, intense structural loading, and extremely high surface temperatures present a formidable challenge to the aerodynamic, propulsion, structural, and materials engineers. The vehicle must continually operate at the highest possible dynamic pressures to maintain propulsive efficiency while avoiding destructive structural loads or temperatures. The demand for a single–stage–to–orbit capability also imposes conflicting objectives on the vehicle design. Aerodynamic properties that are beneficial for efficient hypersonic flight are detrimental for the subsonic portion of the trajectory. The same difficulty arises for the engine, since the principles of scramjet propulsion are inoperable at subsonic or low supersonic speeds.

Flight control design for an AHSV is a particularly difficult task for several reasons. First of all, for a wide range of hypersonic flight conditions the aerodynamic, propulsive, and control coefficients may not be fully predictable from theory or experimental data. This is because several theoretical aspects of hypersonic flow are not adequately understood, and the high Mach number flight conditions cannot be reproduced in available wind tunnels. Furthermore, due to the coupling between the aerodynamic behavior and the performance of the scramjet engine, the overall vehicle dynamics will be highly nonlinear and very sensitive to attitude changes while at hypersonic speeds. In addition, the aggressive nature of the SSTO trajectory for an AHSV implies that the control system will be required to maintain tight tolerances on the state tracking errors relative to a desired trajectory. The controller will also be limited by a variety of constraints which restrict the range of admissible control actions at any time. These include state constraints, actuator limits and maximum rates, and others (such as bounds on temperature, dynamic pressure, or maximum acceleration), all of which may vary with flight conditions.

In short, the flight control problem for an AHSV is to closely track a desired trajectory from Earth to orbit, given a set of highly nonlinear, multivariable, time–varying dynamics with uncertain coefficients, in the presence of multiple constraints, and disturbances. Moreover, a control strategy is desired for which stability, tracking convergence, and robustness to uncertainty can be guaranteed, and for which a solution can be obtained in real time. Collectively, these objectives eliminate from consideration most, if not all, conventional control system methodologies.

The considerations mentioned above (and discussed further in Chapter 2) motivate the need for an advanced flight control design that can quickly determine the most appropriate control action, while accounting for the system's nonlinearities, uncertainties, and constraints, as well as the future changes in these properties. This thesis presents an approach that utilizes all available *a priori* and in–flight information to perform robust real–time short–term trajectory planning. Stable tracking of a desired trajectory is achieved through the repetitive solution of a receding–horizon nonlinear optimal control problem. A viable correction trajectory is generated, followed for a short interval of time, and then recomputed. The flight control approach combines an enhanced A^* optimization algorithm with a Lyapunov stability criterion in order to perform an intelligent search for a solution that guarantees tracking performance in the presence of uncertainty, while observing all physical constraints.

1.2 Research Objectives

The flight control challenge presented by air–breathing hypersonic vehicles was the motivation for this research effort. The overall objective of this thesis has therefore been to develop representative models and a simulation for AHSV vehicles, design an effective control strategy, and demonstrate its operation in flight using a simulated vehicle. The specific objectives are listed below:

- To construct reasonably complete models to represent the geometry, mass properties, aerodynamics, propulsion, controls, and operating environment of an air–breathing hypersonic vehicle, such as the National Aerospace Plane, in hypersonic flight.

- To develop a computer program for the parametric design and dynamic simulation of an AHSV, and for the evaluation of candidate flight control algorithms.

- To develop a new approach to handle nonlinear multivariable control problems with complex constraints and uncertain dynamics. This approach is based on combining Lyapunov stability theory with intelligent optimization techniques in an algorithm that is suitable for parallel processing.

- To demonstrate the capabilities of this new Robust Intelligent Flight Control method (RIFC) through its application to the flight control problem of NASP type vehicles.

- To evaluate the performance of the RIFC algorithm in a variety of simulated flight experiments.

- To compare the RIFC control algorithm to a compatible, existing, alternative approach based on the concept of Single–Step Optimal Control (SSOC).

1.3 Contributions

- The development of a new Robust Intelligent Flight Control approach that is applicable to vehicles with uncertain nonlinear dynamics, capable of utilizing all available *a priori* and on–line information, and suitable for parallel implementation on advanced flight computers.

- A demonstrated solution to the tracking flight control problem of air–breathing hypersonic vehicles, (including the nature of the dynamics, constraints and uncertainties specific to AHSV's).

- The combination of an enhanced A^* optimization algorithm with Lyapunov stability theory in order to provide stability, convergence, and robustness guarantees, within a prescribed memory limit.

1.4 Organization Of Thesis

Chapter 2 presents some introductory material on hypersonic vehicles and the National Aerospace Plane. Many of the technological challenges associated with the development of the NASP are examined qualitatively. In particular, those issues relevant to the flight control problem are discussed. This chapter also gives a historical perspective on the development of flight control methods, and it concludes with a discussion of the merits and limitations of several approaches in the context of the AHSV application.

Chapter 3 describes the main objectives for a trajectory controller for an AHSV, and it presents a suitable structure for the overall control system. This chapter then gives a complete qualitative description of the Robust Intelligent Flight Control system that is developed in this thesis. In addition, an overview section summarizes the nature of the models and simulation software used to represent AHSV configurations and dynamic behavior. A brief discussion of the scope of this research is also included.

Chapter 4 develops all of the models for AHSV's in some detail. The vehicle geometry, mass properties, and atmospheric models are presented first, followed by the aerodynamic and propulsion models which are more involved and require some preliminary theoretical background. This chapter also derives the full six degrees–of–freedom equations of motion for an AHSV in a spherical rotating reference frame. The dynamic model used by the flight controller is derived from these general equations.

Chapter 5 explains the operation of the simulation software used to design realistic AHSV configurations and to test candidate flight control algorithms. The important capabilities and options are mentioned, and a few specifics of the implementation are discussed.

Chapter 6 provides the theoretical background necessary to understand and to derive the properties of the RIFC trajectory control algorithm. The first section explains the basics of Lyapunov stability theory and leads up to the specific definitions and theorems that are later needed in Chapter 7 to develop the RIFC approach. In the second section, the fundamental concepts of heuristic optimization techniques are discussed with an emphasis on the theoretical properties of the A^* algorithm and its variations. The definitions and theorems given will be required for the following chapters. Also note that the application of an intelligent search approach to the AHSV control problem is largely motivated by the discussion in this section.

Chapter 7 begins by formally stating the AHSV flight control problem. It then presents four propositions which establish the properties of controllability, stability, convergence, and robustness to uncertainty under some reasonable assumptions and conditions. These propositions are verified by supporting proofs, derivations, and physical arguments. The fundamental structure of the RIFC control algorithm is also given.

Chapter 8 discusses several issues relevant to the practical implementation of the RIFC controller. Computer memory and speed limitations are addressed through a number of enhancements to the fundamental search algorithm. A specialized node storage scheme is also developed which can guarantee convergence of the search within a prescribed memory limit. The overall controller logic is described and a parallel implementation of the algorithm is proposed. Other topics include a discussion of the effects of quantization in the states, controls, and time, and an analysis of the numerical complexity of the RIFC algorithm.

Chapter 9 presents the results of simulations using the RIFC controller under a variety of test conditions. Experiments are performed to evaluate and demonstrate the controller's performance in the presence of uncertainty and disturbances. Comparisons are made to the performance of the SSOC controller.

Chapter 10 discusses the conclusions and some suggestions for further research.

Chapter 2

Hypersonic Flight

2.1 The National Aerospace Plane

In January of 1986, in his State of the Union address, President Reagan announced that "*We are going forward with research on a new 'Orient Express' that could, by the end of the next decade, take off from Dulles Airport and accelerate up to 25 times the speed of sound, attaining low–earth–orbit or flying to Tokyo within two hours.*" With these words began the National Aerospace Plane Program, which has as its ultimate objective the development and demonstration of a fully reusable horizontal takeoff and landing (HTOL) aircraft that can accelerate to hypersonic speeds and reach orbit with a single stage. In his speech, the president actually referred to a combination of what is now considered two different research programs. The development of an experimental NASP prototype falls under the heading of the "X–30" technology demonstrator, and will be the first aircraft to actually test air–breathing hypersonic flight. As for commercial transport vehicles, NASA is actually pursuing a separate High–Speed Civil Transport (HSCT) study, but this research will also undoubtedly benefit from technological developments of the NASP program. [Wil1, Voel]

The main objective of the National Aerospace Plane is to develop and demonstrate a cheaper more efficient mode of transportation to low–earth–orbit.

By taking off and landing horizontally, and using atmospheric oxygen to burn propellant, rather than carrying onboard oxidizer, air–breathing hypersonic vehicles (AHSV's), such as the NASP, are intended to operate more like conventional aircraft than like rockets. The savings in oxidizer weight and the benefit of aerodynamic lift are the principal advantages of this design. The additional objective of single–stage–to–orbit operation means that NASP derived vehicles could benefit from a fully reusable design. This translates into minimal maintenance and ground support requirements when compared to previous launch support vehicles. These operational advantages are expected to drastically reduce the cost of access to LEO, as well as to increase the frequency at which flights can be launched, thus rendering the mission to orbit a routine operation. [Key1]

The National Aerospace Plane is only one of a wide range of possible hypersonic vehicle concepts that have all contributed to the recent resurgence of interest in hypersonic research [Han1]. Several other applications that may also involve some degree of hypersonic maneuvering, include reentry vehicles, spacecraft for interplanetary missions, and other vehicles or probes that employ aero–braking or aero–capture maneuvers. For many space missions, significant fuel savings are possible if atmospheric maneuvering can be used to effect orbital changes. One type of vehicle, known as a Hypersonic Waverider, has recently attracted special attention due to significant advances made in hypersonic fluid dynamics. By taking into account viscosity effects in hypersonic flow, a new method for tailoring vehicle geometry to optimize aerodynamic performance has been developed. The resulting hypersonic designs exhibit exceptionally high lift–to–drag ratios. This has generated much enthusiasm for a myriad of ambitious planetary missions which involve hypersonic maneuvering through the atmosphere of one or several planets. For such missions, and for many others, extraordinary benefits are possible, (in terms of fuel savings, mission flexibility, maximum payload, time, and cost), given the capability to operate a spacecraft under the extreme conditions of hypersonic flight. [And2, Lew1, Bow2, Cor1]

There are actually several hypersonic vehicle concepts that would directly benefit from NASP research experience. In fact, many vehicles can be viewed as scaled–down variations of the NASP with different, and usually less complex, mission requirements. For example, the military is interested in developing a new high–speed interceptor, as well as in replacing the now retired SR–71 fleet with new hypersonic reconnaissance vehicles [Voe1]. Although these aircraft will have their own design difficulties, their mission is less demanding than the Earth–to–orbit trajectory required by an AHSV. The Hypersonic Waverider for interplanetary exploration is another good example. This vehicle would operate in a similar environment to the AHSV, and experience gained with hypersonic aerodynamics, materials, and controls, would be directly applicable. The waverider, however, would not be as complex as the AHSV in the sense that it would not use an air–breathing propulsion system, (it is essentially a glider), and it would not be required to operate at the lower Mach numbers [Lew1]. As a final example, the next generation of commercial transport vehicles would also certainly benefit from NASP research, particularly in the areas of structures, materials, and propulsion [Dri1, Dem2, Lea1]. Since the challenge of designing an AHSV is, in several ways, a more difficult problem than many of the other hypersonic applications, the NASP is perhaps a vehicle that is representative of the technology required for a whole class of new advanced flight vehicles.

Despite the promise of all these applications, much research is still needed in the areas of materials, structures, aerodynamics, hypersonic gasdynamics, and flight controls, before many vehicle concepts can actually be tested. Although hypersonic flight spans a wide range of Mach numbers and atmospheric densities, the flow conditions surrounding a vehicle throughout this flight regime are, in general, quite inhospitable to materials and mechanical systems. Whether a vehicle accelerates to hypersonic speeds from the surface, or it enters the upper atmosphere of a planetary body at orbital speeds, it would be subject to extreme temperatures and high structural loading. Design tolerances for any type of hypersonic vehicle are likely to be stringent, and unfortunately the physics of hypersonic flow is not yet understood well enough to predict local flow properties with a high degree of accuracy. [Kor1]

2.2 The NASP Technological Challenge

The development of the National Aerospace Plane presents a tremendous technological challenge to the aerospace community. Never before has any vehicle depended so critically on the combination of the latest advances in so many fields of engineering. As such, there are many significant obstacles and issues which complicate the development of the NASP. Most of these difficulties can be attributed to factors related to its required flight trajectory, characteristics of the hypersonic environment, single–stage–to–orbit operation, and the integration of the aerodynamic and propulsion design. [Gre1]

2.2.1 NASP TRAJECTORY

The objective of an Earth–to–orbit trajectory is a major element contributing to the design challenge of the NASP vehicle. To successfully fly as a winged aircraft from a low–speed horizontal launch to the altitude and velocity required for orbital flight, the NASP must perform effectively at subsonic, transonic, supersonic, and hypersonic speeds. These very different flight regimes impose conflicting objectives on the aerodynamic and propulsive design [Joh1].

For low (subsonic) speeds, best performance demands a high aspect–ratio configuration for sufficient lift, and an engine equipped with a compressor to generate adequate thrust. At higher (supersonic) speeds, shock waves and high dynamic pressure encourage lower aspect ratios, larger wing sweep angles, and a tapered fuselage design to reduce wave drag penalties. Then, once the vehicle reaches a flight condition with a high enough dynamic pressure, (around Mach *3*), it becomes possible to use a ramjet configuration for propulsion. This type of engine uses the air ram–pressure to compress the flow through a normal shock wave, rather than requiring a compressor prior to mixing and combustion. Finally, at still higher Mach numbers (hypersonic flight), the aerodynamic and propulsive design is complicated by extremely high temperatures, high dynamic pressure, and a chemically reacting ionized flow. Issues such as shock impingement, aerodynamic shadowing, structural loading, boundary layer thickness, laminar/turbulent transition, as well as cooling considerations, dominate the design requirements. In this flight regime, the propulsive efficiency of a ramjet sharply declines and the capability for supersonic combustion is required in order to have air–breathing propulsion.

To meet its objectives, the NASP design will clearly have to balance the tradeoffs between all of these flight regimes. That is, in order to achieve best overall performance, (e.g. payload to orbit, range, etc.), design parameters must be chosen to optimize cumulative measures covering the entire mission from Earth to orbit. The final aerodynamic/propulsive configuration, as well as the vehicle's flight trajectory, must therefore depend not only upon mission requirements, but also upon opposing design considerations associated with the various flight regimes that will be encountered. [Ada1, San1]

2.2.2 THE HYPERSONIC ENVIRONMENT

Extended operation under hypersonic flight conditions is another major challenge for the NASP. Thermal management and structural integrity are primary issues to be addressed. The hypersonic environment is characterized by ionized chemically reacting gases and extremely high temperatures. Due to these conditions, there is a delicate tradeoff between airframe and propulsion system requirements. For best propulsive efficiency, a trajectory of high dynamic pressure is most desirable, however, this same trajectory threatens the vehicle's structural integrity, and is limited by airframe thermal and structural constraints. New advanced materials capable of handling temperatures reaching up to *3000* °F, while supporting loads as high as 1500 psf, will be required. On the vehicle nose and leading edge surfaces temperatures may exceed *4000* °F, suggesting the need for some form of active cooling to handle thermal loading. [New1, Jac1, Ten1]

Extreme temperatures and pressures also present a serious problem for external sensors as well as control actuators. New instruments will be required to measure important flow properties without extending probes into the flow.[1]

[1] Traditional instruments such as pitot tubes or other protruding sensors would experience extremely high surface temperatures and could not survive in this environment.

Devices currently under development include laser–based optical sensors, and indirect sensors based on heat flow measurements. Aerosurface loading during hypersonic flight would exceed that experienced by any previous aircraft. In addition, some degree of aerodynamic shadowing[2] is unavoidable at hypersonic speeds, and this will adversely affect the control authority of aerosurfaces while operating in this flight regime. Other control actuators will therefore be required to offset aerosurface loading and to provide needed control authority at the higher Mach numbers. Thrust vectoring and/or reaction control jets will probably be required. [Kan1, Hon1, Mil3]

A final problem related to the hypersonic environment is the modelling and prediction of aerodynamic and propulsive behavior. At low speeds one can simplify the analysis by treating air as an inviscid fluid. At hypersonic speeds, however, viscosity becomes much too important to ignore. In this flight regime, viscosity is sensitive to flow chemistry as well as to local temperatures, and it is directly responsible for the character of the flow within boundary–layers both external to the vehicle and internal to the engine. As such, it has a profound influence upon the aerodynamic, propulsion, and control coefficients while in hypersonic flight. Theoretical modelling of hypersonic flow is very difficult, as it must account for viscosity, heat transfer, and chemistry, in addition to the complexity of a three–dimensional high Reynolds number compressible supersonic flow. Many aspects of hypersonic flow are still not fully understood, such as turbulence, fuel/air mixing, skin friction in a mixing/reacting flow, finite–rate chemistry, and the effects of heat transfer, pressure gradients, three–dimensionality, chemical reactions, and shock waves on the transition of hypersonic boundary layers.

To further complicate matters, design evaluation for the NASP will have to be done using CFD (Computational Fluid Dynamics) numerical codes, since hypersonic flight conditions beyond Mach 8 are not fully reproducible in currently available

[2] Aerodynamic shadowing refers to a condition where part of the vehicle blocks the flow which would otherwise encounter other downstream surfaces. This is particulary significant at hypersonic speeds where the mean free path of molecules in the free stream can take on dimensions of the order of the vehicle size. The degree of shadowing depends upon Mach number, Reynolds number, and vehicle orientation.

wind tunnels. The state–of–the–art CFD hypersonic modelling codes are capable of accounting for effects due to chemistry, shock/boundary–layer interactions, flow spillage for scramjet engines, three–dimensionality of the flow, and more, but several important factors have not yet been incorporated. Current research in CFD is aimed at understanding and modelling phenomena such as hypersonic boundary–layer transition, turbulence, and combustion flow chemistry. Validation of hypersonic CFD codes is another serious problem since very little empirical data is available for the higher Mach numbers. As a result, CFD codes may not be correct for some portion of the flight envelope.

It seems that, at best, predictions of aerodynamic (and propulsive) behavior will be approximate for flight at hypersonic conditions. This, in effect, complicates the NASP development since adequate performance must be assured in the presence of potential modelling errors that cannot be resolved until the vehicle is actually tested in flight. In fact, an evolution of the vehicle design may be necessary as the operational flight envelope is expanded. [Dwo1, Pov1, Dav1].

2.2.3 SINGLE–STAGE–TO–ORBIT

Additional challenges to the development of the NASP can be attributed to the objective of having a single–stage–to–orbit vehicle. The main purpose of this goal is to assure maximum reusability while minimizing the operational complexity and expense associated with disposable hardware and refurbishing. A NASP derived SSTO transport vehicle could allow a quick turn around time between flights, minimal ground support requirements, and (not least of all) the elegance and simplicity of a single one–piece aircraft capable of routine flight between Earth and space. Unfortunately, however, the SSTO requirement presents a serious obstacle to the design of AHSV's due to several other factors.

Having only one stage, the NASP must carry the entire propellant load within the volume of the spacecraft itself. In addition, the propellant will necessarily be hydrogen, since this is the only fuel known that can burn quickly enough for hypersonic air–breathing propulsion, and has a high enough specific impulse to lift itself plus a payload into low–earth–orbit. One problem is that some preliminary estimates indicate that the required mass of fuel required for an Earth–to–orbit trajectory may exceed *90* % or more of the vehicle's gross takeoff weight. This figure implies that the payload lift capacity of the NASP may be severely limited, and that the vehicle mass properties will change dramatically (as fuel is burned) over the course of a flight, thus creating potential problems for flight control. In addition, the low specific density of condensed hydrogen creates a conflict between the fuel volume requirements and the objective of minimizing projected frontal area to reduce aerodynamic drag. To offset these difficulties, the NASP design will clearly have to emphasize the development of extremely lightweight (high strength & heat resistant) materials for the airframe structure, as well as the achievement of the highest possible propulsive efficiencies to reduce fuel requirements. The storage of hydrogen fuel in the form of a cold slush (rather than as a liquid) is also under consideration to help alleviate these problems. [Marl]

Other flight systems on the NASP are also burdened by additional overhead with the requirement of a SSTO design. For example, over most of the trajectory the propulsion system is required to deliver sufficient thrust to continually accelerate the vehicle. The scramjet engine is expected to provide this propulsion over a wide range of Mach numbers (from Mach *6* to perhaps Mach *25*), however, it is not capable of operation at the lower Mach numbers, nor is it capable of boosting the vehicle into low earth orbit. In effect, a turbojet type engine mode will be required to accelerate the craft from a runway up to supersonic speeds (Mach *2* or *3*), at which time a ramjet propulsion mode could be used to accelerate up to about Mach *6*. The scramjet can then potentially attain velocities close to that required for orbit, but some form of rocket will be required for orbital insertion. Clearly a vehicle design which carried four different engines for each of the four flight regimes would not be very practical. Ideally, one multi–mode engine would be designed to operate as a turbojet, ramjet, scramjet, and rocket engine, depending upon the current flight condition. Although much research has been done to investigate the feasibility of combined cycle engines, their potential is difficult to assess in the context of this thesis due to security restrictions. [Marl]

Another SSTO design challenge is the control actuator requirements for flight through a wide range of environmental conditions. Just as with the propulsion system, a multitude of actuators will be needed to assure adequate control authority over the entire trajectory. Aerosurface controls would be useful for subsonic, supersonic, and part of the hypersonic segments of the flight. For phases of the trajectory which include very low dynamic pressures, however, the aerosurfaces would be quite ineffective. For on–orbit attitude control of the NASP, a number of jet thrusters, (like those used for the Shuttle), will be required at strategic points around the vehicle exterior. These jets could also be used for attitude control on ascent at the higher altitudes to supplement aerosurface controls, and on reentry until aerosurface control authority is recovered. At the lower altitudes, however, attitude control jets would not be practical for two reasons: the jets would waste fuel while aerosurfaces could meet control requirements; and the disruption to the external flow could cause unpredictable effects on the aerodynamic and propulsive behavior of the vehicle. Finally, some degree of thrust vectoring will most likely be needed in order to provide control authority in flight conditions for which the aerosurfaces experience shadowing, and to account for the changing vehicle dynamics as fuel is used and flight conditions change.[3]

2.2.4 AERODYNAMIC & PROPULSIVE COUPLING

A final characteristic of the NASP that is a major factor contributing to its technological challenge is the close integration of the aerodynamic geometry and the propulsion system design. In sharp contrast to other aerospace propulsion systems currently in use, the performance of a scramjet engine is strongly dependent upon the shape and orientation of the vehicle which it propels. The flow entering the engine is initially compressed through oblique shocks emanating from the undersurface of the entire forebody of the vehicle. The inlet Mach number, flow density, pressure, temperature, and boundary layer characteristics are completely determined by the external vehicle geometry and attitude with respect to the free stream. Likewise, the geometry of the aft undersurface acts as a nozzle and completely determines the expansion of the flow exiting from the engine. The magnitude and direction of the thrust vector are critically dependent upon the

[3] Time-varying aerodynamic, propulsion, and mass properties will affect the center of mass (CM) center of pressure (CP), and the aerodynamic/propulsive pitching moments.

properties of this expansion, (as is the propulsive efficiency). The performance of the scramjet engine is therefore directly related to the geometry of the vehicle. Of course, the airframe geometry is also the primary influence upon the aerodynamic behavior, and as a result the propulsion system and the aerodynamic configuration cannot be designed independently. This coupling between aerodynamics and propulsion is also troubling in the context of maneuvering the vehicle in flight. In general, aircraft attitude and engine thrust are used to control speed and altitude. For the NASP, these inputs cannot be controlled independently since the propulsive performance is sensitive to vehicle attitude. [Wei1, Edw1, Wal1]

2.3 Flight Control For Hypersonic Vehicles

For many reasons, the flight control problem for AHSV's (such as the NASP) goes far beyond the difficulties encountered with previous aircraft and spacecraft. The control obstacles are related to many of the same issues already discussed in the preceding section. The vehicle design (aerodynamics/propulsion), the desired trajectory (high dynamic pressure, high temperature, SSTO), and the performance requirements (tracking, tolerances, constraints, robustness), all contribute to the challenges faced by the flight control system. Just as with many other high performance aircraft, the AHSV dynamics exhibit a nonlinear, multivariable, time–varying, nonminimum phase type of behavior. But while various assumptions, approximations, and techniques are often applicable to simplify the control task for other vehicles, certain characteristics of the AHSV seem to defy most approaches to the problem.

In the past, it has been common practice to use linearized (approximate) equations of motion to represent an aircraft's dynamic behavior. Since the airspeed and altitude are typically the main sources of variation in the dynamics, the linearized model and/or the parameters of the control system are often scheduled as a function of the flight condition. This works well enough for many applications, since the operating range of the other important variables is often small enough that the linearized approximations are justified. The advantage of using a linear model is, of course, that a wide range of analysis and design tools are available for linear systems.[4]

In problems with severely nonlinear or quickly time–varying dynamics, however, a linearized approach would fail since the system would continually operate outside the applicable range of the linearization. High performance flight at high angles of attack is a good example of a current research area in which the dynamics are extremely nonlinear and not well suited for a linearized treatment.

With AHSV's, the difficulty is related to the highly coupled nature of the aerodynamics and the propulsion system. Forces and moments from the scramjet engine are directly related to external flow properties in the vicinity of the forebody and aft surfaces of the vehicle. In these areas shock angles, boundary layers, flow temperatures, pressures, and densities will all vary with the angle of attack. This creates a highly nonlinear dynamic dependence on vehicle attitude which changes on a time scale which is much faster than that of the usual scheduling variables (such as velocity and/or altitude). Considering both the time scale and the severity of the attitude nonlinearities for an AHSV, a linearized approach to the flight control problem is not practical. This tends to complicate the design of a flight control system for AHSV configurations, since the available tools for nonlinear multivariable systems are very limited.

Another AHSV characteristic that distinguishes it from most other classes of aircraft is the number of critical constraints which must be observed during the course of a flight. An aggressive trajectory from Earth–to–orbit is required for an AHSV to sustain optimum propulsion efficiency and achieve orbital velocities. But structural integrity and passenger comfort impose limitations on the maximum dynamic pressure, external skin temperatures, and maximum acceleration. Mission objectives, therefore, call for routine operation at or near these constraint

[4] Properties of linear systems, primarily superposition, make possible the use of Laplace Transform Theory, eigenvalue/eigenvector analysis, pole placement algorithms, all frequency domain methods, such as Bode plots (gain and phase margins), Nyquist methods, Singular value analysis for multivariable systems, and more.

boundaries. In addition, tight tolerances in tracking a preprogrammed trajectory will be important, since time–varying dynamics (e.g. changing mass properties) could prevent a large deviation from being correctable. This implies constraints on the maximum allowable tracking errors in velocity, flight path angle, and angle of attack. Limitations on scramjet engine performance will also impose constraints on the angle of attack to assure sufficient combustion pressure, prevent engine un–starts, and avoid excessive flow spillage at the inlet. Note that the constraints mentioned above are *not* typically encountered on other high performance aircraft. The additional complexity for an AHSV can be attributed to its unique mission objective of accelerating to near orbital speeds using an air–breathing propulsion system.

Another, more common, type of constraint involves the physical limitations of the actuators used to execute the commands of the control system. Aerosurfaces have a limited angular range as well as maximum deflection rates. Similar limitations exist for throttle controls, thrust vectoring, and attitude control jets (if available). Although these constraints are common to any flight vehicle, the AHSV controls involve other complications. The hypersonic flight environment introduces additional factors which influence the behavior of the controls. In this flight regime, local temperatures can have a profound effect upon local pressures and boundary layer transition, which in turn affects the performance of partly embedded aerosurfaces. Likewise, these same effects influence engine behavior (and thus engine controls). Furthermore, aerodynamic shadowing can occur, reducing the effectiveness of control surfaces that are blocked by other parts of the vehicle. All of these effects are strongly dependent on the vehicle's attitude, and so the control authority for each actuator tends to be a highly nonlinear function of not only control input, but also of the vehicle state.

Perhaps the most troublesome obstacle for an AHSV flight control system is that the vehicle aerodynamic, propulsion, and control coefficients will be uncertain for much of the operational envelope. For most high performance aircraft to date it has been possible to verify the design using theoretical, numerical, and/or experimental methods. Subsonic and supersonic aircraft operate in a flow environment which can be approximated by assuming an ideal inviscid gas with a thin viscous boundary layer attached to vehicle surfaces. This makes it possible to perform crude theoretical calculations or a more refined numerical analysis of aerodynamic behavior. In addition, particular designs can be carefully evaluated in wind tunnel experiments before testing a vehicle in actual flight. Furthermore, huge quantities of empirical data are available from flight experience with previous aircraft. This information can be used to design and predict aerodynamic behavior for new vehicles.

For AHSV's, however, much of the planned flight envelope will include flight conditions for which none of the above holds true. The theory of hypersonic flow is not completely developed, there is little or no empirical data in this flight regime[5], and it is beyond the capabilities of available wind tunnels to fully reproduce very high Mach number conditions (above Mach *8*). Most of the analysis will therefore have to be done using numerical flow codes, which are at best approximate, since (as discussed earlier) many aspects of modelling hypersonic flow are still poorly understood. Moreover, the numerical analysis is very expensive computationally due to the extremely high Reynolds numbers involved. As a result, the flight control system for an AHSV will probably have to operate using a model of the system which includes some uncertainty in the parameters. Robustness to this uncertainty will be an important factor in the design.

From all the preceding discussion it is evident that the National Aerospace Plane presents a unique challenge to the aerospace community on many fronts. In particular, the extreme nature of hypersonic flight and the special problems posed by AHSV designs motivate the need for a new advanced flight control strategy capable of handling highly nonlinear dynamics with complex physical constraints, while accounting for parametric uncertainties, in the setting of a real–time application.

[5] The Space Shuttle does operate as a winged aircraft at hypersonic speeds during reentry, however it is an unpowered glider and follows a trajectory through flight conditions that are different from those that would be experienced by the NASP.

2.4 Previous Flight Control Experience

Historically, aeronautical applications have been by far the most significant motivating force behind the development of Control Theory. One simple reason for this might be that, for aircraft, adequate control is a necessity to avoid disaster. In general, however, flight control has always been a special problem in several ways: the dynamics of aircraft are relatively complex, the response to inputs and disturbances is relatively fast, the system behavior is variable and depends upon flight conditions, and instability can lead to the loss of a vehicle and possibly human lives[6].

Flight control really began with the first successful powered flight at Kitty Hawk on December 17, 1903. The Wright brothers were the first to realize that an efficient control system was the key to preventing the kinds of accidents that plagued (and killed) their predecessors[7]. Eventually, improvements in engine efficiency made it possible for aircraft to access a wider range of speeds and altitudes. By 1944, the first turbojet driven aircraft was fully operational. Then in the 1950's, with the demand for increased performance over an ever expanding flight envelope, it became apparent that vehicle behavior significantly changed as a function of flight condition. It was in this era that the concept of Adaptive Control really began. (Classical Control Theory had already been developed and was in use for simple regulating autopilots). An additional design objective for the autopilot now included compensation for plant variations with flight condition. The first "adaptive" controllers were systems that simply adjusted single–loop compensator gains according to air–data measurements. [Kre1, Ryn1, Bar2, Kal1]

[6] In sharp contrast, the regulation of room temperature, or the control of chemical concentration in a brewing process, are slowly changing non-critical processes with invariable behavior.

[7] In the 1890's, Otto Lilienthal was the first man to achieve sustained flight in an unpowered glider. After nearly 2000 flights, he crashed when a gust of wind stalled one of his wings. In 1900, the Wright brothers pioneered the use of wing warping to prevent stall and to control bank angle. They also included elevators and a rudder in later designs to control their aircraft in pitch and yaw. By 1902, they had a completely controllable glider. Their first powered flight (1903) was piloted by Orville, and flew 200 ft.

Eventually, two distinct approaches to the problem emerged. The first is known as Gain Scheduling, and is essentially the blending of several autopilot designs as the flight conditions change from one regime to another. Parameters of the control system are optimized to yield the best performance at several operating conditions within the flight envelope. These parameters are then stored, and later retrieved during actual flight. Typically, Gain Scheduling is done using a linearized model of the dynamics for different speeds and altitudes. In most cases, a linear feedback law with adjustable gains is used to effectively position the closed–loop system poles to achieve desired performance specifications[8]. These gains are then scheduled as a function of flight condition (ie. control gains are interpolated for current speed and altitude).

The other approach is the formal concept of Adaptive Control, in which the control system is able to recognize that the dynamics are changing, and in some way adapt to these changes on–line. Adaptive Control is subdivided into two categories, namely direct and indirect methods. Indirect Adaptive Control divides the problem into two parts: one of identifying some changing parameters of the dynamic model, and one of using the best estimate of the model to compute gains for the controller. Direct Adaptive Control involves directly identifying parameters of the controller to yield the best performance. In either case, adaptive control always involves the on–line adjustment of internal parameters in conjunction with providing feedback control. Adaptive control is an enticing concept because of its "black box" nature; that is, the idea that an ideal adaptive controller might be able to control a class of systems without any a *priori* knowledge of the dynamics.

In the late 1950's, Model Reference Adaptive Control (MRAC) was introduced by H.P. Whitaker at MIT [Whi1, Whi2]. This very popular method allowed an autopilot to minimize the error between the aircraft behavior and the response of an ideal aircraft model. The controls were chosen in such a way as to reduce the error in the direction of its negative gradient. Unfortunately, this algorithm was not based on any theoretical arguments, and it was later shown that a MRAC autopilot

[8] Such as gain margin, phase margin, overshoot, response time, etc...

could be destabilized with certain inputs [Ast4, Par3]. This realization inspired much of the more recent research in the stability theories of Lyapunov and Popov. Meanwhile, the promise of adaptive control was appealing, and through the 1960's experimental research in this area was actively continued with aircraft like the F–4, F–101, F–111, and the X–15 [Hal1, Ste1]. It was during this time that many bad experiences, and an X–15 flight test that ended in complete disaster, gave adaptive control a poor reputation [Ryn1].

Eventually, Gain Scheduling became the preferred method of flight control for high performance aircraft. This was particularly true because of improvements in the design of sensors for measuring air–data, and better predictions of aerodynamic derivatives from wind tunnel tests [Ste1]. With Gain Scheduling, autopilots could be designed to operate effectively and reliably over a wide range of flight conditions without the risk of instability.

In the 1970's, the trends toward digital flight computers, redundant systems, increased performance requirements, and wider flight envelopes, shifted the emphasis once again. By this time, major developments in control theory had been made in the areas of stability theory, system identification, estimation, stochastic control, state–space design and analysis, and more. The MRAC method had been redeveloped using Lyapunov stability methods, and interest in adaptive control resurfaced [Par3]. Many flight control concepts of this era involved some type of parameter estimation or system identification coupled with a model–following or other feedback control algorithm [Ise1]. Several algorithms of this type, with a variety of identifier and controller combinations were demonstrated in simulations during this period [Kau1, Kau2]. In the early 80's, however, it was discovered that unmodelled dynamics and additive disturbances could cause instability in these systems [Roh1, Roh2], and this generated interest in a new topic called Robust Control Theory. Major results were obtained in this area during the 1980's (see for example references [Mor1, Nar1, Loz1, Nar2]).

Current research in flight control emphasizes many areas. More general algorithms for the design of multivariable control systems are of interest, particularly for systems having some kind of structured uncertainty. Nonlinear control is also a very active area because of the trend toward high angle of attack maneuvering for advanced fighter aircraft[9]. The use of digital computers for the implementation of most current flight control systems has spurred much investigation in the area of discrete–time control for multivariable systems [Goo1, Goo2, Lan1]. Adaptive control activity is directed at achieving results for multivariable systems, and toward robust adaptive algorithms [Bit1, Nar3]. Other research areas in adaptive control, with applications to flight control, include control of partially known plants, or systems with bounded parametric uncertainty [Cla2, Oss1].

2.5 Control Methods & Limitations

At present, the state–of–the–art in control theory can handle a wide range of possible problems. For linear multivariable systems, there are control methods to deal with additive disturbances, measurement noise, and even structured parametric uncertainty. For nonlinear systems, there are several approaches to handle single–input–single–output (SISO) problems, or problems with only one nonlinear component. There are even robust nonlinear control methods for SISO systems or restricted multi–input–multi–output (MIMO) systems.

In the context of the flight control system for the National Aerospace Plane, however, none of the available control methodologies are really able to deal with the full complexity of the problem. A large number of control design techniques make use of frequency domain concepts such as pole placement, Nyquist stability, and Bode plots, and there are now MIMO extensions of these classical methods based on singular value concepts. Another main category of control design methods are based upon quadratic minimization of a state–space performance measure, such as the Linear Quadratic Gaussian – Loop Transfer Recovery (LQG–LTR) approach, and its many extensions. Robust variations of these techniques have also been

[9] Applications in robotics and many other areas have also motivated much research in nonlinear control theory.

developed for systems with modelling uncertainty. Less conservative approaches such as μ–synthesis can be used to handle problems with structured uncertainty. Unfortunately, the common feature of all of these control system design methods is that they all depend on the assumption that the system can be approximated as a linear time–invariant plant over some useful range. As discussed above, this is not a reasonable assumption for the AHSV problem.

Optimal control theory can, in general, address nonlinear problems; the difficulty is to find a practical method for solving a nonlinear constrained optimization problem on–line. Adaptive control algorithms do exist for multivariable systems, however, current results are limited to a restrictive class of problems. Nonlinear design methods for multivariable systems are extremely limited. Input/Output linearization is one interesting approach that transforms a nonlinear problem into an equivalent linear one. Several restrictions apply on the original system and there is no direct way to map control constraints, or parametric uncertainty, to the equivalent system. Another approach for nonlinear systems is the so–called Sliding Mode Control (SMC) method. The advantage of Sliding Mode is that it can track a desired trajectory, and is robust to modelling uncertainty. It is also applicable to a restricted class of square multivariable systems. Unfortunately, the SMC technique requires the nonlinear system to have a few special properties, (for example – the controls must enter the dynamics in a linear fashion $\dot{\underline{x}}(t) = f(\underline{x},t)+g(\underline{x},t)\underline{u}(t)$), and the AHSV dynamics do not meet these conditions. In short, the combined difficulties of highly nonlinear MIMO dynamics, modelling uncertainty, and complex physical constraints, render most modern control methods ineffective for the AHSV flight control problem.

Chapter 3

Intelligent Control Approach

3.1 Flight Control Objectives

It is clear from the discussion in the previous chapter that, for many reasons, flight control for air–breathing hypersonic vehicles is a formidable challenge. Before presenting a candidate approach to solving this problem, it is useful to summarize some of the important issues that any controller proposed for this application would have to address.

The most apparent AHSV flight control obstacle is the fact that the system dynamics are severely nonlinear. Aerodynamic, propulsive, and control coefficients, and therefore the vehicle dynamics, are strongly dependent on vehicle attitude; as will be demonstrated dramatically in Chapter 9, a linearized model of the form $\dot{x}(t) = Ax(t)+Bu(t)$ would experience parametric variations (Aij, Bij) on a time scale similar to some of the states. It would, therefore, not be reasonable to design an AHSV trajectory control system based on a locally linearized model. The controller should be able to account for the nonlinear behavior as part of its design. Moreover, the expectation of strict performance requirements, suggests that an approach that deals with these nonlinearities directly would be preferable to one which bounded them in some overly conservative fashion.

Constraints are another important issue to be addressed by the controller. As discussed earlier, several types of constraints exist for AHSV's. Each control actuator is limited by a bounded range and maximum rates, and the states (velocity, flight path, pitch rate, and attitude) are constrained by tracking error tolerances and other physical limitations. Additional dynamic constraints are imposed, such as dynamic pressure, and maximum acceleration. For an AHSV, it is critical that these constraints are carefully observed by the controller, because the vehicle will often operate in flight conditions at or near these boundaries. In addition, the nature of the AHSV trajectory from Earth to orbit amplifies the problem of dealing with these constraints. It is **not** good enough to simply provide a feedback law which avoids all state and control constraints at the current state, since that would **not** guarantee that (in the process of tracking a desired trajectory) constraints can continue to be avoided in the near future. Ideally one would like an autopilot that is capable of "looking ahead" to assure that control actions taken now will lead to an entire error–correcting future trajectory that is admissible in the states and controls.

Another important objective is to have a control system for which vehicle stability and tracking convergence are also guaranteed. If a linear approach to the problem were considered, the fact that some coefficients in the linearized dynamics might change as quickly as some of the states, presents a problem for stability analysis. A linear, locally stable, feedback law may result in instability due to the fact that the linear dynamics are time–varying. A nonlinear control approach would anticipate the dynamic variation with attitude. The only other time–varying components for an AHSV are related to the slow change in vehicle mass and inertia matrix due to fuel expenditure[1]. In any event, instability is clearly intolerable since it would most likely mean loss of the vehicle. However, guaranteeing closed–loop stability for a nonlinear MIMO system, with modelling uncertainty, and external disturbances, is nontrivial. For an AHSV, it can be further complicated by the requirement that the vehicle track a preprogrammed trajectory. Stability and convergence in tracking would then be critically important as well. Given a

[1] The dynamic variation with flight condition, (ie. Mach number and altitude), is, in a sense, time-varying over the course of a trajectory, but it is really a slowly changing state dependency.

disturbed state, offtrack from the desired trajectory, the flight control system would have to be able to find a short term restoring trajectory with the guarantee of convergence at some future time. Nonlinear dynamics, multiple constraints, parametric uncertainty, and a desired trajectory which is no doubt time–varying, make this a very ambitious requirement.

A final control objective is the robustness of stability and tracking performance to the presence of modelling uncertainty. The type of uncertainty expected for the AHSV dynamics is primarily due to the inability to accurately predict the aerodynamic, propulsion, and control coefficients, for the hypersonic flight regime. Since flow below around Mach *6* is well understood and can be reproduced in wind tunnels, accurate values for these coefficients, as a function of Mach number, altitude (density), and vehicle orientation, can be obtained for the lower speeds. As the Mach number (and flow temperature) increases, the uncertainty in any ground based (a *priori*) predictions will increase. For the AHSV, it is critically important that the flight control system be robust to these uncertainties.

One way to improve performance is to try to identify these coefficients while the vehicle is flying (as in adaptive control). However, for best performance, and for the most reliable operation, it would be wise to combine the best available a *priori* predictions of these coefficients with any additional information that can be obtained on–line. An important advantage of this approach is that any on–line identification can be checked against predictions to assure that values fall within reasonable ranges. Predictions and uncertainty bounds for dynamic and control coefficients could be tabulated as a function of flight condition (Mach number, altitude, angle of attack). The objective then would be to design a flight control system that is robust to parametric uncertainty within these bounds, and which can provide better performance if the uncertainty can be decreased using on–line information. This structure is also consistent with the most recent adaptive control philosophy (see [Roh3]) of coupling a robust control design with an identification scheme that works to reduce the set of plants for which the controller must be stabilizing.

Based on consideration of all the control objectives a conceptual design for the overall flight control system is defined. The structure shown in Figure 3.1 represents, in a rough sense, a combination of an indirect adaptive control scheme with a scheduling approach in order to obtain the best properties of each. Control gains are not tabulated, however. Instead, a *priori* estimates and uncertainty bounds are stored for all of the aerodynamic, propulsion, and control coefficients. These estimates can be combined or compared with parameter values obtained from on–line estimates. The results could be used to obtain an improved instantaneous model, or to reject spurious on–line information. This table, therefore, provides both a reliable worst–case parameter envelope, and a means for cross checking on–line identification. A robust nonlinear control design can then determine its response on the basis of the best available information at any given time. When parametric uncertainty is reduced due to the use of new information, performance should improve.

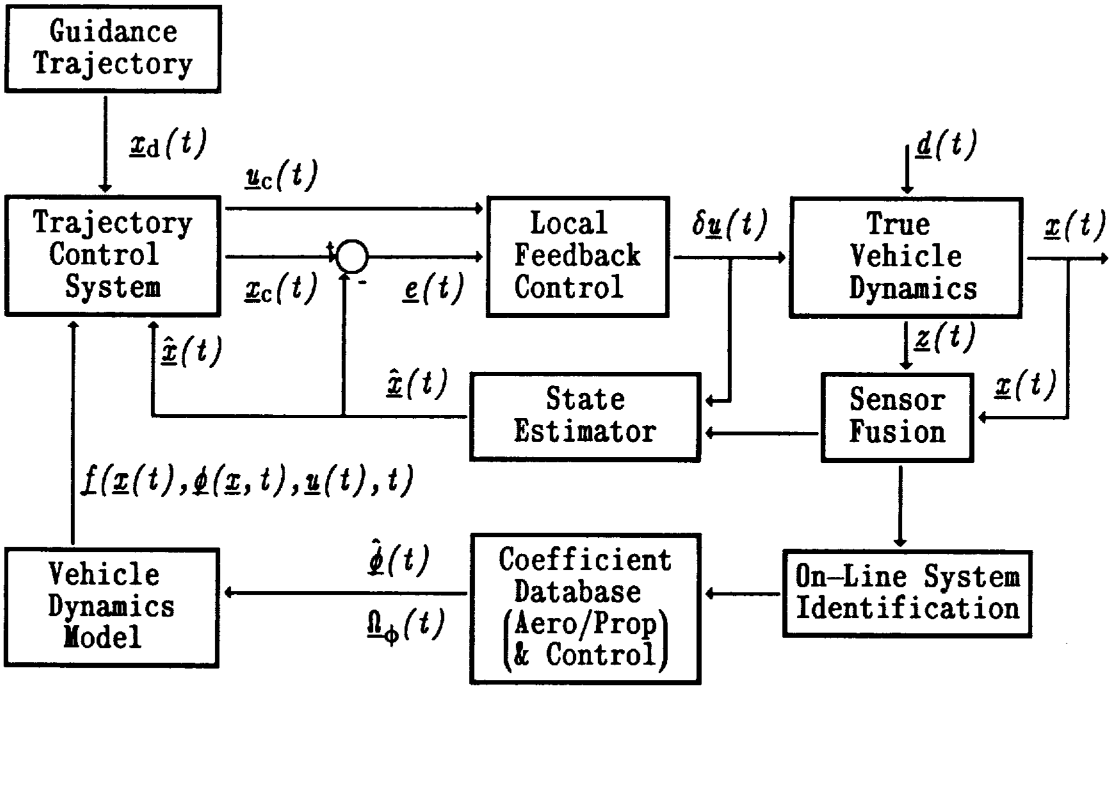


Figure 3.1: Conceptual Design for an AHSV Flight Control System

In Figure 3.1, the **Guidance Trajectory** represents the desired flight path from Earth–to–orbit for an AHSV. This path is essentially a complete time–history for each of the state variables. It might be precomputed off–line as the solution to a global optimization problem, or it may be computed (or adjusted) on–line, depending upon the approach taken for the overall guidance scheme. The problem of tracking this desired trajectory is addressed by the **Trajectory Control System**. This block solves a short–term trajectory planning and control problem based on the desired trajectory, the current state, and a model of the vehicle dynamics. The objective of the **Trajectory Control System** is to find a *viable*[2] control solution to correct for any deviations from the desired guidance trajectory. This problem is solved repeatedly, and the solution is treated as a low frequency feedback law for the stabilization of tracking error. Depending upon the bandwidth of the **Trajectory Control System** and the nature of the disturbances, a **Local Feedback** Control subsystem may be needed for the rejection of higher frequency disturbances. The **Vehicle Dynamics Model** is used by the **Trajectory Control System** to predict the system's response to candidate control input functions. This model includes the full nonlinear equations of motion, and is parameterized by the aerodynamic, propulsion, and control coefficients. All modelling uncertainty is lumped into the uncertainty in these coefficients. The vehicle model accesses predicted parameter values, and uncertainty bounds, from the **Coefficient Database**. This information is tabulated as a function of flight condition, and is based on *a priori* knowledge from wind–tunnel measurements, empirical data, numerical flow analysis, or theoretical results. Coefficient estimates, and their associated uncertainties, can also potentially be improved on–line using information collected in flight. An **On–Line System Identification** subsystem would perform this function using information gathered in the **Sensor Fusion** block, which includes environmental as well as inertial measurements. A **State Estimator** is shown in the figure, but for the purposes of this thesis it is assumed that accurate full state information is available. It is also assumed that the **Coefficient Database** always represents the best combination of all information available (from *a priori* and on–line sources), thus the problem of system identification is not addressed as part of this work. The focus

[2] Candidate solutions are considered viable if they are stable, meet all state and control constraints, and provide some degree of tracking convergence.

of this thesis is on the **Trajectory Control System**, which will also frequently be referred to as the **Trajectory Controller**. This block is responsible for achieving the desired tracking performance, maintaining stability, and accounting for all the nonlinearities, constraints, and uncertainties.

3.2 Robust Intelligent Flight Control

Ideally, one can imagine that the best control solution could be obtained if somehow, at each time, one could choose from among the set of all possible future trajectories, a path that meets all the constraints (for all time) and converges to the desired trajectory in some predefined optimal sense. If this selection process was continued repeatedly, responding to disturbances by re–optimizing the future trajectory, it is clear that the best possible performance would be achieved. If the maximum "look ahead" time is limited to some $t_{\max}$, and terminal constraints are imposed at $t_{\max}$, then, in fact, this is referred to as a Receding Horizon Optimal Control (RHOC) Problem [May1]. Unfortunately, with multivariable nonlinear dynamics (with constraints, changing desired states, etc...), the problem is far too complex to solve analytically. However, if the state space is discretized at intervals of time Δt, and the control inputs are divided into quantized values, it is possible to solve this problem numerically. A dynamic programming approach would typically be used, but in this case, the solution to the problem would require such intense computational effort that it would not normally be considered for on–line applications.

In this thesis an alternative approach is developed using intelligent search techniques for solving optimization problems. This approach will be seen to drastically reduce the required search space, (as compared with dynamic programming), and is structured such that it can take full advantage of parallel processing computer hardware. This new method combines Lyapunov stability theory with a modified A^* (A–Star) optimization algorithm, to guarantee stepwise stability of the system, and to assure convergence of the optimization to a solution within a prescribed memory limit.

The flight control approach taken in this thesis, which will be called Robust Intelligent Flight Control (RIFC), has several important advantages. First of all, as the algorithm searches for a solution, it generates candidate trajectories by predicting forward in time the response of the full nonlinear dynamics to sequences of control inputs. No approximations are necessary, so that the actual trajectory can be expected to behave as predicted (with the exception of disturbances and uncertainty). Furthermore, parametric uncertainty is easily included in the optimization, (as will be shown in Chapter 7), by predicting not only the nominal trajectory, but also a growing uncertainty envelope around it. A robust control solution is obtained by minimizing the worst case tracking error rather than the nominal error. This approach is also particularly well suited to using a tabular representation of the predictions of the dynamic and control coefficients along with interval uncertainty bounds. Another important advantage is that the state, control, and dynamic constraints actually simplify the problem. All constraints are automatically included, and they effectively reduce the search space explored by the optimization algorithm. Only trajectories which observe all constraints (for all $t \leq t_{\max}$) are considered as candidate solutions. Another beneficial property of this approach is its ability to consider a variety of actuator types simultaneously. It is also reconfigurable in the case of identifiable changes in–flight. Not only can it take advantage of improved coefficient estimates based on in–flight measurements, but it can easily handle hardware failures, or other changes in actuator authorities, when these changes are detectable by other onboard systems. Finally, this algorithm has inherent guarantees of stability and tracking convergence, and it is suitable for parallel computation, making its real–time implementation feasible in (not too distant) future flight computers.

Qualitatively, the RIFC trajectory control algorithm operates as a single–step feedback controller with the added capability to look ahead a number of steps to be sure that the controls chosen now will lead to tracking convergence (without constraint violations) in the future. It is implemented as a short–term planning algorithm, which repeatedly solves for a multi–step trajectory during each single–step of actual flight time (Δt_s). It can therefore also be viewed, (during any single–step), as a low bandwidth feed–forward control solution[3], where the bandwidth is determined by the control cycle–time (Δt_s).[4] The choice of this trajectory control bandwidth is influenced by several factors. It would **not** be sensible to re–optimize a correction trajectory at very high frequencies just to counteract small high–frequency disturbances. Nor would it be practical (or computationally feasible) to re–solve the problem at very short intervals of time. The control cycle–time should be short enough, however, to capture the effects of time–varying coefficients and atmospheric variations on the vehicle dynamics. In contrast, the upper bound on Δt_s is limited by the tradeoff between desired tracking precision and control quantization (discussed further in Chapter 8), as well as the deterioration of prediction accuracy due to uncertainties and future disturbances. The time–step Δt_s should be small enough to prevent environmental disturbance terms from dominating other terms in the state equations, particularly those related to control inputs.

In short, the trajectory control system is intended to provide a stabilizing (limited bandwidth) feedback law which accounts for the system's nonlinearities, constraints, and uncertainties. An inner feedback loop can then be used to track the outer–loop commands by rejecting disturbances between the outer–loop control time–steps. For a prudent choice of the bandwidth separation between inner and outer loops, the control design of the inner–loop can be relatively simple. Although a robust linear control method could be used, a better approach (suggested in Chapter 8), is to utilize a single–step version of the RIFC controller. This method has the advantage that all nonlinearities and uncertainties could be included in the feedback loop, while very little additional software would be required. For the purposes of this thesis, however, the existence of an appropriate inner–loop feedback controller is assumed, and the emphasis will be placed on the development and analysis of the (outer–loop) trajectory control system.

[3] It is not strictly a feed-forward controller, however, since only one time-step of the feed-forward trajectory is used before the problem is solved again.

[4] Note that the integration step size used for trajectory prediction (within the trajectory controller) can be much smaller than Δt_s, or even variable, to attain a desired degree of precision.

The theoretical foundation of the RIFC algorithm is based on a combination of Lyapunov stability theory and the properties of a modified A^* optimization technique. The A^* approach performs an intelligent search for a solution which minimizes a cumulative cost function, guided by estimates of that cost function for incomplete trajectories. In the basic algorithm, candidate trajectories are generated and explored in an order consistent with a best–first rule. That is, trajectories which are estimated to have a lower overall cost are explored first. The details of A^*, and of several enhancements to the basic A^* method to improve its efficiency, will be explained in Chapter 6. The important point for this discussion is that given a simple property of the cost estimating function, the A^* search can be shown to converge to the optimal solution. The combination of a Lyapunov criterion with the A^* optimization essentially assures that the algorithm converges to a solution with guaranteed stability properties.

The development of the RIFC algorithm begins with the determination of constraints on the state space that assure controllability. A Lyapunov function of the tracking error is defined, and the existence of a solution that stabilizes the tracking error dynamics is verified. It is then shown that if the A^* cost function is chosen as a cumulative sum of tracking errors measured by the Lyapunov function (which is a norm of the errors), then the A^* search will converge to a solution that tracks the desired trajectory. Robustness to parametric uncertainty is then attained through a redefinition of the Lyapunov function to include a maximization over the parameter space. State space constraints also have to be recomputed to account for uncertainty. The resulting search guarantees robustness by performing a minimax type of optimization, where the worst case tracking error is minimized. Finally, a scheme is developed to manage the optimization such that a solution can be guaranteed with limited available memory.

One way to view the RIFC algorithm is as an enhanced Single–Step Optimal Controller (SSOC). The SSOC approach is also based on the fact that solving a complete dynamic programming problem on–line is not feasible. The SSOC method compromises by optimizing only one step at a time. With this type of approach, it is hoped that a sequence of single–step optimal trajectories will approximate the overall multi–step optimal trajectory. Otherwise, the SSOC is applicable to the same kinds of problems. It can be made to predict forward with the nonlinear equations (one step), account for uncertainty, and avoid constraints. It cannot, however, probe ahead to be sure that its single–step optimal solution does not lead the system toward trouble a some later time–step.

The advantage of the RIFC trajectory control algorithm is that it considers multi–step trajectories, and chooses its single–step decisions based on the optimality and constraint avoidance of a full multi–step correcting trajectory. It is more computationally expensive than SSOC, but far less burdensome than dynamic programming. The required computational effort can also be significantly reduced by combining a number of search enhancements to find admissible suboptimal trajectories very quickly. Since the objective of performing the optimization is future admissibility and convergence of the single–step decision, and since the problem will be solved again each time–step, a suboptimal solution meeting all constraints may be perfectly acceptable. An enhanced A^* algorithm, could find a viable solution and then spend any remaining allotted time improving the cost. In any case, the RIFC method is effectively a SSOC approach with enough additional foresight to guarantee certain properties of its solution . As such, for the purposes of this thesis, the SSOC method will be used as a basis for comparison. It will be shown later (Chapter 9), that situations may arise in which the RIFC controller succeeds, while the SSOC controller fails.

3.3 Overview

The research in this thesis can be divided into four major areas: modelling, simulation, control, and experiments. These topics are presented in detail in the following chapters. It is useful, however, to summarize the efforts in each area to give the reader an overview of the entire work. In this section, the main ideas from each of the topics above will be briefly reviewed.

3.3.1 MODELLING

Although flight control for air–breathing hypersonic vehicles was the main focus of this research, a significant amount of effort was devoted to developing models of the vehicle geometry, mass properties, aerodynamics, propulsion, controls, and environment, in order to provide the basis for a realistic AHSV simulation. Actually, it is important to distinguish between two separate classes of models that have been used. First, there are the high fidelity models used by the simulation to try to reproduce the actual flight behavior of the vehicle. These models include modules such as the flow analysis code, or the scramjet engine code, which compute component forces and torques on the vehicle as a function of many flight condition and design variables. The second class of models includes the nonlinear equations used by the flight control system to predict vehicle behavior. Appearing in these equations are all of the aerodynamic, propulsive, and control coefficients, which are tabulated (along with uncertainty bounds) in the Coefficient Database as a function of flight condition. Each of these models will be briefly described in turn below.

Truth Model For Simulation

To provide a reasonable characterization of the behavior of an actual AHSV, all the component models work with a fairly general parameterized vehicle design. The three dimensional (3–D) geometry of an AHSV is represented by *50* intersecting polygonal surfaces. A wide range of possible shapes can be obtained by varying *18* geometric parameters, such as length, wing span, sweep angle, combustor length, nozzle angle, fuselage aspect ratio, forebody ramp angles, and inlet height. The software automatically constructs the polygonal surfaces to generate a 3–D vehicle geometry. A mass properties model then makes certain assumptions about the materials, structure, propellant distribution, and equipment, to integrate mass distribution throughout the complex 3–D shape, to arrive at an overall vehicle mass and inertia matrix. These values will, in fact, change as propellant is used over the course of a simulation.

The atmospheric model is based on data from the space shuttle simulation at the NASA Dryden Flight Research Center. Temperature, density, pressure, and speed of sound are tabulated as a function of altitude at increments of *1* Km up to *244* Km. The atmosphere is assumed to be stationary with respect to the Earth, and disturbances, such as wind gusts or density variations, are modelled by adjustable random processes that are added during flight simulation. The aerodynamic model uses a hypersonic Newtonian flow approximation to compute pressure coefficients on each of the *50* vehicle surfaces. Laminar to turbulent transition is predicted over the vehicle surface, and skin friction coefficients are determined based on local flow properties. Oblique shock and expansion fan models are used to determine local flow. Net force vectors for each panel are combined to produce overall vehicle aerodynamic forces and moments in six degrees–of–freedom (6–DOF). Similarly, the scramjet propulsion model performs a full inlet, combustor, nozzle analysis depending upon the flight condition and *23* variables which specify the engine design. This engine model (and code) was adapted from some work done at MIT on the analysis of scramjet engines [Ren1]. Propulsive efficiencies and flow properties are used to compute overall specific impulse, a net thrust vector, and the thrust moment on the vehicle.

All force and moment contributions from the models are combined and integrated by the dynamics model which includes the full 6–DOF translational and rotational equations of motion for an arbitrary latitude, and longitude, and direction, in the atmosphere of a rotating spherical Earth. All centripetal and coriolis effects are included, since they are not insignificant at hypersonic speeds. The rotational dynamics use a Quaternion representation to keep track of attitude. Control surfaces are modelled directly as part of the aerodynamic model in terms of pressure and frictional forces. Aerodynamic shadowing effects are also added to the model to account for conditions where part of a control surface is blocked by the rest of the vehicle. Together, all the component models can predict forces, moments, and the dynamic response of any particular NASP design (geometry and engine) in most realistic hypersonic flight conditions. The scope of the work in this thesis is limited to flight in the hypersonic regime.

Analytical Model For Controller

The flight control research in this thesis is limited to consideration of longitudinal motions only. As such, the controllers model consists of four differential equations describing the rates of change of the states $\underline{x}^{\mathrm{T}} = [\mathrm{V} \quad \gamma \quad \omega \quad \theta]$ (velocity, flight path angle, pitch rate, and attitude). These equations are nonlinear functions of the states, parameters, controls, and time.

$$\dot{\underline{x}}(t) = \underline{f}(\underline{x}(t),\underline{\phi}(\underline{x},t),\underline{u}(t),t) \tag{3.1}$$

Implicit in equation (3.1) is the uncertainty of the aerodynamic, propulsion, and control coefficients, which are represented by the parameter vector $\underline{\phi}$. These equations are derived from the full 6–DOF nonlinear equations used for the simulation truth model. Appropriate simplifications are made for the longitudinal case, such as equatorial launch to orbit, and zero bank/yaw angles, roll/yaw rates, and so on. For the controller, a tabulated model of coefficients along with associated uncertainties is used. Values are derived from numerical simulation data taken at various Mach numbers, and angles of attack. Fictitious uncertainty in the coefficients is then created by adding bounded noise to these values.

3.3.2 SIMULATION

The AHSV simulation is an interactive software package developed for the purpose of vehicle design, flight simulation, and the analysis of flight control systems. The user interface consists of pulldown menus to access all options, and a real–time 3–D graphical display of an AHSV in simulated flight. Other displays present flight data or any combination of real–time graphs to display variables of interest. Options are available to design the vehicle, select different models, set the initial flight conditions, add disturbances, test different control algorithms, display different variables, record or play back flights, analyze scramjet engine performance, change to different display modes, use different integration algorithms, and more. The software is divided into two tasks, namely Design and Flight. Altogether, 39 variables have to be chosen to specify an AHSV design, and the interaction between aerodynamics and propulsion makes it very difficult to design a balanced vehicle. Special design software has been written to aid in the search for a set of parameters which constitute a balanced configuration that is optimized for a given flight condition. Other routines build tabulated models for a specific design in order to speed up some of the calculations during simulated flight. The flight software then coordinates all of the models discussed in the previous section to actually create a real–time flight environment. Finally, flight control software is tested as if it were operating in the actual vehicle.

3.3.3 CONTROL

The theoretical development of the RIFC algorithm begins with a statement of the problem in precise mathematical terms. The algorithm is then derived and justified through a series of propositions that are followed by supporting arguments. Guarantees of controllability, stability, convergence, and robustness are demonstrated on the basis of certain reasonable assumptions, and the properties of Lyapunov stability theory and A^* optimization techniques. A number of enhancements are then made to the algorithm in order to improve its overall efficiency in the face of practical computational considerations.

3.3.4 EXPERIMENTS

Demonstration of the RIFC controller is carried out through a series of experimental flight simulations. A particular AHSV design is chosen, and then used as the basis for evaluating the flight controller under different circumstances. The properties of the optimization are examined, and the controller's performance in the presence of uncertainty and disturbances is demonstrated. The RIFC autopilot is compared to the SSOC method for a number of cases.

3.4 Thesis Scope

The following list summarizes the assumptions and limitations adopted in this research:

- The operational envelope for the simulation is limited to the hypersonic flight regime.
- Flight control is limited to longitudinal dynamics.
- Certain models are limited to a range of useful Mach numbers and angles of attack.
- Coefficients in the database are assumed to be the best available combination of *a priori* and in–flight information. On–line system identification is not addressed.
- Perfect state information is assumed. (However, measurement noise can be added to the uncertainty accumulated in the state predictions used by the optimization).
- Heat transfer has not been modelled nor is temperature included among the constraints.
- Controls are limited to elevons, throttle, and thrust vectoring.
- Actuator dynamics have been neglected.
- Aerodynamic, propulsion, and control coefficients are available in a tabulated form as a function of flight condition.
- Parametric uncertainty is in the form of interval bounds.
- The modelling of atmospheric disturbances for hypersonic flight has not been addressed.

Chapter 4

Vehicle Modelling

In this chapter, the AHSV flight environment and the dynamic behavior of the vehicle are characterized. For the purpose of simulating a realistic flight, models are developed to represent the atmosphere, the geometry and mass properties of the vehicle, as well as the aerodynamic and propulsive behavior. These models are combined to generate overall forces and moments, which are then integrated by a dynamics model to obtain the translational and rotational response of the vehicle. The simulation is capable of reproducing the hypersonic portion of a six–degree–of–freedom SSTO trajectory over a spherical Earth for a class of parameterized AHSV vehicles. For longitudinal flight control, a reduced order model of the dynamics is developed. This model predicts the response of the vehicle to control inputs using a tabulated representation of the aerodynamic, propulsion, and control coefficients. It also comprises all of the information that is available to the control system, while the higher fidelity models are treated as the true environment.

4.1 Vehicle Geometry Model

The need for good performance over a wide range of flight conditions, and the close coupling between the aerodynamic and propulsive requirements, makes the geometric design of an AHSV critically important. The model developed for this research was motivated by the desire to adequately characterize the important physical features of an AHSV, while taking into consideration the computational effort required to analyze the flow field (over a complex geometry) in a simulation that strives to operate in near real–time. The result is a three–dimensional solid–object vehicle geometry represented by *50* polygonal surfaces. These surfaces are generated as functions of *18* design variables of which *14* are independent. These parameters can be freely changed to create a wide range of possible vehicle realizations. A typical AHSV design is shown in Figure 4.1.

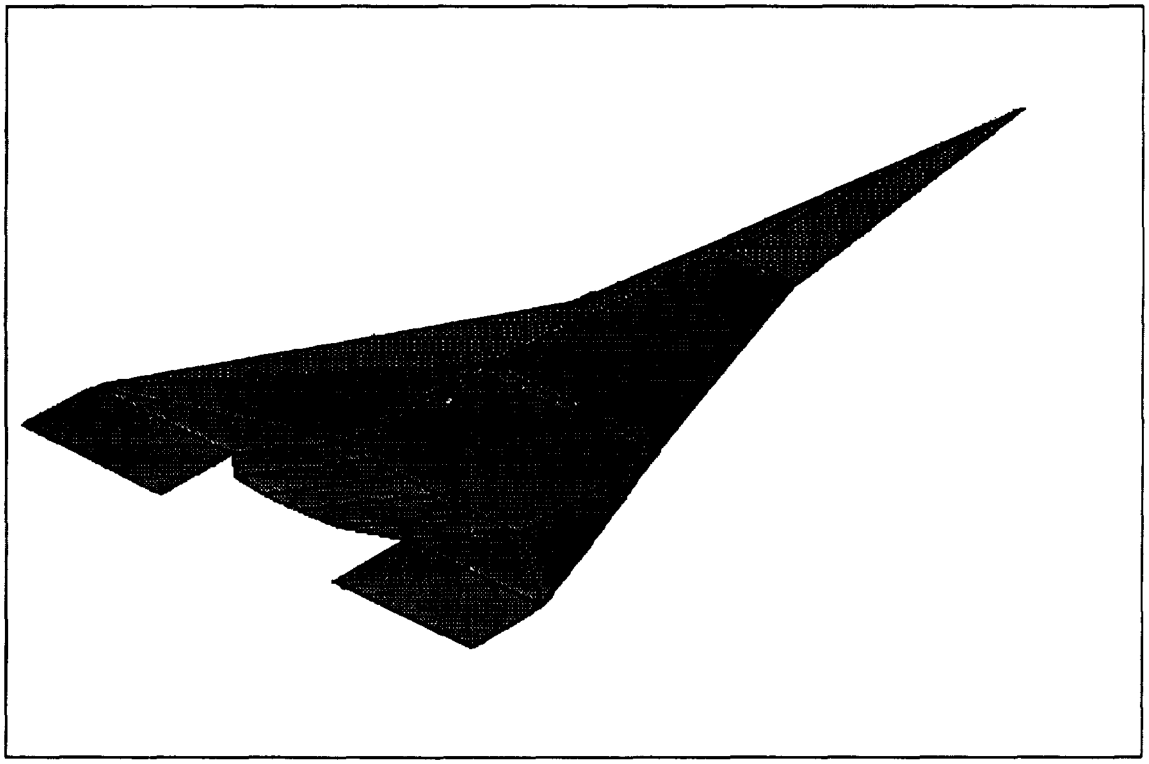


Figure 4.1: A Typical AHSV Geometric Design

The geometric design variables, listed below in Table 4.1, are a combination of important aerodynamic and propulsion parameters, some of which are specific to air–breathing hypersonic vehicles.

(1) Vehicle Length (l_v)
(2) Wing Span (b)
(3) Wing Sweep Angle (δ)
(4) Body Aspect Ratio (AR_B)
(5) Combustor *Width (ω_c)*
(6) Combustor Length (l_c)
(7) Capture Height (h_c)
(8) Ramp Angle 1 (γ_1)
(9) Ramp Length 1 (l_{f1})
(10) Ramp Angle 2 (γ_2)
(11) Ramp Length 2 (l_{f2})
(12) Nozzle Angle (β)
(13) Nozzle Length (l_N)
(14) Nozzle Lip Length (l_l)
(15) Inlet Height (h_i)
(16) Inlet Angle (γ_3)
(17) Elevon Length (l_e)
(18) Center Of Mass Bias (CM_b)

Table 4.1: Geometric Design Variables

These design variables, along with some other useful dimensions, are illustrated in the following figures.

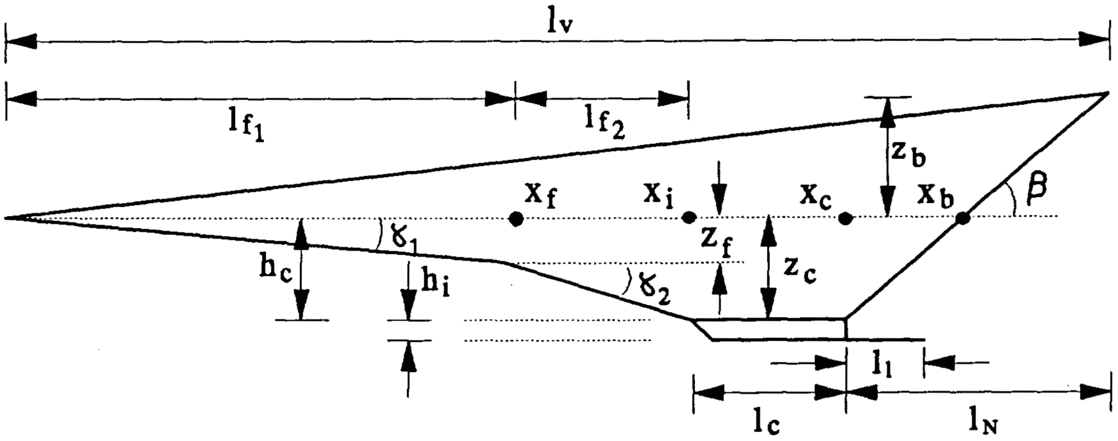


Figure 4.2a: Physical Dimensions of an AHSV – Side View

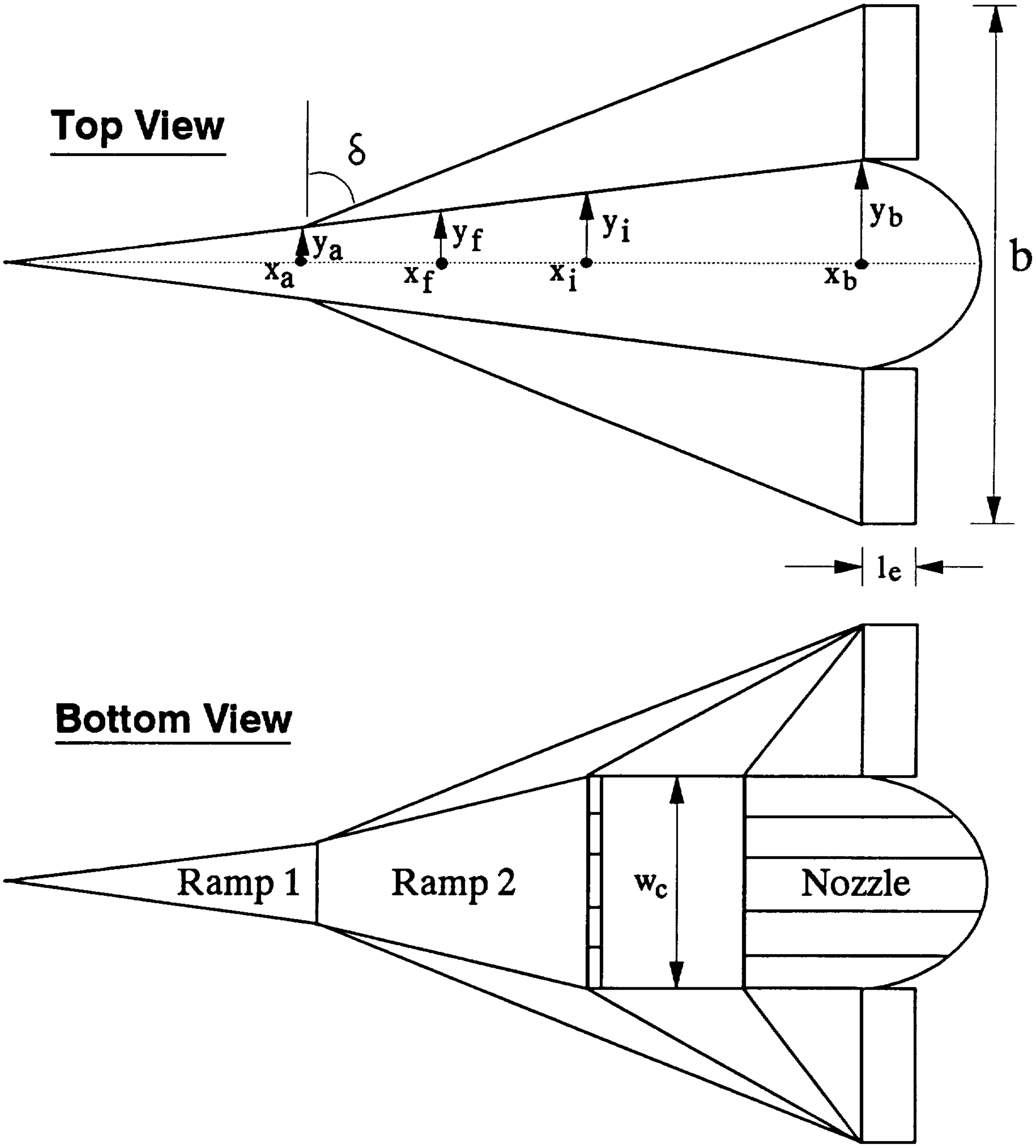


Figure 4.2b: Physical Dimensions of an AHSV – Top/Bottom View

The following equations define all of the dimensions in the figure above in terms of the independent design variables:

$$l = l_{f1} + l_{f2} + l_c + l_N \tag{4.1a}$$

$$x_f = l_{f1} \tag{4.1b}$$

$$z_f = x_f \; tan(\gamma_1) \tag{4.1c}$$

$$x_i = x_f + l_{f2} \tag{4.1d}$$

$$z_c = z_f + l_{f2} \; tan(\gamma_2) \tag{4.1e}$$

$$x_c = x_i + l_c \tag{4.1f}$$

$$x_b = x_c + z_c/tan(\beta) \tag{4.1g}$$

$$z_b = (l-x_b) \; tan(\beta) \tag{4.1h}$$

$$\phi_n = tan^{-1}(z_b/l) \tag{4.1i}$$

$$AR_B = (x_b \; z_b)/(\omega_c \; l/2) \tag{4.1j}$$

$$y_b = \omega_c/2 \tag{4.1k}$$

$$y_f = (y_b/x_b) \; x_f \tag{4.1l}$$

$$y_i = (y_b/x_b) \; x_i \tag{4.1m}$$

$$x_a = \frac{b/2 \; - \; x_b tan(90-\delta)}{y_b/x_b \; - \; tan(90-\delta)} \tag{4.1n}$$

$$y_a = (y_b/x_b) \; x_a \tag{4.1o}$$

$$x_l = x_i + h_i/tan(\gamma_3) \tag{4.1p}$$

$$z_l = z_c + h_i \tag{4.1q}$$

The vehicle geometry is constructed by generating the three dimensional coordinates of the vertices for all surface panels based on the values of the design variables.[1] These coordinates are then grouped and ordered in an array of polygonal elements. The ordering of the coordinates is done in a manner that identifies the direction of the outward normal for each surface panel. This is important, since the aerodynamic force on each panel will depend on its orientation with respect to the external flow, and which side of the panel is exterior to the vehicle. The unit normal vector for each panel is obtained through the cross product of the two vectors which connect the pairs of opposing vertices.

[1] Each surface panel is a four sided polygon.

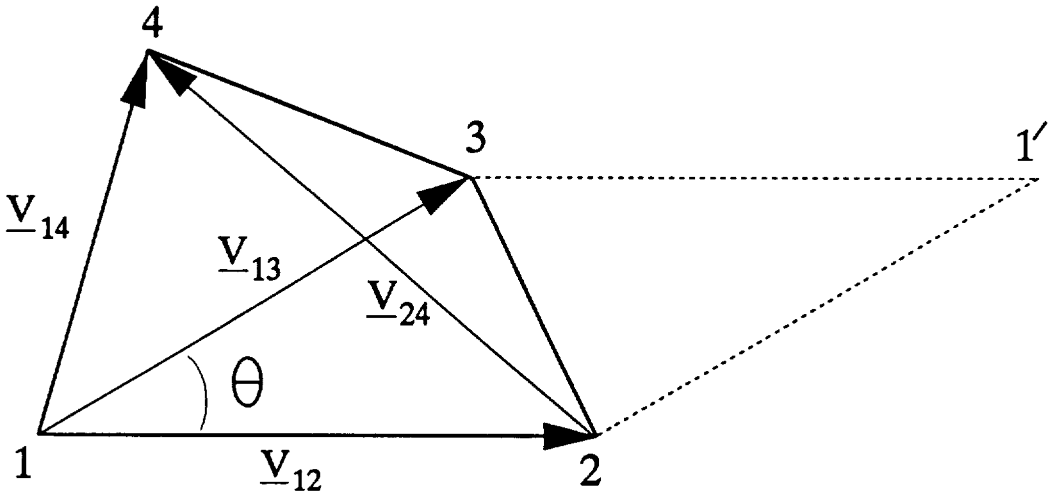


Figure 4.3: Normal Vector Calculation for Surface Panels

$$\hat{n} = \frac{\underline{V}_{13} \times \underline{V}_{24}}{\| \underline{V}_{13} \times \underline{V}_{24} \|} \tag{4.2}$$

The surface area for each panel is calculated by adding the areas of the two triangles Δ_{123} and Δ_{134}. Since, for example, $\| \underline{V}_{12} \times \underline{V}_{13} \| = |\underline{V}_{12}| \cdot |\underline{V}_{13}| \; sin(\theta)$ is the area of the parallelogram 1–2–1′–3, the triangle 1–2–3–1 has half this area. Therefore the surface area for a panel is given by,

$$S_{1234} = \tfrac{1}{2}\| \underline{V}_{12} \times \underline{V}_{13} \| + \tfrac{1}{2}\| \underline{V}_{13} \times \underline{V}_{14} \| \tag{4.3}$$

The centroid of each panel, which will also be required, can be found by averaging the coordinates of the vertices.

$$\bar{x} = \frac{[x_1 + x_2 + x_3 + x_4]}{4} \tag{4.4a}$$

$$\bar{y} = \frac{[y_1 + y_2 + y_3 + y_4]}{4} \tag{4.4b}$$

$$\bar{z} = \frac{[z_1 + z_2 + z_3 + z_4]}{4} \tag{4.4c}$$

The full three–dimensional geometry of the vehicle is therefore characterized by an array of points (coordinates of the vertices), an array of surfaces (groups of pointers to elements of the array of points), and three additional arrays for the normal vectors for each panel, the panel areas, and their centroids. An important advantage of this representation is that the orientation of the entire vehicle (all individual surfaces), with respect to any other coordinate frame, can be easily obtained through the simple matrix multiplication of these arrays by transformation cosine matrices.[2] The geometry model, therefore, only has to generate the required arrays for a particular design once, at the beginning of an AHSV flight simulation.[3]

4.2 Mass Properties Model

The dynamic response of the AHSV to aerodynamic and propulsive forces and moments depends upon the mass properties of the vehicle. This section presents a representative model of the vehicle mass, inertia tensor, and the location of the center of mass. The estimation of these mass properties can be divided into contributions due to the vehicle structure, fuel, engine, payload, and other equipment. For convenience, the internal volume is divided into fuselage and wing components. Specific assumptions regarding the location of the payload, fuel, or equipment are avoided. Instead, all internally located mass contributions are combined and represented by one average internal density. An adjustable bias for the location of the center–of–mass (CM_b) is used to represent the (limited) design freedom in terms of the mass distribution.[4]

[2] Moreover, since the array of surfaces only contains pointers to the coordinates in the array of points (i. e. it does not directly contain any dimensions), this array never requires any transformations. The array of surface areas is also clearly invariant to any changes in the coordinate system.

[3] With the exception that subsequent changes to the vehicle geometry, such as a deflection of the elevon aerosurfaces, do require some of the surfaces to be recalculated.

[4] For a more detailed discussion on the estimation of mass properties for space transportation systems see reference [Mac1].

In this section, all values will be referenced to a *vehicle–geometry* coordinate system as shown in Figure 4.4. The origin of these axes is at the vehicle nose with the positive x–axis toward the rear of the vehicle; the y–axis is in the starboard direction; and the z–axis points up.

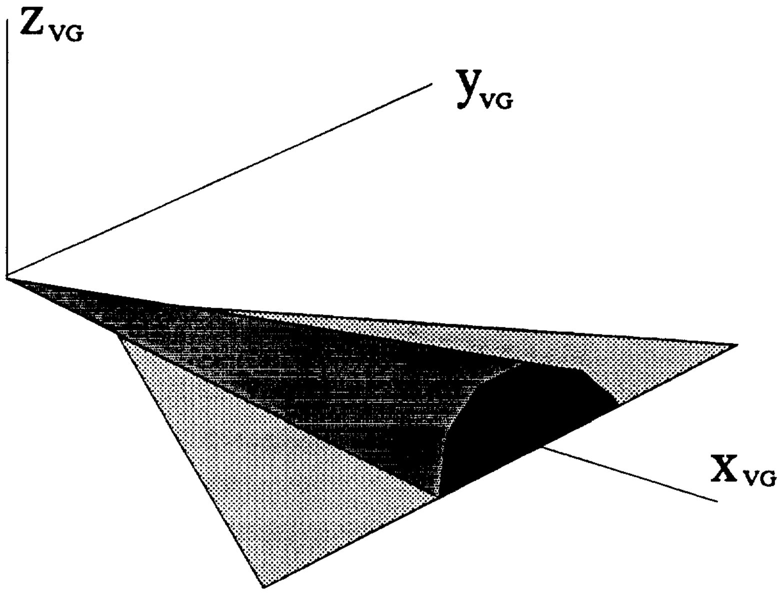


Figure 4.4: Vehicle–Geometry Coordinate System

Beginning with the vehicle fuselage and wing structure, the total surface area (S_T) can be calculated from,

$$S_T = \sum_{i=1}^{N} S_i \tag{4.5}$$

where S_i is the surface area of the *i*th of *N* surfaces.

Using ρ_s as the density of the structural material, the total surface structural mass is given by,

$$M_s = \rho_s \, S_T \tag{4.6}$$

and the center of mass for the external structure can be expressed as,

$$\bar{x}_s = \frac{1}{S_T} \sum_{i=1}^{N} S_i \, \bar{x}_i \qquad \bar{y}_s = \frac{1}{S_T} \sum_{i=1}^{N} S_i \, \bar{y}_i \qquad \bar{z}_s = \frac{1}{S_T} \sum_{i=1}^{N} S_i \, \bar{z}_i \tag{4.7}$$

where the coordinate $(\bar{x}_i, \bar{y}_i, \bar{z}_i)$ is the centroid of the *i*th surface.

The structural moments of inertia are,

$$I_{xs} = \sum_{i=1}^{N} (\bar{y}_i^2 + \bar{z}_i^2) \, S_i \, \rho_s \tag{4.8a}$$

$$I_{ys} = \sum_{i=1}^{N} (\bar{x}_i^2 + \bar{z}_i^2) \, S_i \, \rho_s \tag{4.8b}$$

$$I_{zs} = \sum_{i=1}^{N} (\bar{x}_i^2 + \bar{y}_i^2) \, S_i \, \rho_s \tag{4.8c}$$

$$I_{xzs} = -\sum_{i=1}^{N} \bar{x}_i \, \bar{z}_i \, S_i \, \rho_s \tag{4.8d}$$

$$I_{xys} = I_{yzs} = 0 \quad \textit{(from symmetry)} \tag{4.8e}$$

In order to estimate the remaining mass properties, the interior volume of the vehicle is required. The geometry of the fuselage has been modelled as half of an elliptic cone (see Figure 4.1). The dimensions of the cone are specified by the vehicle length (l), the combustor width (ω_c), and the body aspect ratio ($\mathcal{AR}_B$). Integrating the interior of the half-cone and omitting the details, the volume of the fuselage is given by,

$$V_f = \frac{1}{24} \pi \, \mathcal{AR}_B \, \omega_c^2 \, l \tag{4.9}$$

and the mass of this section is,

$$M_{\mathrm{f}} = \rho_{\mathrm{v}}\ V_{\mathrm{f}} \tag{4.10}$$

where ρ_{v} is the mean density of the fuel and payload mass contributions.

The center of mass for the fuselage is taken to be at,

$$(\bar{x}_{\mathrm{f}},\ \bar{y}_{\mathrm{f}},\ \bar{z}_{\mathrm{f}}) = (3l/4,\ 0,\ 0) \tag{4.11}$$

which utilizes the formula for the centroid of a cone to obtain $\bar{x}_{\mathrm{f}}$. The value of $\bar{z}_{\mathrm{f}}$ is taken to be zero since it is assumed to be balanced by the z–component centroid for the wing mass. By symmetry, $\bar{y}_{\mathrm{f}} = 0$.

Integrating the moments of inertia[5] for the fuselage gives,

$$I_{\mathrm{xf}} = \left[\frac{3}{20}\right] M_{\mathrm{f}}\ \omega_{\mathrm{c}}^{2} \tag{4.12a}$$

$$I_{\mathrm{yf}} = I_{\mathrm{zf}} = \left[\frac{3}{10}\right] M_{\mathrm{f}} \left[(\omega_{\mathrm{c}}/4)^{2} + l^{2}\right] \tag{4.12b}$$

$$I_{\mathrm{xzf}} = -\left[\frac{1}{24}\right] \rho_{\mathrm{v}}\ \mathcal{AR}_{\mathrm{B}}^{2}\ l^{2}\ \omega_{\mathrm{c}}^{3} \tag{4.12c}$$

To simplify the mass property estimates for the wing, the chordwise cross section is approximated by a rectangular section with the same area. The wing section, shown in Figure 4.5, has a root thickness ($\bar{t}$) equal to the average actual root thickness (see also Figure 4.2b).

$$\bar{t} = \left[\left[\frac{x_{\mathrm{i}} - x_{\mathrm{a}}}{2}\right] + l_{\mathrm{c}} + \left[\frac{x_{\mathrm{b}} - x_{\mathrm{c}}}{2}\right]\right] \frac{z_{\mathrm{c}}}{[x_{\mathrm{b}} - x_{\mathrm{a}}]} \tag{4.13}$$

[5] In general, the moments of inertia are given by the following integrals,

$$I_{\mathrm{x}} = \int_{V} (y^{2} + z^{2})\, \rho_{\mathrm{v}}\, dV \quad I_{\mathrm{y}} = \int_{V} (x^{2} + z^{2})\, \rho_{\mathrm{v}}\, dV \quad I_{\mathrm{z}} = \int_{V} (x^{2} + y^{2})\, \rho_{\mathrm{v}}\, dV$$

$$I_{\mathrm{xz}} = -\int_{V} x\, z\, \rho_{\mathrm{v}}\, dV \quad I_{\mathrm{yz}} = -\int_{V} y\, z\, \rho_{\mathrm{v}}\, dV \quad I_{\mathrm{xy}} = -\int_{V} x\, y\, \rho_{\mathrm{v}}\, dV$$

where V represents the volume of the region of interest.

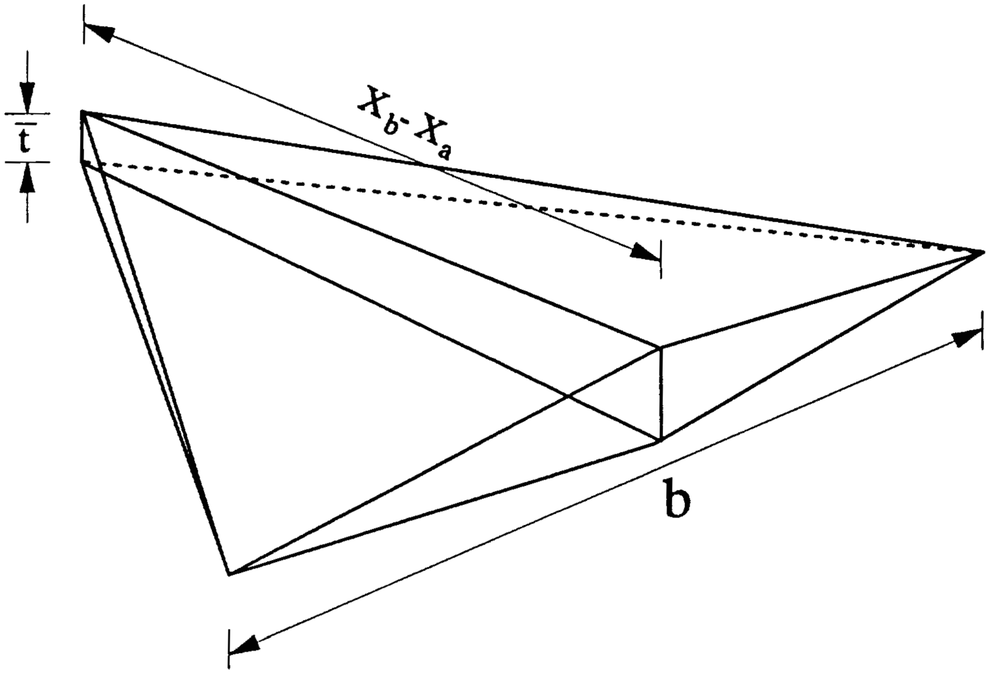


Figure 4.5: Mass Properties Model of Wing Volume

Integrating the interior of this geometry results in a wing volume and mass given by,

$$V_w = \frac{1}{3}\, \bar{t}\, b\, (x_b - x_a) \tag{4.14}$$

$$M_w = \rho_v\, V_w \tag{4.15}$$

The center of mass for the wing is found to be at,

$$\bar{x}_w = \frac{(3x_a \;+\; 5x_b)}{8} \tag{4.16a}$$

$$\bar{y}_w = \bar{z}_w = 0 \tag{4.16b}$$

(again $\bar{y}_w$ by symmetry, and $\bar{z}_w$ is assumed to be
balanced by the fuselage contribution $\bar{z}_f$)

Integrating the inertias for the wing geometry gives the formulae,[6]

$$I_{xw} = \left[\frac{1}{120}\right] \bar{t}\, b^3\, \rho_v\, (x_b - x_a) \tag{4.17a}$$

$$I_{yw} = \bar{t}\, b\, \rho_v\, (x_b - x_a) \left[\frac{1}{3} x_a^2 + \frac{5}{12} x_a (x_b - x_a) + \frac{3}{20} (x_b - x_a)^2\right] \tag{4.17b}$$

$$I_{zw} = \bar{t}\, b\, \rho_v\, (x_b - x_a) \left[\frac{39\, b^2 + 260\, x_a^2 + 480\, x_a x_b + 660\, x_b^2}{3600}\right] \tag{4.17c}$$

$$I_{xzw} = \bar{t}^2\, b\, \rho_v\, (x_b - x_a) \left[\frac{109\, x_a + 186\, x_b}{1800}\right] \tag{4.17d}$$

For the scramjet engine, the mass properties are estimated using a characteristic mass per inlet area (ρ_p). The inlet area is given by $\omega_c \times h_i$, therefore, the engine mass is given by,

$$M_p = \omega_c\, h_i\, \rho_p \tag{4.18}$$

The center of mass for the engine is assumed to be at the center of the combustion chamber (see Figures 4.2a–b), so that

$$(\bar{x}_p, \bar{y}_p, \bar{z}_p) = (x_i + \tfrac{1}{2} l_c,\ 0,\ -z_c - \tfrac{1}{2} h_i) \tag{4.19}$$

The inertias for the engine are given by,

$$I_{xp} = M_p\, (\bar{z}_p^2 + \bar{y}_p^2) \tag{4.20a}$$

$$I_{yp} = M_p\, (\bar{x}_p^2 + \bar{z}_p^2) \tag{4.20b}$$

$$I_{zp} = M_p\, (\bar{x}_p^2 + \bar{y}_p^2) \tag{4.20c}$$

$$I_{xzp} = -\bar{x}_p\, \bar{z}_p\, M_p \tag{4.20d}$$

The results above for the structure, fuselage, wing, and propulsion components are now combined to give the mass properties for the entire vehicle.

<u>**Total Mass:**</u> $$M_T = M_s + M_f + M_w + M_p \tag{4.21}$$

[6] These integrals were somewhat involved due to the complex geometry of the wing section. The Mathematica software package was used to verify the results.

Center Of Mass:

$$\bar{x}_{\mathrm{cm}} = \frac{1}{M_{\mathrm{T}}} \left[M_{\mathrm{s}}\, \bar{x}_{\mathrm{s}} + M_{\mathrm{f}}\, \bar{x}_{\mathrm{f}} + M_{\mathrm{w}}\, \bar{x}_{\mathrm{w}} + M_{\mathrm{p}}\, \bar{x}_{\mathrm{p}} \right] \tag{4.22a}$$

$$\bar{y}_{\mathrm{cm}} = \frac{1}{M_{\mathrm{T}}} \left[M_{\mathrm{s}}\, \bar{y}_{\mathrm{s}} + M_{\mathrm{f}}\, \bar{y}_{\mathrm{f}} + M_{\mathrm{w}}\, \bar{y}_{\mathrm{w}} + M_{\mathrm{p}}\, \bar{y}_{\mathrm{p}} \right] \tag{4.22b}$$

$$\bar{z}_{\mathrm{cm}} = \frac{1}{M_{\mathrm{T}}} \left[M_{\mathrm{s}}\, \bar{z}_{\mathrm{s}} + M_{\mathrm{f}}\, \bar{z}_{\mathrm{f}} + M_{\mathrm{w}}\, \bar{z}_{\mathrm{w}} + M_{\mathrm{p}}\, \bar{z}_{\mathrm{p}} \right] \tag{4.22c}$$

Moments Of Inertia:

$$I_{\mathrm{x}} = I_{\mathrm{xs}} + I_{\mathrm{xf}} + I_{\mathrm{xw}} + I_{\mathrm{xp}} \tag{4.23a}$$

$$I_{\mathrm{y}} = I_{\mathrm{ys}} + I_{\mathrm{yf}} + I_{\mathrm{yw}} + I_{\mathrm{yp}} \tag{4.23b}$$

$$I_{\mathrm{z}} = I_{\mathrm{zs}} + I_{\mathrm{zf}} + I_{\mathrm{zw}} + I_{\mathrm{zp}} \tag{4.23c}$$

$$I_{\mathrm{xz}} = I_{\mathrm{xzs}} + I_{\mathrm{xzf}} + I_{\mathrm{xzw}} + I_{\mathrm{xzp}} \tag{4.23d}$$

Also, since the moments of inertia will later be needed with respect to the center of mass, the shifted values are given below:

$$I_{\mathrm{xcm}} = I_{\mathrm{x}} - M_{\mathrm{T}}\, (\bar{y}_{\mathrm{cm}}^{\;2} + \bar{z}_{\mathrm{cm}}^{\;2}) \tag{4.24a}$$

$$I_{\mathrm{ycm}} = I_{\mathrm{y}} - M_{\mathrm{T}}\, (\bar{x}_{\mathrm{cm}}^{\;2} + \bar{z}_{\mathrm{cm}}^{\;2}) \tag{4.24b}$$

$$I_{\mathrm{zcm}} = I_{\mathrm{z}} - M_{\mathrm{T}}\, (\bar{x}_{\mathrm{cm}}^{\;2} + \bar{y}_{\mathrm{cm}}^{\;2}) \tag{4.24c}$$

$$I_{\mathrm{xzcm}} = I_{\mathrm{xz}} + M_{\mathrm{T}}\, \bar{x}_{\mathrm{cm}}\, \bar{z}_{\mathrm{cm}} \tag{4.24d}$$

Clearly, the final mass properties for a particular vehicle design will depend upon the values for the three densities ρ_{s}, ρ_{p}, and ρ_{v}. A basis for estimating the structural mass density for the wing and fuselage of a single–stage–to–orbit vehicle is presented in reference [Hat3]. A value of *34.5* kilograms per square meter of external surface area is obtained (ρ_{s} = *34.5* kg/m^2). The mass of the propulsion system per unit of inlet area (ρ_{p}) is derived in reference [Hat4]. A representative value for ρ_{p} is found to be *1,435* kg/m^2.

Also following [Hat3], the mean internal density ρ_{v} can be estimated using some approximate relationships based on Space Shuttle data (and a few assumptions). First, note that the AHSV interior volume is the sum of the fuselage and wing volumes:

$$V_{\mathrm{total}} = V_{\mathrm{f}} + V_{\mathrm{w}} \tag{4.25}$$

For simplicity, an AHSV payload capacity of *30,000 lbs* ($M_{\text{payload}} = 13{,}600\ kg$) is assumed. Based on the shuttle capacity and volume (*65,000 lbs* and *10,600 ft*^{3}), this corresponds to a payload volume of *4,890 ft*^{3} ($V_{\text{payload}} = 138.6\ m^3$). For the crew compartment *2,000 ft*^{3} ($V_{\text{crew}} = 56.7\ m^3$) are reserved (compared to the shuttle's *2,525 ft*^{3}). The remaining internal volume is divided between the propellant, propellant storage tanks, other equipment, and unusable space. It is assumed that *80%* of this volume is available for propellant storage.

$$V_{\text{prop}} = 0.80 \times (V_{\text{total}} - V_{\text{payload}} - V_{\text{crew}}) \qquad (4.26)$$

The mass of the cryogenic storage tanks is approximately *10.3 kg* for each cubic meter of propellant, so that

$$M_{\text{tanks}} = (10.3\ kg/m^3) \times V_{\text{prop}} \qquad (4.27)$$

and *20,000 lbs* of fixed equipment is assumed ($M_{\text{equip}} = 9{,}090\ kg$). Also from [Hat3], the mass of the orbital maneuvering system (OMS) is approximately *15%* of the vehicle dry mass.

$$M_{\text{oms}} = 0.15 \times [M_s + M_p + M_{\text{tanks}} + M_{\text{equip}}] \qquad (4.28)$$

It is also assumed that *5%* of the hydrogen fuel is required for the final exo–atmospheric portion of the trajectory to orbit. The mass of on–board oxidizer required to burn this fuel is approximately five times the mass of hydrogen.[7] Solving for the propellant mass,
from,

$$V_{\text{hydrogen}} + V_{\text{oxygen}} = V_{\text{prop}}$$
$$\rho_{\text{oxygen}}\ V_{\text{oxygen}} = 5\ (0.05\ \rho_{\text{hydrogen}}\ V_{\text{hydrogen}})$$

we have,

$$V_{\text{hydrogen}} = V_{\text{prop}}/(1 + 0.25\ \rho_{\text{hydrogen}}/\rho_{\text{oxygen}})$$
$$V_{\text{oxygen}} = V_{\text{prop}} - V_{\text{hydrogen}}$$

and then,

$$M_{\text{prop}} = \rho_{\text{hydrogen}}\ V_{\text{hydrogen}} + \rho_{\text{oxygen}}\ V_{\text{oxygen}} \qquad (4.29)$$

where from [Thr1,Hat3],

$$\rho_{\text{hydrogen}} \approx 80.0\ kg/m^3 \ \ \textit{(for hydrogen slush)}$$
$$\rho_{\text{oxygen}} = 1{,}145.\ kg/m^3 \ \ \textit{(liquid)}$$

[7] For the rocket mode, 0.625 times the stoichiometric mass ratio is required [Hat3].

Finally, the mean internal density for the vehicle is obtained by the following expression:

$$\rho_{\mathrm{v}} = \frac{M_{\mathrm{prop}} + M_{\mathrm{tanks}} + M_{\mathrm{equip}} + M_{\mathrm{payload}} + M_{\mathrm{oms}}}{V_{\mathrm{total}}} \qquad (4.30)$$

Now that the densities ρ_{s}, ρ_{p}, and ρ_{v} have been specified, all of the terms in equations (4.21–4.24) are completely defined in the calculations of this section. The mass properties can therefore be computed for any AHSV design geometry. Since the actual vehicle design process allows some flexibility in the placement of cargo, fuel, and equipment, an additional term is included to bias the location of the center of mass (*C.M.*) for the overall vehicle. The value of CM_{b} is restricted to the range ($0 \leq CM_{\mathrm{b}} \leq 1.0$), which corresponds to a forward and down *C.M.* shift given by,

$$\bar{x}_{\mathrm{cm}} \longrightarrow \bar{x}_{\mathrm{cm}} - CM_{\mathrm{b}}\ (l/5) \qquad (4.31a)$$

$$\bar{z}_{\mathrm{cm}} \longrightarrow \bar{z}_{\mathrm{cm}} - CM_{\mathrm{b}}\ z_c \qquad (4.31b)$$

As a final note, the mass properties model must be continually updated throughout the course of a simulation due to the effect of the expenditure of fuel on the total mass and inertias. This is simply accounted for by adjusting the mean internal density ρ_{v} accordingly and recomputing the affected equations.

4.3 Atmospheric Model

The properties of the atmosphere at any given altitude are represented by a model which was derived from the shuttle simulation software at the NASA Dryden Flight Research Center. The atmospheric data from this model is reproduced in reference [Hat2]. Ambient temperature, pressure, density, and speed of sound are tabulated as a function of altitude at increments of *1* kilometer up to a maximum of *244* kilometers. The atmospheric profiles are shown in Figure 4.6.

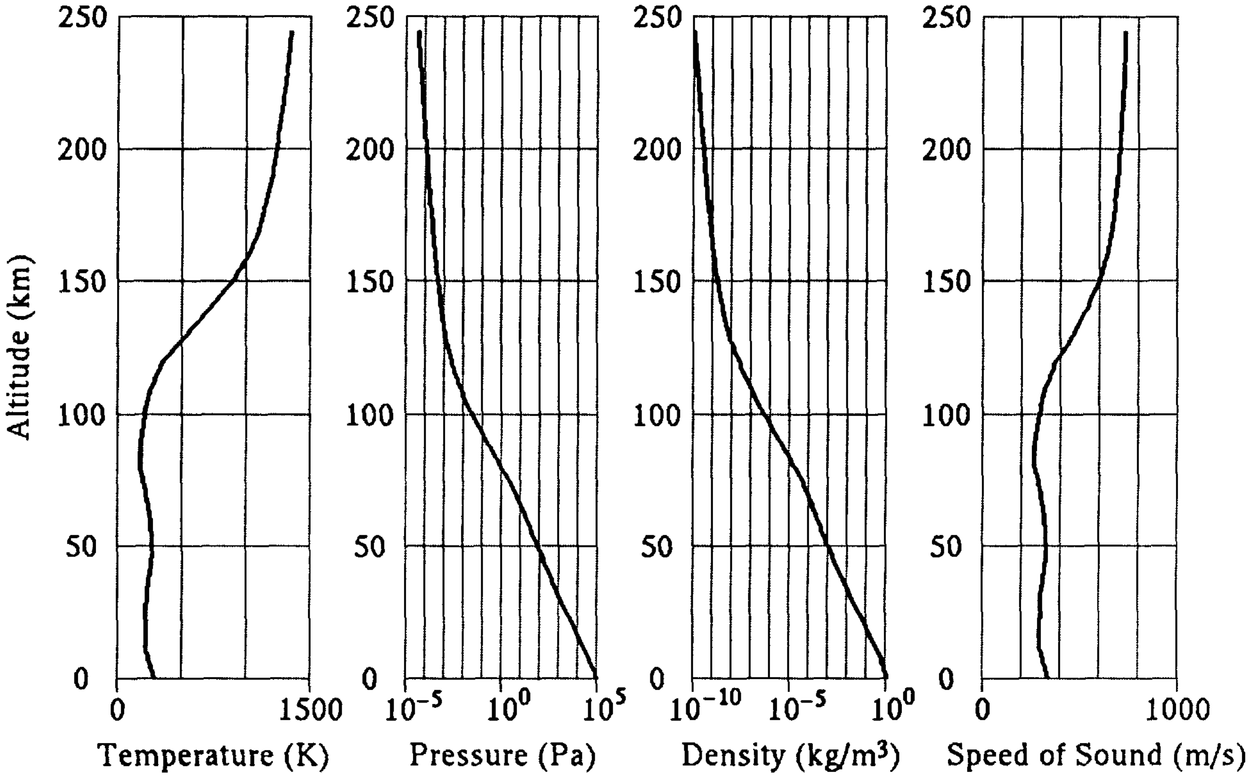


Figure 4.6: Tabulated Atmospheric Data from Dryden Shuttle Simulation

For the AHSV application, it is assumed that the atmosphere is stationary with respect to the rotating Earth's surface. Wind data, which is part of the original shuttle simulation model, is therefore not used. Atmospheric disturbances are included, however, and this is achieved by artificially adding noise to the air density, ambient temperature, angle of attack, and airspeed. These disturbances are modelled as gauss–markov processes with adjustable variance and correlation time constants.

The atmospheric model is called repeatedly during the course of an AHSV simulation. A linear interpolation scheme is used to determine the ambient conditions for the current altitude. In the interest of storage and computational efficiency, a subset of the entire data set, representing a 10 km range of altitudes, is stored in memory at any given time. As the vehicle transitions beyond this range a new set of data is loaded into memory prior to the next interpolation.

4.4 Aerodynamics Model

In this section a model is developed for the aerodynamic forces and moments experienced by an AHSV. In order to limit the scope of the modelling effort required for this research, only the hypersonic portion of the Earth–to–orbit trajectory is considered. This is the flight regime that is unique to AHSV's, and is perhaps the most challenging (in terms of structures, materials, propulsion, and controls) for many of the reasons already discussed in Chapter 2. Since the conditions of high Mach number flight cannot be completely reproduced in currently available wind tunnels, and since many of the complex theoretical aspects of hypersonic flow remain poorly understood, the accurate prediction of aerodynamic behavior, by empirical or numerical means, is a difficult task. Moreover, the complexity of a high temperature, high Reynolds number, chemically reacting flow within shock and boundary layers, means that any numerical flow analysis model would be computationally expensive (even for the evaluation of one flight condition).

For the purposes of this research, however, a representative model is desired which captures the important hypersonic aerodynamic characteristics of a particular vehicle design, while being simple enough to be applicable to a real–time simulation. This is achieved by using a Newtonian flow approximation to obtain pressure coefficients for each of the *50* panels on the vehicle surface, and by estimating the frictional forces based on laminar and turbulent flow models. Boundary layer transition is predicted from the local flow properties which are derived using oblique shock and expansion fan calculations.

4.4.1 HYPERSONIC FLOW

The most important distinction between hypersonic flow and subsonic or supersonic flow, is that the kinetic energy of a hypersonic gas far exceeds the thermal energy. Since the probabilistic thermal speed of the gas molecules is given by $\sqrt{2RT}$, the hypersonic condition can be expressed as,

$$\frac{\sqrt{2RT}}{V} << 1 \tag{4.32a}$$

Noting that the Mach number is defined by,

$$Ma = \frac{V}{\sqrt{\gamma_a RT}} = \frac{\sqrt{2}}{\sqrt{\gamma_a}} \cdot \frac{V}{\sqrt{2RT}} \tag{4.32b}$$

it is clear that the Mach number is inversely proportional to the ratio in (4.32a). Therefore, for hypersonic flow, equations (4.32a–b) imply that the Mach number must be large $Ma >> 1$. Usually this is taken to mean that the hypersonic flow regime begins at approximately Mach *6* and is well developed by Mach *10*. [Kapl]

When the kinetic energy of a gas exceeds the thermal energy, the result is extremely high gas temperatures behind shock waves and inside boundary layers. In fact, the gas temperature rises with the square of any velocity changes imparted to the flow, and if the temperature is high enough, the simple ideal gas model is no longer valid (due to excitation and dissociation of the gas molecules) and the assumption that the atmosphere behaves as a continuum breaks down.[8] Hypersonic flow can therefore also be characterized as a condition in which chemically reacting gases occur within the shock and boundary layers.

The flow surrounding a vehicle can only be considered to be in a state of chemical and kinetic equilibrium (i.e. a continuum flow) as long as the rate of molecular collisions is high. Once the molecular mean free path approaches the dimensions of the vehicle the result is a free molecular flow. A quantitative parameter used to distinguish between continuum and nonequilibrium flow is the

[8] This is also true for low density flows.

Knudsen number defined by,

$$Kn = \frac{Molecular\ Mean\ Free\ Path}{Vehicle\ Characteristic\ Length} = \frac{\lambda}{l} \tag{4.33}$$

If the Knudsen number is high (typically $Kn > 5$) the vehicle is operating in a regime of free molecular flow. Under these conditions the aerodynamic characteristics of the vehicle are governed by the kinetics of the collisions between individual gas molecules and the vehicle surfaces. [Kap1]

Another distinguishing characteristic of hypersonic flow is the so–called *viscous interaction* phenomenon. Considering a flat plate in a hypersonic flow, much of the flow's kinetic energy is dissipated within the boundary layer giving rise to very high temperatures. The flow density and viscosity coefficient within the boundary layer increase accordingly, and this results in a rapidly growing boundary layer. In fact, the thickness of a laminar hypersonic boundary layer is proportional to the square of the freestream Mach number and inversely proportional to the square root of the local Reynolds number ($\delta \propto Ma^2/\sqrt{Re_x}$). The extreme thickness of the laminar hypersonic boundary layer may cause an interaction with the outer inviscid flow that is not significant for subsonic or supersonic conditions. For surfaces inclined to the freestream flow, high Mach numbers correspond to thin shock layers. It is therefore possible for the shock and boundary layers to merge. Under these conditions the entire shock layer behaves as a viscous flow and the conventional boundary layer analysis methods are no longer applicable.[9] [And2]

The fact that high Mach number flows result in very thin shock layers is a very useful property. It can be shown from oblique shock theory (section 4.4.2) that for a given flow turning angle (corresponding to the inclination of a surface with respect to the flow) the shock angle decreases with increasing Mach number. For high Mach numbers this means that the flow streamlines do not turn until they are very close to the surface, and once beyond the shock they move parallel to the surface. The situation is illustrated in Figure 4.7. [And2]

[9] The viscous interaction phenomenon may not be a major problem for air-breathing hypersonic vehicles, since the dynamic pressure requirements of the engine demand a trajectory of high enough air density that most of the boundary layer is expected to be turbulent.

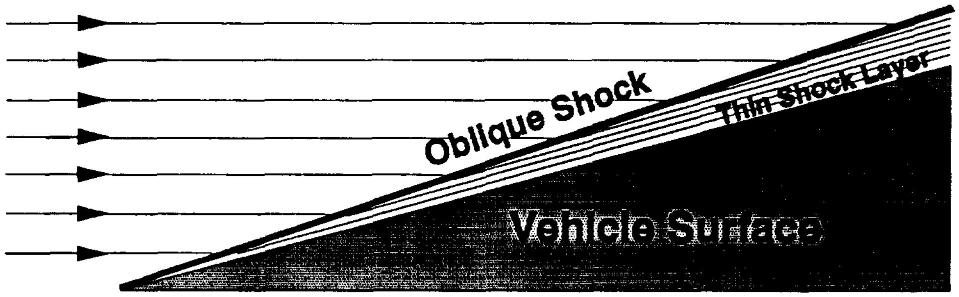


Figure 4.7: The Hypersonic Shock Layer

The behavior described above for the hypersonic streamlines motivates the application of a Newtonian flow approximation to the modelling of the aerodynamics for hypersonic vehicles. Newton's method assumes that the flow can be treated as a stream of particles. When a flow particle strikes another surface, the momentum normal to the surface is transmitted, the tangential momentum is preserved, and the flow continues in a direction parallel to the surface. While this approximation is invalid for flow at low speeds, it has been shown to work extremely well for predicting aerodynamic forces under hypersonic conditions. This approach is commonly used to predict pressure forces and to determine aerodynamic coefficients for vehicles experiencing high Mach number or free molecular flows.

4.4.2 HYPERSONIC AERODYNAMICS MODEL

For the prediction of the aerodynamic forces experienced by an AHSV in hypersonic flight, there are two contributing effects that must be considered: pressure forces, and shear forces. The pressure forces are associated with the momentum changes of the flow as it is forced to turn by the presence of the vehicle geometry. The shear forces are a result of the viscous flow properties, which for hypersonic flow are associated with large velocity and temperature gradients. Heat transfer, in this case, is the dominating factor influencing boundary layer separation (laminar/turbulent transition) and therefore the frictional forces. In this section, a Newtonian flow model is used to approximate the pressure forces. A local flow analysis is performed to predict transition and to estimate the shear forces from the local skin friction coefficients. Flow calculations are made on a panel by panel basis, and the overall aerodynamic forces and moments are obtained through a summation of force vectors (applied at different locations).

Pressure Forces

The principal advantage of using Newtonian flow theory to approximate hypersonic flows is that the pressure coefficient (C_p) is a function only of the local geometry.

$$C_p = k(\underline{\hat{v}}\cdot\underline{\hat{n}})^2 = k\,sin^2(\psi) \tag{4.34}$$

In this equation, $\underline{\hat{v}}$ is the freestream unit velocity vector, $\underline{\hat{n}}$ is the local surface normal unit vector (pointing into the surface), ψ is the angle of incidence between the flow and the surface, and k is the proportionality constant. In the classical Newtonian theory $k = 2$. This value comes from the assumption that all flow momentum normal to the surface is lost, while the tangential momentum is retained.

The definition of the pressure coefficient is given by,

$$C_p = \frac{P - P_0}{\frac{1}{2}\rho_0 U_0^2} \tag{4.35}$$

and from the momentum equation (for the assumption above),

$$P - P_0 = \rho_0\, U_0^2\, sin^2(\psi) \tag{4.36}$$

where the subscript 'o' refers to the freestream values. Substituting (4.36) into (4.35) gives,

$$C_p = 2\, sin^2(\psi) \tag{4.37}$$

and therefore $k = 2$.

Other values for k are often used in order to best fit empirical data corresponding to particular shapes. For example, experimental data suggests that for blunt bodies the best value for k is the maximum pressure coefficient behind normal shocks. For low incidence angles, a value of $k = \gamma+1$ gives good results and approaches the exact shock calculation for C_{p} at high Mach numbers. For complex geometries an approach can be taken in which different values of k are used for each surface component. Since the geometric configuration of the AHSV can be generally classified as a slender design, the modified Newtonian value of $k = \gamma+1$ will be used in this model. For a detailed coverage of Newtonian flow theory see references [Hay1, Col1, Jas1, and Pik1]. The application and validation of this approach to the determination of forces on arbitrary three–dimensional shapes is well documented in reference [Gen1].

The total pressure force exerted by the flow on the vehicle can be expressed as,

$$\underline{F}_{\mathrm{p}} = \tfrac{1}{2}\rho V^2 \int_A C_{\mathrm{p}}(x,y,z)\, \hat{\underline{n}}(x,y,z)\, dA \qquad (4.38)$$

where the pressure coefficient C_{p} and the normal vector $\hat{\underline{n}}$ are functions of position (x,y,z), and 'A' represents the exterior surface of the vehicle. Not explicitely shown in equation (4.38) is the complication that not all exterior surfaces have unobstructed projections into the flow. For any orientation of the vehicle, some of the surfaces will face downstream, and others may be blocked by other upstream surfaces. The first situation is easily rectified by not integrating external surface regions with downstream normal projections (i.e. $\hat{\underline{v}}\cdot\hat{\underline{n}} \leq 0$). The second situation, which is referred to as *aerodynamic shadowing*, can occur only for geometric shapes that are not convex. Typically, for hypersonic aircraft, this problem affects rearward aerodynamic surfaces which can be shadowed by the body or wings.

Since the geometric model represents the AHSV configuration using a number of polygonal panels, the integral in equation (4.38) can be replaced by the following sum:

$$\underline{F}_{\mathrm{p}} = \sum_{i=1}^{N} \underline{F}_{\mathrm{pi}} = \tfrac{1}{2}\rho V^2 \sum_{i=1}^{N} S_{\mathrm{i}}\, C_{\mathrm{pi}}\, \hat{\underline{n}}_{\mathrm{i}} \qquad (4.39)$$

Note that the pressure coefficient and the normal vector only have one value per panel (C_{pi}, $\underline{\hat{n}}_{\mathrm{i}}$), since each of the N panels is planar (with surface area S_{i}). The value of C_{pi} is given by,

$$C_{\mathrm{pi}} = (\gamma+1)\cdot(\underline{\hat{v}}\cdot\underline{\hat{n}}_{\mathrm{i}})^2 = (\gamma+1)\ sin^2(\psi_{\mathrm{i}}) \tag{4.40}$$

where the angle ψ_{i} is the incidence angle between the ith surface and the freestream flow.

For convenience, the AHSV geometry model actually keeps track of *outward* pointing normals (denoted by $\underline{\hat{n}}_{\mathrm{o}}$) for each panel. An array of normal vectors is stored in body–coordinates, and a transformation cosine matrix between body–coordinate and wind–coordinate axes is continually updated during simulation. The flow incidence angle for each panel is then easily calculated by transforming the normal vectors to the wind–coordinate system and finding the complement to the angle between the normal vector and the freestream velocity vector. The geometry is illustrated in Figure 4.8.

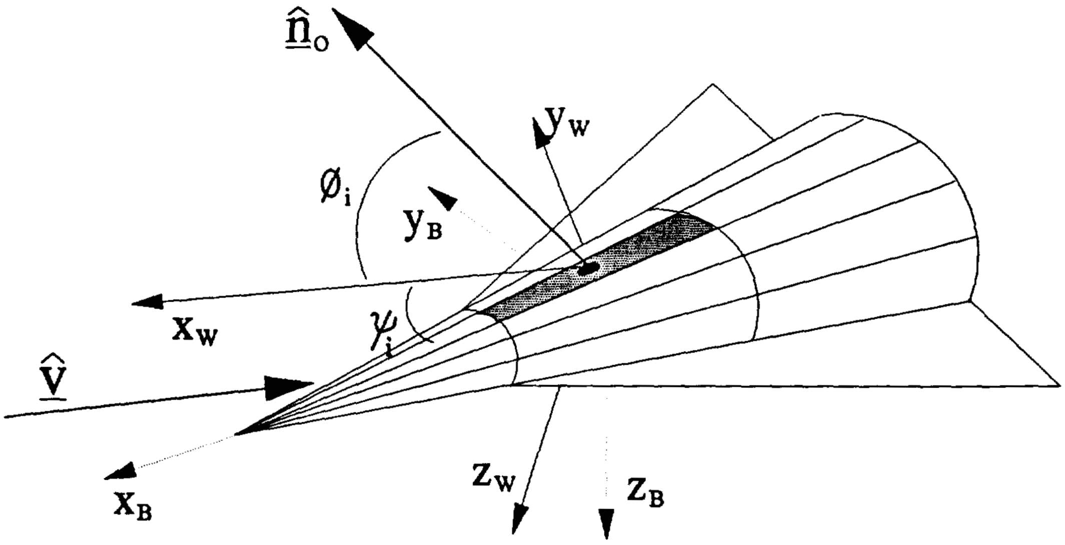


Figure 4.8: Angle of Incidence for Surface Panels

Since the vector $\hat{\underline{n}}_o$ is a unit vector, when expressed in the wind–coordinate system its elements are simply the cosines of the angles between $\hat{\underline{n}}_o$ and the coordinate axes. Therefore, the angle ϕ_i in the figure is the inverse cosine of the first element of $\hat{\underline{n}}_o$, and if $\hat{\underline{n}}_o = [n_{ox}\ n_{oy}\ n_{oz}]^T$ (in wind axes), the incidence angle is given by,

$$\psi_i = \pi/2 - cos^{-1}(n_{ox}) \tag{4.41}$$

In order to facilitate the calculation of aerodynamic moments, it is also assumed that the force on each panel acts at the surface centroid. The force vector for each panel can be expressed in body coordinates as,

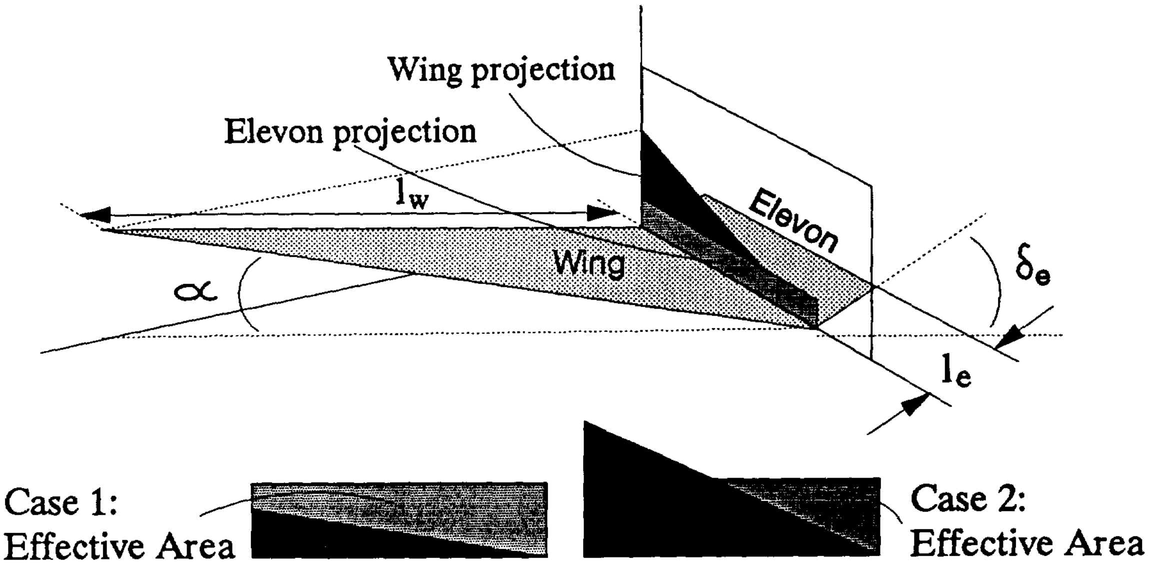
$$\underline{F}_{pi} = \tfrac{1}{2}\rho V^2\ S_i\ (\gamma+1)\ sin^2(\psi_i)\ \hat{\underline{n}}_i \tag{4.42}$$

Denoting $\underline{r}_i$ as the vector from the vehicle center of mass to the centroid of the ith panel, the total aerodynamic moment (due to pressure forces) is then given by,

$$\underline{M}_p = \sum_{i=1}^{N} \underline{r}_i \times \underline{F}_{pi} = \tfrac{1}{2}\rho V^2 \sum_{i=1}^{N} S_i\ (\gamma+1)\ sin^2(\psi_i)\ (\underline{r}_i \times \hat{\underline{n}}_i) \tag{4.43}$$

Aerodynamic shadowing is included in the model[10] by projecting the wing planform area and the rear elevon surface areas into a plane perpendicular to the freestream flow. By examining the overlapping projections, the degree of elevon shadowing and the resulting effective area can be computed. For a given vehicle geometry (wing sweep angle and elevon length), the fraction of elevon surface that is hidden behind the triangular shadow of the wing depends on the angle of attack and elevon deflection angle. There are two possible scenarios, shown in Figure 4.9, for which the effective elevon area is decreased. In the figure, S_e denotes the elevon surface area, and $\hat{S}_e$ is the area of the effective (unshadowed) portion.

[10] Since large sideslip angles are not expected to be part of the AMSV trajectory, shadowing of the aerosurfaces due to the fuselage is not included in the model.



$$\hat{Se} = Se\,[1 - \tfrac{1}{2}(\tfrac{l_w}{l_e})\sin(\alpha)/\sin(\delta_e - \alpha)] \qquad \hat{Se} = Se\,\tfrac{1}{2}(\tfrac{l_e}{l_w})\sin(\delta_e - \alpha)/\sin(\alpha)$$

Figure 4.9: Aerodynamic Shadowing of the Elevons

A final correction is now made to the overall Newtonian flow model for the AHSV in order to improve the pressure coefficient estimate for the second forebody ramp and the elevon surfaces. In computing pressure coefficients based only on local geometry, the simple Newtonian approximation assumes that the flow over each surface is independent. In actual fact, upstream surfaces do affect the flow over downstream surfaces, however, empirical data suggests that the Newtonian approximation works quite well for geometries that are convex. The same is not true for concave geometries, and the junction between the first and second ramp of the AHSV configuration presents such a surface. Since the second ramp area intercepts a large fraction of oncoming flow, the effect of an improved value for the pressure coefficient may be significant. The only other surfaces that may end up downstream of a concave corner are the elevons. Since these surfaces determine the pitch attitude control authority for the vehicle, an improved estimate for these pressure coefficients is also desirable.

For the forebody, a direct approach to improving the model would be to correct the properties of the flow incident on the second ramp by accounting for the changes caused by the first ramp.[11] Figure 4.10 defines the terms which are used in the following discussion.

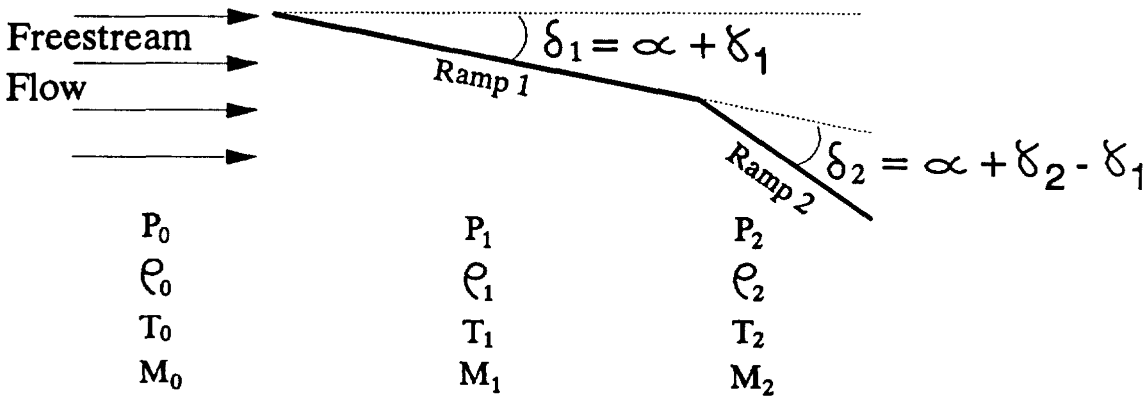


Figure 4.10: Flow Properties at the First and Second Ramp

Directly applying the simple Newtonian flow approximation to estimate the pressure coefficients for each ramp would give,

$$C_{p1} = \frac{P_1 - P_0}{\frac{1}{2}\rho_0 U_0^2} = (\gamma+1)\ sin^2(\delta_1) \tag{4.44a}$$

$$C_{p2} = \frac{P_2 - P_0}{\frac{1}{2}\rho_0 U_0^2} = (\gamma+1)\ sin^2(\delta_1+\delta_2) \tag{4.44b}$$

The corrected value for C_{p2} is obtained as follows:

$$C_{p2} = \frac{P_2 - P_0}{\frac{1}{2}\rho_0 U_0^2} = \frac{P_1 - P_0}{\frac{1}{2}\rho_0 U_0^2} + \frac{P_2 - P_1}{\frac{1}{2}\rho_0 U_0^2}$$

or,

$$C_{p2} = C_{p1} + \frac{P_2 - P_1}{\frac{1}{2}\rho_0 U_0^2}$$

Now, from the momentum equation (and the Newtonian flow assumption),

$$P_2 - P_1 = \rho_1\ U_1^2\ sin^2(\delta_2)$$

[11] There is an implicit assumption that the sideslip angle is small enough that the first ramp encounters the freestream flow first.

and using $U_1 \approx U_0 \, cos(\delta_1)$,

$$C_{p2} = C_{p1} + 2\left[\frac{\rho_1}{\rho_0}\right] cos^2(\delta_1)\, sin^2(\delta_2)$$

For a perfect gas $P = \rho \; R \; T$, and therefore,

$$\left[\frac{\rho_1}{\rho_0}\right] = \left[\frac{P_1}{P_0}\right]\left[\frac{T_0}{T_1}\right]$$

which becomes,

$$\frac{\rho_1}{\rho_0} = \frac{1 + \gamma_a M_0{}^2 \, sin^2\delta_1}{1+\frac{1}{2}(\gamma_a - 1) \; M_0{}^2 sin^2\delta_1}$$

by making use of the momentum and energy equations,

$$P_1 - P_0 = \rho_0 \; U_0{}^2 \, sin^2(\delta_1)$$

and,

$$T_1 - T_0 = \frac{U_0{}^2}{2\,C_p} \, sin^2(\delta_1)$$

Finally, C_{p2} can be expressed as,[12]

$$C_{p2} = C_{p1} + 2\left[\frac{1 + \gamma_a M_0{}^2 \, sin^2\delta_1}{1+\frac{1}{2}(\gamma_a - 1) \; M_0{}^2 sin^2\delta_1}\right] cos^2(\delta_1)\, sin^2(\delta_2) \tag{4.45}$$

The pressure coefficients for the elevon surfaces can be corrected in exactly the same manner. In that case, C_{p1} would represent the pressure coefficient for the wings, and the correction would only be necessary, of course, for certain combinations of the angle of attack and elevon deflection angle.

Shear Forces

In hypersonic flight, the most difficult aerodynamic forces to predict accurately are those due to friction. A complete theoretical understanding of the three–dimensional hypersonic boundary layer does not yet exist, and approximate numerical codes are computationally expensive. In this model, an engineering approach is taken to estimate the local skin friction coefficients. The entire AHSV

[12] This correction approaches the correct two-shock calculation for high Mach numbers. It should be used with caution, however, since it can actually be worse than the simple Newtonian estimate for low Mach numbers.

geometry is approximated (in cross section) by a wedge composed of two flat plates that bound the upper and lower surfaces. Local flow conditions for any point on the vehicle are then estimated using an inviscid flow analysis based upon oblique shock or expansion fan calculations. The local Mach and Reynolds numbers are then used to predict the location of the boundary layer transition on the basis of empirical data. In this manner, the distribution of laminar and turbulent flow over the vehicle surface panels is established. A reasonable value for the vehicle surface temperature is assumed, and the local laminar/turbulent skin friction coefficients are predicted from theoretical boundary layer results for flat plates. As with the aerodynamic pressure forces, these coefficients are integrated over the vehicle surfaces to obtain net forces and moments. Reference [Genl] utilizes a similar approach for the analysis of viscous forces on arbitrary three–dimensional bodies in hypersonic flows.

The shape of the wedge that encompasses the AHSV geometry is shown in Figure 4.11. The upper surfaces are contained within a flat plate at an angle equal to the angle of the upper fuselage. The lower surfaces are bounded by another plate which extends from the vehicle nose at an angle that intersects the bottom of the inlet lip.

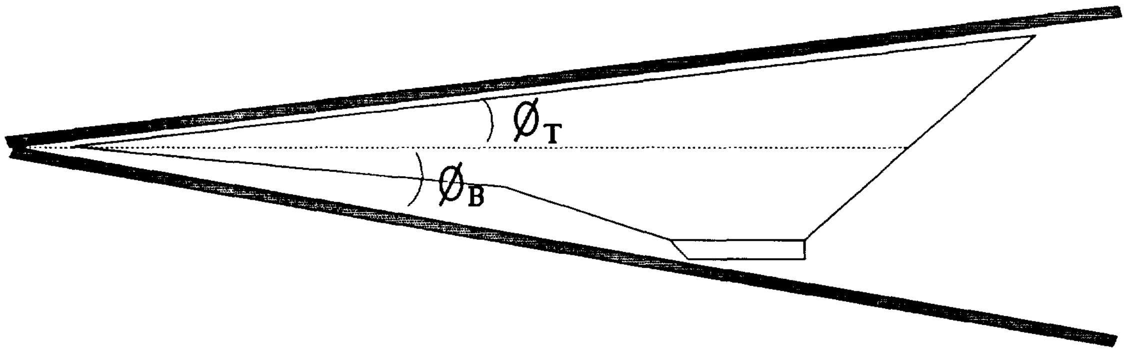


Figure 4.11: Approximate Wedge Representation for the AHSV Geometry

The flow turning angles for the top and bottom surfaces are approximated by,

$$\theta_T = \phi_T - \alpha \qquad (4.46a)$$

$$\theta_B = \phi_B + \alpha \qquad (4.46b)$$

where α is the angle of attack for the vehicle. In order to estimate local inviscid flow properties (for both the top and bottom surfaces), a shock calculation is done if the turning angle is greater than zero, and an expansion fan calculation is done if it is less than zero. These procedures are now described in detail.

Oblique Shock Calculation

An excellent coverage of the subject of supersonic flow and a discussion of normal and oblique shock theory is found in reference [Liel]. Here, an algorithm is developed for the determination of the flow conditions downstream of an oblique shock for a given flow turning angle and Mach number. A typical oblique shock configuration for the approximate AHSV wedge geometry is illustrated in Figure 4.12. The flow turning angle is θ, and β is the shock angle.

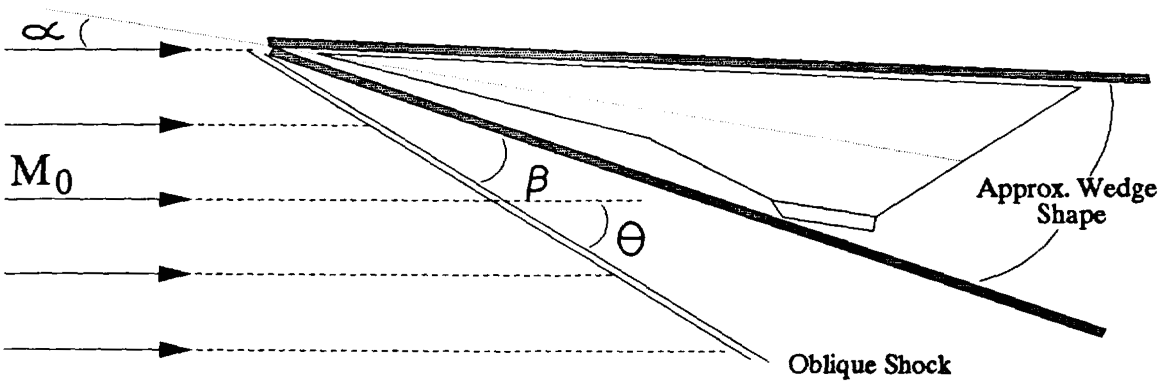


Figure 4.12: A Typical Oblique Shock for the AHSV Wedge Geometry

The relationship between the turning angle, the shock angle, and the Mach number for an oblique shock is given by,

$$tan(\theta) = \frac{2cot\,(\beta)(M^2 sin^2(\beta)-1)}{2+M^2(\gamma_a + cos(2\beta))} \qquad (4.47)$$

If the angle of attack is too large for a given wedge angle and Mach number, it may not be possible to satisfy equation (4.47). This means that the required flow turning angle is not possible with an oblique shock. Physically what occurs is that the shock detaches from the bow and locally forms a normal shock that tapers back and becomes oblique farther downstream. The flow in the region of the bow becomes subsonic and can therefore negotiate the larger turning angle. Since the flow properties are much more difficult to analyze under these conditions, the shock model for the AHSV is restricted to the case of attached shocks. Since this is also a desirable condition for the scramjet engine, it does not present a serious limitation.

A test for the existence of an oblique shock solution can be derived from equation (4.47) by differentiating with respect to β. [13] This gives an expression for the maximum possible turning angle θ_{max}.

$$\theta_{max} = tan^{-1}\left[\frac{2\left[\frac{1}{T}-1\right]^{\frac{1}{2}}(M^2\,T-1)}{M^2\,(\,\gamma_a+1-2\,T\,)+2}\right] \qquad (4.48a)$$

where,

$$T = \frac{1}{4\gamma_a M^2}\left[(\gamma_a+1)\,M^2 - 4 + \sqrt{\gamma_a+1}\,\left[(\gamma_a+1)\,M^4 + 8\,(\gamma_a-1)\,M^2 + 16\right]^{\frac{1}{2}}\right] \qquad (4.48b)$$

Now, if $\theta > \theta_{max}$ then the vehicle attitude is beyond the range of validity for the model. Otherwise, equation (4.47) is solved iteratively for the shock angle. A close estimate of β for small turning angles is given below, and can be used as a starting point in the iteration.

$$\beta = sin^{-1}\left[\frac{1}{M} + \frac{(\gamma_a+1)\;M}{4\;(M^2-1\,)^{\frac{1}{2}}}\,tan(\theta)\right] \qquad (4.49)$$

[13] The curves of shock angle, as a function of Mach number, have peaks at the maximum turning angle. See figure 4. 2, page 87, of reference [Liel].

Newton's method is used to find the exact shock angle. Rewriting equation (4.47) in a more useful form by making the substitutions $\tau = sin^2(\beta)$ and $\varphi = tan(\theta)$ gives,

$$f(\tau) = 2\sqrt{\frac{1-\tau}{\tau}}\,(M^2\tau-1) - M^2\varphi\,(\gamma_a+1-2\tau) - 2\varphi = 0 \tag{4.50}$$

and the derivative,

$$\frac{df}{d\tau}(\tau) = \frac{1}{\tau^2}\,(1-M^2\tau)\left[\frac{\tau}{1-\tau}\right]^{\frac{1}{2}} + 2\,M^2\left[\frac{\tau}{1-\tau}\right]^{-\frac{1}{2}} + 2\,M^2\varphi \tag{4.51}$$

Solving for τ iteratively,

$$\tau_{n+1} = \tau_n - f(\tau_n)\left[\frac{df}{d\tau}(\tau_n)\right]^{-1} \tag{4.52}$$

and the shock angle is obtained from $\beta = sin^{-1}(\sqrt{\tau})$.

The downstream conditions are then given by the following oblique shock relations once β is known:

$$M_1 = \frac{1}{sin(\beta-\theta)}\left[\frac{2 + (\gamma_a-1)\;M^2\;sin^2(\beta)}{2\gamma_a M^2 sin^2(\beta) - (\gamma_a-1)}\right]^{\frac{1}{2}} \tag{4.53a}$$

$$P_1 = \frac{1}{\gamma_a+1}\left[2\gamma_a M^2 sin^2(\beta) - (\gamma_a-1)\right] P_0 \tag{4.53b}$$

$$\rho_1 = \left[\frac{(\gamma_a+1)\;M^2\;sin^2(\beta)}{(\gamma_a-1)\;M^2\;sin^2(\beta) + 2}\right]\rho_0 \tag{4.53c}$$

$$T_1 = \left[\frac{P_1}{P_0}\right]\left[\frac{\rho_0}{\rho_1}\right] T_0 \tag{4.53d}$$

$$a_1 = \sqrt{\gamma_a R T_1} \tag{4.53e}$$

$$U_1 = M_1\,a_1 \tag{4.53f}$$

Expansion Fan Calculation

If the flow turning angle is negative ($\theta < 0$), the flow expands through a series of Mach waves that originate from the vehicle nose. The downstream Mach number can be found by using the Prandtl–Meyer expansion fan equation,

$$\omega(M) = \left[\frac{\gamma_a+1}{\gamma_a-1}\right]^{\frac{1}{2}} tan^{-1}\left[\left[\frac{\gamma_a-1}{\gamma_a+1}\right](M^2-1)\right]^{\frac{1}{2}} - tan^{-1}(\sqrt{M^2-1}\,) \qquad (4.54)$$

The difference between the upstream and downstream Prandtl–Meyer angles is equal to the expansion angle θ.

$$\theta = \omega(M_1) - \omega(M_0) \qquad (4.55)$$

The value of M_1 can be found iteratively using Newton's method by making the following simplifications:

$$\text{Let} \quad c = \left[\frac{\gamma_a+1}{\gamma_a-1}\right]^{\frac{1}{2}}, \quad \tau = \sqrt{M^2-1}\,, \quad K = \theta + \omega(M_0)$$

Equation (4.55) can then be written as $\omega(M_1) - K = 0$ or,

$$f(\tau) = c\ tan^{-1}\left[\frac{\tau}{c}\right] - tan^{-1}(\tau) - K = 0 \qquad (4.56)$$

where,

$$\frac{df}{d\tau}(\tau) = \frac{c^2}{\tau^2 + c^2} - \frac{1}{\tau^2 + 1} \qquad (4.57)$$

and iterating for τ,

$$\tau_{n+1} = \tau_n - f(\tau_n)\left[\frac{df}{d\tau}(\tau_n)\right]^{-1} \qquad (4.58)$$

the downstream Mach number is obtained from $M_1 = \sqrt{\tau^2+1}$.

Once the downstream Mach number is known, since the expansion of the flow is an isentropic process, the one–dimensional adiabatic compressible flow equations can be used to obtain the remaining downstream flow properties:

$$P_1 = \left[\frac{2 + (\gamma_a-1)\ M_0^2}{2 + (\gamma_a-1)\ M_1^2}\right]^{\frac{\gamma_a}{\gamma_a-1}} P_0 \qquad (4.59a)$$

$$\rho_1 = \left[\frac{2 + (\gamma_a-1)\ M_0^2}{2 + (\gamma_a-1)\ M_1^2}\right]^{\frac{1}{\gamma_a-1}} \rho_0 \qquad (4.59b)$$

$$T_1 = \left[\frac{2 + (\gamma_a-1)\ M_0^2}{2 + (\gamma_a-1)\ M_1^2}\right] T_0 \qquad (4.59c)$$

$$a_1 = \sqrt{\gamma_a R T_1} \tag{4.59d}$$

$$U_1 = M_1\, a_1 \tag{4.59e}$$

Boundary Layer Transition

It is difficult to accurately predict the location of the transition point for hypersonic boundary layers since very little data is available on this subject, particularly for the higher Mach numbers. According to [Lew3], published wind tunnel transition data only extends up to Mach *8*, and even these results conflict with flight test data. With very little else to work with, the transition model for the AHSV is based upon the sharp cone transition data from reference [She1], which extends from Mach *1* all the way up to Mach *16*. [14] This data gives transition point as a function of the local Mach and Reynolds numbers. An excellent fit to this data for Mach numbers in the hypersonic range is found to be given by,

$$Re_x \approx 4.87 \times 10^5\, e^{-0.333\, M} \tag{4.60}$$

where,

$$Re_x \equiv \frac{\rho\ U\ x}{\mu} \tag{4.61}$$

is the local Reynolds number, μ is the viscosity coefficient for air, and x is the local length along the surface. Since the variation of viscosity with altitude is very small compared to the changes in density and velocity along the AHSV trajectory, a constant value of $\mu = 1.58 \times 10^{-5}\ kg/(m\ s)$ is used. This approximation is within *10%* of the actual value for a range of altitudes up to at least *100,000 ft*. The transition length (T_x) for a given flight condition is therefore,

$$T_x = 4.87 \times 10^5 \left[\frac{\mu}{\rho\ U}\right] e^{-0.333\, M} \tag{4.62}$$

where the density, velocity, and Mach number (ρ, U, M) are the local values obtained from the shock or expansion fan equations for the upper and lower wedge surfaces. The criteria for laminar and turbulent boundary layer flows are,

$$\text{If} \begin{Bmatrix} x < T_x \\ x \geq T_x \end{Bmatrix} \text{the flow is} \begin{Bmatrix} \text{laminar} \\ \text{turbulent} \end{Bmatrix} \tag{4.63}$$

[14] How closely this cone data represents transition on a flat plate is uncertain.

where the length x is the distance from the vehicle nose to a local position on one of the vehicle surface panels.

The procedure for determining the distribution of laminar and turbulent panels begins by testing the four vertices of each surface against the transition length. If all corners are within the laminar flow regime then the entire panel is designated as a laminar surface. Similarly, the panel can be designated as being entirely turbulent. For these cases, the appropriate boundary layer model is used to compute skin friction coefficients for each of the panel vertices, and the composite coefficient for the panel is taken to be the mean of these values. In the case that the panel is split (partially laminar and partially turbulent), the appropriate skin friction coefficients are evaluated for each corner, and the overall coefficient for the panel is estimated by a weighted mean.

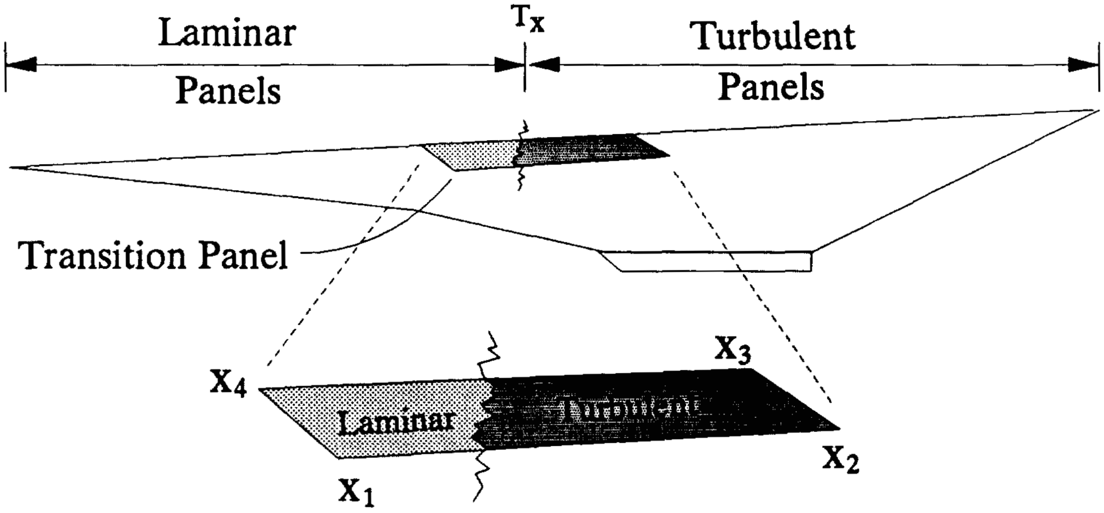


Figure 4.13: Distribution of Laminar and Turbulent Surface Panels

Laminar Panel: $\bar{C}_f = \Sigma\, C_{fi}/4$ *(4.64a)*

Turbulent Panel: $\bar{C}_f = \Sigma\, C_{fi}/4$ *(4.64b)*

Split Panel: $$\bar{C}_f = \frac{\Sigma \; C_{f\,i}/(|x_i - T_x|)}{\Sigma \; 1/(|x_i - T_x|)} \qquad (4.64c)$$

Laminar Skin Friction Coefficient

For surface coordinates that are within the laminar flow regime, the skin friction coefficient is calculated using a laminar compressible boundary layer model for flat plates with zero pressure gradients. The details of the model are given in reference [Mar1], which presents a chart of friction coefficient as a function of local Reynolds number and Mach number. A notable characteristic of the model is that the results are only slightly sensitive to wall temperature. Therefore, the effects of surface cooling can be safely neglected for this case. An accurate fit to the model is found to be given by the following relationship, which is reproduced in Figure 4.14 below.

$$C_{f_{\text{laminar}}} = \frac{1}{\sqrt{Re_x}}\left[0.3 + 0.364\, e^{-0.1\, M}\right] \qquad (4.65)$$

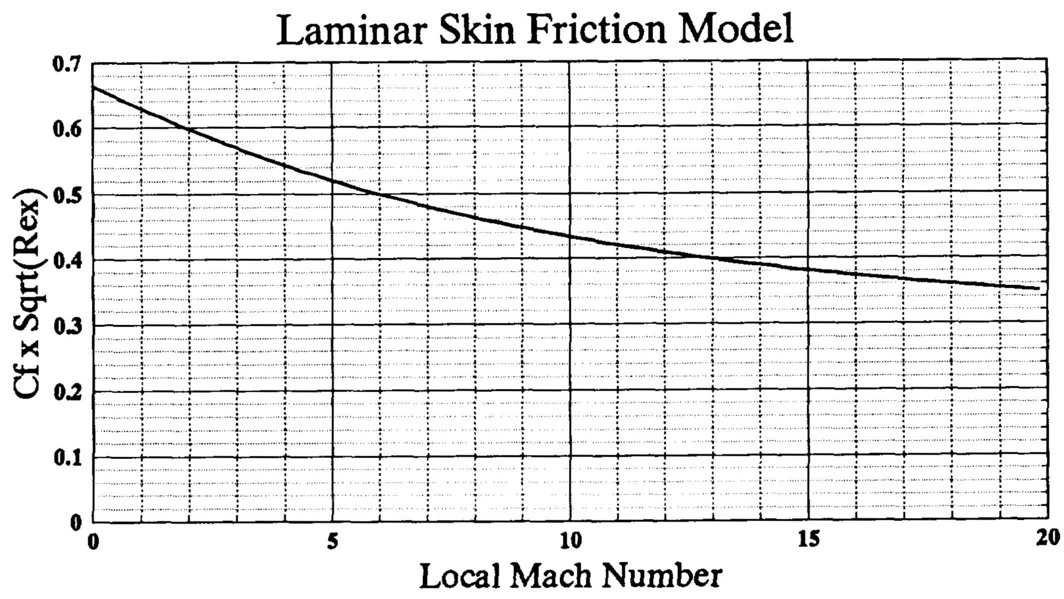


Figure 4.14: Skin Friction for Laminar Compressible Boundary Layers

Turbulent Skin Friction Coefficient

For the turbulent surface regions, reference [Marl] presents the flat plate compressible turbulent boundary layer model of Van Driest. For turbulent flow, the friction coefficient is seen to decrease rapidly with increasing wall temperature (T_w) as well as with Mach number. Therefore, in this case, the skin friction coefficient depends upon local Reynolds number, Mach number, and the temperature ratio (T_w/T_0), where T_0 is the freestream temperature. The chart, reproduced in Figure 4.15, gives the skin friction coefficient in terms of the Reynolds number based on boundary layer thickness (rather than length).

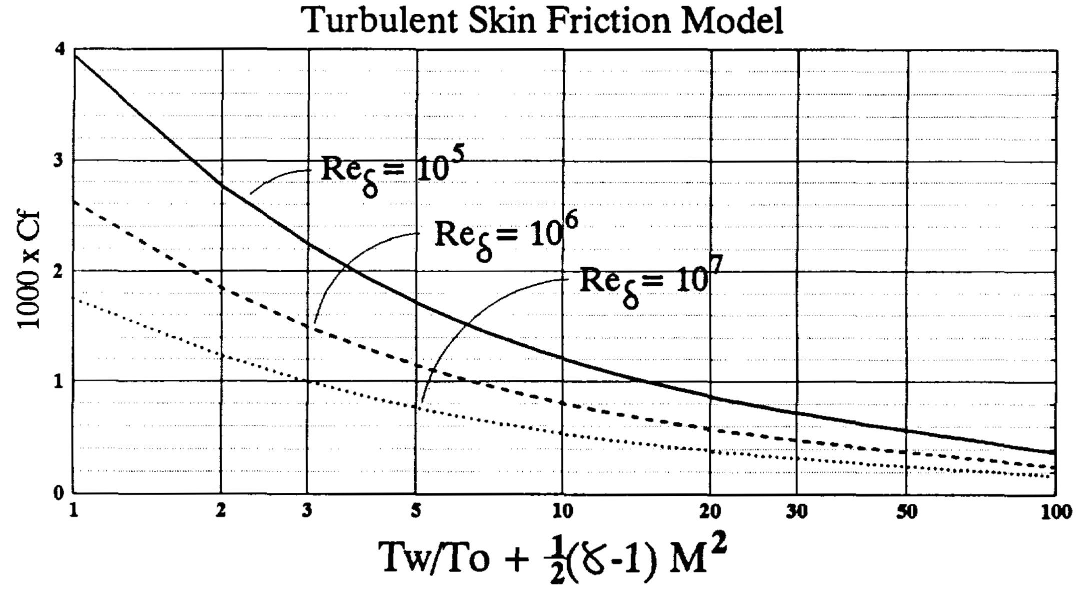


Figure 4.15: Skin Friction for Turbulent Compressible Boundary Layers

The local Reynolds number is first converted to a thickness value using (from [Marl]),

$$\log_{10}(Re_\delta) = 0.95\log_{10}(Re_x) - 1.42 \tag{4.66}$$

and the skin friction coefficient is obtained using the following curve fit to the model:

$$C_{f_{\text{turbulent}}} = \frac{1}{1000}\left[\frac{2}{3}\right]^{(\log_{10}(Re_\delta)-5)}\left[b_0 + b_1\chi + b_2\chi^2 + b_3\chi^3\right] \tag{4.67}$$

where,
$$\chi = \log_{10}\left[\frac{T_w}{T_0} + \left(1+\frac{\gamma_a - 1}{2} M^2\right)\right] \qquad (4.68)$$

and the coefficients $b_0 \ldots b_3$ are given by,

$$[b_0, b_1, b_2, b_3] = [3.935, -4.459, 2.129, -0.3955] \qquad (4.69)$$

Net Forces & Moments

Once the skin friction coefficients have been estimated for each surface panel, the overall friction forces and moments for the vehicle can be computed. The shear force on the ith panel can be expressed as,

$$\underline{F}_{si} = \tfrac{1}{2}\rho_i V_i^2 \, S_i \, C_{fi} \, \hat{f}_i \qquad (4.70)$$

where the density and velocity are local values, and the unit vector $\hat{f}_i$ is in the direction of the friction force. From geometry it is not too difficult to show that (for the outward pointing unit normal vector defined by $\hat{\underline{n}}_o = [n_{ox} \; n_{oy} \; n_{oz}]^T$),

$$\hat{f}_i = -cos(\psi_i) \, \hat{\underline{x}}_w + cos(\xi_i) sin(\psi_i) \, \hat{\underline{y}}_w + sin(\xi_i) sin(\psi_i) \, \hat{\underline{z}}_w \qquad (4.71)$$

where ψ_i is the angle of incidence ($\psi_i = \pi/2 - cos^{-1}(n_{ox})$), ($\hat{\underline{x}}_w$, $\hat{\underline{y}}_w$, $\hat{\underline{z}}_w$) are the unit axes for wind coordinates, and the angle ξ_i is given by $\xi_i = tan^{-1}(n_{oz}/n_{oy})$. The total shear force on the vehicle is,

$$\underline{F}_s = \sum_{i=1}^{N} \underline{F}_{si} = \sum_{i=1}^{N} \tfrac{1}{2}\rho_i V_i^2 \, S_i \, C_{fi} \, \hat{f}_i \qquad (4.72)$$

and the shear moment is given by,

$$\underline{M}_s = \sum_{i=1}^{N} \underline{r}_i \times \underline{F}_{si} = \sum_{i=1}^{N} \tfrac{1}{2}\rho_i V_i^2 \, S_i \, C_{fi} \, (\underline{r}_i \times \hat{f}_i) \qquad (4.73)$$

where $\underline{r}_i$, once again, is the vector from the vehicle center of mass to the centroid of the ith panel.

Finally, the total aerodynamic forces and moments are now given as the sum of the pressure and shear components:

$$\underline{F}_{\text{aero}} = \underline{F}_{\text{p}} + \underline{F}_{\text{s}} \tag{4.74a}$$

$$\underline{M}_{\text{aero}} = \underline{M}_{\text{p}} + \underline{M}_{\text{s}} \tag{4.74b}$$

4.5 Propulsion Model

As mentioned earlier, in this research the models for the AHSV are limited to the hypersonic portion of the trajectory from Earth to orbit. This is, in part, to limit the scope of the problem, but also because the hypersonic flight regime remains largely unexplored and is of primary interest for this class of vehicles. For the range of Mach numbers that will be considered (Mach *6* to Mach *16*), the scramjet engine is currently the only viable option for air–breathing propulsion. In this section, the important features of a representative scramjet propulsion model are presented.

4.5.1 THE SUPERSONIC COMBUSTION RAMJET (SCRAMJET)

As flight speed increases beyond Mach *3*, the utility of conventional aircraft propulsion systems declines rapidly due to the effects of extreme temperatures on compressor blades and other turbomachinery. The maximum flight velocity for a turbojet engine is limited by the temperature constraints required to assure the integrity of the internal components. At higher Mach numbers, the ramjet propulsion system outperforms the turbojet, since the required flow compression can be achieved by ramming the airflow past compression surfaces, without expending the energy to drive turbines or a compressor. Beyond Mach *6*, however, the ramjet engine also suffers from extreme temperatures in the inlet, and becomes inefficient due to losses in total pressure across the compression shocks. At very high Mach numbers, it is possible to achieve air–breathing propulsion through the use of a scramjet propulsion cycle. The scramjet engine avoids the problems above by mixing and burning the fuel within a supersonic airstream. A comparison of the relative efficiencies of the various propulsion cycles, measured in terms of ISP (specific impulse), is presented in Figure 2.1 of reference [Matl]. The applicable range of Mach numbers for each type of propulsion is also discussed in detail.

The fundamental principle behind the operation of a scramjet engine is the compression of the inlet flow through a series of oblique shocks and the subsequent injection and combustion of the fuel at supersonic speeds. Since the flow through the combustor section is supersonic, the residence time of the flow and, therefore, the time available for the combustion process is typically very short. Hydrogen is therefore considered to be the only viable fuel for a SSTO vehicle, due to its high combustion rate and high specific impulse.

Experimental results with scramjet engines have demonstrated that a large mass flow of air and a large expansion surface are necessary for efficient operation at hypersonic speeds. This has lead to the current concept of an airframe–integrated scramjet engine [Matl]. For the AHSV configuration, this means that the entire forebody of the vehicle acts as an inlet compression surface, and the entire aftbody undersurface acts as an expansion nozzle. The successful integration of the airframe (aerodynamics) and propulsion system is one of the most challenging design issues facing the development of an AHSV. Some discussion on this topic was presented in Chapter 2, and it is addressed in detail in references [Edwl, Weil].

Figure 4.16 illustrates the geometry of an airframe–integrated scramjet propulsion system. In general, the forebody compression can be accomplished with a smooth surface, or through a series of compression ramps as shown in the figure.[15] Oblique shocks are formed as a result of the flow turning angles that are enforced by the forebody geometry and orientation with respect to the flow. These shocks compress the incoming flow, decrease the Mach number, and guide the flow toward the inlet to the combustor. Inside the combustor, fuel struts inject hydrogen into the airstream, and the combustion process adds energy to the flow by increasing its

[15] For the analysis in this research the forebody is modelled as a series of two ramps.

temperature. Thrust is generated from the pressure forces on the aft undersurface of the vehicle as the flow expands and cools in the nozzle (converting thermal energy into kinetic energy).[16]

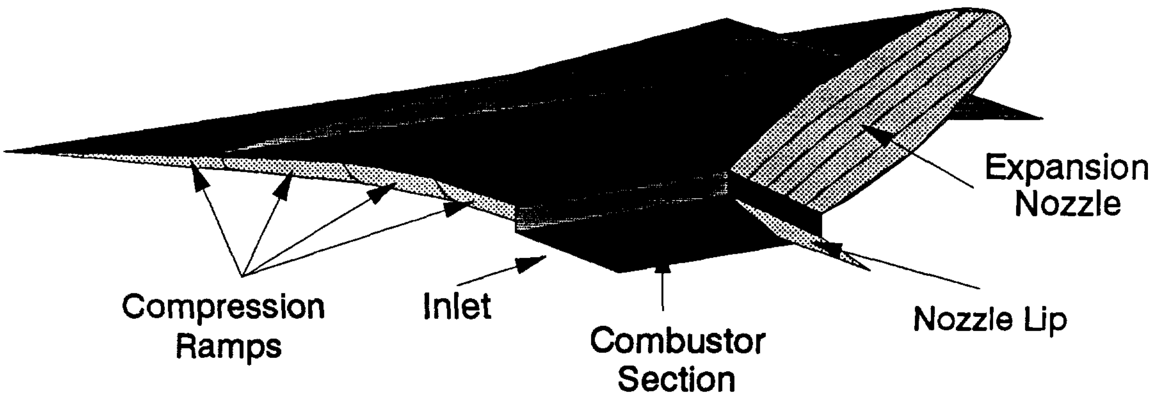


Figure 4.16: An Airframe–Integrated Scramjet Propulsion System

Although the scramjet engine is the primary development that makes it possible to achieve hypersonic air–breathing flight, it is also the source of many difficulties. The fact that the engine's performance is so closely coupled to the airframe design also means that it is very sensitive to attitude. The vehicle's angle of attack profoundly affects the character of the forebody shocks, which in turn affect the character of the flow entering the combustor. Inlet temperature, pressure, and mass flow directly vary with attitude, as do the effects of local heating, spillage drag, shock ingestion, and inlet boundary layer thickness. The one–sided nature of

[16] Note that for the AHSV (or airframe-integrated) configuration, a large fraction of the nozzle is only one sided. This property is the source of a myriad of design problems ranging from trim penalties associated with undesirable propulsion moments, to the complication of the interaction between exhaust gases and the freestream flow under different flight conditions.

the nozzle also means that the characteristics of the flow expansion are susceptible to effects related to angle of attack. Good performance is also difficult to achieve for a wide range of flight conditions, since the character of the inlet shocks depends directly on Mach number as well. For a fixed geometry, there is only one combination of Mach number and angle of attack for which performance is optimal (a design condition where all shocks coalesce directly on the inlet lip). In all other flight conditions performance will be degraded (due to flow spillage, losses, etc.). The capability for a variable geometry may be necessary to achieve adequate performance over the full trajectory. For an in–depth discussion of the difficulties associated with scramjet engine design for hypersonic vehicles see references [Lew3, Mar1, Mat1].

Even with the capabilities of today's supercomputers, a numerical algorithm for the complete three–dimensional flow analysis of a scramjet propulsion system is beyond the state–of–the–art. Deficiencies in the current theory, finite computational speed, and the scarcity of hypersonic (in–flight) data required for code validation, limit the ability of current CFD codes to predict the detailed flow properties.

The important characteristics of such an engine, however, can be modelled using approximate methods. One such approach is presented in reference [Mat1], which performs a detailed numerical analysis of the flow through the inlet, combustor, and nozzle sections of a scramjet engine. Some assumptions and approximations are made to facilitate the analysis (such as neglecting finite rate chemistry), but the results are still useful in comparing the performance of different designs.

For the purposes of this research, a propulsion model is desired that can realistically represent the behavior of a scramjet engine in response to on–line control commands in a simulated flight. For this application, it is clearly not feasible to perform a detailed flow analysis for each new flight condition. An alternative approach to modelling a scramjet engine was developed at M.I.T. by Chiang–Hwa Ren under the supervision of Prof. Manuel Martinez–Sanchez [Ren1].

This model avoids the computational burden of a CFD approach by making certain assumptions, and using approximate analysis methods to estimate the flow properties in the inlet, combustor, and nozzle sections of the engine.

The propulsion model for the AHSV is based on the work presented in [Ren1]. Modifications to the algorithms have been made to compute overall forces and moments by integrating local pressures over the internal engine surfaces; to account for some missing contributions in the overall thrust; to obtain some desired quantities that did not appear in the original model; to provide extensive error checking to handle conditions for which the model fails; and finally, to provide a consistent interface between the propulsion model and the geometric and aerodynamic models. Otherwise, however, the fundamental analysis methods are exactly as presented in [Ren1].

4.5.2 HYPERSONIC PROPULSION MODEL

As with the geometric and aerodynamics models, the scramjet propulsion model performs an analysis for a parameterized configuration. The performance of a wide range of possible engine designs can therefore be reproduced for any desired flight condition. There are *23* design variables that are required to specify a particular engine design. These parameters (listed below in Table 4.2) include items related to the geometric, atmospheric, and internal specifications corresponding to a design flight condition for which the engine will be optimized. The model decides upon the optimal inlet ramp lengths, combustor area ratio, and nozzle geometry in order to achieve maximum performance. Once designed, *17* variables are needed to specify a set of off–design conditions (Table 4.3). The model then performs an off–design analysis, and generates overall thrust and moment estimates along with *8* other useful outputs such as ISP and fuel mass flow rate (Table 4.4).

Since the scramjet engine is designed for optimal performance at a certain flight condition, the design process is an iterative procedure that involves the aerodynamic design as well. That is, given a set of geometric specifications for an AHSV (Table 4.1), the engine design for a certain choice of the design Mach number and angle of attack may require modifications to the forebody or nozzle geometry.

(1) Design Mach number

(2) Design Angle of Attack

(3) Ramp Angle 1

(4) Ramp Angle 2

(5) Capture Height

(6) Combustor Length

(7) Combustor Width

(8) Combustor Wall Temperature

(9) Nozzle Length

(10) Nozzle Lip Length

Inlet Wall Boundary Layers (Laminar or Turbulent)

(11) Wall 1 – 1st Ramp

(12) Wall 2 – 2nd Ramp

(13) Wall 3 – Combustor Top

(14) Wall 4 – Combustor Bottom

Inlet Wall Temperatures

(15) Wall 1 – 1st Ramp

(16) Wall 2 – 2nd Ramp

(17) Wall 3 – Combustor Top

(18) Wall 4 – Combustor Bottom

(19) Design Hydrogen Injection Temperature

(20) Stoichiometric Oxidizer/Fuel Mass Ratio

(21) Design Altitude

(22) Ambient Pressure

(23) Ambient Temperature

Table 4.2: Scramjet Propulsion Design Parameters

(1) Flight Mach number

(2) Actual Angle of Attack

(3) Nozzle Lip Length

(4) Combustor Wall Temperature

Inlet Wall Boundary Layers (Laminar or Turbulent)

(5) Wall 1 – 1st Ramp

(6) Wall 2 – 2nd Ramp

(7) Wall 3 – Combustor Top

(8) Wall 4 – Combustor Bottom

Inlet Wall Temperatures

(9) Wall 1 – 1st Ramp

(10) Wall 2 – 2nd Ramp

(11) Wall 3 – Combustor Top

(12) Wall 4 – Combustor Bottom

(13) Hydrogen Injection Temperature

(14) Oxidizer/Fuel Mass Ratio

(15) Flight Altitude

(16) Ambient Pressure

(17) Ambient Temperature

Table 4.3: Off–Design Conditions for the Scramjet Engine

(1) Horizontal Thrust	*(2) Vertical Thrust*
(3) Propulsive Pitch Moment	*(4) Specific Impulse*
(5) Flow Expansion Fraction	*(6) Combustion Temperature*
(7) Combustion Pressure	*(8) Fuel Mass Flow Rate*
(9) Ideal Thrust from ISP	*(10) Design Nozzle Angle*

Table 4.4: Outputs from the Scramjet Model

This, however, would change the aerodynamic characteristics. The overall design objective, therefore, is to find a configuration for which the aerodynamic and propulsion forces (and moments) are in equilibrium for one particular (design) flight condition. If a nominally accelerating design is desired then the configuration must include a net force while balancing aerodynamic and propulsive moments.

This iterative design process has been partially automated as part of the AHSV simulation. Starting with a set of aerodynamic and propulsion design values, the propulsion model is used to reconfigure the vehicle geometry for optimal performance at the design flight condition. All models are used to predict net forces and moments, and if the vehicle is not in the desired state, several of the geometric variables and the design angle of attack are adjusted accordingly. The process is repeated until an acceptable design is found.[17]

Even though the vehicle may never even encounter the exact design flight condition, it still serves as an important reference point from which off–design performance can be measured. Experience with the model has also shown that the design Mach number influences the range of Mach numbers for which good performance can be achieved.

[17] This is not a simple task, and often requires several initial configurations before converging to a successful design.

Inlet Analysis

The modelling of the scramjet inlet, as presented in [Ren1], is accomplished by assuming that the forebody flow can be approximated by a two–dimensional geometry. The classical assumption of a separate inviscid outer flow and a thin viscous boundary layer is used to estimate the flow properties along each ramp and inside the inlet. For the design cycle, a shock matching condition at the lip of the inlet is used as a criterion to assure optimal inlet performance. The oblique shock relationships are used to compute the external flow, which is used to provide boundary conditions for the boundary layer calculations. The boundary layer thickness is approximated by a linear growth, and the ramp surfaces are pushed back so that the displacement thickness of the boundary layer does not interfere with the external flow. This, combined with the shock matching condition, determines the design geometry of the inlet.

For the off–design calculations, the solution is obtained through an iteration between the oblique shock and boundary layer equations (since the geometry is now fixed). Here, the inviscid flow turning angles are adjusted on the basis of the new boundary layer thickness, and the shock calculations are repeated and followed again by the boundary layer calculations until the solution is found.

The off–design character of the inlet flow depends on whether the flight condition is above or below the design Mach number. At higher Mach numbers the shocks are swept into the inlet, while at lower Mach numbers they do not even reach the lip (causing flow spillage and creating additive drag). The variation with angle of attack is also significant, causing the shocks to either coalesce before the lip, or move apart. In each of these cases, a control volume approach is used to determine the exit conditions of the inlet (i.e. conditions at the entrance to the combustor). The possible inlet shock conditions are illustrated in Figure 4.17.

For laminar flow, the inlet boundary layer is modelled using the Crocco relations and the integrals for momentum and boundary layer thickness. The predicition of boundary layer transition is avoided in this analysis by assuming that the 2nd ramp is turbulent, and the effective velocity approach of van Driest is used to estimate the thickness of the turbulent boundary layer. [Renl]

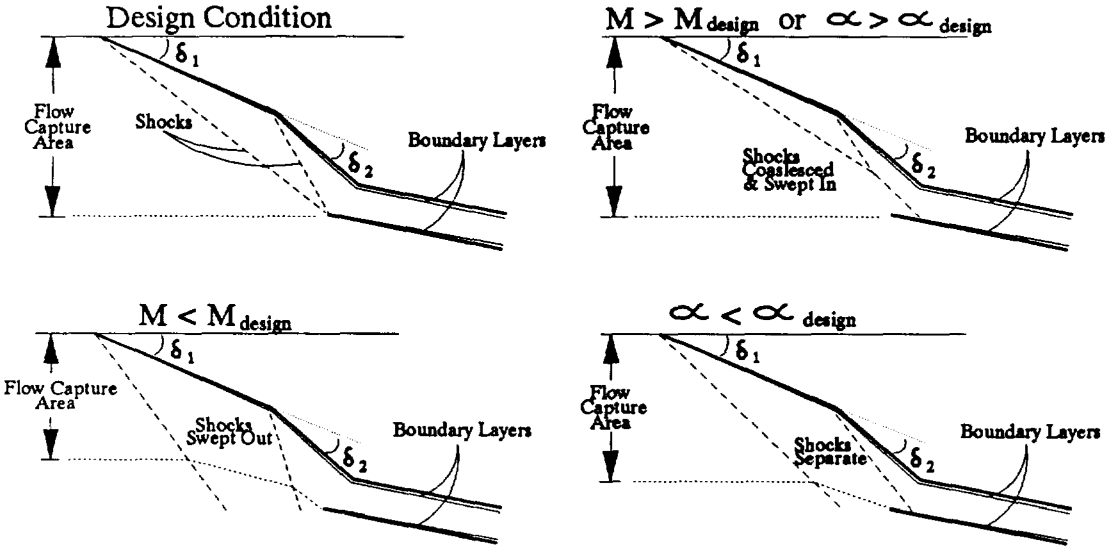


Figure 4.17: Design and Off–Design Inlet Conditions

A detailed description of the inlet analysis is presented in [Renl], where it is also pointed out that the results of this approach agree well with higher fidelity models.

Combustor Analysis

The combustion process is the most difficult part of the scramjet propulsion system to model. The effects of high temperature, finite–rate chemistry, and hypersonic boundary layers, along with the uncertainties associated with turbulent mixing and combustion, are more pronounced within the combustor than with the external flows. Since many of these issues are still beyond the state of the current theory, the model used for the AHSV is correspondingly approximate. This analysis circumvents the details of the combustion process by utilizing a control volume approach to determine the properties of the flow exiting the combustor.

The approach taken in [Ren1] is fundamentally based on the Crocco relationship for the integral analysis of a combustor,

$$\frac{P_2}{P_1} = \left[\frac{A_1}{A_2}\right]^{\frac{\epsilon}{\epsilon - 1}} \tag{4.75}$$

which relates the pressure distribution to the cross sectional area. The constant ϵ is an unknown constant to be determined.

The model uses mass and momentum conservation for a control volume inside the combustor, to obtain the exit velocity and pressure as a function of the inlet conditions, the parameter ϵ, and the oxidizer/fuel mass ratio. An energy balance is then applied to determine the enthalpy at the exit. Assuming chemical equilibrium at the exit, the exit temperature can be related to the composition of the products of combustion. Therefore, the correct exit temperature is found through an iteration until the combined enthalpy of all the exit species matches the calculated value. Finally, the value of ϵ is obtained through another iteration until the resulting area ratio is correct. A correction to this analysis is applied to account for the boundary layer that travels into the combustor from the inlet. Refer to [Ren1] for more details.

Nozzle Analysis

The performance of the nozzle is also evaluated using a control volume approach. For the design condition, pressure matching at the lip and a parallel exhaust flow are used as conditions to design the nozzle geometry. The method of waves is then used to compute the local flow properties throughout the expanding flow field. For the purpose of analysis, the upper contour of the nozzle (vehicle aft undersurface) is shaped as a curve which forces a compression turn to cancel any expansion waves reflected from the lip. This simplifies the calculations and results in a parallel flow (for the design condition). An approximate linear fit to the curved nozzle is passed back to the AHSV simulation, and the force calculations are actually done by integrating local pressure over the approximate flat nozzle surface.

The nozzle analysis accounts for the effects of an expansion which includes both a chemical equilibrium and a frozen flow component. For the frozen flow, an isentropic expansion is assumed, and an iterative procedure to find the exit temperature is developed from the enthalpy equation. This is done for two cases: pressure matching (for design), and a known area ratio (for off–design). A chemical kinetics code is used to determine the freezing point pressure, and a chemical equilibrium analysis, similar to that used for the combustor, is used for the equilibrium flow section.

For off–design conditions, the flow streamlines may no longer emanate straight back from the nozzle lip. Whether the nozzle flow is over or under expanded is determined by the area ratio from the control volume analysis. The model includes an additive drag component for each of these cases. Again, for more details refer to [Ren1].

Propulsive Forces & Moments

The computation of thrust and thrust moment for the scramjet propulsion system is accomplished through the integration of pressure forces along the interior surfaces of the combustor and nozzle. The forces and moments on the inlet are already included as part of the aerodynamics model. The entrance and exit conditions for each section (combustor and nozzle) are known from the results of the scramjet model. The Crocco relationship (4.75) is now used to march along the interior of the engine (downstream), and compute the local pressure as a function of the cross sectional area. Positive contributions to the thrust can only occur in sections with a diverging cross section. A net vertical force occurs in the last stage of the nozzle where there is no lower lip. Figure 4.18 illustrates these contributions to the overall thrust. Also shown in the figure is the location of the vehicle nose which is used as a reference point for the thrust moment calculations. The moment is later transferred to the vehicle center of mass by the dynamics model.

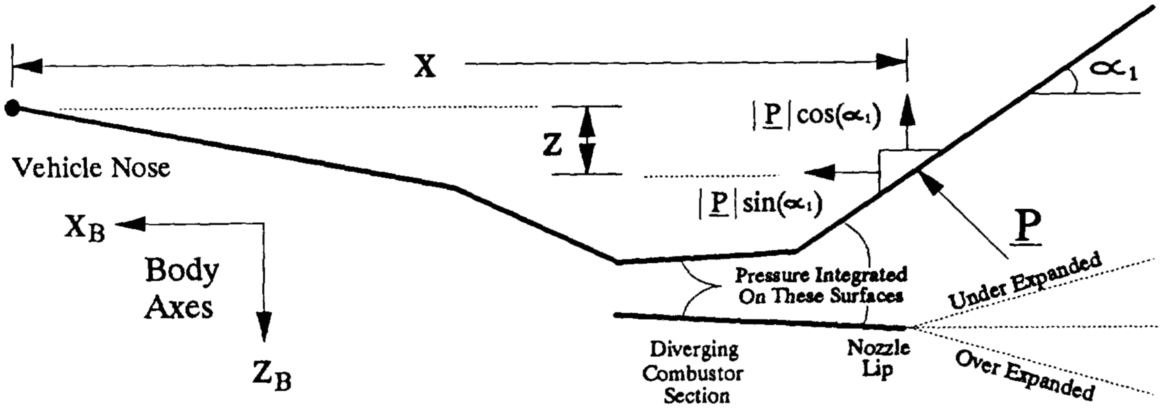


Figure 4.18: Calculation of Propulsive Forces and Moments

The total thrust can be expressed as, [18]

$$\underline{F}_{\mathrm{T}} = \int_S \underline{P}\; dS \tag{4.76}$$

with the horizontal and vertical components given by,

$$\underline{F}_{\mathrm{Tx}} = \int_S |\underline{P}|\; sin(\alpha_1)\; \omega_{\mathrm{c}}\; dz \tag{4.77a}$$

$$\underline{F}_{\mathrm{Tz}} = -\int_S |\underline{P}|\; cos(\alpha_1)\; \omega_{\mathrm{c}}\; dx \tag{4.77b}$$

where α_1 is the angle of the surface and ω_c is the combustor width (also the nozzle width). The thrust moment is found to be,

[18] Note that the integration of pressure along the interior surfaces of the engine $\int \underline{P}\, dS$ accounts for both the pressure and momentum changes in the flow. For example, in the horizontal direction, using A for the the cross sectional area, the integral $\int P\, dA$ can be written as $\int d(PA) - \int A\, dP$. From conservation of mass, it is easy to show that locally $A\, dP = -\dot{m}\, du$ ($\dot{m}$ = mass flow rate, u = velocity). Therefore, $\int P\, dA = (PA)_{\mathrm{out}} - (PA)_{\mathrm{in}} + \int \dot{m}\, du$, or $Thrust = (PA)_{\mathrm{out}} - (PA)_{\mathrm{in}} + \dot{m}\, (u_{\mathrm{e}} - u_{\mathrm{i}})$.

$$\underline{M}_{\mathrm{T}} = \int_S (\underline{r} \times \underline{P})\, dS$$
$$= \int_S z\, |\underline{P}|\; sin(\alpha_1)\; \omega_c\; dz + \int_S x\, |\underline{P}|\; cos(\alpha_1)\; \omega_c\; dx \qquad (4.78)$$

A correction is now made to account for the thrust due to the momentum change of the fuel as it is injected into the combustion chamber. Since the hydrogen fuel is preheated[19], its temperature and pressure are very high just prior to injection. In fact, the fuel may even be injected at a greater velocity than the air flow in the combustor. This effect is not included in the original model, and the simple integration of pressure on the walls of the combustor and nozzle overlooks the forces on the walls inside the fuel reservoir.

The contribution to the thrust is obtained by assuming that the fuel pressure matches the combustor pressure at the injection point [Marl]. Beginning with the energy equation,

$$(u_f)_{\mathrm{inj}} = [2\; Cp_f\; (T_{f0} - T_{fi})]^{\frac{1}{2}} \qquad (4.79)$$

where $(u_f)_{\mathrm{inj}}$ is the injection fuel velocity, Cp_f is the specific heat of the fuel (for hydrogen $Cp_f \approx 14{,}500\ J/(kg\ K)$), T_{fo} and T_{fi} are the reservoir and injection temperatures of the fuel. For an adiabatic expansion to the injection point this can be expressed as,

$$(u_f)_{\mathrm{inj}} = \left[2\; Cp_f\; T_{fo} \left[1 - \left[\frac{P_{fi}}{P_{fo}}\right]^{\frac{\gamma-1}{\gamma}}\right]\right]^{\frac{1}{2}} \qquad (4.80)$$

where P_{fo} and P_{fi} are the reservoir and injection pressures. For a fuel mass flow rate of $\dot{m}_f$, the momentum equation gives a force of $F_{\mathrm{inj}} = \dot{m}_f\; (u_f)_{\mathrm{inj}}$ which must be distributed over the walls of the fuel reservoir. This additional thrust is therefore given by,

$$F_{\mathrm{inj}} = \dot{m}_f \left[2\; Cp_f\; T_{fo} \left[1 - \left[\frac{P_{fi}}{P_{fo}}\right]^{\frac{\gamma-1}{\gamma}}\right]\right]^{\frac{1}{2}} \qquad (4.81)$$

[19] It is used for cooling other components.

For simplicity, the reservoir is assumed to be at a pressure (P_{f_0}) of *3* atmospheres, and a temperature (T_{f_0}) of *1200* K for fuel that has already been used for cooling the vehicle. These values are reasonable, and if more details for a specific engine design were available then more accurate values could be used.

For the AHSV simulation other assumptions were made with regard to the propulsion system model. The boundary layers were assumed to be laminar for the 1st ramp and the lower surface of the combustion chamber. The 2nd ramp and the upper surface of the combustor were assumed to be turbulent. In addition, the temperature of all the walls for the inlet were assumed to be maintained at *1000* K. The combustor wall temperature was fixed at *2000* K. These assumptions were also made in reference [Ren1] for various reasons which are motivated therein.

A final modification to the original scramjet model is the inclusion of a thrust vectoring capability. Due to the effects of shadowing on the elevon control surfaces, it was determined that some degree of directed thrust control is necessary. For the AHSV propulsion model this is accomplished using a deformable nozzle geometry. Since the shape of the vehicle's aft undersurface would be difficult to change appreciably, the length of the lower nozzle lip is assumed to be controllable. The simulation passes a lip length control input to the propulsion model, which performs the thrust calculation for the new geometry.

For completeness, a few representative curves from the propulsion model are reproduced in this section. For comparative purposes, all of the plots are for the same engine design. The design conditions were Mach *10*, altitude *30 km*, $\alpha = 4.3^o$, $\gamma_1 = 4^o$, $\gamma_2 = 6^o$, $l_v = 62\ m$, $l_c = 12\ m$, $\omega_c = 15\ m$, $\beta = 25^o$, $h_c = 5\ m$, $l_N = 12.5\ m$ (see Figures 4.2a–b).

Figure 4.19 charts the horizontal thrust (in wind axes) as a function of Mach number. Note that the altitude is held constant, so that a higher Mach number corresponds to a higher velocity. The inlet shocks are swept in, and, for a constant fuel mixture ratio, the additional mass flow of air is matched by additional fuel, therefore, increasing the thrust. The vertical thrust is not shown but it also increases with Mach number. The moment for this particular case was pitch down and decreasing with Mach number.

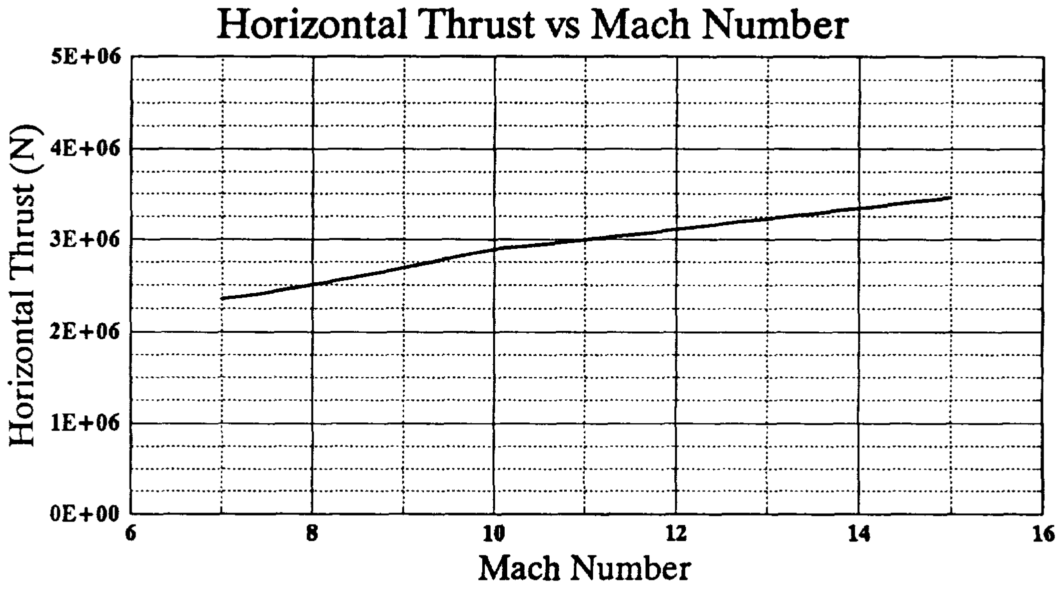


Figure 4.19: Typical Scramjet Behavior (Horizonal Thrust vs Mach Number)

In Figure 4.20 the horizontal thrust is plotted as a function of angle of attack. A larger capture area results in a larger mass flow and more thrust. Again, this is for a constant stoichiometric ratio. Of course, the gain in thrust is offset by an increase in aerodynamic drag which is not visible here. The vertical thrust also increases with angle of attack, and the pitch moment becomes larger pitch down.

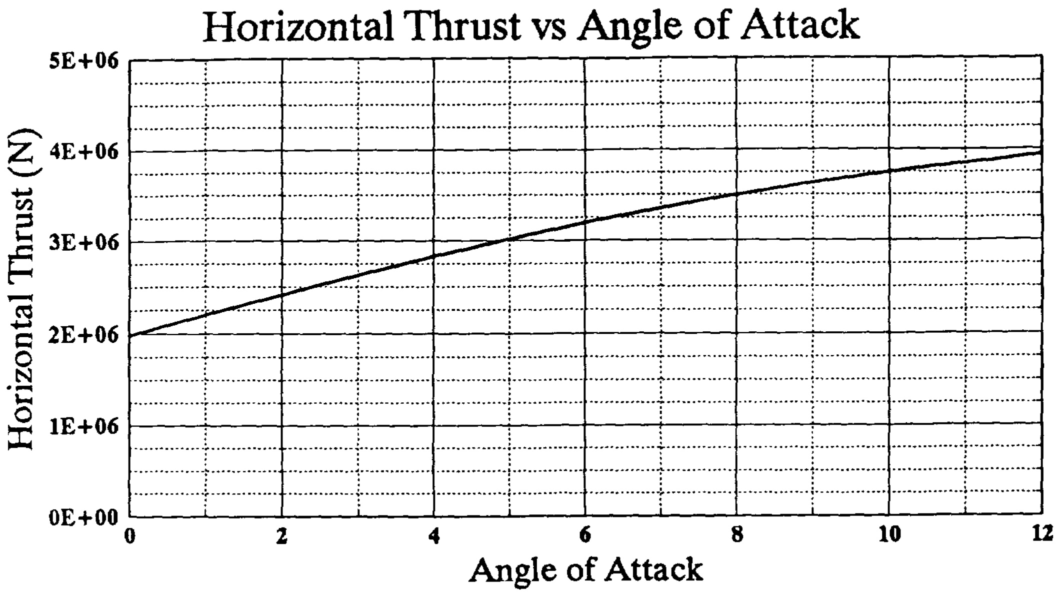


Figure 4.20: Typical Scramjet Behavior (Horizontal Thrust vs Angle Of Attack)

Figures 4.21 through 4.23 illustrate the variation of horizontal thrust, combustor temperature, and specific impulse, with the setting of the throttle ratio (ϕ). An air mass to fuel ratio of *34.32/1.0* is the stoichiometric condition, and the lower ratios correspond to the fuel rich case. Notice that thrust increases for mixtures below the stoichiometric ratio. This is because additional fuel means a higher mass flow and therefore additional thrust. This cannot continue indefinitely, however, because the additional mass causes the temperature in the combustor to drop rapidly (Figure 4.22). At some point, the temperature becomes too low to support combustion. In the model, this effect causes the combustor analysis to fail and valid results cannot be obtained below a certain mixture ratio ($\sim \phi < 8$). Also note that (in Figure 4.23) the specific impulse begins to drop off sharply as the excess fuel is wasted ($\phi <$ *34.32*).

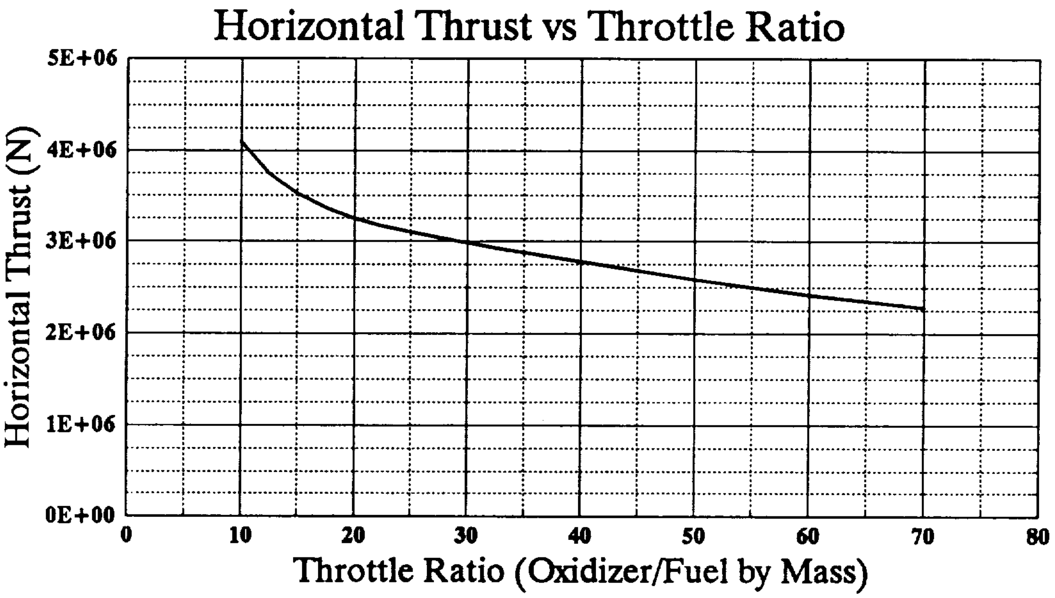


Figure 4.21: Typical Scramjet Behavior (Horizontal Thrust vs Throttle Ratio)

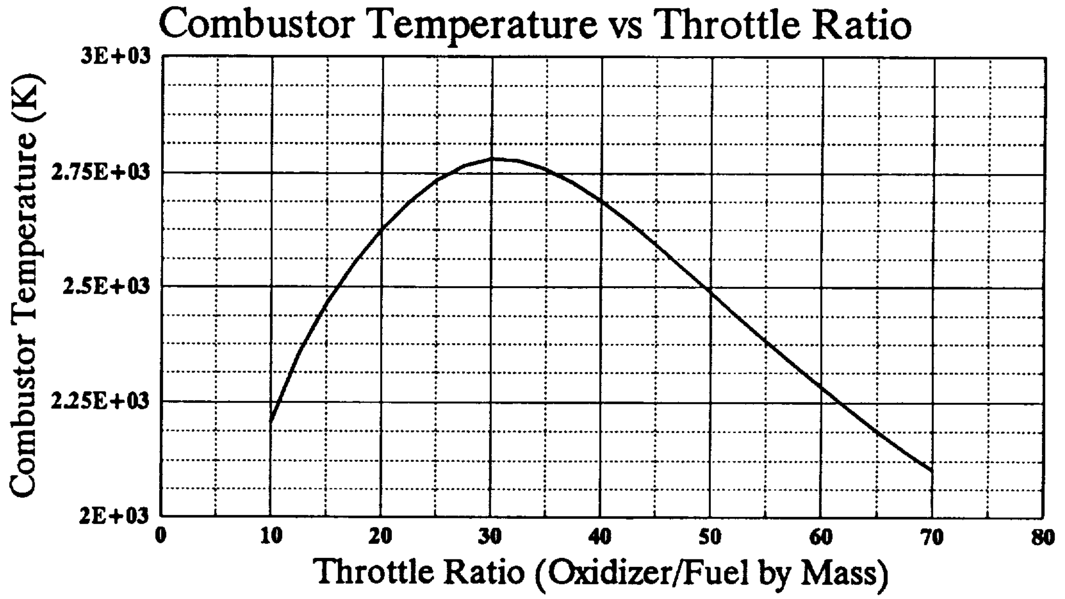


Figure 4.22: Typical Scramjet Behavior (Combustor Temp. vs Throttle Ratio)

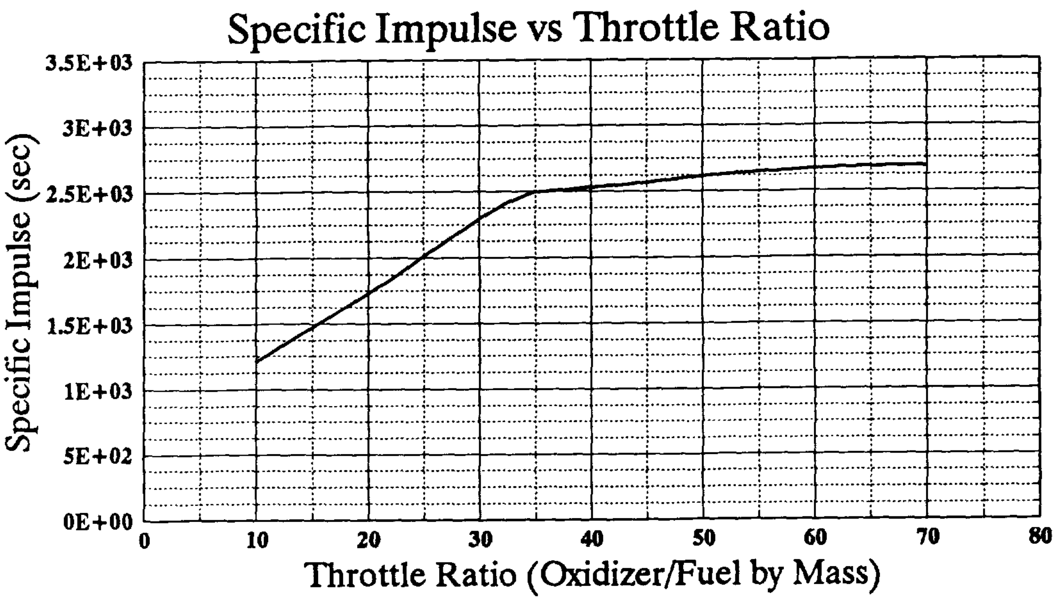


Figure 4.23: Typical Scramjet Behavior (Specific Impulse vs Throttle Ratio)

As a final example, the propulsive pitching moment is shown as a function of the nozzle lip length in Figure 4.24. From this illustration it is clear that the nozzle lip can make an effective thrust vector control actuator.

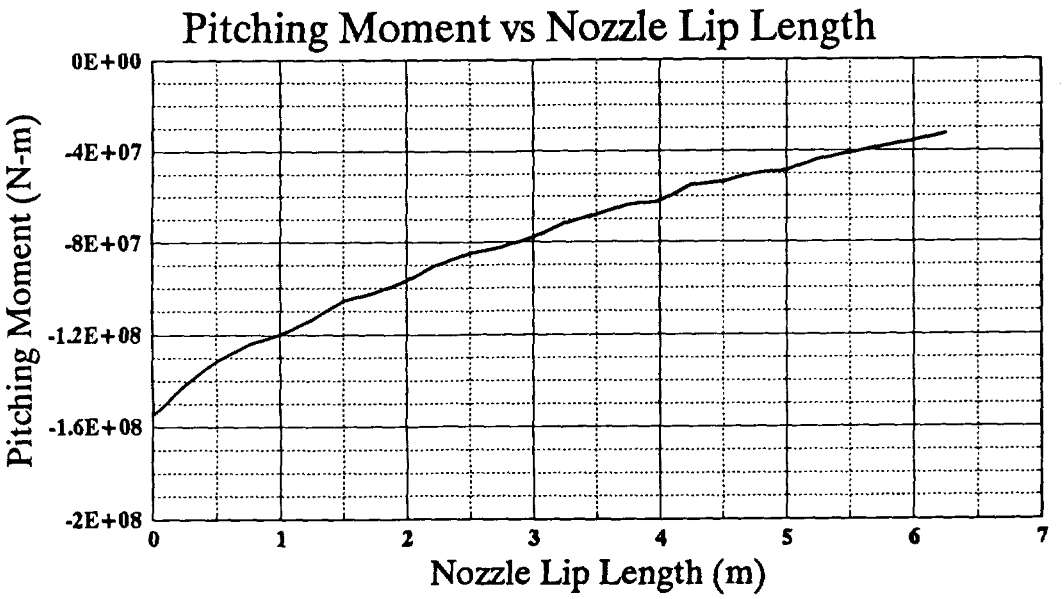


Figure 4.24: Thrust Vector Control Using Nozzle Lip Length

4.6 Flight Dynamics Model

The geometric, atmospheric, aerodynamic, and propulsion models presented in this chapter provide the basis for calculating the overall forces and moments acting on a hypersonic vehicle for a wide range of design configurations and flight conditions. This section develops the dynamics model which (in conjuction with the mass properties model) determines the vehicle's translational and rotational response to these forces and moments.

This model includes the full six–degree–of–freedom equations of motion for an arbitrary direction and location in the atmosphere of a rotating spherical Earth. All centripetal and Coriolis effects are included, and are significant at hypersonic speeds. In addition, the gravitational model accounts for variation with altitude. A stationary atmosphere is assumed, and only rigid body motions are considered.[20]

Beginning with the fundamental relationship for the rate of change of a vector ($\underline{\lambda}$) in a rotating reference frame, [21]

$$\left[\frac{d\lambda}{dt}\right]_I = \left[\frac{d\lambda}{dt}\right]_R + \underline{\omega} \times \underline{\lambda} \qquad (4.82)$$

the translational and rotational dynamic equations are derived by substituting the velocity vector ($\underline{V}$) and angular momentum vector ($\underline{L}$) in place of $\underline{\lambda}$.

$$\left[\frac{dV}{dt}\right]_I = \left[\frac{dV}{dt}\right]_R + \underline{\omega} \times \underline{V} \qquad (4.83a)$$

$$\left[\frac{dL}{dt}\right]_I = \left[\frac{dL}{dt}\right]_R + \underline{\omega} \times \underline{L} \qquad (4.83b)$$

Since $\underline{V} = d\underline{r}/dt$, and $(d\underline{r}/dt)_I = (d\underline{r}/dt)_R + \underline{\omega} \times \underline{r}$ from (4.82), equation (4.83a) can

[20] Note that the AHSV modelling and simulation are complete 6-DOF representations of the vehicle. The RIFC controller developed in Chapters 7 and 8, however, has been limited to longitudinal dynamics in order to simplify the problem.

[21] I refers to an inertial frame and R to a rotating frame.

be written as,

$$\left[\frac{d\underline{V}}{dt}\right]_I = \left[\frac{d}{dt}\left[\left[\frac{d\underline{r}}{dt}\right]_R + \underline{\omega} \times \underline{r}\right]\right]_R + \underline{\omega} \times \left[\left[\frac{d\underline{r}}{dt}\right]_R + \underline{\omega} \times \underline{r}\right] \tag{4.84}$$

or,

$$\left[\frac{d\underline{V}}{dt}\right]_I = \left[\frac{d\underline{V}}{dt}\right]_R + 2\,\underline{\omega} \times \underline{V} + \dot{\underline{\omega}} \times \underline{r} + \underline{\omega} \times (\underline{\omega} \times \underline{r}) \tag{4.85}$$

Now, for a vehicle mass of M and an external force given by the vector $\underline{F}$, the inertial acceleration is determined from $\underline{F} = M\,(d\underline{V}/dt)_{\mathrm{I}}$. Substituting in for $\underline{F}$ and noting that the angular acceleration of the rotating frame is zero ($\dot{\underline{\omega}} = 0$), since it is the rotation of the Earth that is of interest ($\underline{\omega}_{\mathrm{E}}$), equation (4.85) becomes,

$$M\left[\frac{d\underline{V}}{dt}\right]_R = \underline{F} - 2M\,\underline{\omega}_{\mathrm{E}} \times \underline{V} - M\,\underline{\omega}_{\mathrm{E}} \times (\underline{\omega}_{\mathrm{E}} \times \underline{r}) \tag{4.86}$$

Equation (4.86) is a general expression for translational dynamics of a body in a reference frame rotating with the Earth. The AHSV dynamic equations are derived by expressing the vectors $\underline{F}$ and $\underline{V}$ in terms of the flight variables, and writing out the entire equation in the rotating coordinate system.

The rotational dynamics are obtained from (4.83b) using the fact that the moment $\underline{M} = (d\underline{L}/dt)_{\mathrm{I}}$. Also, since $\underline{L} = \underline{I}\,\underline{\omega}$ (4.83b) becomes,

$$\underline{M} = \underline{I}\,\dot{\underline{\omega}} + \underline{\omega} \times \underline{I}\,\underline{\omega} \tag{4.87}$$

where,

$$\underline{I} = \begin{bmatrix} I_{\mathrm{x}} & 0 & I_{\mathrm{xz}} \\ 0 & I_{\mathrm{y}} & 0 \\ I_{\mathrm{xz}} & 0 & I_{\mathrm{z}} \end{bmatrix} \tag{4.88}$$

is the inertia matrix. In this case, the vector ($\underline{\omega}$) refers to the angular rate of the body axes with respect to an inertial coordinate system. Substituting $\underline{\omega} = [\omega_{\mathrm{x}}\ \omega_{\mathrm{y}}\ \omega_{\mathrm{z}}]^{\mathrm{T}}$ and $\underline{M} = [M_{\mathrm{x}}\ M_{\mathrm{y}}\ M_{\mathrm{z}}]^{\mathrm{T}}$ gives the angular accelerations which result from an applied external moment,

$$\dot{\omega}_{\mathrm{x}} = \frac{A\ I_{\mathrm{z}} - B\ I_{\mathrm{xz}}}{I_{\mathrm{x}}\ I_{\mathrm{z}} - I_{\mathrm{xz}}{}^2} \tag{4.89a}$$

$$\dot{\omega}_{\mathrm{y}} = \frac{1}{I_{\mathrm{y}}}\left[M_{\mathrm{y}} - I_{\mathrm{xz}}(\omega_{\mathrm{z}}{}^2 - \omega_{\mathrm{x}}{}^2) - (I_{\mathrm{x}} - I_{\mathrm{z}})\omega_{\mathrm{x}}\omega_{\mathrm{z}}\right] \tag{4.89b}$$

$$\dot{\omega}_z = \frac{B\, I_x \;-\; A\, I_{xz}}{I_x\, I_z \;-\; I_{xz}^2} \tag{4.89c}$$

where,

$$A = M_x - I_{xz}\omega_x\omega_y - (I_z - I_y)\omega_y\omega_z \tag{4.90a}$$

$$B = M_z - I_{xz}\omega_y\omega_z - (I_y - I_x)\omega_x\omega_y \tag{4.90b}$$

Returning now to the translational equations, the expansion of equation (4.86) requires the definition of several coordinate systems. Figure 4.25 illustrates an inertial coordinate frame with the z–axis pointing toward the North pole. The local–horizontal–local–vertical frame (LVLH) is also shown at an arbitrary longitude, latitude, and radius from the origin (r). The relationship between the wind axes coordinate system, the body frame, and the LVLH frame is shown in Figure 4.26. The following definitions are used:

$\phi_l \equiv$ *Latitude* $\qquad \theta_l \equiv$ *Longitude*

$r \equiv$ *Radius to Earth center* $\qquad \gamma \equiv$ *Flight path angle*

$\psi \equiv$ *Heading angle* $\qquad \alpha \equiv$ *Angle of Attack*

$\sigma \equiv$ *Roll (Bank) angle* $\qquad \beta \equiv$ *Yaw Angle*

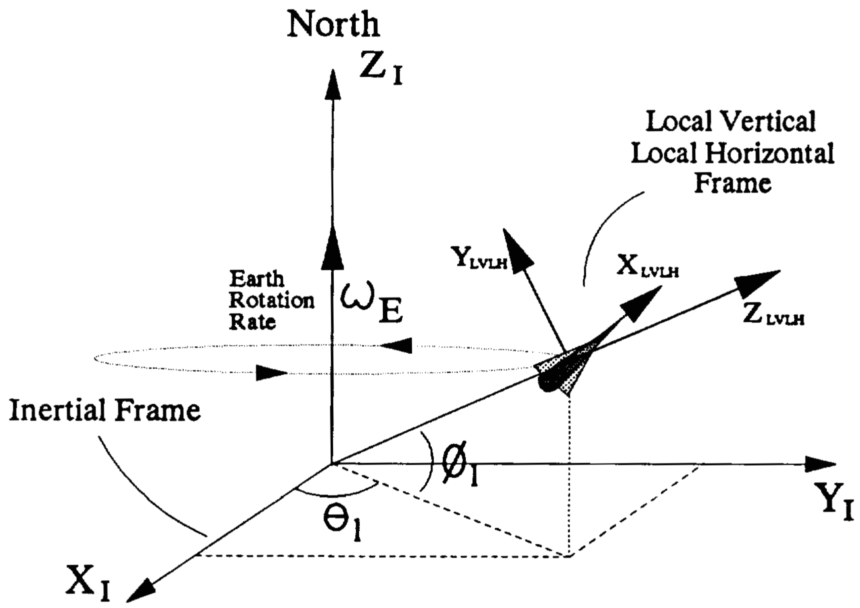


Figure 4.25: Inertial Coordinates and the Local–Vertical–Local–Horizontal Frame

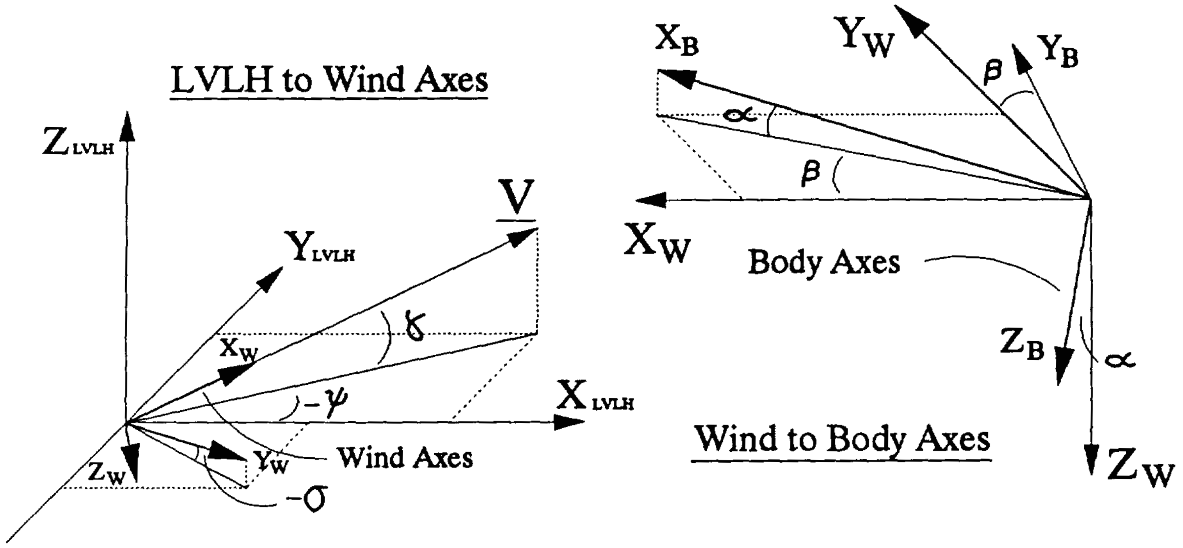


Figure 4.26: The Relationship Between Wind, Body, and LVLH Coordinates

From these figures the following vectors can be defined in the LVLH coordinate frame:

$$\underline{r} = r\,\hat{k} \tag{4.91a}$$

$$\underline{V} = V\cos(\gamma)\cos(\psi)\,\hat{i} + V\cos(\gamma)\sin(\psi)\,\hat{j} + V\sin(\gamma)\,\hat{k} \tag{4.91b}$$

$$\underline{\omega} = \omega_E\cos(\phi_1)\,\hat{j} + \omega_E\sin(\phi_1)\,\hat{k} \tag{4.91c}$$

where ω_E is the angular rate of the Earth. Before substituting into (4.86), the external (propulsive and aerodynamic) forces are decomposed into a tangential component (along the velocity vector) and a normal component (perpendicular to the velocity vector). Bypassing the details,

$$\underline{F}_T = F_T\cos(\gamma)\cos(\psi)\,\hat{i} + F_T\cos(\gamma)\sin(\psi)\,\hat{j} + F_T\sin(\gamma)\,\hat{k} \tag{4.92a}$$

$$\underline{F}_N = -F_N\,[\cos(\sigma)\sin(\gamma)\cos(\psi) + \sin(\sigma)\sin(\psi)]\,\hat{i} - F_N\,[\cos(\sigma)\sin(\gamma)\sin(\psi) - \sin(\sigma)\cos(\gamma)]\,\hat{j} + F_N\cos(\sigma)\cos(\gamma)\,\hat{k} \tag{4.92b}$$

and the gravitational term is expressed as $\underline{F}_g = -\,M\,g(r)\,\hat{k}$.

Substituting the above equations into (4.86), expressing $d\underline{V}/dt$ in the LVLH frame, and performing a substantial amount of algebra (see [Bus1] or [Etk1]) gives the translational dynamic equations of motion:

$$\frac{dV}{dt} = \frac{1}{M} F_T - g(r)\,sin(\gamma) + \omega_E^2 r\,cos(\phi_l)[sin(\gamma)cos(\phi_l) - cos(\gamma)sin(\phi_l)sin(\psi)] \tag{4.93a}$$

$$V\frac{d\gamma}{dt} = \frac{1}{M} F_N cos(\sigma) - g(r)\,cos(\gamma) + \frac{V^2}{r}cos(\gamma) + 2\omega_E V\,cos(\phi_l)cos(\psi) + \omega_E^2 r\,cos(\phi_l)[cos(\gamma)cos(\phi_l) + sin(\gamma)sin(\phi_l)sin(\psi)] \tag{4.93b}$$

$$V\frac{d\psi}{dt} = \frac{1}{M}\frac{F_N sin(\sigma)}{cos(\gamma)} - \frac{V^2}{r}cos(\gamma)cos(\psi)tan(\phi_l) + 2\omega_E V\,[tan(\gamma)cos(\phi_l)sin(\psi) - sin(\phi_l)] - \omega_E^2 r\,sin(\phi_l)cos(\phi_l)cos(\psi)/cos(\gamma) \tag{4.93c}$$

where,

$$g(r) \equiv \frac{GM_E}{r^2},\quad F_T = T_V - D,\ \ and\quad F_N = T_N + L$$

$$L \equiv Lift,\ D \equiv Drag,\ T_V \equiv Thrust\ along\ \underline{V},\ T_N \equiv Thrust \perp to\ \underline{V}$$

Also, by expressing $d\underline{r}/dt$ in the LVLH frame and comparing to (4.91b), the kinematic relationships for global position are found to be,

Radius from Earth center: $$\frac{dr}{dt} = V\,sin(\gamma) \tag{4.94a}$$

Longitude from launch: $$\frac{d\theta_l}{dt} = \frac{V\ cos(\gamma)cos(\psi)}{r\ cos(\phi_l)} \tag{4.94b}$$

Latitude: $$\frac{d\phi_l}{dt} = \frac{V\ cos(\gamma)sin(\psi)}{r} \tag{4.94c}$$

Translational velocities and positions are obtained in the AHSV simulation by simple numerical integration of equations (4.93a–c) and (4.94a–c). Standard Euler integration and an adaptive step–size Runge Kutta algorithm are both available. The angular velocities of the vehicle can likewise be obtained by integrating equations (4.89a–c). Attitude, however, cannot be determined in this way since it is not a directly integrable quantity. A quaternion integration approach has been chosen for this purpose.

Quaternions (or Euler parameters) have the advantage of only depending on four parameters that require integration (whereas cosine matrices require *9* integrations). Numerically, quaternions are very efficient, and a constraining relationship between the parameters provides a convenient way to enforce a consistent result. In order to obtain the attitude variables, perform aerodynamic calculations, and simulate the vehicle in flight (including display), cosine matrices between the different coordinate systems are used. The elements of these transformation matrices are calculated directly from the quaternion parameters. See reference [Hugl] for the quaternion integration equations and the relationships between quaternions and cosine matrices.

4.7 Flight Controller's Model

Since the scope of this research is limited to longitudinal flight control, the controller's model of the dynamics can be obtained from the equations in the preceding section for the case of equatorial launch to orbit with all lateral variables set to zero ($\phi_l = \sigma = \psi = \omega_x = \omega_z = 0$). Also, expressing the forces along the velocity vector[22] as ΣF_{xw}, the normal forces as ΣF_{zw}, and the total pitching moment as ΣM_y, equations (4.89), (4.93), and (4.94) reduce to,

$$\frac{dV}{dt} = \frac{1}{M}(\Sigma F_{xw}) - g(r)\sin(\gamma) + \omega_E^2 r \sin(\gamma) \tag{4.95a}$$

$$V\frac{d\gamma}{dt} = -\frac{1}{M}(\Sigma F_{zw}) - g(r)\cos(\gamma) + \frac{V^2}{r}\cos(\gamma) + 2\omega_E V + \omega_E^2 r \cos(\gamma) \tag{4.95b}$$

$$\frac{d\omega_y}{dt} = \frac{1}{I_y}(\Sigma M_y) \tag{4.95c}$$

$$\frac{dr}{dt} = V\sin(\gamma) \tag{4.95d}$$

$$\frac{d\theta_l}{dt} = V\cos(\gamma)/r \tag{4.95e}$$

For the controller, the forces and moments (ΣF_{xw}, ΣF_{zw}, ΣM_y) are not obtained using the other models in this chapter. The simulation uses the models of sections

[22] Which is also along $\hat{x}_w$, the unit wind coordinates x-axis·

4.1 through 4.6 to generate the "truth" environment. The control system is assumed to have its own model based on a database of aerodynamic, propulsion, and control coefficients as a function of flight condition. This database is comprised of a *priori* and on–line information, and it provides a best estimate of the vehicle characteristics given limited knowledge. An uncertainty bound for each coefficient is also assumed to be given by this model. Each of the coefficients described below is nondimensionalized by the dynamic pressure, the wing reference area, and the root mean chord length in the case of moment coefficients.

$C_{\mathrm{L}}(Ma,\alpha)$ ≡ *Aerodynamic lift coefficient*

$C_{\mathrm{D}}(Ma,\alpha)$ ≡ *Aerodynamic drag coefficient*

$C_{\mathrm{Ma}}(Ma,\alpha)$ ≡ *Aerodynamic pitch moment coefficient*

$C_{\mathrm{Tv}}(Ma,\alpha)$ ≡ *Thrust coefficient along velocity vector*

$C_{\mathrm{Tn}}(Ma,\alpha)$ ≡ *Thrust coefficient normal to velocity vector*

$C_{\mathrm{Mp}}(Ma,\alpha)$ ≡ *Thrust pitch moment coefficient*

$C_{\mathrm{X}_{\delta e}}(Ma,\alpha,\delta e)$ ≡ *Elevon control force coefficient in* $\hat{\underline{x}}_{\mathrm{w}}$ *direction*

$C_{\mathrm{Z}_{\delta e}}(Ma,\alpha,\delta e)$ ≡ *Elevon control force coefficient in* $\hat{\underline{z}}_{\mathrm{w}}$ *direction*

$C_{\mathrm{M}_{\delta e}}(Ma,\alpha,\delta e)$ ≡ *Elevon pitch moment coefficient in* $\hat{\underline{y}}_{\mathrm{w}}$ *direction*

$C_{\mathrm{X}_{\delta_{\mathrm{T}}}}(Ma,\alpha,\delta_{\mathrm{T}})$ ≡ *Throttle control force coefficient in* $\hat{\underline{x}}_{\mathrm{w}}$ *direction*

$C_{\mathrm{Z}_{\delta_{\mathrm{T}}}}(Ma,\alpha,\delta_{\mathrm{T}})$ ≡ *Throttle control force coefficient in* $\hat{\underline{z}}_{\mathrm{w}}$ *direction*

$C_{\mathrm{M}_{\delta_{\mathrm{T}}}}(Ma,\alpha,\delta_{\mathrm{T}})$ ≡ *Throttle pitch moment coefficient in* $\hat{\underline{y}}_{\mathrm{w}}$ *direction*

$C_{\mathrm{X}_{\delta v}}(Ma,\alpha,\delta v)$ ≡ *Thrust vector force coefficient in* $\hat{\underline{x}}_{\mathrm{w}}$ *direction*

$C_{\mathrm{Z}_{\delta v}}(Ma,\alpha,\delta v)$ ≡ *Thrust vector force coefficient in* $\hat{\underline{z}}_{\mathrm{w}}$ *direction*

$C_{\mathrm{M}_{\delta v}}(Ma,\alpha,\delta v)$ ≡ *Thrust vector pitch moment coefficient in* $\hat{\underline{y}}_{\mathrm{w}}$ *direction*

Note that each coefficient is a function of flight Mach number (Ma) and angle of attack (α). The control coefficients are also functions of their respective input values (δe, δ_{T}, δv). The controller's model of the dynamics is now expressed in terms of the coefficients above.

$$
\begin{aligned}
\dot{V}(t) &= - \left[\frac{GM_{\mathrm{E}}}{r^2} - \omega_{\mathrm{E}}^2 r\right] sin(\gamma) + (\tfrac{1}{2}\rho V^2)\frac{S_{\mathrm{ref}}}{M(t)}\Big[C_{\mathrm{Tv}}(Ma,\alpha) - C_{\mathrm{D}}(Ma,\alpha)\Big] \\
&\quad + (\tfrac{1}{2}\rho V^2)\frac{S_{\mathrm{ref}}}{M(t)}\Big[C_{\mathrm{X}_{\delta e}}(Ma,\alpha,\delta e) + C_{\mathrm{X}_{\delta_{\mathrm{T}}}}(Ma,\alpha,\delta_{\mathrm{T}}) \\
&\quad + C_{\mathrm{X}_{\delta v}}(Ma,\alpha,\delta v)\cdot[C_{\mathrm{Tv}}(Ma,\alpha) + C_{\mathrm{X}_{\delta_{\mathrm{T}}}}(Ma,\alpha,\delta_{\mathrm{T}})]\ \Big]
\end{aligned}
\tag{4.96a}
$$

$$
\begin{aligned}
\dot{\gamma}(t) &= - \frac{1}{V}\left[\frac{GM_{\mathrm{E}}}{r^2} - \omega_{\mathrm{E}}^2 r\right] cos(\gamma) + \left[\frac{V}{r}\right] cos(\gamma) + 2\omega_{\mathrm{E}} \\
&\quad + (\tfrac{1}{2}\rho V^2)\left[\frac{1}{V}\right]\frac{S_{\mathrm{ref}}}{M(t)}\Big[C_{\mathrm{Tn}}(Ma,\alpha) + C_{\mathrm{L}}(Ma,\alpha)\Big] \\
&\quad - (\tfrac{1}{2}\rho V^2)\left[\frac{1}{V}\right]\frac{S_{\mathrm{ref}}}{M(t)}\Big[C_{\mathrm{Z}_{\delta e}}(Ma,\alpha,\delta e) + C_{\mathrm{Z}_{\delta_{\mathrm{T}}}}(Ma,\alpha,\delta_{\mathrm{T}}) \\
&\quad + C_{\mathrm{Z}_{\delta v}}(Ma,\alpha,\delta v)\cdot[C_{\mathrm{Z}_{\delta_{\mathrm{T}}}}(Ma,\alpha,\delta_{\mathrm{T}}) - C_{\mathrm{Tn}}(Ma,\alpha)]\ \Big]
\end{aligned}
\tag{4.96b}
$$

$$
\begin{aligned}
\dot{\omega}(t) &= + (\tfrac{1}{2}\rho V^2)\frac{S_{\mathrm{ref}}\ \bar{c}}{I_{\mathrm{y}}(t)}\Big[C_{\mathrm{Ma}}(Ma,\alpha) + C_{\mathrm{Mp}}(Ma,\alpha)\Big] \\
&\quad + (\tfrac{1}{2}\rho V^2)\frac{S_{\mathrm{ref}}\ \bar{c}}{I_{\mathrm{y}}(t)}\Big[C_{\mathrm{M}_{\delta e}}(Ma,\alpha,\delta e) + C_{\mathrm{M}_{\delta_{\mathrm{T}}}}(Ma,\alpha,\delta_{\mathrm{T}}) \\
&\quad + C_{\mathrm{M}_{\delta v}}(Ma,\alpha,\delta v)\cdot[C_{\mathrm{Mp}}(Ma,\alpha) + C_{\mathrm{M}_{\delta_{\mathrm{T}}}}(Ma,\alpha,\delta_{\mathrm{T}})]\ \Big]
\end{aligned}
\tag{4.96c}
$$

$$\dot{\theta}(t) = \omega(t) \tag{4.96d}$$

with the auxiliary equations,

$$\dot{r}(t) = V\,sin(\gamma) \tag{4.96e}$$

$$\alpha(t) = \theta(t) - \gamma(t) \tag{4.96f}$$

$$Ma(t) = V/\sqrt{\gamma_{\mathrm{a}} R T} \tag{4.96g}$$

These are the final equations used by the controller to predict the vehicle response to a candidate set of inputs. Note that the equation $\dot{\theta}(t) = \omega(t)$ is also included as a formal state equation. The longitude equation (4.95e) is omitted only because the performance objective considered later in Chapter 7 does not depend on downrange position. Finally, an important characteristic of equations (4.96a–c) is that the thrust vector coefficient appears (in a nonlinear manner) as a multiplier of other coefficients. These equations will be analyzed in more detail in Chapter 7.

Chapter 5

Flight Simulation

5.1 Simulation Objectives

In order to facilitate the evaluation of flight control strategies for hypersonic vehicles an extensive software package was developed to provide a realistic flight simulation capability. A tool was desired with which vehicle designs could be evaluated, modified, and tested under various flight conditions. It was designed with a modular structure so that different vehicles, models, conditions, disturbances, controllers, and many other options could be selected and combined as needed for a particular test run. The ability to access simulation data in various forms was also important. An on–line plotting capability, full–state numeric display, and graphical vehicle representations were therefore included. The final AHSV simulation software is a fully integrated menu–driver design and analysis package which includes all of the hypersonic vehicle modelling effort presented in Chapter 4, as well as the flight control algorithm and logic developed in Chapters 7 and 8. In addition, it provides capabilities such as automated parametric vehicle design, stability coefficient determination, local linearization and linear simulation, standard Euler and adaptive stepsize Runge Kutta integration, programmable atmospheric and wind disturbances, scramjet engine analysis and design, various display and warning modes, a trajectory store and replay option, and more. These options and the other capabilities described in this chapter form the basis of a realistic test–bed for hypersonic flight control algorithms such as the RIFC autopilot developed in this thesis.

5.2 Simulation Structure

At its highest level, the AHSV simulation code can be described by the simple flow diagram shown in Figure 5.1. A **Set Defaults** block first initializes a number of settings, and then an **Options Menu** is displayed from which all capabilities of the software can be accessed. These options are used to either **Design** the vehicle, Configure the simulation, or actually **Run** a simulated flight. At the beginning of each simulation an **Initialization** procedure is invoked to reset all initial conditions, construct the vehicle geometry, and compute the mass properties. Once initialized, the simulation then cycles through a primary loop which executes the following general routines: **Integration**, **Environment**, **Models**, **Dynamics**, **Control**, **Estimation**, Display, and **Interrupts**. The **Integration** module advances the simulation by one time–step using the selected integration scheme. The **Environment**, **Models**, and **Dynamics** modules are called by the integrator to determine the rates of change for all variables at any time. All blocks call the proper subroutines depending upon the specific configuration.[1] If the simulation is in **Playback** mode then several routines are bypassed and the flight data is read directly from a previously stored file. The main loop is repeated continually as the flight trajectory evolves from a specified initial condition. User interrupts can change a variety of run–time options or end the simulation.

The remainder of this section describes an expanded view of each of the blocks shown in Figure 5.1. The intention is to provide a functional overview of the AHSV simulation as well as a brief explanation of the software capabilities. Toward this end, an outline of the logical structure of the major components for each block is presented. The inputs and outputs are given for important modules, and any relevant options are discussed.

[1] That is, the configuration determines which models are called by the **Models** block, or which control algorithm is called by the **Control** block, etc.

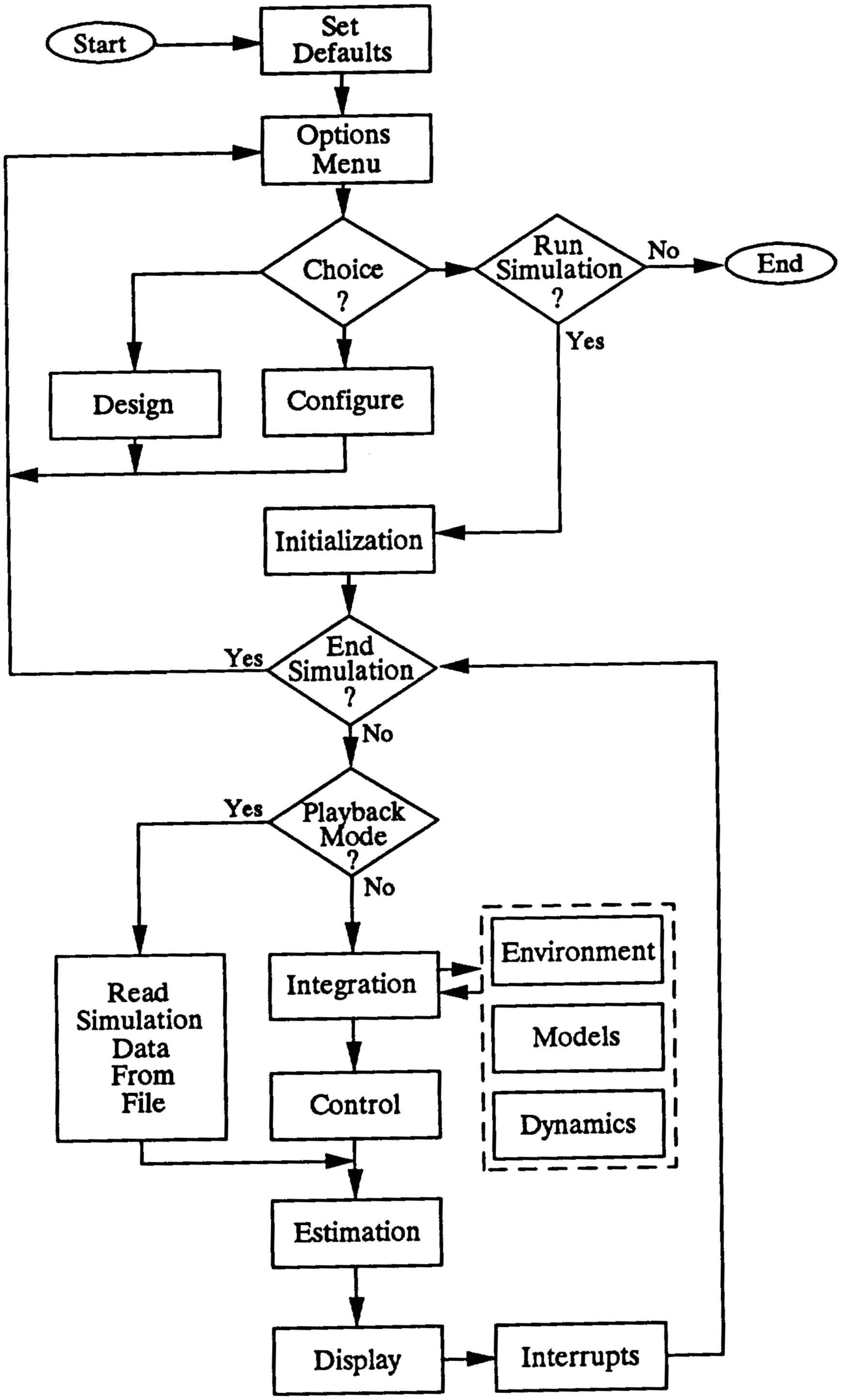


Figure 5.1: High Level AHSV Flight Simulation Flow Diagram

For the purpose of specifying inputs and outputs to various modules, it is convenient to define the following categories of simulation variables:

Attitude:	$\underline{\theta}$	Roll, pitch, yaw attitude, angle of attack, flight path, etc...
Attitude Rates:	$\underline{\omega}$	Roll, pitch, yaw rates.
Position:	$\underline{x}$	Altitude, longitude, latitude.
Velocity:	$\underline{V}$	Velocity, ascent rate, etc...
Flow Variables:	$\underline{\phi}$	Temperature, density, speed of sound, pressure, viscosity, Mach number, dynamic pressure.
Forces:	$\underline{\sigma}$	Aerodynamic, propulsion, and control, forces and moments.
Inputs:	$\underline{u}$	Elevon, throttle, and thrust vector settings.
Geometry:	$\underline{\Gamma}$	Arrays of (x,y,z) coordinates to specify vehicle shape, surface polygons, normal vectors, centroids, and areas. $\underline{\Gamma}_B$ and $\underline{\Gamma}_W$ are the same arrays expressed in the body and wind coordinates respectively.
Mass Properties:	$\underline{\mu}$	Vehicle mass, inertia matrix, fuel flow rate.
Transformations:	$\underline{C}$	Between inertial, LVLH, wind, body, and viewer's coordinates.
Design Parameters:	$\underline{P}$	Aerodynamic and propulsion design values (i.e. wing span, sweep angle, fuel injection temperature, etc...).
Tabulated Data:	$\underline{T}$	Scramjet engine tables or the coefficient database model.

Set Defaults

The first function of the **Set Defaults** block (Figure 5.2) is to load a number of configuration settings from a default file when the simulation is started. This file is used to store previous configurations so that commonly selected options do not have to be reselected every time. Defaults include all simulation modes as well as options for integration, modelling, control, and display. Other flags, display defaults, and the disturbance settings are also intialized at start–up. In addition, a set of vehicle design parameters are loaded from a default design file. Depending on the start–up configuration other files may be loaded as well; an example would be the currently active scramjet data file.

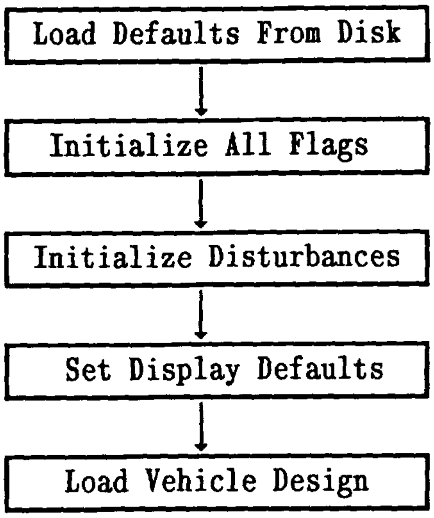


Figure 5.2: Set Defaults Block

Options Menu

This block (Figure 5.3) provides an extensive pulldown–menu interface which is used to access vehicle design and analysis tools, configure all simulation options, and begin an actual simulation. The menu options are divided into the following categories: **Display**, **Simulation**, **Control**, **Estimation**, **Design**, **Models**, **Conditions**, **Inputs**, and **Defaults**. Each of these categories includes a number of options that either set flags, select modes, load/save models, choose algorithms, specify conditions, perform specialized tasks, or present additional suboptions. Some options lead to entirely new pulldown menus, such as the **Scramjet Analysis** or **Vehicle Design** options. All selections ultimately return to the main menu, including the option which runs the simulation. In that case, however, the menu will not appear until the simulation is complete or terminated by user request.

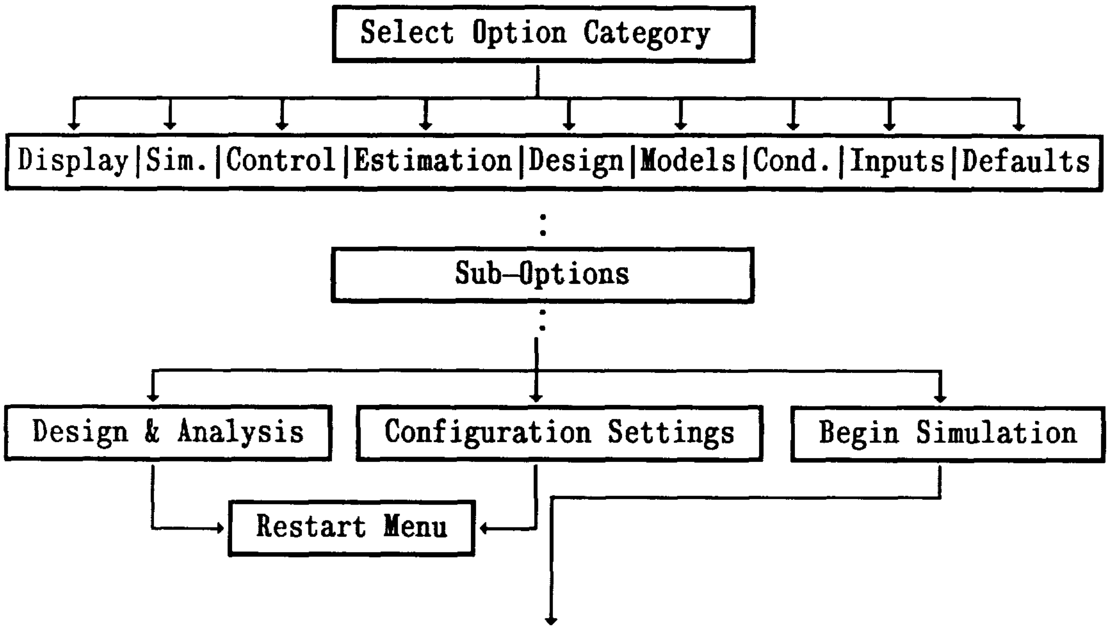


Figure 5.3: Options Menu Block

Design

This block (Figure 5.4) includes all of the functions used to produce viable AHSV designs as well as tabulated models used by the simulation at run time. The two major design functions are **Vehicle Design** and **Scramjet Analysis**, each of which presents an entirely new set of options. In **Vehicle Design** the vehicle geometry, flight conditions, and scramjet engine design parameters can all be changed to create new designs. Various designs can be stored, retrieved, and modified by adjusting any of the design parameters discussed in Chapter 4 (i.e. wing span, sweep angle, combustor length, etc.). A **Force Balance** option evaluates each design using all of the models to determine if the vehicle is in equlbrium for a particular flight condition. An **Automatic Parameter Search** option will search for a balanced design automatically through iterative adjustments of a selected set of parameters. This design process is limited to optimizing the vehicle for one particular flight condition. The off–design operating range will depend upon the available control authority to trim the vehicle at other Mach numbers and angles of attack.

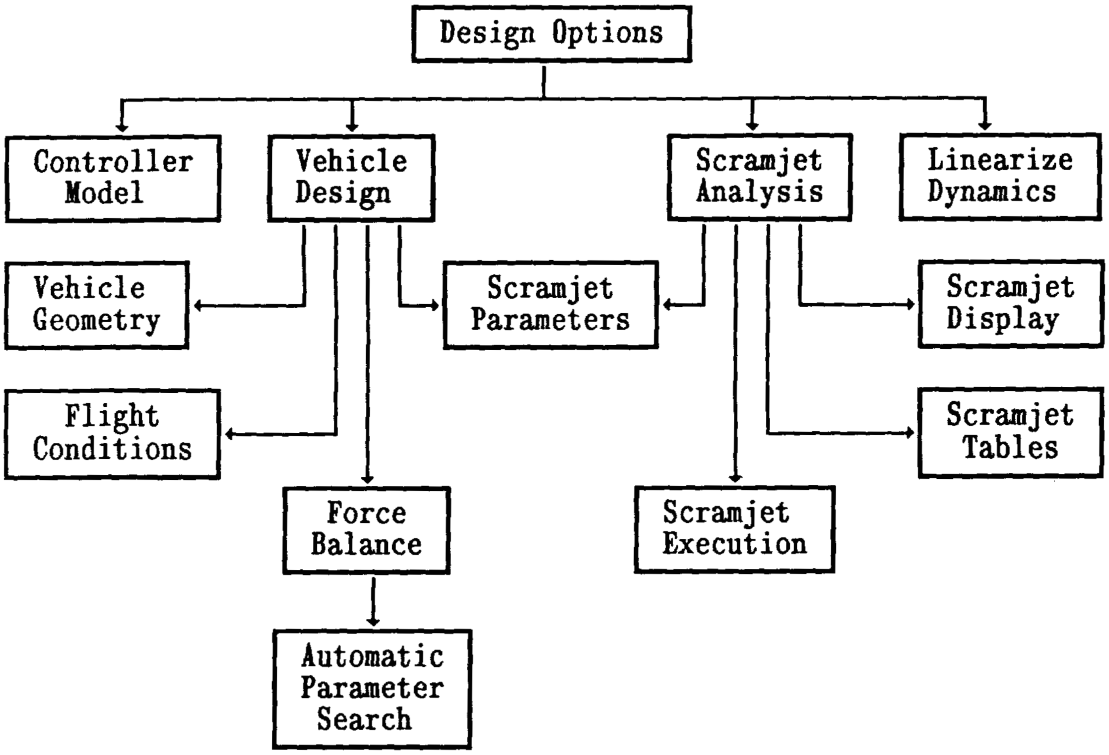


Figure 5.4: Design Block

The **Scramjet Analysis** software provides the interface between the AHSV simulation code and the scramjet propulsion model. Available options are divided into the following categories: **Scramjet Parameters**, **Scramjet Display**, **Scramjet Execution**, and **Scramjet Tables**. The **Scramjet Parameters** options include initializing design or off–design conditions, and editing all scramjet engine inputs (21 variables – see Chapter 4). **Scramjet Display** options are used to select dependent and independent plotting variables, generate plots, or view engine model outputs. Possible outputs include thrust, moment, specific impulse, fuel flow rate, combustor temperature, etc. Possible independent variables are Mach number, angle of attack, throttle setting, etc. The **Scramjet Execution** options perform the actual calls to the scramjet analysis code (or tabulated models) to generate engine outputs for one or a series of input conditions. Finally, the **Scramjet Tables** options are used to load tabulated scramjet data files, and to construct tables for later use with the simulation. Tables are used because the scramjet analysis code is too complex (i.e. too slow) to include in the actual simulation. Options for a multivariable linear or cubic spline interpolation scheme are also included. Both algorithms have been implemented to provide smooth extrapolation in any or all of five dimensions.[2]

Another **Design** option, called **Controller Model**, is used to build up a database of coefficients by sampling the results of all the models in the AHSV simulation. This database, combined with the nonlinear dynamic equations (4.96a–g), becomes the controller's model of the system. Since the controller should not realistically have a perfect model, this data is then corrupted with noise (the bias and distribution of which is programmable). The resulting database of coefficients is stored along with associated interval uncertainties. The table consists of aerodynamic, propulsion, and control coefficients for various Mach numbers, angles of attack, and control input settings.

The last **Design** function is called **Linearize Dynamics** and provides the capability to linearize the vehicle dynamics in any flight condition. This is done numerically by perturbing the states and controls and executing all models to determine the resulting rates. That is, from $\dot{\underline{x}}(t) = f(\underline{x}(t),\underline{u}(t))$, the linear system $\dot{\underline{x}}(t) = A\underline{x}(t)+B\underline{u}(t)$ is obtained by numerically computing the following derivatives:

$$A = \frac{\partial f}{\partial \underline{x}}\bigg|_{\underline{x}_0,\underline{u}_0} \quad \text{and} \quad B = \frac{\partial f}{\partial \underline{u}}\bigg|_{\underline{x}_0,\underline{u}_0} \tag{5.1}$$

The matrices (A,B) are then displayed, and the system can be saved to disk. The file format used is compatible with popular design packages (such as MatLab or MatrixX) so that linear analysis and control system design tools can be applied. Other options allow the linear system to be selected as the true AHSV dynamics model for the simulation.

[2] The scramjet tables store the horizontal and vertical components of thrust, thrust moment, and fuel flow rate, as a function of altitude, Mach number, angle of attack, throttle setting, and thrust vector input (which is the nozzle lip length). A warning message is displayed when the scramjet model is extrapolating beyond the range of the tabulated data.

Configure

This block (Figure 5.5) includes all options in the main menu that are used to configure the simulation prior to execution. These options can be divided into the following categories: **Simulation Modes**, **Flight Conditions**, **External Inputs**, and **Simulation Defaults**.

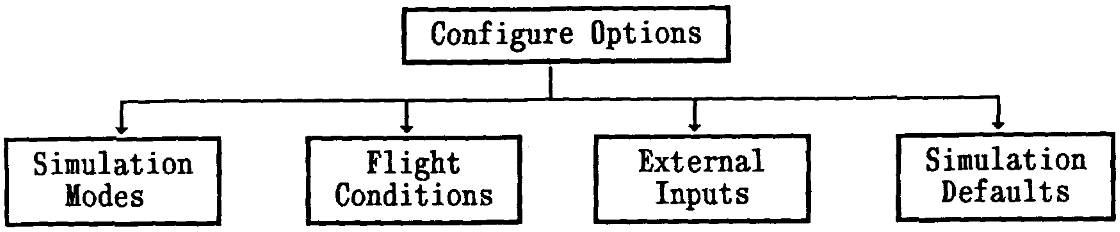


Figure 5.5: Configure Block

There are two categories of **Simulation Modes**: static and dynamic flight. The dynamic modes actually execute all models and integrate the dynamics based on calculated rates for all variables. **Manual Flight Control** and **Automatic Flight Control** are both dynamic simulation modes. In Manual the control actuators can be operated from the keyboard, while in Automatic mode one of the control algorithms is implemented. For the static modes no integration is performed and the vehicle attitude is controlled directly from the keyboard. Rotations can be referenced to the user's "computer" frame, or to the body axes of the vehicle (i.e. pilot's frame). The most useful static mode actually calls all of the models (to determine forces, moments, coefficents, rates, etc.) but then does not integrate the dynamics. This is useful, for example, to examine or plot aerodynamic, propulsion, and control derivatives, as a function of user controlled variations in other variables, such as the states or control inputs.

The **Flight Conditions** option is simply used to set the flight conditions prior to a simulation run. The starting altitude, Mach number, flight path angle, angle of attack, elevon deflection, throttle setting, and thrust vector angle, can all be initialized to any desired values. Initial conditions can be stored and retrieved from files which may correspond to different designs, models, or test cases. It is also possible to specify default flight conditions which can be set or reset at the beginning of each simulation.

The selection of **External Inputs** is further divided into **Noise Generation**, **Command Inputs**, and **Control Dithering**. The available disturbances include lateral variations in atmospheric density and temperature, as well as wind gusts in the form of angle of attack and velocity variations. A ficticious disturbance can also be used to directly affect pitch attitude. All noises are modelled as Gauss–Markov signals (i.e. normally distributed white noise passing through a first order filter). The variance and correlation time constant for each noise can be specified arbitrarily. The **Command Inputs** option is used to specify the desired flight conditions. Constant reference commands can be chosen if regulation about a fixed flight condition is desired. Otherwise a desired trajectory which has been previously tabulated can be loaded from a file. A random trajectory generator can also be selected. The **Control Dithering** option is used in conjunction with the **Recursive Least** Squares algorithm for linear controller models (see the **Estimation** block).

The **Simulation Defaults** category is used to configure many of the run–time simulation options. In addition, several debugging tools are available as defaults. In particular, the **Dump**, **Trace**, and **Debug** modes can be used to dump variables from anywhere in the simulation, display messages for each module as it is executed, and perform other programmable debugging functions. Other defaults include a speed factor to slow down the simulation (used for manual flight), a graphics mode setting, and an option to defeat the run–time error messages generated by all of the models. Additional options to store simulation trajectory data or playback a previous simulation are also available. The default settings can be reset at any time, or stored and retrieved from user specified files.

Initialization

At the beginning of each simulation all vehicle states must be initialized to the proper values. This includes three dimensional position, velocity, attitude, and attitude rates. Control inputs are also set to their nominal values (which can be specified). From the initial state, the coordinate transformation matrices can then be determined. This initialization stage is also where the entire vehicle structure is generated. Based on the design parameters, the **Vehicle Geometry Model** computes the coordinates, panels, normal vectors, centroids, and areas of the exterior vehicle surface in a structural coordinate frame. The **Mass Properties Model** then integrates the volume and surface densities to determine overall vehicle mass, center of mass, and inertia properties.[3] Finally, the geometric vehicle structure ($\underline{\Gamma}$) is translated into body and wind axes ($\underline{\Gamma}_B$, $\underline{\Gamma}_W$) using the center of mass as the origin.

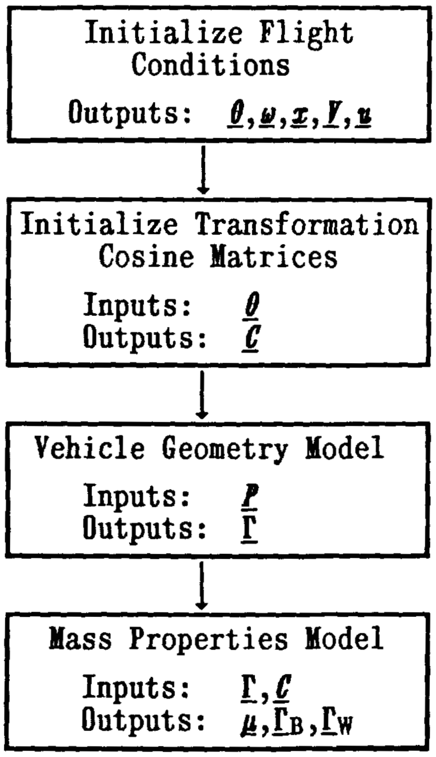
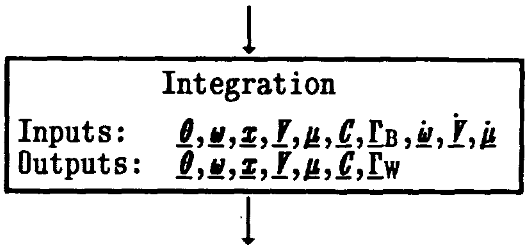

Initialize Flight Conditions
Outputs: $\underline{\theta}, \underline{\omega}, \underline{x}, \underline{V}, \underline{u}$
Initialize Transformation Cosine Matrices
Inputs: $\underline{\theta}$
Outputs: $\underline{C}$
Vehicle Geometry Model
Inputs: $\underline{P}$
Outputs: $\underline{\Gamma}$
Mass Properties Model
Inputs: $\underline{\Gamma}, \underline{C}$
Outputs: $\underline{\mu}, \underline{\Gamma}_B, \underline{\Gamma}_W$

Figure 5.6: Initialization Block

[3] Refer to Chapter 4 for a detailed description of the geometry and mass properties models.

Integration

To simulate the dynamics there is a choice of using either standard **Euler Integration** or an adaptive stepsize 4th order **Runge Kutta Integration**. These options are available regardless of which models are in effect (i.e. the high–order 6–DOF aerodynamic and propulsion models, down to the approximate linear dynamics). Euler integration uses a fixed time–step which can be any desired fraction of the simulation output time–step.[4] The Runge Kutta method attempts to meet a desired accuracy specification between simulation output time–steps. This accuracy is entered as a scalar fraction which is internally multiplied by a vector of nominal values for each state. The algorithm adaptively changes its stepsize to assure that the cumulative errors for each state independently meet the requirements. An option for a minimum Runge Kutta stepsize is also provided to limit the time required for an integration. In this case an error message would indicate a failure to meet the desired precision.


Integration

Inputs: $\underline{\theta},\underline{\omega},\underline{x},\underline{V},\underline{\mu},\underline{C},\underline{\Gamma}_B,\underline{\dot{\omega}},\underline{\dot{V}},\underline{\dot{\mu}}$

Outputs: $\underline{\theta},\underline{\omega},\underline{x},\underline{V},\underline{\mu},\underline{C},\underline{\Gamma}_W$

Figure 5.7: Integration Block

[4] The output time-step is the interval at which simulation data is available for display, storage, plotting, etc. It is also the control cycle-time. That is, the update rate for the control system inputs and outputs.

Environment

The **Environment** block determines the external flow properties ($\underline{\phi}$) as a function of altitude and velocity. Flow temperature, density, speed of sound, pressure, and viscosity are obtained by the **Atmospheric Model** through interpolation of tabulated data (see section 4.3). The flight velocity is required as an input since the model is also used to compute Mach number and dynamic pressure. Disturbances to the ambient flow temperature, density, velocity, and direction are then added by the **Disturbance Model**.

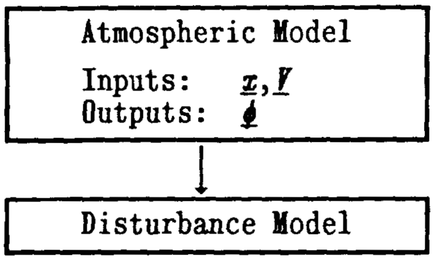

Atmospheric Model
Inputs: $\underline{x}$,$\underline{V}$
Outputs: $\underline{\phi}$
Disturbance Model

Figure 5.8: Environment Block

Models

A choice of several models are available for simulation of the AHSV dynamics (Figure 5.9). There are two classes of models: those which are intended to represent the actual flight behavior of the vehicle, and those which comprise the controller's model. At run–time the "truth model" and the controller's model must be completely isolated from one another. In this way, the simulation reproduces the conditions of an actual flight in which the controller is limited by its approximate model plus any information that can be obtained on–line.

The standard configuration for the AHSV simulation is to use the **Aerodynamics Model** combined with the **Scramjet Propulsion Model** as the true system, while the **Coefficient Database Model** provides the controller with its model of the system. In this configuration the simulation computes forces and moments

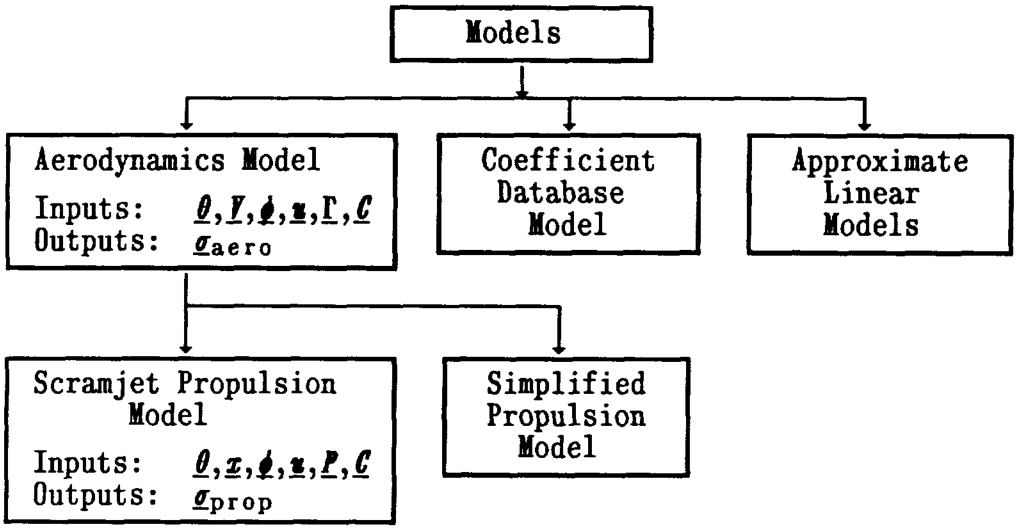

Models
Aerodynamics Model
Inputs: $\underline{\theta},\underline{V},\underline{\phi},\underline{u},\underline{\Gamma},\underline{C}$
Outputs: $\underline{\sigma}_{\mathrm{aero}}$
Coefficient Database Model
Approximate Linear Models
Scramjet Propulsion Model
Inputs: $\underline{\theta},\underline{x},\underline{\phi},\underline{u},\underline{P},\underline{C}$
Outputs: $\underline{\sigma}_{\mathrm{prop}}$
Simplified Propulsion Model

Figure 5.9: Models Block

using the full hypersonic aerodynamics[5] and scramjet analysis codes. The controller bases its decisions on the dynamic equations given by (4.96a–g) and the predictions of aerodynamic, propulsion, and control coefficients stored in the database along with associated uncertainties. Various scramjet tables and coefficient tables can be loaded from stored files, while the aerodynamics code always executes on–line. An option also exists to substitute a simple (*Thrust* = *Drag*) model for the propulsion system. Alternatively, it is possible to configure the simulation to use the controller's model as if it were the truth model. This is a useful feature for testing the control system, since in this case the controller has a perfect model of the dynamics.

Other configurations for the simulation use **Approximate Linear Models** to represent the vehicle dynamics. Some simple second or third order systems can be selected as examples. Moreover, a complete 5–DOF linear system can be used as

[5] i. e. Newtonian flow to obtain pressure coefficients for each panel, shocks and expansion fans to obtain local flow properties, and skin friction coefficients from laminar and tubulent boundary layer calculations for each panel.

the truth model to represent the full longitudinal dynamics. Typically, these models are based on a linearization of the full nonlinear system (see **Design**), and can be stored, retrieved, and modified as desired.

Dynamics

The **Dynamics** block is responsible for determining the derivatives of all the states at any given time. For the full nonlinear simulation this means combining all aerodynamic and propulsion forces and moments to compute linear and angular accelerations for the current mass properties (see section 4.6). The derivatives for the mass properties are also obtained by the **Dynamics Model** based on the fuel flow rate given by the scramjet model. For simulation modes using approximate linear models, or the coefficient database model, the rates are obtained by simply evaluating the right hand side of the dynamic equations (i.e. $A\underline{x}(t)+B\underline{u}(t)$ for linear, or $f(\underline{x}(t),\underline{u}(t))$ for nonlinear dynamics).

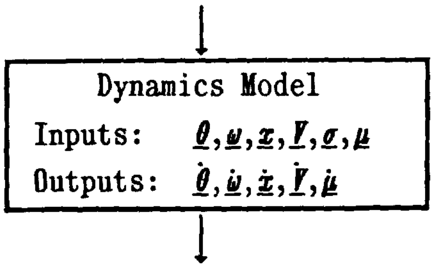

Dynamics Model
Inputs: $\underline{\theta},\underline{\omega},\underline{x},\underline{V},\underline{\sigma},\underline{\mu}$
Outputs: $\underline{\dot{\theta}},\underline{\dot{\omega}},\underline{\dot{x}},\underline{\dot{V}},\underline{\dot{\mu}}$

Figure 5.10: Dynamics Block

Control

This block utilizes all available information to determine what control action should be taken to meet some desired objectives. Other inputs, not shown in Figure 5.11, may include a desired trajectory, dynamic constraints, and actuator

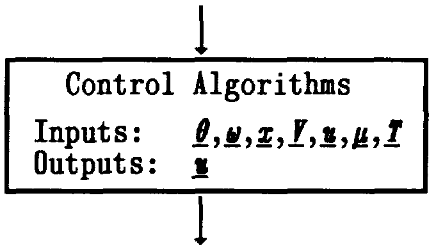

Control Algorithms
Inputs: $\underline{\theta},\underline{\omega},\underline{x},\underline{V},\underline{u},\underline{\mu},\underline{T}$
Outputs: $\underline{u}$

Figure 5.11: Control Block

bounds and rate limits. For the RIFC algorithm, the control inputs are chosen to minimize a worst case integrated tracking error with respect to the uncertainty in the dynamics, while observing all constraints. A variety of other control algorithms have also been implemented and are available as controller options. Among these are a simple PD controller, a discrete adaptive controller, an LQR approach, and the single step optimal controller (SSOC) which is compared to the RIFC algorithm in Chapter 9.

Estimation

Since the objective of this research is focused on the control problem, only a few options are available in this category. An interface for state estimation and parameter identification is included, and future efforts may utilize this capability more extensively. In this work, perfect full state feedback is assumed, and the controller's database model of coefficients is assumed to represent the best combination of a *priori* and in–flight information. Nevertheless, an **Extended Kalman Filter**, and a **Recursive Least Squares** parameter identification algorithm, have been implemented for use with the approximate linear models and discrete linear controllers. An algorithm for real–time learning or improvement of the aerodynamic, propulsion, and control coefficient estimates would be the next logical addition to the AHSV simulation.

Display

While a simulation is in progress, the information that is presented to the user will depend on the display configuration settings. The **Vehicle Display** option, for example, displays an exterior 3–D solid object representation of the vehicle in flight with indicators to display altitude, ascent rate, downrange distance, Mach number, throttle setting, thrust vectoring, and elevon deflections. The **Status Panel** option is a full numeric display showing all forces, moments, positions, rates, coefficients, controls, etc. Another primary option is for **On–Line Plots** which have the capability to do multivariable plots of any of *50* simulation variables as functions of time or any other variable. Altogether over *30* different display formats are possible, including multiple data entry forms[6]; vehicle design displays; scramjet engine geometry; and control system performance displays, such as state tracking errors, constraint violations, trajectory cost, and search progress.

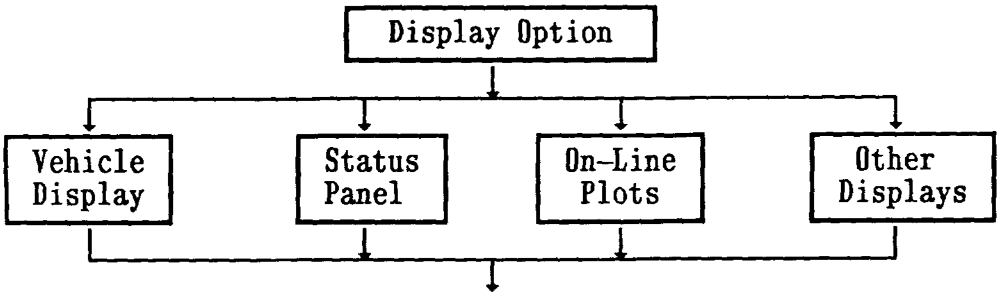


Figure 5.12: Display Block

For modes which display the vehicle, varying degrees of graphical fidelity can be selected. The simplest representation is a wireframe vehicle, followed by a normal projection model, and finally a solid–object (full hidden surface removal) rendering. Normal projection means that only surfaces with outward pointing

[6] Data entry forms are used, for example, to specify geometric design parameters, flight conditions, scramjet design parameters, disturbance magnitudes, control requirements such as constraints or tolerances, etc.

normal vectors having a projection toward the viewing direction are visible. For completely convex objects, normal projection works quite well. The solid object mode, however, will correctly display only the panels that should be visible, regardless of the shape, and is better suited for more complex geometries.[7] Naturally, the speed of the simulation decreases with increasing fidelity, however, the hidden surface algorithm used was specially tailored to meet the requirements of a real–time simulation. A specialized surface sorting algorithm that takes advantage of nearly ordered sets is combined with a polygon blanking approach to provide almost perfect (but very fast) projections.[8]

There are several other options relative to vehicle display. One of these is an observer's reference frame. The simulation actually keeps track of *5* different coordinate systems including body, wind, LVLH, inertial, and observer's axes. The view seen on the display can be moved to any position around the vehicle, and the orientation and size of the vehicle is displayed accordingly. Other options include a mass properties display, differential shading of laminar and turbulent panels, selectable background and foreground colors, and a choice of display resolutions.

Interrupts

The final block in Figure 5.1 is for **Interrupts**, which are user generated keyboard commands that are entered while a simulation is running. Many of the simulation options can be reconfigured "on the fly", including display modes, debugging trace and dump modes, error messages, 3–DOF or 6–DOF simulation, flight control modes, and all graphics options. In addition, an interrupt is available to terminate the simulation.

[7] Note that these projection algorithms are not only for display. The Newtonian hypersonic aerodynamics model uses the normal projection approach to determine pressure coefficients for each surface panel (see Chapter 4).

[8] Errors can only occur for one unusual circumstance, from a certain viewing direction, in which a small panel overlays a panel which is much larger. For the AHSV geometry this situation cannot occur. For general geometries, the algorithm works best for panels of approximately the same size, and it becomes infallible as the panels get smaller.

Chapter 6

Theoretical Background

The purpose of this chapter is provide the theoretical background that is required for the development of the RIFC controller in Chapter 7. The first section covers the fundamental concepts of Lyapunov Stability Theory. The important definitions and theorems are presented in a manner which leads to the specific results that are directly applicable to the flight control problem for air–breathing hypersonic vehicles. The reader familiar with this subject may wish to read only the material related to the stability of discrete–time systems beginning with Definition 6.9. The following section in this chapter presents a full discussion of heuristic optimization methods and the A^* algorithm. The detailed A^* search procedure is explained, and the theoretical properties of A^* using different classes of heuristic information are developed. The notation, definitions, and theorems in this section will be important for Chapter 7. The discussion is guided by the objective of motivating the application of A^* to the solution of discrete optimal control problems, since this is how it is used in the RIFC controller.

6.1 Lyapunov Stability Theory

For any control problem, regardless of the application, the most important issue to be addressed is the stability of the closed–loop system. Unstable systems are typically useless and can often lead to disaster[1]. In general, there are two fundamental approaches available for the analysis of the stability of dynamic systems. The first, Lyapunov Stability Theory, is based on the work of Alexander Michhailovich Lyapunov, which was published in 1892 (*The General Problem of Motion Stability*). Lyapunov stability is applicable to the analysis of dynamic systems that are initially in a nonequilibrium state, or are momentarily disturbed from equilibrium. All the available tools known for linear control theory depend upon Lyapunov's theorems for justification of their application to the analysis and control of locally linearized nonlinear systems. The second approach to stability analysis is called Input–Output Stability, and is applicable to problems for which it is desired to characterize a system's output behavior given known properties of its inputs. Signals are classified into sets defined by the L_p–Spaces, and stability is defined in terms of the properties of a system's input–output mapping. For the case of L_∞ bounded signals this kind of stability is called Bounded–Input–Bounded–Output (BIBO) stability. This thesis is concerned with stability in the sense of Lyapunov, and the reader is referred to reference [Vid1] for a complete discussion of both Lyapunov and Input–Output Stability.

The concepts in Lyapunov stability theory are divided into two basic methods called Lyapunov's Indirect Method (or First Method), and Lyapunov's Direct Method (or Second Method). The First Method characterizes the properties of a nonlinear system in the neighborhood of an equilibrium point based on the properties of the locally linearized system. This method is the one which provides the theoretical justification for using linear stability theory to control nonlinear systems for small perturbations from equilibrium.

THEOREM 6.1: Lyapunov's Indirect Method

Given the autonomous[2] nonlinear system

$$\dot{\underline{x}} = \underline{f}(\underline{x}) \qquad \text{with } \underline{f}(0) = 0 \tag{6.1a}$$

[1] This is particularly true in the context of the present problem.

[2] An autonomous system is defined as one for which the dynamic equations are not explicitely dependent on time.

which can be expressed as [3]

$$\dot{\underline{x}} = \left(\frac{\partial f}{\partial \underline{x}}\right)_{\underline{x}=0} \underline{x} + f_{\text{HOT}}(\underline{x}) \qquad (6.1b)$$

and the linearized approximate system

$$\dot{\underline{x}} = A\underline{x} \qquad \text{where } A = \left(\frac{\partial f}{\partial \underline{x}}\right)_{\underline{x}=0} \qquad (6.1c)$$

the following statements are true:

- *If the linearized system in (6.1c) is strictly stable, (all eigenvalues $\lambda_i(A)$ are in the left–half plane (LHP)), then the equilibrium is asymptotically stable for the original nonlinear system.*
- *If the linearized system is unstable, (at least one eigenvalue $\lambda_i(A)$ is in the right–half plane (RHP)), then the equilibrium point is unstable for the original nonlinear system.*
- *If the linearized system is marginally stable, (all $\lambda_i(A)$ are in the LHP except at least one on the jw–axis), then no conclusion can be drawn regarding the stability of the original nonlinear system.*

REMARKS:

(1) For proof, refer to reference [Vid1].

(2) Note that this theorem also applies to the more general system described by $\dot{\underline{x}} = f(\underline{x},\underline{u})$, with a nonzero equilibrium at $\underline{x} = \underline{x}_e$. A redefinition of the state as $\underline{z} = \underline{x} - \underline{x}_e$, and the representation of the feedback law as $\underline{u} = \underline{g}(\underline{z})$, gives a system of the form $\dot{\underline{z}} = \underline{h}(\underline{z})$ with the equilibrium $\underline{h}(0) = 0$, which fits the description of (6.1a). This property will hold for all Lyapunov theorems presented in this section.

(3) This theorem applies only to local stability in an arbitrarily small neighborhood of the equilibrium point. For control purposes, the linearized system may not be useful if the operating range goes beyond the radius from equilibrium for which the stability results apply.

[3] Where $f_{\text{HOT}}(\underline{x})$ refers to higher-order-terms of the Taylor's expansion for $f(\underline{x})$.

Lyapunov's Direct (Second) Method is the more general stability analysis tool, since it is capable of directly analyzing the properties of nonlinear time–varying systems. This approach is based on the concept of the energy of a system. Consider an isolated system which has some state that can be identified as an equilibrium condition. Also, suppose that a suitable "energy" function can be defined in such a way that it is zero at the equilibrium state, and positive for all other states. Now suppose that the system is initially disturbed from its equilibrium. If the system is purely dissipative, the energy stored must decrease with time until the system has reached a state with the lowest possible energy. In other words, if the total energy of a physical system is continually dissipated, then the system must eventually end up at its equilibrium state. Thus, it is possible to draw conclusions about the stability of a system by examining the behavior of a simple scalar function (the system's energy). The utility of Lyapunov's Second Method is that it gives conditions on the stability of a system without having to directly solve the dynamic equations, (which can be very difficult for many systems). [Slo1]

Before presenting Lyapunov's theorems, it is important to define exactly what is meant by stability. For linear systems, stability can be simply defined in terms of the location of the poles of the closed–loop system. For nonlinear systems, however, the notion of stability becomes much more complicated, and several definitions are required to fully describe the stability of a system [Hed1].

DEFINITION 6.1: Lyapunov Stability

The equilibrium state $\underline{x} = \underline{0}$ of the dynamic system $\dot{\underline{x}} = f(\underline{x},t)$ is ***stable*** *(in the sense of Lyapunov) if, for every real $R > 0$, there exists an $r(R,t_0) > 0$, such that $\| \underline{x}(t_0) \| < r$ implies that $\| \underline{x}(t; \underline{x}_0,t_0) \| < R$ for all $t \geq t_0$. Otherwise, the equilibrium point is unstable.* [4]

[4] The notation $\underline{x}(t;\underline{x}_0, t_0)$ is used to recognize that for non-autonomous systems the trajectory $\underline{x}(t)$ is a function of the initial state and time. Since for autonomous systems the response is time invariant, and the dependence on initial state is understood, the trajectory is expressed simply as $\underline{x}(t)$.

REMARKS:

(1) Essentially, stability means that small disturbances result in small deviations from the equilibrium. The system trajectory can be kept close to the equilibrium state provided the system starts sufficiently close to it.

(2) If $r(R,t_0)$ is independent of t_0 the system is uniformly stable.

(3) If $r(R,t_0)$ includes the entire state space, then the equilibrium is said to be globally stable.

DEFINITION 6.2: Attractivity

The equilibrium state $\underline{x} = \underline{0}$ of the dynamic system $\dot{\underline{x}} = \underline{f}(\underline{x},t)$ is **attractive**, *if for some $\epsilon > 0$ and all $\delta > 0$ (where $\delta < \epsilon$), there exists some $T(\delta,t_0)$ such that, for all $\| \underline{x}(t_0) \| < \epsilon$, $\| \underline{x}(t) \| \leq \delta$ (for $t-t_0 \geq T$).*

REMARKS:

(1) Attractivity means that all trajectories starting in a neighborhood of the equilibrium eventually converge to the equilibrium ($\underline{x}(t) \longrightarrow \underline{0}$ as $t \longrightarrow \infty$).

(2) If $T(\delta,t_0)$ is independent of t_0 the system is uniformly attractive.

(3) If $\| \underline{x}(t_0) \| < \epsilon$ includes the whole state space, then the equilibrium is said to be globally attractive.

DEFINITION 6.3: Asymptotic Stability

The equilibrium state $\underline{x} = \underline{0}$ of the dynamic system $\dot{\underline{x}} = \underline{f}(\underline{x},t)$ is asymptotically stable if it is both stable and attractive.

REMARK: See [Hed1] for examples of attractive but unstable systems.

DEFINITION 6.4: Equiasymptotic Stability

The equilibrium state $\underline{x} = \underline{0}$ *of the dynamic system* $\dot{\underline{x}} = f(\underline{x},t)$ *is equiasymptotically stable if it is both stable and uniformly attractive with respect to* $\underline{x}_0$.

DEFINITION 6.5: Uniform Asymptotic Stability

The equilibrium state $\underline{x} = \underline{0}$ *of the dynamic system* $\dot{\underline{x}} = f(\underline{x},t)$ *is uniformly asymptotically stable if it is both uniformly stable and uniformly attractive.*

DEFINITION 6.6: Exponential Stability

The equilibrium state $\underline{x} = \underline{0}$ *of the dynamic system* $\dot{\underline{x}} = f(\underline{x},t)$ *is exponentially stable if there exist* $a > 0$, $\lambda > 0$, *for some* $\| \underline{x}(t_0) \| < r$ *such that*

$$\| \underline{x}(t) \| \leq a \| \underline{x}(t_0) \| \, e^{-\lambda(t-t_0)} \quad \forall \, t \geq t_0$$

GENERAL REMARKS:

(1) All the above stability definitions are global if the conditions hold over the entire state space.

(2) For linear time–invariant (LTI) systems the above definitions are reduced to only one type of stability, since linear stability is always global, asymptotic, and exponential.

A final stability definition that will be important for the remainder of this chapter is the concept of robust stability. So far, stability has been defined in terms of the properties of a known system in the neighborhood of an equilibrium point. In general, however, we are interested in the stability of a class of closed–loop systems of the form $\dot{\underline{x}}(t) = f(\underline{x},\underline{\phi},t)$ for $\underline{\phi} \in \underline{\Omega}_{\phi}$, where $\underline{\phi}$ is a parameter vector from the set of all possible parameter values $\underline{\Omega}_{\phi}$. The regions of the parameter space for which the system remains stable is of primary importance.

DEFINITION 6.7: Robust Stability

The equilibrium state $\underline{x} = \underline{0}$ of the dynamic system $\dot{\underline{x}} = f(\underline{x},\underline{\phi},t)$ is robustly stable with respect to the set of parameters $\underline{\Omega}_{\Phi}$, if it is stable for all vectors $\underline{\phi}$, such that $\underline{\phi} \in \underline{\Omega}_{\Phi}$.

REMARK: Robust definitions (with respect to a set of parameters) for the properties of attractivity, asymptotic, uniform, and exponential stability, all follow similarly to the definition above.

A final definition is required to describe the concept of positive definite functions, which will be needed for Lyapunov's Direct Method.

DEFINITION 6.8: Positive Definite Functions [Vid1]

A continuous function $V(\underline{x},t)$ which maps $\mathbb{R}^n \times \mathbb{R}_+ \longrightarrow \mathbb{R}$ is said to be locally positive definite (l.p.d.) if there exists a continuous nondecreasing function $\alpha : \mathbb{R} \longrightarrow \mathbb{R}$ such that,

(i) $\alpha(0) = 0,\ \alpha(\xi) > 0,\ \forall\ \xi > 0$

(ii) $V(\underline{0},t) = 0,\ \forall\ t \geq 0$

(iii) $V(\underline{x},t) \geq \alpha(\| \underline{x} \|),\ \forall\ t \geq 0$, and $\forall\ \underline{x} \in B_r$,

where B_r is the n–dimensional ball defined by
$B_r = \{\ \underline{x}: \| \underline{x} \| \leq r\ \},\ r > 0$

V is said to be globally positive definite (p.d.) if (iii) holds for all $\underline{x} \in \mathbb{R}^n$ and, in addition, $\alpha(\xi) \longrightarrow \infty$ as $\xi \longrightarrow \infty$.

REMARKS:

(1) $V(\underline{x},t)$ is negative definite if $-V(\underline{x},t)$ is positive definite.

(2) $V(\underline{x},t)$ is positive semi–definite if $V(\underline{0},t) = 0$ and $V(\underline{x},t) \geq 0\ \ \forall\ \underline{x} \neq 0.$

(3) $V(\underline{x},t)$ is negative semi–definite if $-V(\underline{x},t)$ is positive semi–definite.

With the various types of stability and the properties of positive definite functions defined above, it is now possible to present the fundamental theorem of Lyapunov's Direct Method for stability analysis. The most general form of the theorem will be presented first, followed by a discussion of its application to more specific cases. It is assumed without loss of generality (See Remark 2 of Theorem 6.1) that the dynamic system can be expressed in the form $\dot{\underline{x}} = \underline{f}(\underline{x},t)$ with an equilibrium point at $\underline{x} = 0$. The basic approach is to define a positive definite (energy) function of the states, $V(\underline{x},t)$, and to show that this function decreases with time (since that would imply that the system converges toward an equilibrium). This idea is expressed more formally below:

THEOREM 6.2: Lyapunov's Direct Method [Vidl]

Consider the nonautonomous nonlinear dynamic system

$$\dot{\underline{x}} = \underline{f}(\underline{x},t) \quad \text{with } \underline{f}(\underline{0},t) = \underline{0},\ \forall\ t \geq 0 \qquad (6.2)$$

If, within a ball $\underline{x} \in B_r$ $(B_r \equiv \{\ \underline{x}: \|\ \underline{x}\ \| \leq r\ \},\ r > 0)$ *around the equilibrium point* $\underline{x} = \underline{0}$, *there exists a scalar function* $V(\underline{x},t)$ *defined for* $t \geq 0$, *with continuous partial derivatives, and satisfying*

(i) $V(\underline{x},t)$ *is positive definite*

$$V(\underline{x},t) \geq \alpha(\|\ \underline{x}\ \|) > 0 \qquad (6.2a)$$

(where $\alpha(\xi)$ *is p.d.)*

(ii) $\dot{V}(\underline{x},t)$ *is negative semi–definite*

$$\dot{V}(\underline{x},t) = \frac{\partial V}{\partial t} + \nabla V \cdot \underline{f}(\underline{x},t) \leq 0 \qquad (6.2b)$$

then the equilibrium state $\underline{x} = \underline{0}$ *is stable in the sense of Lyapunov and the function* $V(\underline{x},t)$ *is called a Lyapunov function of the system.*

If, furthermore,

(iii) $V(\underline{x},t)$ *is a "decrescent" function*

$$V(\underline{x},t) \leq \beta(\|\ \underline{x}\ \|) \qquad (6.2c)$$

(where $\beta(\xi)$ *is p.d.)*

then the equilibrium state $\underline{x} = \underline{0}$ is ***uniformly stable.***

If condition (ii) is strengthened by requiring that

(iv) $\dot{V}(\underline{x},t)$ *is negative definite*

$$\dot{V}(\underline{x},t) = \frac{\partial V}{\partial t} + \nabla V \cdot \underline{f}(\underline{x},t) \leq -\gamma(\|\underline{x}\|) \qquad (6.2d)$$

(where $\gamma(\xi)$ is p.d.)

then the equilibrium state $\underline{x} = \underline{0}$ is ***uniformly asymptotically stable.***

If, in addition,

(v) $\underline{x} \in B_r$ *is replaced by* $\underline{x} \in \mathbb{R}^n$, *and* (6.2e)

(vi) $V(\underline{x},t)$ *is radially unbounded* (6.2f)

$$a(\|\underline{x}\|) \longrightarrow \infty \quad as \ \|\underline{x}\| \longrightarrow \infty$$

then the equilibrium state $\underline{x} = 0$ is ***globally uniformly asymptotically stable.***

REMARKS:

(1) The proof of this theorem is based on the fact that $\dot{V}$ being negative definite implies that $V(\underline{x}(t),t) < V(\underline{x}(t_0),t_0)$, $\forall\, t > t_0$, since

$$V(\underline{x}(t),t) - V(\underline{x}(t_0),t_0) = \int_{t_0}^{t} \dot{V}(\underline{x}(\tau),\tau)d\tau, \quad \text{for } t > t_0$$

As $V(\underline{x}(t),t)$ decreases $\underline{x}(t) \longrightarrow 0$ since this is the value for which V has its minimum. See references [Vid1,Slo1] for the complete proof.

(2) The conditions for stability presented above are sufficient conditions only. If these conditions fail for a particular choice of the function $V(\underline{x},t)$, called a candidate Lyapunov function, then nothing is implied about the stability of the system. The system may indeed be stable, and a better choice of the candidate Lyapunov function could indicate this. The disadvantage of Lyapunov's Direct Method is that, in general, necessary conditions for stability cannot be ascertained. Furthermore, for general nonlinear systems there is no systematic approach for selecting good Lyapunov functions.

(3) Lyapunov's Direct Method consists of selecting candidate Lyapunov functions, testing for stability with the conditions above, and either concluding some degree of stability, or repeating the process with another candidate Lyapunov function.

One of the conditions above for asymptotic stability, item (*iv*), is often very difficult to meet. A very important extension to Theorem 6.2 above, called LaSalle's Theorem, provides a means to prove asymptotic stability for systems and Lyapunov functions which satisfy condition (*ii*) but not the stronger condition (*iv*).

THEOREM 6.3: LaSalle's Theorem

Consider the ***autonomous*** *nonlinear dynamic system*

$$\dot{\underline{x}} = f(\underline{x}) \quad \text{with } f(\underline{0}) = \underline{0} \tag{6.3}$$

If, within a ball $\underline{x} \in B_r$ *around the equilibrium point* $\underline{x} = \underline{0}$*, there exists a scalar function* $V(\underline{x})$ *with continuous partial derivatives, and satisfying*

(i) $V(\underline{x})$ *is positive definite* *(6.3a)*

(ii) $\dot{V}(\underline{x}(t;\underline{x}_0,t_0)) \leq 0, \ \forall \ \underline{x}$ *(6.3b)*
(negative semi–definite)

(iii) $\dot{V}(\underline{x}(t;\underline{x}_0,t_0)) \not\equiv 0, \ \forall \ t \geq t_0$ *(6.3c)*
along trajectories of $\dot{\underline{x}} = f(\underline{x})$ *with* $\underline{x}_0 \not\equiv 0$

then the equilibrium state $\underline{x} = \underline{0}$ *is* ***asymptotically stable.***

If, in addition,

(iv) $\underline{x} \in B_r$ *is replaced by* $\underline{x} \in \mathbb{R}^n$ *(6.3d)*

(v) $V(\underline{x})$ *is radially unbounded* *(6.3e)*

then the equilibrium state $\underline{x} = 0$ *is* ***globally asymptotically stable.***

REMARKS:

(1) The basic idea behind LaSalle's Theorem is that asymptotic stability can be concluded even if $\dot{V}$ is not negative definite. As long as $\dot{V} \leq 0$ (negative semi–definite) and $\dot{V} \not\equiv 0$ along any system trajectories from

nonzero initial conditions, then it is impossible for the system to get "trapped" in any states other than the equilibrium state. For the complete proof see [Vid1].

(2) Note that LaSalle's Theorem is applicable only to systems with time–invariant (or periodic) dynamics.

(3) For nonautonomous systems, a similar theorem can be obtained using Barbalat's Lemma, which relates the asymptotic properties of functions to the properties of their time derivatives (see [Slo1]).

Lyapunov's Direct Method has been presented for the general case of nonlinear time–varying dynamic systems. The conditions for stability of linear or autonomous systems are somewhat less restrictive. For LTI systems, in fact, Lyapunov's Theorem can be used to generate necessary as well as sufficient conditions for stability. It can be shown that, if the solution to the so called Lyapunov Equation[5] is a symmetric positive definite matrix, then the corresponding LTI system is strictly stable. Many other stability analysis tools are available for linear systems, such as eigenvalue analysis, Routh Hurwitz tests, Bode plots (gain & phase margins), and Nyquist techniques. The equivalence of these methods and the criteria for Lyapunov stability is discussed in [Hed1]. A number of approaches also exist for the analysis of SISO–LTI systems in combination with nonlinear (memoryless) feedback elements. The Circle criterion is applicable to these systems, and is an extension of the Nyquist criterion for a class of nonlinear elements called sector (1st and 3rd quadrant) nonlinearities. Popov's criterion is another frequency domain approach based on the properties of positive real transfer functions. An important result known as the Kalman–Yacubovich Lemma provides the relationship between the existence of Lyapunov functions and the conditions for stability in the frequency domain; it is instrumental for the derivation of Popov's criterion, and many other results. The reader is referred to references [Vid1] and [Hed1] for a thorough coverage of the topics mentioned above, which are limited to LTI, or SISO nonlinear systems.

[5] For $\dot{\underline{x}} = A\underline{x}$, $V = \underline{x}^T P\underline{x}$, and $\dot{V} < -\underline{x}^T Q\underline{x} < 0$, the Lyapunov Equation is given by $A^T P + PA = -Q$.

In this thesis, we are interested in stability for the more general problem of MIMO nonlinear systems. As will be discussed in section 7.1, however, the AHSV trajectory control problem can be simplified by making some reasonable assumptions for which the plant dynamics are effectively time–invariant (i.e. autonomous). The stability criteria for the (simplified) case of autonomous nonlinear systems are presented below.

THEOREM 6.4: **Lyapunov's Direct Method for Autonomous Systems**

Consider the autonomous nonlinear dynamic system

$$\dot{\underline{x}} = f(\underline{x}) \quad \text{with } f(\underline{0}) = \underline{0} \qquad (6.4)$$

If, within a ball $\underline{x} \in B_r$ around the equilibrium point $\underline{x} = \underline{0}$, there exists a scalar function $V(\underline{x})$ with continuous partial derivatives, and satisfying

(i) $V(\underline{x})$ *is locally positive definite* *(6.4a)*

(ii) $\dot{V}(\underline{x})$ *is locally negative semi–definite* *(6.4b)*

then the equilibrium state $\underline{x} = \underline{0}$ is ***stable.***

If, instead,

(ii) $\dot{V}(\underline{x})$ *is locally negative definite*

then the equilibrium is **asymptotically stable.**

If, in addition,

(iii) *the ball B_r is replaced by $\mathbb{R}^n$* *(6.4c)*

(iv) $V(\underline{x}) \longrightarrow \infty$ *as* $\| \underline{x} \| \longrightarrow \infty$ *(6.4d)*

then the equilibrium state $\underline{x} = 0$ is ***globally asymptotically stable.***

REMARKS:

(1) The main difference between this theorem and Theorem 6.2 is that for autonomous systems there is no need for a uniformity condition (see item *(iii)* Theorem 6.2).

(2) Note also that LaSalle's Theorem applies directly to the autonomous case.

It is also useful to mention, that in addition to Lyapunov criteria for the stability of systems, there are theorems which guarantee the *instability* of a system. One might expect, in view of the above discussion, that if there exists a positive definite function $V(\underline{x})$ for which $\dot{V}(\underline{x})$ is also positive definite, in the neighborhood of the equilibrium $f(\underline{0}) = \underline{0}$ of the autonomous system $\dot{\underline{x}} = f(\underline{x})$, then the equilibrium point is unstable. Similar criteria for the instability of nonautonomous systems also exist.

Another important category of theorems related to Lyapunov stability, are the existence theorems. It can be shown that if a nonautonomous system given by $\dot{\underline{x}} = f(\underline{x},t)$ has an equilibrium state that is uniformly asymptotically stable then there exists a Lyapunov function $V(\underline{x},t)$ that meets the conditions of Theorem 6.2 (see [Hed1]).

In all of the discussion above, stability has been defined, and the criteria for the stability of a system have been expressed, in terms of continuous functions and their derivatives. The flight control system in this thesis, however, necessarily operates as a discrete–time system. The controller is implemented on a digital computer, and all inputs and outputs are values pertaining to discrete time intervals. Definitions for stability and a statement of Lyapunov's Direct Method are therefore needed for the case of discrete–time systems. A proof of Lyapunov's Theorem for discrete systems is also included, since this theorem is directly utilized later in Chapter 7.

DEFINITION 6.9: Discrete Lyapunov Stability

The equilibrium state of the discrete dynamic system $\underline{x}(t+\Delta t) = f_d(\underline{x}(t))$ *with* $f_d(\underline{0}) = \underline{0}$ *is* **stable** *if, for every real* $R > 0$*, there exists an* $r(R) > 0$*, such that* $\| \underline{x}(0) \| < r$ *implies that* $\| \underline{x}(k\Delta t) \| < R$ *for all* $k \in \mathbb{Z}_+ \equiv \{1,2,3,\ldots\infty\}$

DEFINITION 6.10: Discrete Asymptotic Stability

The equilibrium state of the discrete dynamic system $\underline{x}(t+\Delta t) = f_d(\underline{x}(t))$ *with* $f_d(\underline{0}) = \underline{0}$ *is* ***asymptotically stable*** *if it is stable, and if there exists an* $r > 0$*, such that* $\| \underline{x}(0) \| < r$ *implies that* $\| \underline{x}(k\Delta t) \| \longrightarrow 0$ *as* $k \longrightarrow \infty$*, (i.e. there exists* $k > 0$, $k \in \mathbb{Z}_+$*, such that* $\| \underline{x}(k\Delta t) \| = 0$*).*

THEOREM 6.5: Lyapunov's Direct Method for Discrete–Time Autonomous Systems

Consider the discrete–time autonomous nonlinear system

$$\underline{x}(t+\Delta t) = f_d(\underline{x}(t)) \quad \text{with } f_d(\underline{0}) = \underline{0} \tag{6.5}$$

If, within a ball $\underline{x} \in B_r$ *there exists a scalar function* $V(\underline{x})$ *with continuous partial derivatives, and satisfying*

(i) $V(\underline{x})$ *is locally positive definite* (6.5a)

(ii) $V(\underline{x}(t+\Delta t)) - V(\underline{x}(t)) \leq 0$ (6.5b)

then the equilibrium state $\underline{x} = \underline{0}$ *is* ***stable.***

If, instead,

(iii) $V(\underline{x}(t+\Delta t)) - V(\underline{x}(t)) < 0$ (6.5c)

for all $\underline{x} \neq \underline{0}$*, then the equilibrium* is ***asymptotically stable.***

PROOF OF DISCRETE STABILITY:

Since $V(\underline{x})$ *is positive definite and continuous, the following are true:*

(1) $V(\underline{0}) = 0$

(2) $\exists\ V_m > 0$ such that $V_m = \underset{\| \underline{x} \| = R}{Min} \left\{ V(\underline{x}) \right\}$ for any R, $0 \leq R \leq R_0$

where R_0 *defines the ball* $\underline{x} \in B_{R_0}$ *for which conditions (i) and (ii) above hold true.*

(3) Also, we have from (i) that $\left\{ \begin{array}{lcl} V \geq V_m & & \| \underline{x} \| \geq R \\ & \textit{for} & \\ V \leq V_m & & \| \underline{x} \| < R \end{array} \right\}$

From (1), (2), and the continuity of $V(\underline{x})$*, there exists an* r*:* $0 < r < R$ *such that for* $\| \underline{x} \| \leq r \Longrightarrow V(\underline{x}) < V_m$*.*

Also, since $V(\underline{x}(t+\Delta t)) - V(\underline{x}(t)) \leq 0$, $V(\underline{x})$ is a nonincreasing function along any trajectory of the system and therefore $V(\underline{x}(k\Delta t)) \leq V(\underline{x}(0))$.
Now, if $\| \underline{x}(0) \| \leq r$ then $V(\underline{x}(0)) < V_m$, which also implies that $V(\underline{x}(k\Delta t)) < V_m$, and this requires that $\| \underline{x}(k\Delta t) \| < R$ for all $k \in \mathbb{Z}_+$ from (3) above.
Therefore, $\| \underline{x}(0) \| \leq r \Longrightarrow \| \underline{x}(k\Delta t) \| < R$, which implies stability from Definition 6.9 above.

PROOF OF DISCRETE ASYMPTOTIC STABILITY:

This proof is based on showing that the assumption that $\underline{x}(t)$ is not asymptotically stable leads to a contradiction for systems meeting conditions (i), (ii), and (iii) in Theorem 6.5.
Since asymptotic stability implies that $\| \underline{x}(k\Delta t) \| = 0$ (Definition 6.10) for some $k \in \mathbb{Z}_+$, it is initially assumed that $\underline{x}(k\Delta t) \neq \underline{0}$ for all $k \in \mathbb{Z}_+$.
Now, condition (iii), $V(\underline{x}(t+\Delta t)) - V(\underline{x}(t)) < 0$, implies that along any particular path $\underline{x}(k\Delta t)$, $\underline{x}(k\Delta t) \neq \underline{0}$ (for any k), there exists some $\beta > 0$ such that $V(\underline{x}(t+\Delta t)) - V(\underline{x}(t)) < -\beta$ for all $k \in \mathbb{Z}_+$. Extrapolating this inequality out to time $t=k\Delta t$ gives $V(\underline{x}(k\Delta t)) \leq V(\underline{x}(0)) - \beta k$.
The assumption that $\underline{x}(k\Delta t) \neq 0$ for all k, and the fact that $V(\underline{x})$ is positive definite, means that $V(\underline{x}(k\Delta t)) > 0$, therefore,

$$0 < V(\underline{x}(k\Delta t)) \leq V(\underline{x}(0)) - \beta k$$

But this inequality leads to a contradiction in the limit as $k \longrightarrow \infty$ (since the signs of β, k, and V are all positive).
Therefore, there must be a k such that $\underline{x}(k\Delta t) = 0$. In other words, there exists $k > 0$, $k \in \mathbb{Z}_+$, such that $\| \underline{x}(k\Delta t) \| = 0$, which implies asymptotic stability from Definition 6.10.

Finally, since it is also the objective of the AHSV flight controller to provide stability robustness, the only remaining task in this section is to modify the definitions and stability theorem above to include parametric uncertainty.

DEFINITION 6.11: Robust Stability For Discrete Systems

The equilibrium state of the discrete dynamic system $\underline{x}(t+\Delta t) = f_d(\underline{x}(t),\underline{\phi})$ *with* $f_d(\underline{0},\underline{\phi}) = \underline{0}$ *is* ***robustly stable*** *with respect to the set of parameters* $\underline{\Omega}_\phi$*, if it is stable (according to Definition 6.9) for all vectors* $\underline{\phi}$*, such that* $\underline{\phi} \in \underline{\Omega}_\phi$*. It is* ***robustly asymptotically stable*** *with respect to* $\underline{\Omega}_\phi$*, if it is asymptotically stable (according to Definition 6.10) for all* $\underline{\phi} \in \underline{\Omega}_\phi$*.*

THEOREM 6.6: Lyapunov's Theorem For Robust Stability Of Discrete–Time Autonomous Systems

Consider the discrete–time autonomous nonlinear system

$$\underline{x}(t+\Delta t) = f_d(\underline{x}(t),\underline{\phi}) \quad \text{with } f_d(\underline{0},\underline{\phi}) = \underline{0} \tag{6.6}$$

If, within a ball $\underline{x} \in B_r$ *there exists a scalar function* $V(\underline{x},\underline{\phi})$ *with continuous partial derivatives, for all* $\underline{\phi} \in \underline{\Omega}_\phi$*, and satisfying*

(i) $V(\underline{x},\underline{\phi})$ *is locally positive definite* *(6.6a)*

(ii) $V(\underline{x}(t+\Delta t),\underline{\phi}) - V(\underline{x}(t),\underline{\phi}) \leq 0$ *(6.6b)*

then the equilibrium state $\underline{x} = \underline{0}$ *is* ***robustly stable.***

If, in addition,

(iii) $V(\underline{x}(t+\Delta t),\underline{\phi}) - V(\underline{x}(t),\underline{\phi}) < 0$ *(6.6c)*

then the equilibrium is ***robustly asymptotically stable.***

REMARK: The proof of Theorem 6.6 exactly parallels the proof already given for Theorem 6.5 with the addition that each statement must apply for all $\underline{\phi} \in \underline{\Omega}_\phi$.

The main objectives of this section have been to review the important concepts of Lyapunov Stability Theory, and to develop the specific theorems that are directly applicable to the flight control problem for AHSVs. It is important to note that all of the results presented here apply only to unforced systems.

Therefore, Theorems 6.5 and 6.6 would actually be used in a manner that determines which feedback law, when applied to the open–loop system, results in stability for the closed–loop. Also note that the desired state for the flight control problem is not at $\underline{x} = \underline{0}$, but along a continuously changing desired trajectory. Since the objective is to track this trajectory, the results of this section can be applied to the dynamic equations in terms of the state errors ($\dot{\underline{e}}(t) = \underline{h}(\underline{e}(t),\underline{\phi})$ or $\underline{e}(t+\Delta t) = \underline{h}_d(\underline{e}(t),\underline{\phi})$). For these error dynamics, the desired state is always at $\underline{e} = \underline{0}$, independent of the desired trajectory or actual state values. The details of how to apply Lyapunov's theorems to the development of the AHSV flight controller will be presented in Chapter 7.

6.2 The A* Optimization Method

Despite the fact that optimal control theory is a well developed field with powerful capabilities (see [Kirl], and [Bryl]), it is often impractical in the application to real problems. Many real systems are multivariable, nonlinear, constrained by complicated state and input inequality constraints, and are often, to some degree, unknown (see section 7.1). For these kinds of systems, the (Hamilton–Jacobi–Bellman) HJB–equation, or the Calculus of variations, usually results in a series of coupled two–point boundary value problems that are extremely difficult to solve, especially for high order systems. Dynamic programming is an alternative approach (applicable to discretized systems) that essentially floods the state space with optimal trajectories (obtained through use of the principle of optimality) and then interpolates among these for the optimal solution from a specified initial state. Unfortunately, however, the curse of dimensionality is a major handicap for dynamic programming with high order systems[6]. This also prevents dynamic programming from being very useful for real–time applications. Another drawback of optimal control theory is that there is no mechanism for the inclusion of additional information regarding the specific properties of the solution to a particular problem. This kind of information is often available, but not in the form of constraints or costs, and can be very useful in reducing the size of the solution space and in guiding the search for an optimal trajectory. Finally, in the context of practical real–time problems, it may be desirable (or necessary) to accept a suboptimal solution in order to reduce the required computational effort. A quantifiable tradeoff between optimality and feasibility is not directly possible using classical optimal control methods.

6.2.1 INTELLIGENT CONTROL

With the trend toward cheaper, high–speed computers with parallel processing architectures, has come the possibility to approach complex control problems in new ways that offer an improved degree of flexibility, simplicity, and capability.

[6] The memory and computational requirements for dynamic programming increase dramatically with the number of states and controls.

Intelligent Control techniques attempt to combine some of the concepts from Control Theory, Artificial Intelligence, and Operations Research, in order to simplify the task of control system design. One kind of intelligent control concept, presented in [Gue1], provides a general framework for the application of heuristic search strategies to the solution of optimal control problems. A structure is defined for an algorithm (called Heuristically Enhanced Optimal Control (HEOC)) which would use an A^* type search strategy to solve optimal control problems represented in the form of a graph or tree. The original problem is discretized in time, quantized in the states and controls, and the graph represents the entire space of possible sequences of future states (depending on the control inputs) up to some final time. Compared to dynamic programming, this strategy has the advantage of a more efficient method of exploring the solution space. Under certain conditions discussed below, the A^* algorithm is also guaranteed to find the optimal solution, and with far less computational effort than with dynamic programming. The RIFC algorithm developed in this thesis fits into this general HEOC framework.

A few other studies have also investigated the use of heuristic search techniques for applications to control problems. Reference [Par1] presents a path planning controller for control moment gyros used to control the attitude of spacecraft. A heuristic search is used to find appropriate gimbal motions in order to achieve desired control torques while avoiding singularities present in the actuator space. Another application [Nii1] performs a trajectory optimization for spacecraft proximity operations using an A^* search technique.

It is important to point out that the term Intelligent Control has been used to describe a wide variety of control schemes. Some of these are heuristic search techniques [Pea1], neural networks/connectionist control [Bar3], rule–based heuristic algorithms [Asa1], and expert control systems [Ast3]. Many of these methods have in common the idea of reducing the complexity of a problem by using a memory intensive approach. There is a great deal of variation among these "intelligent" schemes, however, in terms of theoretical properties that can be guaranteed in the context of a control problem. For example, an informed search algorithm can use heuristic information to solve an optimization problem while retaining certain guaranteed properties. A gradient method is a good example, since it is known to have theoretical guarantees of local convergence, and it is essentially an algorithm that uses the heuristic "always move in the direction of the negative gradient of the function." On the other hand, a heuristically based set of rules in a software control program may not be able to guarantee anything about the actual behavior of a dynamic system. Likewise, theoretical guarantees of stability or convergence remain elusive in the application of neural networks to the control of dynamic systems [Mil2]. In the context of this thesis, Intelligent Control is meant to refer to the solution of an optimal control problem using an informed (heuristically guided) search technique. It will be shown later that this approach can be made to exhibit the desired properties important for control, (such as convergence to a solution, guaranteed stability, and robustness to uncertainty).

6.2.2 HEURISTIC OPTIMIZATION METHODS

The A^* algorithm is only one of a class of heuristic search techniques for solving problems with the objective of optimization and/or the satisfaction of constraints. In general these techniques are applicable to any problem that can be put in a form that presents choices among a multitude of possible options available in different states of a system. There must be a goal and/or some measure by which to evaluate and compare potential solutions. All of the following examples fit this description: path planning problems such as the road–map or travelling salesman problem [Hel1]; game playing problems such as for chess or checkers; puzzles or maze problems; or any general (discrete) optimization problem.

The distinguishing feature of heuristic search techniques is a systematic and informed strategy for finding a solution to a problem. Complex problems often involve an immense number of possibilities, and heuristic information can be used to minimize the search space and reduce the number of evaluations required to obtain a solution. *A heuristic is defined as a criterion, method, rule, or principle for deciding which among several alternative courses of action promises to be the most effective in order to achieve some goal.* In the context of problem solving, heuristics can be divided into two categories: Problem Domain heuristics, which provide information about the solution from knowledge about the problem; and Algorithmic heuristics, which are rules that direct the search procedure. Heuristics in the first category can often be used to drastically simplify the problem, however, this is sometimes at the risk of missing the solution altogether. The second category of heuristics are really the defining characteristics of a particular search algorithm (such as the gradient search method). Depending on the choice of these "rules" certain properties of the search algorithm can be guaranteed independent of the problem. Fortunately, it is often possible to include both types of heuristics in a search for a solution without compromising the properties of the algorithm. [Peal]

Besides providing a more efficient means for finding the solution to optimization problems (that may have otherwise been intractable), heuristic search techniques have other advantages as well. The ability to incorporate any kind of additional information that can help to simplify a problem is an inherent property of this type of approach. Furthermore, it is possible (see section 6.2.5), to control a tradeoff between the optimality of the solutions and the required computational effort in obtaining them. Another advantage is flexibility, since any special cases, unusual constraints or costs, discontinuities, failures, contingencies, and other problem specific complexities can be easily included in the problem. This general framework can include almost any mathematical or other type of criterion in an optimization problem. Finally, these algorithms can directly utilize the parallel processing capability of modern computers to achieve greater performance. The main disadvantage of intelligent search strategies, despite all their devices to search as efficiently as possible, is that these methods are still computationally intensive and may require a considerable amount of computer memory. These issues will be addressed in later sections.

Central to any heuristic search strategy for problem solving are the following elements: a means to represent candidate solutions, a method for evaluating these candidates, and an algorithm to direct the search in the most promising directions. These elements are the basis of discussion in the following paragraphs.

The most natural representation for many problems is in terms of subsets of potential solutions organized in the form of a graph (or tree) as shown in Figure 6.1.

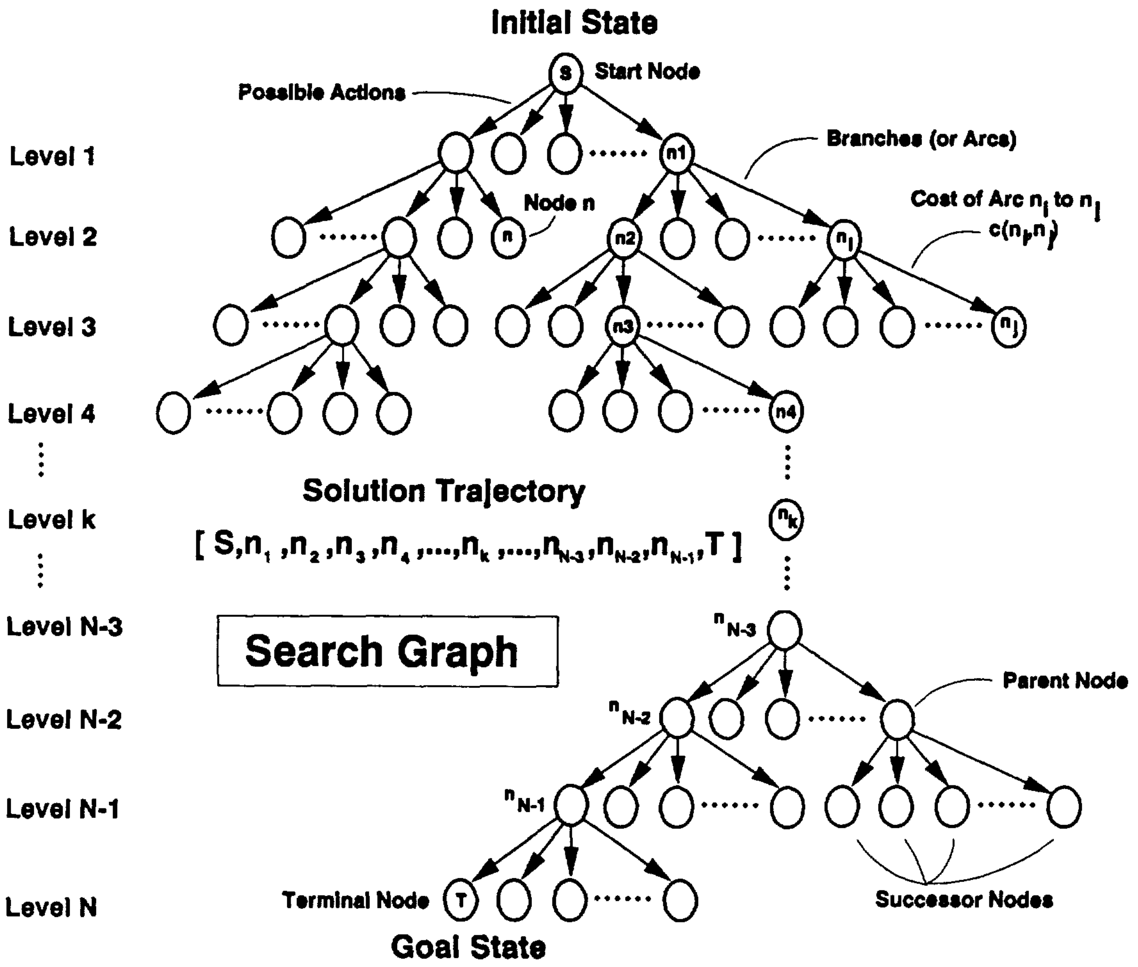


Figure 6.1: Graph Structure for Search Problems

Nodes in this graph structure represent states of a system, and the branches from a particular node represent the possible actions available from that state. For any node n_i, the nodes n_j, representing the achievable states resulting from these actions, are called successor (or offspring) nodes, and the node n_i is referred to as the parent (or predecessor) node. A tree is defined as a graph in which each node has at most one parent. The system's initial condition (or initial problem) is designated as the start node S, and any terminating (solution) nodes are identified as T nodes, which are elements of the set of goal nodes Γ (i.e. $T \in \Gamma$). A sequence of nodes n_1, n_2, ... n_k, where each n_i is the successor of n_{i-1} is called a path of length k.

A path from node S to node T is, therefore, a solution to the problem. Any specified path from the start node S to any other node n represents a unique subset of candidate solutions. It also represents a unique subproblem, since the remaining path from n to a terminating node T is still unspecified, and the number of possible paths from node n to T are a subset of those from S. The branching degree of each node, denoted N_B, is defined as its maximum number of possible offspring; and a *uniform graph* is one for which N_B is the same for all nodes. When the branching degree is a finite number the graph is called *locally-finite.* We will only be interested in *uniform locally-finite graphs.*

Candidate solutions are compared using a cost function which in some way evaluates the merit of any particular path. Usually, the branches (or arcs) of the graph are assigned individual weightings or costs $c(n_i,n_j)$, and these are associated with the overall objective that is to be accomplished or optimized. The cost of any path is a function of the costs along its individual arcs, and the optimal solution (from S to T) is, therefore, the admissible path with the minimum total cost.

The selection of the cost function is a highly problem–specific task. The properties of the cost function, in combination with the search algorithm itself, are the distinguishing features of the various heuristic search techniques. This is an important point, because it is possible to use heuristics in the cost function as well as in the search algorithm, and the difference can be dramatic. For example, consider a chess playing application. Certain heuristic rules related to the game (such as *trade a Knight for a Queen*) could be directly included as weighted terms in a cost function to be minimized. Problem domain information used in this fashion, however, allows the meaning of what is called an optimal solution to be affected by the heuristics. A search algorithm that minimizes the resulting cost function might easily lose a game simply because it optimized the wrong objective. If the true objective is to win the game, then the algorithm can fail. In certain problems, as in chess, the objective is very difficult to define, and there is little choice but take this approach and accept the risk. Unfortunately, the occasional failures of algorithms using heuristic information in this specific way, has given heuristics, in general, a bad reputation for usually working well, but *not* all of the time.

In sharp contrast to the example above, there are many other problems that lend themselves very well to rigorously defined cost functions. Optimal control problems are typically in this category. For example, consider a servo positioning application where the cost is measured in terms of a time integral of squared position error. Any search that truly minimizes this cost function actually finds the optimal solution for a problem with a precise objective. The difference between this example and the previous one (chess) is that here, heuristic information is *not* used in the specification of the objective function. Note also, that nothing has been said here to preclude the use of heuristics in the decision process of the search algorithm itself. In fact, it will be shown that algorithmic heuristics meeting certain requirements can be used to guarantee various properties of the search procedure. Based on this discussion, it should be clear that for critical applications, where failure is not acceptable, the use of heuristics should be limited to the algorithmic rules of the search itself.

In order to apply heuristic search techniques to discrete optimal control problems, a cost function must be defined as a cumulative function of the states and/or the control actions along a trajectory (or path). The cost of a solution path passing through any node n is then comprised of the interval costs for each arc of the trajectory through all nodes along the entire solution path. Candidate solutions can be compared by using partial cost information from S to their deepest node n, plus an estimate of the cost from n to T (since the actual cost for the unexplored part of the solution path is unknown until the search has been completed). Algorithmic heuristics can help to estimate the cost of terminating a candidate solution. As will be seen later, the better these estimates are, the more efficient the search algorithm becomes.

The search algorithm itself can be viewed as a set splitting procedure. It starts with only the S node, from which the set of all possible paths can be reached. It then chooses to explore various paths leading to other nodes, from which only a subset of the possible paths can be reached (i.e. the subset of paths constrained to pass through that node and its predecessors). The fundamental operations of the search algorithm are generating a successor from any node, handling pointers to relate offspring to parent nodes, evaluating candidate paths, and deciding the order in which nodes should be explored. Other tasks include expanding nodes, which means to generate all direct successors, and truncating them, which prevents any successor nodes from ever being explored. Truncating is performed when the algorithm is sure that the solution path does not pass through a particular node (for example, due to constraints). The process of splitting subsets of solutions and comparing partially completed paths continues until a node is found that is a member of the set of goal states. The path leading to this node is a solution for a constraint satisfaction problem. Better solutions (lower cost) are usually obtained by continuing the search along promising but incomplete paths. The optimal solution is found when all other paths (partial or complete) are sure to have a higher cost. Since in most practical problems, the entire search graph will be far too large to be represented explicitly in memory, a search algorithm will incrementally generate and store only the portions of the graph which are needed at any one time. An efficient well–guided search strategy should be able to avoid exploring the entire graph without unduly discounting viable solution candidates.

There are three categories of search strategies for exploring a graph: hill climbing, uninformed systematic search, and informed directed search. Hill climbing is a local optimization method. The gradient method for function minimization falls into this category. In terms of a graph search procedure, this approach expands a node, evaluates the successors, chooses the best one, and repeats the process at the next level. For simple convex problems this technique works very well. In general, however, it suffers from the drawback that it converges to local extremal solutions and cannot escape. The main types of uninformed systematic search procedures are the breadth–first and depth–first strategies. Breadth–first algorithms explore all possibilities to the same depth before continuing to deeper levels of a graph. Although the optimal solution is guaranteed (if one exists) a large amount of memory and time may be required. Depth–first algorithms search a graph in depth before breadth, generating one node at a time and exploring it before any others. Backtracking is used to redirect the search to unexplored nodes when a particular candidate solution leads to failure. This approach is very economical in terms of storage, and can work well when there are many solutions. It can be very inefficient, however, especially for problems with a large branching degree and many levels in the graph. Both of these methods are simple variations of an exhaustive blind search; that is, the search direction is not guided by the nature of the solution or the costs of the explored paths. The method of dynamic programming also falls into this category, since it is essentially a recursive formulation of a breadth–first search which indiscriminately solves for optimal paths from every point in the state space (see [Peal,Guel]).

An informed directed search is one which uses additional information to decide which nodes to explore first. These are known as best–first (BF) algorithms, since at each step they expand the node that is the most "promising" of all nodes encountered so far in the search (regardless of where it is in the graph). As the search progresses, it jumps from branch to branch on the graph as it pursues the most promising paths. The search terminates when a goal node is found, or it cannot find a more promising node to expand than the last one explored.

The promise of a node n (or a candidate path) can be measured in many ways. In general, a numerical figure of merit called the *heuristic evaluation function* $f(n)$ is defined. The form of $f(n)$ is completely arbitrary and can include any kind of heuristic information in its computation. As explained earlier, however, the choice of $f(n)$, and the type of heuristics used, directly affects the inherent properties of the search algorithm. For this reason, in optimal control applications it is desirable to rigorously define $f^*(n)$ as being exactly equal to the problem's cost function for an optimal path constrained to pass through the node n. Since the terminating cost of an incomplete candidate path is unknown, however, this portion of the cost usually has to be estimated. This means that the search algorithm has to work with path cost estimates, denoted by $f(n)$, rather than with $f^*(n)$ itself. Nodes are then compared using the cumulative costs associated with each state or arc along a path in the graph. During the search, the best node to expand is the one with the lowest estimated cost $f(n)$. A method of estimating the terminating cost for each partial trajectory is required, but it can be shown [Peal] that if these estimates are always optimistic, then the best–first search is guaranteed to converge to the global optimal solution (if one exists). Such a convergence proof is given for the A^* method in Theorem 6.9 of section 6.2.4.

6.2.3 THE A* SEARCH PROCEDURE

Best–first algorithms differ in the way they choose their heuristic evaluation functions $f(n)$. The A* search procedure is a special case of the best–first approach where the objective is to achieve a minimum sum cost. The following properties of $f(n)$ are required by the A* algorithm:

(1) The function $f(n)$ is a cost function of the form

$$f(n) = g(n) + h(n) \tag{6.7}$$

where $g(n)$ represents the cumulative cost along the path from the start node S to the node n, and $h(n)$ represents the estimated cost of completing the path from node n to a terminal node T.

(2) The cost function is additive, that is

$$g(n) = \sum_{i=1}^{N} c_i(n_i, n_{i-1}) \tag{6.8}$$

$$\textit{with } n_0 = \textit{the start node } S$$

$$\textit{and } n_N = \textit{the current node } n$$

$$\textit{and } g(n_j) = g(n_i) + c(n_i, n_j) \tag{6.9}$$

$$\textit{node } n_j \textit{ a descendant of } n_i$$

The A* algorithm was first developed by Hart, Nilson, and Raphael [Har2,Har3], and is one of the most popular search techniques used because of its efficiency and other favorable properties (discussed below). In view of the specific A* characteristics given above, it is also clearly consistent with the objectives of discrete optimal control problems.

The A* algorithm is presented in Figure 6.2. It works by keeping track of two sets of nodes, those that have been generated (OPEN nodes), and those that have already been expanded (CLOSED nodes). It also makes use of pointers which link each node to its predecessor. The search begins with the starting node S (initial

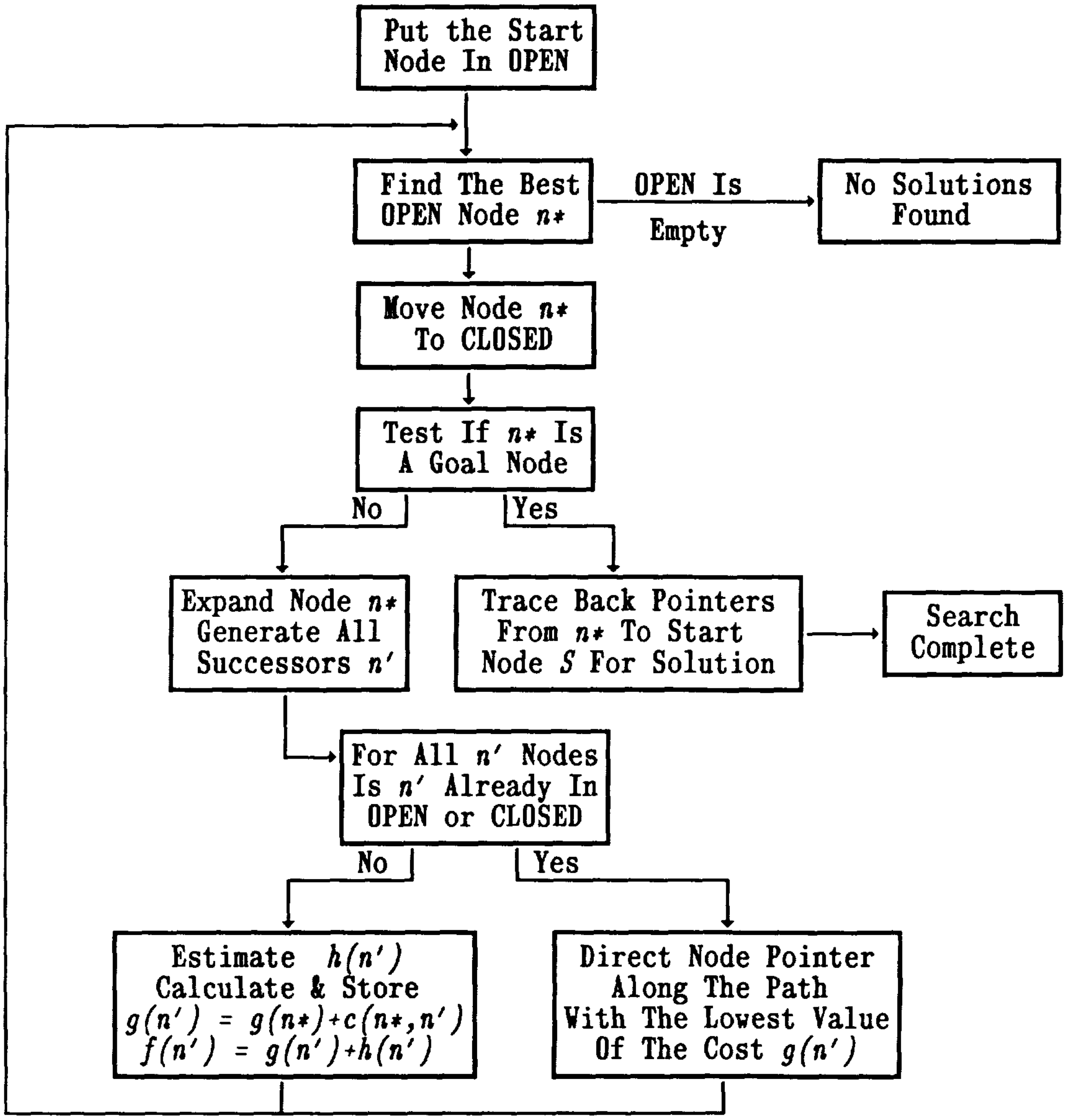

Put the Start Node In OPEN
Find The Best OPEN Node n*
OPEN Is Empty
No Solutions Found
Move Node n* To CLOSED
Test If n* Is A Goal Node
No
Yes
Expand Node n* Generate All Successors n'
Trace Back Pointers From n* To Start Node S For Solution
Search Complete
For All n' Nodes Is n' Already In OPEN or CLOSED
No
Yes
Estimate h(n') Calculate & Store g(n') = g(n*)+c(n*,n') f(n') = g(n')+h(n')
Direct Node Pointer Along The Path With The Lowest Value Of The Cost g(n')

Figure 6.2: The A^* Search Algorithm

state) in the OPEN set, moves it to the CLOSED set, and then generates all immediate (1st level) successors denoted as n'. These new nodes are placed in the OPEN set with pointers directed back to the start node. The cost for trajectories passing through each successor node is estimated as $f(n') = g(n') + h(n')$, where

$g(n')$ is the known cost from S to n', and $h(n')$ is the estimated termination cost from n' to some $T \in \Gamma$. Based on the values of $f(n')$ the next node for expansion $n*$ is selected. Before proceeding, $n*$ is tested to check if it is a goal node. If so, the solution is obtained by tracing back the pointers to recover the full solution path. Otherwise, $n*$ is moved to the CLOSED set, and all its successors are generated repeating the cycle. Depending on the problem, it is possible for newly generated nodes to overlap existing ones (two paths leading to the same state). When this occurs, the path from S to n' with the lowest cost $g(n')$ is preferred, and the pointer is adjusted from n' to the parent node along the preferred path. If the OPEN set ever becomes the empty set, it means that no solution has been found. If the basic A* algorithm is allowed to continue processing until the search terminates, the only way it can exit with failure is if there is no solution. [Peal]

The A* algorithm, as presented thus far, generates sets of nodes (OPEN and CLOSED) which continually grow until a solution is found. There are many mechanisms, however, that can operate during the search to reduce these sets or slow their growth. The constraints of a problem can eliminate nodes and entire sub–trees of the graph either before or after expansion. For optimal control problems, constraints on the control inputs limit the number of nodes per expansion, and constraints on the states eliminate sets of candidate paths. Other ways to reduce the search space fall into the category of *enhancements* to the basic A* algorithm, such as cost–bounded paths, suboptimal search, hybrid methods, and others which will be discussed in section 6.2.5.

6.2.4 PROPERTIES OF THE A* ALGORITHM

It has already been mentioned that the properties of heuristic search algorithms can depend upon the way in which heuristic information is used. The purpose of this section is to present some important properties that can always be guaranteed under certain conditions. Since the A* algorithm is the basis for the RIFC controller, the discussion is focused on the properties specific to A*. The following definitions will be required:[7]

[7] The material and the notation in this section follows the discussion found in reference [Peal],

$P_{n_i-n_j} \equiv$ *Any path from node* n_i *to* n_j $\mathbb{P}_{n_i-n_j} \equiv$ *The set of all such paths*

P_{n-T} $\equiv$ *Any path from a node* n *to a solution node* T, $(T \in \Gamma)$

$\mathbb{P}_{n-\Gamma}$ $\equiv$ *The set of all paths from* n *to the set* Γ *(solution nodes)*

$P^*_{n_i-n_j} \equiv$ *An optimal path from node* n_i *to* n_j

$c^*(n_i,n_j) \equiv$ *The cost of the optimal path from* n_i *to* n_j

$g^*(n) \equiv$ *The optimal cost from* S *to* n, *(i.e.* $c^*(S,n)$*)*

$h^*(n) \equiv$ *The optimal cost from* n *to* T, $\underset{T\in\Gamma}{Min}\{c^*(n,T)\}$

$f^*(n) \equiv g^*(n)+h^*(n) \equiv$ *The cost of the optimal solution passing through* n

C^* $\equiv$ *The cost of the optimal path from* S *to* Γ, *(i.e.* $h^*(S)$*)*

Γ^* $\equiv$ *The subset of solution nodes accessed by optimal paths*

The function $f^*(\cdot)$ has some interesting properties. For an optimal path, $f^*(S) = C^*$ since $f^*(S)$ represents the cost of the entire path. Also note that $f^*(S) = h^*(S) = g^*(T)$ by definition. Now, for a node n^* along an optimal path ($n^* \in P^*_{S-\Gamma}$) it must be true that $f^*(n^*) = C^*$. That is, if a node is on an optimal path, then the best trajectory that is constrained to pass through it will be the optimal path, (with the optimal cost). Conversely, it is also true that if a node is *not* along an optimal path ($n \notin P^*_{S-\Gamma}$) then the cost of the best trajectory constrained to pass through it is greater than that of the optimal solution ($f^*(n) > C^*$). This implies that a solution can only be optimal if every node along its trajectory contains an optimal path passing through it. This is, in effect, a statement of the principle of optimality (see [Kir1]).

Another very important property of $f^*(\cdot)$ is that, if $f^*(n)$ were known exactly for all nodes n, it would provide enough information to lead an A* search directly toward the optimal solution. This is true, since for all nodes in each expansion, $f^*(n)$ gives the best possible cost for pursuing a path passing through each node n.

where a more detailed analysis of the subject can be found.

There would be no need to explore to deeper levels to decide which nodes were best. A search using $f^*(\cdot)$ would only expand nodes along the optimal path and the solution trajectory would be found immediately. Now, for any partially explored path, the function $f^*(n)$ is unknown only because the cost of the terminal portion of the trajectory $h^*(n)$ (from n to T) is unknown. However, $h^*(n)$ can be approximated by some function $h(n)$, and this is the basis for the efficiency of the A^* algorithm. Using $f(n) = g(n)+h(n)$ to approximate $f^*(n)$, A^* is able to discriminate between partial paths which are potential solutions and those which cannot possibly lead to an optimal solution. The more precisely $h(n)$ approximates $h^*(n)$, the more discriminating the search becomes. In sharp contrast, the dynamic programming approach essentially searches the entire space to find the exact value of $h^*(n)$ for each node n. [Peal]

The most fundamental property of the A^* algorithm is its convergence behavior. It is shown below, that if the cost estimate $h(n)$ is always optimistic, that is $h(n)$ underestimates $h^*(n)$, then the A^* algorithm is guaranteed to converge to the optimal solution (if one exists). The cost estimate $h(n)$ is commonly called the heuristic function, since it usually uses some heuristic information to estimate the true cost of completing a particular trajectory. The condition above defines what is meant by an *admissible heuristic.* Note that any kind of information whatsoever can be used to compute $h(n)$, and as long as $h(n)$ is admissible, the A^* algorithm will be guaranteed to converge. This idea is presented more formally in the definitions and theorems below. Theorem 6.9 provides the proof of convergence for the A^* algorithm.

DEFINITION 6.12: *A heuristic function h is said to be admissible if*

$$h(n) \leq h^*(n) \qquad \forall\, n \qquad (6.10)$$

(i.e. the terminating cost estimate is optimistic)

THEOREM 6.7: *The A^* algorithm always terminates with a solution if one exists.*

PROOF: *Referring to Figure 6.2, the search fails only in the case that the set of OPEN nodes becomes empty. Now, assume that n' represents a node along a solution path P_{S-T}.*

Note that,

(1) if a solution path P_{S-T} exists, then, at any time during the search, there must exist a node n' (along P_{S-T}) such that $n' \in$ OPEN.[8]

(2) if P_{S-T} is not found, and OPEN is empty, then the node n' (on P_{S-T}) must have no successors.

This, however, contradicts the assumption that n' is along a solution path, since by definition, any such node (except the terminal node T), must have one successor which is also along P_{S-T}. Therefore, the set OPEN cannot become empty before the solution P_{S-T} has been found.

THEOREM 6.8: *If $h(n)$ is admissible, then at any time during the A^* search, there exists an OPEN node n' along the optimal path P^*_{S-T} such that $f(n') \leq C^*$.*

REMARK: Essentially, this says that if the search has not terminated, one of the nodes available for expansion is part of the optimal solution and has an optimistic cost estimate. (This result is needed to prove Theorem 6.9)

PROOF: *Consider any optimal path $P^*_{S-T} \in \mathbb{P}^*_{S-\Gamma}$, where P^*_{S-T} is defined by the sequence of nodes,*

$$P^*_{S-T} \equiv [S, n_1, n_2, \ldots, n', \ldots, T].$$

[8] This is true, since the search starts with the node S in OPEN. When S is expanded it is moved to the set CLOSED, but one of its successors n' is clearly on the path $P_{S\text{-}T}$. The node n' stays in the set OPEN until it too is expanded, at which time one of its successors (now in OPEN) must also be on $P_{S\text{-}T}$. Therefore, there is always one node on $P_{S\text{-}T}$ in OPEN until T is found.

Also *let n' be the shallowest OPEN node such that $n' \in P^*_{S-T}$.*

(Some node along the optimal path is always in OPEN[9]*)*

All ancestors of n' are therefore in the set CLOSED, and since the partial path defined by $[S, n_1, n_2, ..., n']$ is optimal, it follows that,

$$g(n') = g^*(n')$$

Now, using the fact that $h(n)$ is admissible,

$$f(n') = g(n')+h(n') = g^*(n')+h(n') \leq g^*(n')+h^*(n') = f^*(n')$$

Finally, since n' is along the optimal path, $f^*(n') = C^*$ and therefore,

$$f(n') \leq C^*$$

which verifies the theorem. [Peal]

THEOREM 6.9: Convergence of the A* Algorithm

The A* algorithm using an admissible heuristic function $h(n)$ converges to the optimal solution (if one exists).

PROOF: *From Theorem 6.7 above, we have that the A* algorithm always terminates with a solution if one exists.*

Now, suppose a search terminates with a solution node $T \in \Gamma$ such that $f(T) = g(T) > C^*$, (i.e. a suboptimal solution).

Referring to the A* algorithm in Figure 6.2, the node T could only have been selected if,

$$f(T) \leq f(n) \qquad \forall\, n \in OPEN$$

Since $f(T) > C^*$, this would require that $f(n) > C^*$ for all other nodes in OPEN as well. This, however, contradicts Theorem 6.8, which assures the existence of at least one OPEN node n' with $f(n') \leq C^*$, provided that $h(n)$ is admissible.

[9] Using the same argument as in the previous footnote, but substituting $P^*_{S\text{-}T}$ for $P_{S\text{-}T}$.

Therefore, the terminal node T must have $f(T) = g(T) = C^*$, which implies that the A^* search terminates when it has found the optimal solution path P^*_{S-T}.

Based on the preceding discussion, and on Theorem 6.9 above, it is clearly desirable to have the largest possible terminating cost estimate function $h(\cdot)$ such that $h(n) \leq h^*(n)$ for all n. The inequality guarantees convergence to the optimal solution, while the difference h^*-h determines the efficiency (or discriminating power) of the A^* search. As mentioned earlier, if $h = h^*$ then the A^* algorithm would only explore nodes along the optimal path. Conversely, if no heuristic is used $(h = 0)$ then A^* becomes an exhaustive breadth–first search which expands all nodes for which $g(n) < C^*$. The interval $0 \leq h(\cdot) \leq h^*(\cdot)$ represents the entire spectrum of search effort, from exhaustive search to immediate solution. The ability of the A^* algorithm to reduce the required search effort by excluding nodes from expansion (based on estimated path costs) is called the *pruning power* of $h(\cdot)$. Furthermore, an A^* search using a heuristic h_2 (denoted A^*_2) such that $h_2(n) > h_1(n)$ $(\forall\ n \neq T,\ T \in \Gamma)$ is said to be more informed than the A^*_1 search using h_1. It can also be shown that A^*_2 is a more efficient search than A^*_1 in that it requires less node expansions to find the optimal solution. The following theorem defines which nodes will be expanded in an A^* search.

THEOREM 6.10: *Any node n that is expanded by an A^* search, using an admissible $h(\cdot)$, cannot have a path cost estimate $f(n)$ that exceeds the actual cost of the optimal solution C^*. That is,*

$$f(n) \leq C^* \qquad \text{for all expanded nodes}$$

REMARKS: (1) All nodes for which $f(n) > C^*$ are excluded from expansion.

(2) Note that since $f(n) = g(n)+h(n)$, a larger terminal cost estimate $h(n)$ increases $f(n)$. A larger $f(n)$ implies that more nodes will be excluded from expansion according to (1).

Theorem 6.10 also suggests a possible enhancement to the basic A^* algorithm, using *Cost–bounded paths*, that will be used to advantage in the RIFC control system. If C^* were known a *priori*, all paths passing through nodes for which $f(n) > C^*$ could be eliminated as candidate solutions. Since C^* is not known, if it can be upper bounded such that $C^* < C_{\max}$, then all nodes in OPEN with $f(n) > C_{\max}$ could be eliminated immediately[10]. By requiring all OPEN nodes to meet the criterion $f(n) < C_{\max}$, the search effort and storage space needed by A^* can be drastically reduced. The upper bound $C_{\max}$ can come from an initial suboptimal solution, and can be updated as the search progresses.

Another useful property of the A^* algorithm is related to the concept of a monotonic heuristic function. This will have important implications for the RIFC control strategy developed in Chapter 7 and 8, since the cost function used will fit this description.

DEFINITION 6.13: *A heuristic function $h(\cdot)$ is monotonic if it satisfies:*

$$h(n) \leq c(n,n') + h(n') \tag{6.11}$$

for all n, n' such that n' is a successor to n.

REMARKS: (1) In effect, a monotonic $h(\cdot)$ implies that the cost estimate $f(n)$ for a particular path is more optimistic when the node n is farther from a goal node $T \in \Gamma$.

(2) For deeper nodes (i.e. more thoroughly explored paths) the cost estimate is higher, and as $n \rightarrow T$, $f(n) \rightarrow f^*(n)$.

(3) It is also quite easily shown that every monotonic heuristic function $h(n)$ (satisfying $h(T) = 0$) is also admissible.

THEOREM 6.11: *An A^* algorithm using a monotonic heuristic function finds optimal paths to each expanded node, that is,*

$$g(n) = g^*(n) \qquad \text{for all } n \in CLOSED \tag{6.12}$$

[10] If $f(n) > C_{\max}$ then the cost of the partially explored path P_{S-n} is already so large that a complete path passing through this node could not possibly be optimal.

REMARKS: (1) This theorem states that, of all possible paths P_{S-n} leading from the start node S to a node n, the optimal one P^*_{S-n} will have been found by the A* search before any successor nodes to n are explored.

(2) For the proof, refer to [Peal].

The advantage of having monotonic heuristics is that, since the explored part of any trajectory is always known to be optimal, the A* search will never require a node to be expanded more than once[11]. This can be seen by noting that from Theorem 6.11, the comparison between nodes in the set OPEN is now based on the quantity $f(n) = g^*(n)+h(n)$, which cannot be improved by taking a different path from S to n. Therefore, once a node is selected, expanded, and CLOSED there would never be a need to OPEN it again. This property is useful because it prevents the search from wasting effort re–expanding nodes.

The properties of the A* algorithm with monotonic heuristics also help to illuminate the relationship between the cost estimate $h(n)$ and the discriminating power of A*. Since the actual cost $g^*(n)$ of a partially explored path P^*_{S-n} is known prior to expanding an OPEN node n, the cost estimate $f(n) = g^*(n)+h(n)$ is really dependent only upon $h(n)$. The condition for node expansion, expressed in Theorem 6.10 as $f(n) \le C^*$, is given below for monotonic $h(\cdot)$.

THEOREM 6.12: *The necessary condition for A* to expand a node n, given that the function $h(\cdot)$ is monotonic, is given by*

$$f(n) = g^*(n) + h(n) \le C^* \tag{6.13a}$$

and the sufficient condition is

$$f(n) = g^*(n) + h(n) < C^* \tag{6.13b}$$

[11] Without monotonic $h(n)$ it is possible that a new path, with a lower cost, might be found from the initial state to a node that has already been expanded.

Note here that, since $h(\cdot)$ is monotonic, $g^*(n)$ is known before a node n is expanded, and the cost estimate $f(n)$ is clearly a direct function of $h(n)$. The criterion for excluding a node from OPEN becomes $h(n) > C^* - g^*(n)$, (from $f(n) > C^*$), which will hold for more nodes if $h(n)$ is more informed (larger), and this translates into less node expansions and reduced search effort.

6.2.5 ENHANCEMENTS TO THE A* ALGORITHM

In the previous section it has been shown that the A* algorithm using the cost estimate $f(n) = g(n)+h(n)$ with admissible $h(n)$ is guaranteed to converge to the optimal solution (if one exists). In addition, if $h(n)$ is a monotonic and well informed heuristic, the search effort can be minimized. In fact, the A* algorithm can be shown to be optimal (in the sense that it expands the least number of nodes) when compared to all other optimization algorithms that minimize an additive cost function, with the same amount of information, while guaranteeing convergence [Dec1].

There are two practical problems, however, that may prevent the basic A* algorithm from meeting its objectives: computation time and available memory. If a search is interrupted prematurely because an application cannot wait any longer for a solution, then the guarantees of returning a complete solution are forfeit. Similarly, if any branches of the solution tree are unduly truncated due to limitations on the storage space for expanded nodes, the algorithm may fail. These issues are important, since for most complex problems the search graph for the basic A* algorithm can be quite large. This section mentions several enhancements to the A* algorithm that can be used to reduce the memory and computational requirements. More details are presented in Chapter 8, where all of the concepts below are utilized in some form as part of the RIFC autopilot logic.

The most fundamental way to simplify the task of an A* search is through the definition of the problem. Any constraints on the properties of the solution automatically restrict the number of paths that are considered admissible. In the context of an optimal control problem, constraints on the states and controls naturally limit the branching degree and the graph size. Otherwise, the discretization of the state space, the controls, and time, determine the dimensions of the graph that must be searched. For best performance, the quantization steps used for each variable should be made as large as possible while retaining enough precision for the specific problem.

As mentioned in the previous section, Cost–bounded paths can be used to drastically reduce the size of the search space. Once a complete path is found that meets all the constraints, then all future candidate paths, having an estimated cost greater than the cost of the known path, can be eliminated from the search. This will be discussed in more detail in Chapter 8.

Hybrid search strategies are another way to improve the performance of A^*. There are many variations on this concept, but the main idea is to alternate between depth–first and best–first search patterns. This approach is also known as a staged search because most of these algorithms switch strategies in stages of depth on the graph as the search runs out of memory.

The most important topic in this section is a variation of the A^* algorithm known as the A_ϵ^* search. The basic A^* search typically wastes a considerable amount of effort simply deciding among paths that are very nearly equivalent in terms of cost. Using A_ϵ^* it is possible to sacrifice a controllable degree of optimality in the solution in order to eliminate this computational effort. This is applicable, of course, only in situations where a suboptimal solution that meets all the constraints is acceptable. The A_ϵ^* algorithm is almost identical to the A^* algorithm described in Figure 6.2, with only one exception that is explained below.

DEFINITION 6.14: The A_ϵ^* Algorithm

(1) *A_ϵ^* uses the sets OPEN and CLOSED as described for A^*.*

(2) *It makes use of another set called FOCAL, which is the set of all OPEN nodes such that the estimated cost $f(n)$ is not higher than $(1+\epsilon)$ times the current best cost estimate, that is*

$$FOCAL = \left\{ n: f(n) \leq (1+\epsilon) \underset{n' \in OPEN}{Min} \left[f(n') \right] \right\} \qquad (6.14)$$

(3) Nodes are selected for expansion from the set FOCAL (not OPEN), and the priority for selection is based on a depth–first rule.

REMARKS: (1) This definition actually describes a specific variation of the A_ϵ^* algorithm. In general, the set FOCAL is used to group nodes with approximately similar costs. Which of these nodes is actually explored is then determined by an independent rule. In this case, the rule is chosen to be a depth–first priority.

(2) The result of using the above algorithm can be visualized by defining another set called LOCAL (not FOCAL) as the set of all nodes n' generated from the last expansion of a node n^*. Now, for monotonic $h(n)$, the A_ϵ^* algorithm is essentially biased to expand the best node from LOCAL unless a node on another path has a substantially better cost estimate. In this way, the A_ϵ^* search probes to deeper levels of the graph sooner by avoiding superfluous branching for small improvements in cost.

The degree to which optimality is traded for speed is determined by the value of ϵ, as seen in the following theorem.

THEOREM 6.13: *The A_ϵ^* algorithm converges to a suboptimal path P_{S-T} costing no more than $(1+\epsilon)$ times the optimal cost,*

$$f(T) \leq (1+\epsilon)\, C^* \qquad (6.15)$$

PROOF: See reference [Peal].

Since the cost distribution among paths in any graph is specific to the problem, it is not possible to make any quantitative statements relating an ϵ to a given decrease in required node expansions. It is feasible, however, to adjust the search speed on–line by adjusting the value of ϵ dynamically. In this way, a search could be forced to reach goal–level depths within a prescribed amount of time.

Several other variations and enhancements to the basic A^* algorithm are possible, including some that depend on the nature of the problem. In Chapter 8, it will be shown how certain properties of the AHSV flight control problem can be used to advantage in order to strictly limit the memory requirements for the optimization. In Chapter 7, Lyapunov stability theory is combined with the A^* algorithm to guarantee the other properties that are important for control (tracking stability and robustness to uncertainty).

Finally, regardless of any enhancements to the A^* and A_ϵ^* algorithms, these search techniques are particularly well suited for implementation on computers with parallel computing architectures. The evaluations of $f(n)$, for each newly generated node in an expansion, are completely independent processes that could be done simultaneously. In a problem that had a thousand possible successors for each node, the search could proceed a factor of a thousand times faster on a parallel machine (with a thousand simple processing units). Other tasks, such as managing the sets of nodes (OPEN, CLOSED, etc.), could also be done in parallel, and in the background while the search continues. An evaluation of the computational requirements for the RIFC controller, in the context of the capabilities of today's parallel computers, is given in Chapter 8.

Chapter 7

Control System Development

The vehicle models and the flight simulation, described in Chapters 4 and 5, provide the means to represent the dynamic behavior of an air–breathing hypersonic vehicle. In this chapter, the flight control problem is addressed, and the approach taken in this thesis, called Robust Intelligent Flight Control (RIFC), is developed and justified from a theoretical point of view. First, however, a formal statement of the AHSV flight control problem is presented and discussed. The foundations of this approach are then given as a series of propositions, which are followed by supporting proofs, derivations, and physical arguments. The objective is to demonstrate that the performance of the RIFC algorithm can be guaranteed under some reasonable assumptions and conditions. This chapter concludes with a functional and logical description of the fundamental RIFC algorithm. A discussion of practical enhancements to the algorithm (related to time and memory limitations) is reserved until Chapter 8.

7.1 Formal Problem Description

A qualitative description of the objectives and the structure of an overall AHSV autopilot was given in Chapter 3. In short, the main objectives of the flight control system are to account for the nonlinear dynamics, to observe a variety of state and control constraints, and to assure stability and tracking performance in the presence of modelling uncertainty. In Figure 3.1, the problem was divided into functional subsystems that perform tasks related to sensor data management, state estimation, system identification, dynamic modelling, local feedback, and trajectory control. Refer back to Chapter 3 for further discussion of the flight control objectives and the advantages of the defined autopilot architecture. As discussed earlier, this research focuses on the trajectory control subsystem of the overall autopilot.

The objectives of an AHSV flight control system are now presented more formally in terms of the variables of interest for the problem that is addressed in this thesis. The elements below characterize a general statement of the problem:

Nonlinear Multivariable Time-Varying Dynamics: *(7.1a)*

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

where, $\underline{x}(t) \in \mathbb{R}^n, \quad \underline{x}^T(t) = [V(t)\ \gamma(t)\ \omega(t)\ \theta(t)]$

$\underline{u}(t) \in \mathbb{R}^m, \quad \underline{u}^T(t) = [\delta e(t)\ \delta_T(t)\ \delta v(t)]$

$\underline{\phi}(\underline{x},t) \in \mathbb{R}^p, \quad \underline{\phi}^T(\underline{x},t) = [C_L\ C_D\ C_{Ma}\ C_{Tn}\ C_{Tv}\ C_{Mp}\ C_{X_{\delta e}} \ldots]$

are the states, controls, and parameters (aerodynamic, propulsion, and control coefficients - see section 4.7*).*

Control Actuator Constraints: *(7.1b)*

Bounds: $u_i(t)$: $\quad u_{i_{min}} \le u_i(t) \le u_{i_{max}} \quad (i=1..m)$

Rates: $\frac{d}{dt}(u_i(t))$: $\quad \left| \frac{d}{dt}(u_i(t)) \right| \le \dot{u}_{i_{max}} \quad (i=1..m)$

State Inequality Constraints: *(7.1c)*

$|V(t) - V_d(t)| \le B_V(t)$ *Velocity Tracking Tolerance*

$\lvert \gamma(t)-\gamma_{\mathrm{d}}(t) \rvert \leq B_{\gamma}(t)$	*Flight Path Tracking Tolerance*
$\lvert \omega(t)-\omega_{\mathrm{d}}(t) \rvert \leq B_{\omega}(t)$	*Pitch Rate Constraint*
$\lvert \theta(t)-\theta_{\mathrm{d}}(t) \rvert \leq B_{\theta}(t)$	*Attitude Constraint*

Other Dynamic Constraints: *(7.1d)*

$\frac{1}{2}\rho V^2(t) \leq Q_{\max}$	*Dynamic Pressure Limit*
$[\dot{V}^2+(V\dot{\gamma})^2]^{\frac{1}{2}} \leq 3\ g\text{'s}$	*Maximum Allowable Acceleration*

Tracking Of Preprogrammed Desired Trajectory: *(7.1e)*

$$\underline{x}_{\mathrm{d}}(t) = [V_{\mathrm{d}}(t)\ \gamma_{\mathrm{d}}(t)\ \omega_{\mathrm{d}}(t)\ \theta_{\mathrm{d}}(t)]^{\mathrm{T}}$$

Parametric Uncertainty: *(7.1f)*

$$\phi_{i_{\min}}(\underline{x},t) \leq \phi_i(\underline{x},t) \leq \phi_{i_{\max}}(\underline{x},t) \qquad (i=1..p)$$

Disturbances: *(7.1g)*

$\tilde{\rho}(t)$	$\tilde{T}(t)$	*Density, Temperature*
$\tilde{V}(t)$	$\tilde{\alpha}(t)$	*Velocity, Angle Of Attack*

where $\tilde{\rho}(t)$, $\tilde{T}(t)$ are due to lateral and vertical atmospheric variations, and $\tilde{V}(t)$, $\tilde{\alpha}(t)$ represent the effects of wind gusts.

The flight control objective is to design an on–line control system to track the desired trajectory (7.1e) within the tolerances specified by (7.1c) for the multivariable nonlinear dynamic system in (7.1a) while observing the constraints in (7.1b,7.1d), assuring stability in the presence of the disturbances (7.1g), and guaranteeing robustness to the interval uncertainty on the parameters in (7.1f). At this point, a complete description of the dynamic equations, the definition of a performance measure, a criterion for robust stability, and some specific assumptions, are as yet unspecified. A more specific statement of the problem addressed by the RIFC algorithm requires the clarification of these details.

The longitudinal equations of motion for an AHSV (in 7.1a) have been derived in Chapter 4. These equations represent the full rigid body translational and rotational dynamics with all uncertainty lumped into the unknown parameters (aero/prop/control coefficients). The RIFC autopilot uses these equations combined with a tabulated model of dynamic coefficients to predict the vehicle response to control inputs. These equations are, therefore, the controller's model of the dynamics. Interval uncertainty in these parameters is also stored in the database model, and this information is used to account for uncertainty in the predictions of the state trajectory. The detailed dynamic equations are given below:

For $\underline{x}(t) = [V(t)\ \gamma(t)\ \omega(t)\ \theta(t)]^T$,

$$\dot{\underline{x}}(t) = \underline{f}(\underline{x}(t),t) + \underline{g}(\underline{x}(t),\underline{u}(t),t) \tag{7.2a}$$

or,

$$\begin{aligned}\dot{V}(t) = &- \left[\frac{GM_E}{r^2} - \omega_E^2 r\right] sin(\gamma) + (\tfrac{1}{2}\rho_r V_r^2)\frac{S_{ref}}{M(t)}\Big[C_{Tv}(Ma_r,\alpha_r) - C_D(Ma_r,\alpha_r)\Big] \\ &+ (\tfrac{1}{2}\rho_r V_r^2)\frac{S_{ref}}{M(t)}\Big[C_{X_{\delta e}}(Ma_r,\alpha_r,\delta e) + C_{X_{\delta_T}}(Ma_r,\alpha_r,\delta_T) \\ &+ C_{X_{\delta v}}(Ma_r,\alpha_r,\delta v)\cdot[C_{Tv}(Ma_r,\alpha_r) + C_{X_{\delta_T}}(Ma_r,\alpha_r,\delta_T)]\Big]\end{aligned} \tag{7.2b}$$

$$\begin{aligned}\dot{\gamma}(t) = &- \frac{1}{V}\left[\frac{GM_E}{r^2} - \omega_E^2 r\right] cos(\gamma) + \left[\frac{V}{r}\right] cos(\gamma) + 2\omega_E \\ &+ (\tfrac{1}{2}\rho_r V_r^2)\left[\frac{1}{V}\right]\frac{S_{ref}}{M(t)}\Big[C_{Tn}(Ma_r,\alpha_r) + C_L(Ma_r,\alpha_r)\Big] \\ &- (\tfrac{1}{2}\rho_r V_r^2)\left[\frac{1}{V}\right]\frac{S_{ref}}{M(t)}\Big[C_{Z_{\delta e}}(Ma_r,\alpha_r,\delta e) + C_{Z_{\delta_T}}(Ma_r,\alpha_r,\delta_T) \\ &+ C_{Z_{\delta v}}(Ma_r,\alpha_r,\delta v)\cdot[C_{Z_{\delta_T}}(Ma_r,\alpha_r,\delta_T) - C_{Tn}(Ma_r,\alpha_r)]\Big]\end{aligned} \tag{7.2c}$$

$$\dot{\omega}(t) = + (\tfrac{1}{2}\rho_r V_r^2)\frac{S_{ref}\ \bar{c}}{I_y(t)}\Big[C_{Ma}(Ma_r,\alpha_r) + C_{Mp}(Ma_r,\alpha_r)\Big]$$

$$+ (\tfrac{1}{2}\rho_r V_r^2)\frac{S_{ref}\ \bar{c}}{I_y(t)}\Big[C_{M_{\delta e}}(Ma_r,\alpha_r,\delta e) + C_{M_{\delta_T}}(Ma_r,\alpha_r,\delta_T)$$

$$+ C_{M_{\delta v}}(Ma_r,\alpha_r,\delta v)\cdot[C_{Mp}(Ma_r,\alpha_r) + C_{M_{\delta_T}}(Ma_r,\alpha_r,\delta_T)]\Big] \quad (7.2d)$$

$$\dot{\theta}(t) = \omega(t) \quad (7.2e)$$

with the auxiliary equations,

$$\dot{r}(t) = V \sin(\gamma) \quad (7.2f)$$

$$\alpha(t) = \theta(t) - \gamma(t) \quad (7.2g)$$

$$Ma(t) = V/\sqrt{\gamma_a RT} \quad (7.2h)$$

(where $\gamma_a = Cp/Cv$, the ratio of the specific heats for air)

In these equations, C_L, C_D, and C_{Ma} are the lift, drag, and aerodynamic moment coefficients. Likewise, C_{Tn}, and C_{Tv} are the thrust coefficients in the normal (up) and tangential directions with respect to the velocity vector. The thrust moment coefficient is C_{Mp}. The remaining coefficients ($C_{X_{\delta e}}$, $C_{X_{\delta_T}}$, $C_{X_{\delta v}}$, $C_{Z_{\delta e}}$, $C_{Z_{\delta_T}}$, $C_{Z_{\delta v}}$, $C_{M_{\delta e}}$, $C_{M_{\delta_T}}$, $C_{M_{\delta v}}$) are the control derivatives in each direction (in wind coordinates, i.e. X–along $\underline{V}$, Z–down and $\perp$ to $\underline{V}$, M–pitch up) for the elevon (δe), throttle (δ_T), and thrust vector (δv) controls. Each one of the *15* coefficients above is a nonlinear function of the Mach number, angle of attack, and, for the control derivatives, the position of the respective actuator ($\delta e, \delta_T, \delta v$).

Equations (7.2a–h) have been carefully written to distinguish between terms that are inertial, and others that are affected by atmospheric disturbances, such as variations in density, temperature, wind velocity or direction. The subscript "*r*" refers to a relative value with respect to a nonstationary atmosphere. The terms V_r and α_r refer to the vehicle's wind relative velocity and angle of attack. The term ρ_r is local atmospheric density, which can differ from the expected density profile $\rho(r)$ both laterally and vertically (where this r = *altitude*[1]). The relative Mach number,

[1] Actually, r is the radial distance from the center of the Earth. It will also be loosely referred to as altitude.

Ma_r, is the ratio of the relative velocity V_r to the speed of sound ($Ma_r = V_r/V_s$, $V_s = \sqrt{\gamma_a RT}$), which itself may differ from its expected value for a given altitude due to temperature fluctuations. Also seen in equations (7.2a–h) is the time–varying nature of the AHSV dynamics due to the slowly changing mass and inertia terms, $M(t)$ and $I_y(t)$.

Finally, observe that the thrust vectoring control coefficients ($C_{X_{\delta v}}$, $C_{Z_{\delta v}}$, $C_{M_{\delta v}}$) complicate the dynamics by entering the equations as multipliers of both the nominal thrust coefficients (C_{Tn}, C_{Tv}, C_{Mp}) and the throttle control derivatives ($C_{X_{\delta T}}$, $C_{Z_{\delta T}}$, $C_{M_{\delta T}}$). This nonlinearity prevents the direct representation of the dynamics in the form $\dot{\underline{x}}(t) = \underline{f}(\underline{x}(t),t) + \underline{g}(\underline{x}(t),t)\cdot\underline{u}(t)$ in equation (7.2a).

The flight control problem addressed in this research has been limited to longitudinal motions only. Note that velocity, flight path angle, pitch rate, and pitch attitude are the primary state variables in equation (7.2a). Altitude $r(t)$ is also one of the states, since it appears on the right hand side of the equations. The relationship $\dot{r}(t) = V\,sin(\gamma)$ is listed as an auxiliary equation, however, since it is assumed that the desired AHSV trajectory is parameterizable in terms of altitude rather than time. That is, the flight control objective is defined to be the tracking of a desired velocity vector as a function of altitude along the trajectory from Earth to orbit.[2]

This parameterization implies that the desired velocity vector is more importantly related to the current flight condition (altitude) than to the actual time that the vehicle arrives at that point in its mission. Without this assumption, situations could arise, for example, in which the vehicle's velocity vector is perfect for the current flight condition, but the control system attempts to make drastic

[2] There are desired values for the attitude states as well, namely ω_d and θ_d, but these only represent nominal values that may not even be completely compatible with the desired velocity V_d and flight path angle γ_d. The tracking of ω_d and θ_d is a secondary objective to achieving the desired velocity vector. In fact, vehicle attitude is used as a pseudo-control to achieve translational tracking.

corrections because it is running late! In effect, parameterizing the trajectory by altitude, rather than time, simplifies the problem by removing a degree–of–freedom from the tracking objectives.[3]

Also note that many of the constraint and uncertainty bounds in (7.1a–g) are expressed as slowly time–varying quantities. In fact, these bounds might be more correctly defined as functions of the state, since they are slowly changing functions of the flight condition, which itself varies with time along a mission trajectory.

For the purposes of flight control, then, it will be assumed that the desired trajectory is parameterized as a function of altitude, and the constraint/uncertainty variations are tabulated as a function of flight condition (which includes altitude, but may also include any of the other states). In the interest of clarity, however, the time–varying representations in (7.1a–g) will remain unchanged. Instead, all time dependent bounds should be interpreted as referring to a slow variation with flight conditions along the trajectory.

There are two complications that arise with the assumptions above. The first problem is that the trajectory from Earth–to–orbit may not be a monotonic function of altitude. The second problem is that the mass and inertias for the vehicle vary more directly with time than with flight condition (as fuel is expended). Both of these issues can be addressed by assuming that the overall trajectory is managed by the guidance system which provides unambiguous short–term desired state histories along with appropriate tracking tolerances (7.1c) to the trajectory controller. It is further assumed that these tolerances account for the variation of the dynamics, and assure that the mass properties are never so far from nominal that the desired states are impossible to achieve.

A final point regarding the time–varying nature of these equations is that, for the trajectory control problem, it is assumed that the mass properties are effectively

[3] The trajectory is still determined by integrating the velocity vector and the $\dot{r}$ equation with respect to time.

constant during the short time interval over which the controller predicts ahead to evaluate candidate error–correcting trajectories.

Referring, now, back to equations (7.2a–h), note that the effects of atmospheric disturbances on the dynamics of an AHSV are embedded in the many terms of the state equations. For the analysis in this chapter, it will be useful to rewrite these equations in a form that separates the disturbances from the rest of the dynamics. This can be done, in principle, by replacing the relative terms by their expected values plus a disturbance, and bounding the resulting effect on the system using information on the magnitudes of the individual disturbances.

Substitute,

$$\rho_r = \rho + \tilde{\rho} \qquad with \mid \tilde{\rho} \mid \leq \tilde{\rho}_{\max} \tag{7.3a}$$

$$V_r = V + \tilde{V} \qquad with \mid \tilde{V} \mid \leq \tilde{V}_{\max} \tag{7.3b}$$

$$\alpha_r = \alpha + \tilde{\alpha} \qquad with \mid \tilde{\alpha} \mid \leq \tilde{\alpha}_{\max} \tag{7.3c}$$

$$Ma_r = Ma + \tilde{Ma} \qquad with \mid \tilde{Ma} \mid \leq \tilde{Ma}_{\max} \tag{7.3d}$$

(Note: $\tilde{Ma}$ is a function of $\tilde{V}$ and $\tilde{T}$)

The system can then be written in the following form:

$$\dot{\underline{x}}(t) = \underline{f}(\underline{x}(t),t) + \underline{g}(\underline{x}(t),\underline{u}(t),t) + \underline{d}(t) \tag{7.4}$$

where $\underline{d}(t)$ represents the difference between the actual dynamics (equations 7.2a-h) and the undisturbed system. That is,

$$\underline{d}(t) = \left[\dot{\underline{x}}(t)\right]_{relative} - \left[\dot{\underline{x}}(t)\right]_{nominal} \tag{7.5a}$$

and,

$$|\, d_i(t)| \leq d_{i_{\max}}(t) \tag{7.5b}$$

The task of modelling these atmospheric disturbances is beyond the scope of this thesis. Nevertheless, given bounds ($\tilde{\rho}_{\max}$, $\tilde{V}_{\max}$, $\tilde{\alpha}_{\max}$, $\tilde{Ma}_{\max}$) as a function of flight condition, it would be possible to derive the limits on $d_i(t)$ shown in (7.5b) using the equations (7.2a–h) in conjunction with tabulated or empirical models of the coefficients. In this research, a wide range of possible magnitudes for these disturbances will be arbitrarily assumed in order to test the flight control system under a variety of simulated disturbance conditions. The controller's model of the AHSV dynamics is presented below with all disturbances lumped into additive terms for each state equation.

$$\dot{V}(t) = f_{\mathrm{V}}(\underline{x},t) + g_{\mathrm{V}}(\underline{x},\underline{u},t) + d_{\mathrm{V}}(t) \tag{7.6a}$$

$$\dot{\gamma}(t) = f_{\gamma}(\underline{x},t) + g_{\gamma}(\underline{x},\underline{u},t) + d_{\gamma}(t) \tag{7.6b}$$

$$\dot{\omega}(t) = f_{\omega}(\underline{x},t) + g_{\omega}(\underline{x},\underline{u},t) + d_{\omega}(t) \tag{7.6c}$$

$$\dot{\theta}(t) = f_{\theta}(\underline{x},t) + g_{\theta}(\underline{x},\underline{u},t) + d_{\theta}(t) \tag{7.6d}$$

where,

$$f_{\mathrm{V}}(\underline{x},t) = - \left[\frac{GM_{\mathrm{E}}}{r^2} - \omega_{\mathrm{E}}^2 r\right] sin(\gamma) + (\tfrac{1}{2}\rho V^2)\frac{S_{\mathrm{ref}}}{M(t)}\left[C_{\mathrm{Tv}}(Ma,\alpha) - C_{\mathrm{D}}(Ma,\alpha)\right] \tag{7.7a}$$

$$\begin{aligned} g_{\mathrm{V}}(\underline{x},\underline{u},t) = +\, (\tfrac{1}{2}\rho V^2)\frac{S_{\mathrm{ref}}}{M(t)}\Big[& C_{\mathrm{X}_{\delta e}}(Ma,\alpha,\delta e) \\ & + C_{\mathrm{X}_{\delta_{\mathrm{T}}}}(Ma,\alpha,\delta_{\mathrm{T}}) + C_{\mathrm{X}_{\delta v}}(Ma,\alpha,\delta v)\cdot[C_{\mathrm{Tv}} + C_{\mathrm{X}_{\delta_{\mathrm{T}}}}]\Big] \end{aligned} \tag{7.7b}$$

$$\begin{aligned} f_{\gamma}(\underline{x},t) = & -\frac{1}{V}\left[\frac{GM_{\mathrm{E}}}{r^2}\omega_{\mathrm{E}}^2 r\right] cos(\gamma) + \left[\frac{V}{r}\right] cos(\gamma) + 2\omega_{\mathrm{E}} \\ & + (\tfrac{1}{2}\rho V)\frac{S_{\mathrm{ref}}}{M(t)}\left[C_{\mathrm{Tn}}(Ma,\alpha) + C_{\mathrm{L}}(Ma,\alpha)\right] \end{aligned} \tag{7.7c}$$

$$\begin{aligned} g_{\gamma}(\underline{x},\underline{u},t) = -\, (\tfrac{1}{2}\rho V)\frac{S_{\mathrm{ref}}}{M(t)}\Big[& C_{\mathrm{Z}_{\delta e}}(Ma,\alpha,\delta e) \\ & + C_{\mathrm{Z}_{\delta_{\mathrm{T}}}}(Ma,\alpha,\delta_{\mathrm{T}}) + C_{\mathrm{Z}_{\delta v}}(Ma,\alpha,\delta v)\cdot[C_{\mathrm{Z}_{\delta_{\mathrm{T}}}} - C_{\mathrm{Tn}}]\Big] \end{aligned} \tag{7.7d}$$

$$f_{\omega}(\underline{x},t) = +\, (\tfrac{1}{2}\rho V^2)\frac{S_{\mathrm{ref}}\,\bar{c}}{I_{\mathrm{y}}(t)}\left[C_{\mathrm{Ma}}(Ma,\alpha) + C_{\mathrm{Mp}}(Ma,\alpha)\right] \tag{7.7e}$$

$$\begin{aligned} g_{\omega}(\underline{x},\underline{u},t) = +\, (\tfrac{1}{2}\rho V^2)\frac{S_{\mathrm{ref}}\,\bar{c}}{I_{\mathrm{y}}(t)}\Big[& C_{\mathrm{M}_{\delta e}}(Ma,\alpha,\delta e) \\ & + C_{\mathrm{M}_{\delta_{\mathrm{T}}}}(Ma,\alpha,\delta_{\mathrm{T}}) + C_{\mathrm{M}_{\delta v}}(Ma,\alpha,\delta v)\cdot[C_{\mathrm{Mp}} + C_{\mathrm{M}_{\delta_{\mathrm{T}}}}]\Big] \end{aligned} \tag{7.7f}$$

$$f_{\theta}(\underline{x},t) = \omega(t) \tag{7.7g}$$

$$g_{\theta}(\underline{x},\underline{u},t) = 0 \tag{7.7h}$$

and as before,

$$\dot{r}(t) = V\, sin(\gamma) \tag{7.7i}$$

$$\alpha(t) = \theta(t) - \gamma(t) \tag{7.7j}$$

$$Ma(t) = V/\sqrt{\gamma_{\mathrm{a}} R T} \tag{7.7k}$$

An important element of the AHSV flight control problem that has not yet been discussed, is the definition of a specific measure of tracking performance with which the controller can evaluate the merit of candidate trajectories. Since the problem is formulated essentially as a receding horizon optimal control problem (see section 3.2) it is natural to measure performance in terms of a penalty function of the tracking error. For several reasons, a stepwise cumulative cost function is chosen of the form:

$$J = \sum_{k=1}^{N} J_k[\, \underline{x}(\underline{\phi},k\Delta t_s) - \underline{x}_d(k\Delta t_s),\ \underline{\phi}(\underline{x}),\ \underline{\Omega}_\phi(\underline{x})] \qquad (7.8)$$

where, $N\Delta t_s = t_{max}$ *is the maximum look-ahead time.*

$\underline{x}(\underline{\phi},k\Delta t_s)$, $x_d(k\Delta t_s)$ *are the actual & desired states at* $t = k\Delta t_s$.

$\underline{\phi}(\underline{x})$, $\underline{\Omega}_\phi(\underline{x})$ *are the parameter estimates and associated uncertainty intervals at the current state* $\underline{x}(k\Delta t_s)$.

The specific choice for the function J_k will be motivated in the development of section 7.5; suffice it to say here that it is a positive definite function of the tracking error along the candidate trajectory, and it includes a penalty for the uncertainty in the state predictions. Note that this performance measure is used to evaluate (and compare) candidate trajectories, and does **not** necessarily have to be globally minimized in order to find a viable control solution.

Recall from Chapter 3, that a viable solution is defined as one which is stable, observes all the constraints, and converges toward the desired trajectory. Any viable suboptimal trajectory is perfectly acceptable. Besides, the confidence in what may be the global optimal solution is diminished with future time steps due to the propagation of uncertainty through the dynamics. Furthermore, only one time step of any solution will actually be used for feedback, since a multi–step predictive trajectory is recomputed at each interval of time (Δt_s). The true tracking performance is, therefore, determined by the sequence of single step control actions taken at each time interval. This is the control feedback actually experienced by the vehicle, and it must be guaranteed to lead to a viable tracking trajectory in the near future.

Based on the discussion above, the foremost criterion for a candidate solution is, therefore, its viability in terms of stability, constraints, and convergence. Minimizing the cost function is then a secondary criterion, which selects from the set of viable solutions one that is optimal in terms of tracking performance. This distinction between the viability and optimality of a solution is important in the context of this problem, since it may be desirable to sacrifice some degree of optimality to reduce the computational burden of the optimization. This is discussed in more detail in Chapter 8.

The concept of the viability of a candidate trajectory, as discussed above, refers in part to the property of stability, which has not yet been defined in the context of this problem. Since the main objective of an AHSV flight control system is to track a desired trajectory, stability for this system will be defined in terms of the state tracking error dynamics. It is assumed that the desired trajectory is a sequence of reachable states for the vehicle.[4] The tracking errors are, therefore, deviations from these states, and tracking stability would be implied by stability of the error dynamics. The particular choice of the stability criterion is motivated in section 7.4. It is based on a discrete Lyapunov stability argument for a suitable scalar measure of the total tracking error. The Lyapunov function is a weighted norm of the individual state errors,

$$L(\underline{e}(t),\underline{\phi}) = \underset{\underline{\phi}\,\in\underline{\Omega}_{\phi}}{Max} \left\| \underline{x}(\underline{\phi},t)-\underline{x}_{\mathrm{d}}(t) \right\|_{Q} \tag{7.9}$$

and the stability criterion is given by the following inequality:

$$\underset{\underline{\phi}\,\in\underline{\Omega}_{\phi}}{Max} \left\| \underline{x}(\underline{\phi},t+\Delta t_{\mathrm{L}})-\underline{x}_{\mathrm{d}}(t+\Delta t_{\mathrm{L}}) \right\|_{Q} - \left\| \underline{x}(t)-\underline{x}_{\mathrm{d}}(t) \right\|_{Q} < 0 \tag{7.10}$$

In this equation, the time step Δt_{L} represents a discrete time interval for which it is possible to get a reduction in the Lyapunov function. In general, Δt_{L} is greater than the control cycle–time Δt_{s}. The maximization over the parameter space of the predicted tracking error at time $t+\Delta t_{\mathrm{L}}$ in equation (7.10) makes this a

[4] Constraints for the desired trajectory will be derived later to assure feasibility.

robust stability criterion in the presence of modelling uncertainty. The choice of the norm Q, the time step Δt_L, and the determination of constraints on the system to assure the existence of a solution that meets this criterion, are all issues addressed in the following sections.

The preceding discussion has clarified many of the details which define the specific problem that is addressed by the RIFC trajectory controller. The combination of the items (7.1a–g) at the beginning of this section, with the dynamic equations and disturbances defined by (7.4, 7.5b, 7.6a–d, 7.7a–k), the defining cost function for tracking performance (7.8), and the Lyapunov stability criterion (7.10), comprise a full mathematical statement of the AHSV flight control problem.

7.2 Theoretical Overview

The Robust Intelligent Flight Control algorithm is constructed in such a way as to account for all significant nonlinearities, constraints, and uncertainties present in the AHSV trajectory control problem, while guaranteeing the important properties of stability, convergence, and robustness of the control solution. This is accomplished by treating the task of trajectory tracking as a receding horizon optimal control problem and combining an A^* optimization technique with certain results from Lyapunov stability theory.[5] A Lyapunov stability criterion is chosen for which it can be shown that a solution exists (with reasonable restrictions on the system and trajectory). The optimization problem is then structured and solved in a manner that concentrates the search on the subspace of trajectories that meets the stability criterion, and guarantees convergence to a solution.

The steps below outline the logical sequence of arguments used in the following sections to justify the approach taken by the RIFC algorithm in terms of its theoretical properties:

[5] Refer back to Chapter 3 for a qualitative description of the RIFC approach.

(1) Establish controllability of the tracking error dynamics by limiting the desired trajectory and constraining the admissible state space to an envelope for which the existence of control margins can be assured.

(2) Define a discrete Lyapunov stability criterion in terms of a normed measure of tracking error, such that it can be satisfied by converging trajectories which meet the controllability conditions.

(3) Show that the A^* search algorithm converges to the optimal solution for the optimization problem defined by the constraints and bounds derived in (1), and a cost function equal to the cumulative sum of the stepwise Lyapunov measure of tracking error from (2).

(4) Augment the conditions in (1), the Lyapunov criterion in (2), the cost function in (3), and the solution criteria for the A^* search, to guarantee robustness to the presence of uncertainty in the parameters.

The analysis in the following sections serves to demonstrate that, under some reasonable assumptions and conditions, the performance of the RIFC algorithm can be guaranteed.

7.3 Controllability

Before the questions of stability or tracking performance can be addressed, it is necessary to establish that the AHSV system dynamics represented in equations (7.2a–h) are controllable from the given inputs. Since there are practical limits on the control actuators, tracking tolerances on the states, and other constraints on the system, there is the issue of physical as well as theoretical controllability. Adequate control margins must be assured in order to track, as well as converge to, a desired trajectory. Clearly this implies some limitations on how aggressive the desired trajectory can be before it is no longer feasible.

The objective of this section is to identify the bounds of a controllable envelope in the state space. This envelope defines the capabilities of the vehicle in terms of translational and rotational accelerations and, therefore, limits the tracking performance for a given desired trajectory. Due to the nonlinear nature of the equations, tabulated representations for the coefficients, and constraints on the states and controls, it is very difficult to identify the complete controllable space for the AHSV dynamics. A more conservative envelope can be found, however, based on some reasonable assumptions and by bounding many of the terms in the state equations. These assumptions make it possible to represent the dynamics in a simplified form for which an analysis of controllability can be carried out. Theoretical controllability is confirmed, and the relationships between the tracking tolerances, limits on the desired trajectory, control margins, and constraints, are derived such that controllability of the error dynamics is assured.

PROPOSITION 1: CONTROLLABILITY

Given that there exists an operational envelope for which there is sufficient control authority to achieve, at a minimum, the following nonzero bi-directional accelerations: [6]

1. $\dot{V}^{-}_{\text{marg}} < 0 < \dot{V}^{+}_{\text{marg}}$ *(acceleration tangential to the velocity vector)*
2. $\dot{\omega}^{-}_{\text{marg}} < 0 < \dot{\omega}^{+}_{\text{marg}}$ *(angular acceleration in pitch)*
3. $\dot{\gamma}^{-}_{\text{marg}} < 0 < \dot{\gamma}^{+}_{\text{marg}}$ *(acceleration normal to the velocity vector)*

then, there exist bounds,

$$| V(t)-V_{\text{d}}(t) | \leq \delta_V \qquad | \gamma(t)-\gamma_{\text{d}}(t) | \leq \delta_\gamma \tag{7.11a}$$

$$| \omega(t)-\omega_{\text{d}}(t) | \leq \delta_\omega \qquad | \theta(t)-\theta_{\text{d}}(t) | \leq \delta_\theta \tag{7.11b}$$

and constraints on the desired trajectory,

$$Min[\dot{V}_{\text{d}}(t)] < \dot{V}_{\text{d}}(t) < Max[\dot{V}_{\text{d}}(t)] \tag{7.11c}$$

$$Min[\dot{\gamma}_{\text{d}}(t)] < \dot{\gamma}_{\text{d}}(t) < Max[\dot{\gamma}_{\text{d}}(t)] \tag{7.11d}$$

[6] These accelerations are referred to as marginal values, and the subscript "marg" is used.

$$Min[\dot{\omega}_{\mathrm{d}}(t)] < \dot{\omega}_{\mathrm{d}}(t) < Max[\dot{\omega}_{\mathrm{d}}(t)] \qquad (7.11e)$$

such that, the tracking error space defined by

$$e_V(t) = V(t)\text{-}V_{\mathrm{d}}(t) \quad e_\gamma(t) = \gamma(t)\text{-}\gamma_{\mathrm{d}}(t) \qquad (7.11f)$$

$$e_\omega(t) = \omega(t)\text{-}\omega_{\mathrm{d}}(t) \quad e_\theta(t) = \theta(t)\text{-}\theta_{\mathrm{d}}(t) \qquad (7.11g)$$

is controllable.

ARGUMENT 1:

Referring back to equations (7.6, 7.7), note that the control coefficients ($C_{X_{\delta e}}, C_{X_{\delta_T}}, C_{X_{\delta v}}, C_{Z_{\delta e}}, C_{Z_{\delta_T}}, C_{Z_{\delta v}}, C_{M_{\delta e}}, C_{M_{\delta_T}}, C_{M_{\delta v}}$) are all coupled and interrelated functions of the control inputs ($\delta e, \delta_T, \delta v$). For example, the value of the moment coefficient for elevon deflection $C_{M_{\delta e}}$ cannot be selected independently of the elevon contribution to normal acceleration $C_{Z_{\delta e}}$. All nine control coefficients are determined for any choice of the three control settings. It is clear that selecting any one of the inputs to control one of the states (e.g. $\delta e \longrightarrow \omega$), has a spillover effect on the other state equations (i.e. $C_{X_{\delta e}}$ and $C_{Z_{\delta e}}$ in this case).

The assumptions of Proposition 1, however, imply that the control input space can effectively be mapped into an equivalent input space comprised of two independent control coefficients: one for translational force (δ_F) and another for rotational torque or moment (δ_M). An equivalent normal force control would also be possible, but it is assumed that angle of attack, and not direct force control, will be the primary means of achieving normal acceleration. Figure 7.1 illustrates the input space for the equivalent controls δ_F and δ_M. The rectangle bounded by F1, F2, M1, M2 represents a region in this space for which it is possible to independently select δ_F and δ_M. Note that this mapping does not limit the controller's ability to use to advantage the C_Z derivatives; it is only a device to simplify the analysis in this section. For a given δ_F and δ_M, the spill–over into the C_Z derivatives will be treated as a disturbance.[7]

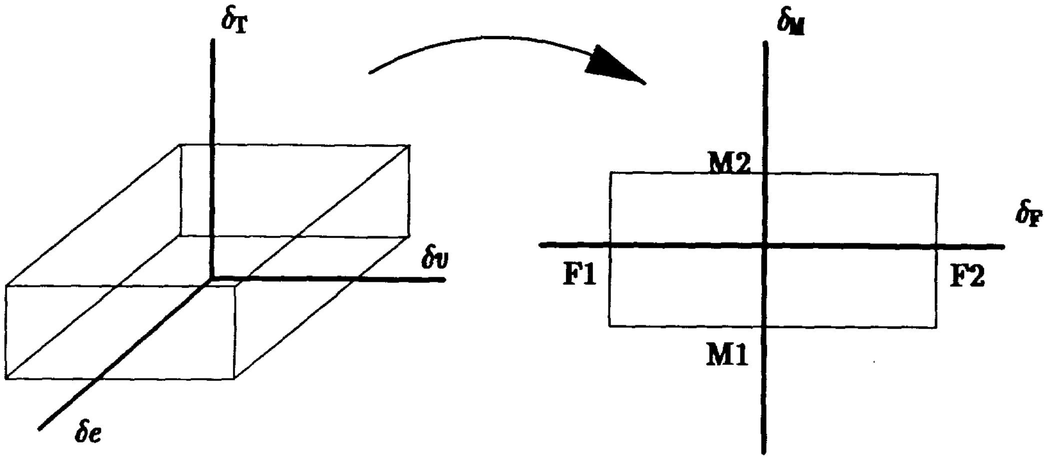


Figure 7.1: Mapping from Actuator Space to Equivalent Control Inputs

The region bounded by F1, F2, M1, M2 in Figure 7.1 is conservative, since, for example, it may be possible to have $\delta_F >$ F2 provided δ_M is less than some value below M2. The advantage of this conservatism is that it allows the treatment of δ_F and δ_M as independent equivalent–controls.

Before deriving expressions for F1, F2, M1, M2, the equations from (7.6, 7.7) are modified in order to lump all spillover effects into the disturbances. The thrust control (δ_T) is the main effector for accelerations along the velocity vector, and the elevon (δe) and thrust vector (δv) controls are primarily for rotational accelerations. The equations are simplified to reflect these properties by examining the worst case perturbations to each state equation, due to "unwanted" control actions in each axis, and adding these contributions to the disturbance terms. Note that this operation is also conservative, since it does not consider the possibility of a coupled multiaxis control effort; but again, this applies only to the analysis and does not limit the options available to the controller.

[7] For analysis, not for control.

Starting with the velocity equation,

$$\dot{V}(t) = f_V(\underline{x},t) + g_V(\underline{x},\delta e,\delta_T,\delta v,t) + d_V(t), \quad \text{with } |d_V(t)| < d_{V\max}(t) \tag{7.12a}$$

where from (7.7b),

$$\begin{aligned} g_V(\underline{x},\delta e,\delta_T,\delta v,t) = + (\tfrac{1}{2}\rho V^2)\frac{S_{\mathrm{ref}}}{M(t)}\Big[& C_{X_{\delta e}}(\underline{x},\delta e) \\ & + C_{X_{\delta_T}}(\underline{x},\delta_T) + C_{X_{\delta v}}(\underline{x},\delta v)\cdot[C_{Tv}(\underline{x}) + C_{X_{\delta_T}}(\underline{x},\delta_T)]\Big] \end{aligned} \tag{7.12b}$$

Now define,

$$g'_V(\underline{x},\delta_T,t) = (\tfrac{1}{2}\rho V^2)\frac{S_{\mathrm{ref}}}{M(t)}\Big[C_{X_{\delta_T}}(\underline{x},\delta_T)\Big] \tag{7.12c}$$

and rewrite $\dot{V}(t)$ as,

$$\dot{V}(t) = f_V(\underline{x},t) + g'_V(\underline{x},\delta_T,t) + d'_V(t) \tag{7.12d}$$

with,

$$|d'_V(t)| < d_{V\max}(t) + \underset{\underline{x}\in\underline{\Omega}_x}{Max}\ \underset{\delta_T}{Max}\ \underset{\delta e}{Max}\ \underset{\delta v}{Max}\Big|g_V(\underline{x},\delta e,\delta_T,\delta v,t) - g'_V(\underline{x},\delta_T,t)\Big| \tag{7.12e}$$

and,

$\underline{x} \in \underline{\Omega}_x$ is defined as the set of states $\underline{x}(t) = [V(t)\ \gamma(t)\ \omega(t)\ \theta(t)]^T$ inside the envelope,

$$|V(t) - V_d(t)| \le \delta_V \qquad |\gamma(t) - \gamma_d(t)| \le \delta_\gamma \tag{7.13a}$$

$$|\omega(t) - \omega_d(t)| \le \delta_\omega \qquad |\theta(t) - \theta_d(t)| \le \delta_\theta \tag{7.13b}$$

where the bounds δ_V, δ_γ, δ_ω, δ_θ have not yet been specified.

For the flight path state equation,

$$\dot{\gamma}(t) = f_\gamma(\underline{x},t) + g_\gamma(\underline{x},\delta e,\delta_T,\delta v,t) + d_\gamma(t), \quad \text{with } |d_\gamma(t)| < d_{\gamma\max}(t) \tag{7.14a}$$

where from (7.7d),

$$\begin{aligned} g_\gamma(\underline{x},\delta e,\delta_T,\delta v,t) = - (\tfrac{1}{2}\rho V)\frac{S_{\mathrm{ref}}}{M(t)}\Big[& C_{Z_{\delta e}}(\underline{x},\delta e) \\ & + C_{Z_{\delta_T}}(\underline{x},\delta_T) + C_{Z_{\delta v}}(\underline{x},\delta v)\cdot[C_{Z_{\delta_T}}(\underline{x},\delta_T) - C_{Tn}(\underline{x})]\Big] \end{aligned} \tag{7.14b}$$

Since normal acceleration is assumed to be achieved through attitude, define

$$g'_\gamma(\underline{x},t) = 0 \tag{7.14c}$$

and rewrite $\dot{\gamma}(t)$ as,

$$\dot{\gamma}(t) = f_{\gamma}(\underline{x},t) + d'_{\gamma}(t) \tag{7.14d}$$

with,

$$|d'_{\gamma}(t)| < d_{\gamma\max}(t) + \underset{\underline{x}\in\underline{\Omega}_{x}}{Max}\ \underset{\delta_{T}}{Max}\ \underset{\delta e}{Max}\ \underset{\delta v}{Max}\left|g_{\gamma}(\underline{x},\delta e,\delta_{T},\delta v,t)\right| \tag{7.14e}$$

Finally, for the pitch rate equation,

$$\dot{\omega}(t) = f_{\omega}(\underline{x},t) + g_{\omega}(\underline{x},\delta e,\delta_{T},\delta v,t) + d_{\omega}(t),\ \ with\ |d_{\omega}(t)| < d_{\omega\max}(t) \tag{7.15a}$$

where from (7.7f),

$$g_{\omega}(\underline{x},\delta e,\delta_{T},\delta v,t) = +\ (\tfrac{1}{2}\rho V^{2})\frac{S_{\text{ref}}\ \bar{c}}{I_{y}(t)}\Big[C_{M_{\delta e}}(\underline{x},\delta e)$$
$$+C_{M_{\delta_{T}}}(\underline{x},\delta_{T})+C_{M_{\delta v}}(\underline{x},\delta v)\cdot[C_{M_{P}}(\underline{x})+C_{M_{\delta_{T}}}(\underline{x},\delta_{T})]\Big] \tag{7.15b}$$

Now define,

$$g'_{\omega}(\underline{x},\delta e,\delta v,t) = (\tfrac{1}{2}\rho V^{2})\frac{S_{\text{ref}}\ \bar{c}}{I_{y}(t)}\Big[C_{M_{\delta e}}(\underline{x},\delta e) + C_{M_{\delta v}}(\underline{x},\delta v)\cdot C_{M_{P}}(\underline{x})\Big] \tag{7.15c}$$

and rewrite $\dot{\omega}(t)$ as,

$$\dot{\omega}(t) = f_{\omega}(\underline{x},t) + g'_{\omega}(\underline{x},\delta e,\delta v,t) + d'_{\omega}(t) \tag{7.15d}$$

with,

$$|d'_{\omega}(t)| < d_{\omega\max}(t) + \underset{\underline{x}\in\underline{\Omega}_{x}}{Max}\ \underset{\delta_{T}}{Max}\ \underset{\delta e}{Max}\ \underset{\delta v}{Max}\left|g_{\omega}(\underline{x},\delta e,\delta_{T},\delta v,t) - g'_{\omega}(\underline{x},\delta e,\delta v,t)\right| \tag{7.15e}$$

Since $g_{\theta}(\underline{x},\underline{u},t) = d_{\theta}(t) = 0$, the result is the following system:

$$\dot{V}(t) = f_{V}(\underline{x},t) + g'_{V}(\underline{x},\delta_{T},t) + d'_{V}(t) \tag{7.16a}$$

$$\dot{\gamma}(t) = f_{\gamma}(\underline{x},t) + d'_{\gamma}(t) \tag{7.16b}$$

$$\dot{\omega}(t) = f_{\omega}(\underline{x},t) + g'_{\omega}(\underline{x},\delta e,\delta v,t) + d'_{\omega}(t) \tag{7.16c}$$

$$\dot{\theta}(t) = f_{\theta}(\underline{x},t) \tag{7.16d}$$

Referring to (7.12c, 7.15c) the equivalent-controls can be defined by,

$$g_{1}(\underline{x},t)\ \delta_{F} \equiv g'_{V}(\underline{x},\delta_{T},t) \tag{7.17a}$$

$$g_3(\underline{x},t)\ \delta_{\mathrm{M}} \equiv g'_{\omega}(\underline{x},\delta e,\delta v,t) \tag{7.17b}$$

where,

$$g_1(\underline{x},t) = (\tfrac{1}{2}\rho V^2)\frac{S_{\mathrm{ref}}}{M(t)} \tag{7.17c}$$

$$g_3(\underline{x},t) = (\tfrac{1}{2}\rho V^2)\frac{S_{\mathrm{ref}}\ \bar{c}}{I_{\mathrm{y}}(t)} \tag{7.17d}$$

The values of F1, F2, M1, M2 from Figure 7.1 can now be obtained as follows:

$$F1 = Min\,[\delta_{\mathrm{F}}] = \underset{\underline{x}\in\underline{\Omega}_{\mathrm{x}}}{Max}\left\{\underset{\delta_{\mathrm{T}}}{Min}\left[g'_{\mathrm{V}}(\underline{x},\delta_{\mathrm{T}},t)\ /\ g_1(\underline{x},t)\right]\right\} \tag{7.18a}$$

$$F2 = Max\,[\delta_{\mathrm{F}}] = \underset{\underline{x}\in\underline{\Omega}_{\mathrm{x}}}{Min}\left\{\underset{\delta_{\mathrm{T}}}{Max}\left[g'_{\mathrm{V}}(\underline{x},\delta_{\mathrm{T}},t)\ /\ g_1(\underline{x},t)\right]\right\} \tag{7.18b}$$

$$M1 = Min\,[\delta_{\mathrm{M}}] = \underset{\underline{x}\in\underline{\Omega}_{\mathrm{x}}}{Max}\left\{\underset{\delta e}{Min}\ \underset{\delta v}{Min}\left[g'_{\omega}(\underline{x},\delta e,\delta v,t)\ /\ g_3(\underline{x},t)\right]\right\} \tag{7.18c}$$

$$M2 = Max\,[\delta_{\mathrm{M}}] = \underset{\underline{x}\in\underline{\Omega}_{\mathrm{x}}}{Min}\left\{\underset{\delta e}{Max}\ \underset{\delta v}{Max}\left[g'_{\omega}(\underline{x},\delta e,\delta v,t)\ /\ g_3(\underline{x},t)\right]\right\} \tag{7.18d}$$

Finally, making the following substitutions,

$$f_1 = f_{\mathrm{V}},\ f_2 = f_{\gamma},\ f_3 = f_{\omega},\ f_4 = f_{\theta},$$
$$d_1 = d'_{\mathrm{V}},\ d_2 = d'_{\gamma},\ d_3 = d'_{\omega}$$

equations (7.16a-d) can be written as,

$$\dot{V}(t) = f_1(\underline{x},t) + g_1(\underline{x},t)\ \delta_{\mathrm{F}} + d_1(t) \tag{7.19a}$$

$$\dot{\gamma}(t) = f_2(\underline{x},t) + d_2(t) \tag{7.19b}$$

$$\dot{\omega}(t) = f_3(\underline{x},t) + g_3(\underline{x},t)\ \delta_{\mathrm{M}} + d_3(t) \tag{7.19c}$$

$$\dot{\theta}(t) = f_4(\underline{x},t) \tag{7.19d}$$

The original assumptions of Proposition 1 can now be explicitly defined in terms of the system variables. The first assumption stated that there is always enough control authority to achieve a positive or negative acceleration tangential to the velocity vector. Referring to equation (7.19a), this means that for any value of the states in an envelope $\underline{x} \in \underline{\Omega}_{\mathrm{x}}$,

<u>Assumption 1</u>

$$\underset{\delta_F}{Max}\left[f_1(\underline{x},t) + g_1(\underline{x},t)\,\delta_F + d_1(t)\right] \geq \dot{V}^+_{marg} > 0 \qquad (7.20a)$$

$$\underset{\delta_F}{Min}\left[f_1(\underline{x},t) + g_1(\underline{x},t)\,\delta_F + d_1(t)\right] \leq \dot{V}^-_{marg} < 0 \qquad (7.20b)$$

where the acceleration margins $\dot{V}^+_{marg}$ and $\dot{V}^-_{marg}$ are defined by,

$$\dot{V}^+_{marg} = \underset{\underline{x}\in\underline{\Omega}_X}{Min}\left[f_1(\underline{x},t)\right] + \underset{\underline{x}\in\underline{\Omega}_X}{Min}\left[g_1(\underline{x},t)\right]\cdot Max[\delta_F] + Min[d_1(t)] \qquad (7.21a)$$

$$\dot{V}^-_{marg} = \underset{\underline{x}\in\underline{\Omega}_X}{Max}\left[f_1(\underline{x},t)\right] + \underset{\underline{x}\in\underline{\Omega}_X}{Min}\left[g_1(\underline{x},t)\right]\cdot Min[\delta_F] + Max[d_1(t)] \qquad (7.21b)$$

(Note: $Max[\delta_F] = F2$, $Min[\delta_F] = F1$)

Equations (7.20a–b) mean that it is always possible to accelerate or decelerate by at least $\dot{V}^+_{marg}$ or $\dot{V}^-_{marg}$ for some choice of δ_F provided the states remain in the set $\underline{x} \in \underline{\Omega}_X$.[8] Likewise, the second assumption states that there is always enough control authority to achieve a positive or negative pitching acceleration. Referring to equation (7.19c), this means that for any value of the states in an envelope $\underline{x} \in \underline{\Omega}_X$,

<u>Assumption 2</u>

$$\underset{\delta_M}{Max}\left[f_3(\underline{x},t) + g_3(\underline{x},t)\,\delta_M + d_3(t)\right] \geq \dot{\omega}^+_{marg} > 0 \qquad (7.22a)$$

$$\underset{\delta_M}{Min}\left[f_3(\underline{x},t) + g_3(\underline{x},t)\,\delta_M + d_3(t)\right] \leq \dot{\omega}^-_{marg} < 0 \qquad (7.22b)$$

where,the pitch acceleration margins $\dot{\omega}^+_{marg}$ and $\dot{\omega}^-_{marg}$ are defined by,

$$\dot{\omega}^+_{marg} = \underset{\underline{x}\in\underline{\Omega}_X}{Min}\left[f_3(\underline{x},t)\right] + \underset{\underline{x}\in\underline{\Omega}_X}{Min}\left[g_3(\underline{x},t)\right]\cdot Max[\delta_M] + Min[d_3(t)] \qquad (7.23a)$$

$$\dot{\omega}^-_{marg} = \underset{\underline{x}\in\underline{\Omega}_X}{Max}\left[f_3(\underline{x},t)\right] + \underset{\underline{x}\in\underline{\Omega}_X}{Min}\left[g_3(\underline{x},t)\right]\cdot Min[\delta_M] + Max[d_3(t)] \qquad (7.23b)$$

(Note: $Max[\delta_M] = M2$, $Min[\delta_M] = M1$)

[8] Note that the assumptions of Proposition 1 assure that there exists enough thrust control authority to swamp out the other terms with the term $g_1(\underline{x},t)\,\delta_F$ to achieve at least the marginal accelerations in each direction.

Equations (7.22a–b) mean that it is always possible to achieve a positive or negative angular acceleration of at least $\dot{\omega}^{+}_{\rm marg}$ or $\dot{\omega}^{-}_{\rm marg}$ for some choice of $\delta_{\rm M}$ provided the states remain in the set $\underline{x} \in \underline{\Omega}_{\rm x}$.

The third assumption from Proposition 1 states that there is enough control authority (within the operational envelope) to achieve a positive or negative normal acceleration. It is also assumed that upward (positive) normal accelerations correspond to angles of attack above the equilibrium value, and downward (negative) normal accelerations correspond to angles of attack below equilibrium. Furthermore, the maximum normal accelerations are experienced at the extreme values of the admissible angle of attack range. Referring to equation (7.19b) the above statements imply the following:

<u>Assumption 3</u>

$$\underset{V}{Min}\;\underset{\gamma}{Min}\;\underset{\omega}{Min}\left\{\underset{\alpha}{Max}\left[f_2(\underline{x},t)\right] + d_2(t)\right\} \geq \dot{\gamma}^{+}_{\rm marg} > 0 \qquad (7.24a)$$

$$\underset{V}{Max}\;\underset{\gamma}{Max}\;\underset{\omega}{Max}\left\{\underset{\alpha}{Min}\left[f_2(\underline{x},t)\right] + d_2(t)\right\} \leq \dot{\gamma}^{-}_{\rm marg} < 0 \qquad (7.24b)$$

where, the normal acceleration margins $\dot{\gamma}^{+}_{\rm marg}$ and $\dot{\gamma}^{-}_{\rm marg}$ are defined by,

$$\dot{\gamma}^{+}_{\rm marg} = \underset{V}{Min}\;\underset{\gamma}{Min}\;\underset{\omega}{Min}\left\{\underset{\alpha}{Max}\left[f_2(\underline{x},t)\right]\right\} + Min[d_2(t)] \qquad (7.25a)$$

$$\dot{\gamma}^{-}_{\rm marg} = \underset{V}{Max}\;\underset{\gamma}{Max}\;\underset{\omega}{Max}\left\{\underset{\alpha}{Min}\left[f_2(\underline{x},t)\right]\right\} + Max[d_2(t)] \qquad (7.25b)$$

With the system dynamics in the simplified form of equations (7.19a–d), and the assumptions expressed mathematically in equations (7.20) through (7.25), the issue of controllability can now approached.

Theoretical Controllability

Controllability is a property that identifies whether a given system can be steered from any state to any other desired state in a finite time with the available control inputs. For linear systems of the form $\dot{\underline{x}}(t) = A\underline{x}(t) + B\underline{u}(t)$ (with $\underline{x} \in \mathbb{R}^n$, $\underline{u} \in \mathbb{R}^m$, $m \leq n$), controllability is determined by examining the rank of the controllability matrix defined by

$$\underline{C} = [\, B \mid AB \mid A^2B \mid \cdots \mid A^{n-1}B] \tag{7.26}$$

If the matrix $\underline{C}$ has rank n then the system is controllable. In effect, this criterion determines if the inputs $\underline{u}$ have an influence on every state or its derivatives. For a complete discussion on linear controllability see reference [Kwa1].

Controllability for nonlinear systems is more complicated. The controllability test is based on the idea that a controllable system must be uniquely transformable into a controllable canonical form. In this form, each state is the derivative of another state, or a function of the states and inputs. Therefore, this test also determines if the inputs $\underline{u}$ have an influence on every state or its derivatives. A nonlinear controllability matrix is used, and must also have rank n for a controllable system. A full treatment of nonlinear controllability is given in references [Her1, Hun1].

In order to present the nonlinear controllability matrix, it is first necessary to define some terms from Lie bracket algebra, which are used to represent the higher derivatives of the system. Consider two vector functions $\underline{f}(\underline{x}) \in \mathbb{R}^n$ and $\underline{g}(\underline{x}) \in \mathbb{R}^n$ (as in the system $\dot{\underline{x}}(t) = \underline{f}(\underline{x}) + \underline{g}(\underline{x})\,\underline{u}$). The Lie bracket operator is defined as,

$$[\underline{f}, \underline{g}](\underline{x}) \equiv \frac{\partial \underline{g}}{\partial \underline{x}}(\underline{x})\cdot \underline{f}(\underline{x}) - \frac{\partial \underline{f}}{\partial \underline{x}}(\underline{x})\cdot \underline{g}(\underline{x}) \tag{7.27a}$$

and the higher order brackets are defined as,

$$ad(\underline{f},\, \underline{g}) = [\underline{f},\, \underline{g}] \qquad (7.27b)$$

$$ad^2(\underline{f},\, \underline{g}) = [\underline{f},\, [\underline{f},\, \underline{g}]] \qquad (7.27c)$$

$$\vdots$$

$$ad^k(\underline{f},\, \underline{g}) = [\underline{f},\, ad^{k-1}(\underline{f},\, \underline{g})] \qquad (7.27d)$$

THEOREM 7.1: Nonlinear Controllability

The multi-input nonlinear system defined by

$$\dot{\underline{x}}(t) = \underline{f}(\underline{x}) + \sum_{i=1}^{m} \underline{g}_i(\underline{x})\, u_i \qquad (7.28a)$$

is controllable if the matrix

$$\underline{C} = [\underline{g}_1, \ldots, \underline{g}_m,\, ad(\underline{f},\, \underline{g}_1), \ldots,\, ad(\underline{f},\, \underline{g}_m),\, \ldots\ldots$$
$$\ldots\ldots,\, ad^{n-1}(\underline{f},\, \underline{g}_1), \ldots, ad^{n-1}(\underline{f},\, \underline{g}_m)] \qquad (7.28b)$$

has n independent column vectors.

Before applying this theorem to the AHSV dynamic equations from (7.19a–d), it is useful to notice that the velocity equation (7.19a) can be controlled independently of the other states. Recall that,

$$\dot{V}(t) = f_1(\underline{x},t) + g_1(\underline{x},t)\, \delta_F + d_1(t) \qquad (7.29a)$$

where from (7.17c),

$$g_1(\underline{x},t) = (\tfrac{1}{2}\rho V^2) \frac{S_{ref}}{M(t)} \qquad (7.29b)$$

In atmospheric flight, $g_1(\underline{x},t)$ will never be equal to zero since ρ, V, M, and S_{ref} are all positive scalars. Also, from Assumption 1, see equations (7.20a–b), there is enough control authority in δ_F to achieve positive or negative accelerations ($\dot{V}$) for all admissible states $\underline{x} \in \underline{\Omega}_x$, disturbances $|d_1(t)| \leq d_{1max}(t)$, and any setting of the equivalent–control δ_M. Therefore, the state V is directly controllable from the equivalent–input δ_F at all times.

At this point, the velocity state will be considered controllable, and the remaining state equations will be tested for controllability using Theorem 7.1. For convenience, the equations for $\dot{\gamma}$, $\dot{\omega}$, and $\dot{\theta}$ are repeated below:

$$\dot{\gamma}(t) = f_2(\underline{x},t) + d_2(t) \tag{7.30a}$$

$$\dot{\omega}(t) = f_3(\underline{x},t) + g_3(\underline{x},t)\, \delta_{\mathrm{M}} + d_3(t) \tag{7.30b}$$

$$\dot{\theta}(t) = f_4(\underline{x},t) \tag{7.30c}$$

where from (7.7c-h),

$$f_2(\underline{x},t) = -\frac{1}{V}\left[\frac{GM_{\mathrm{E}}}{r^2} - \omega_{\mathrm{E}}^2 r\right] cos(\gamma) + \left[\frac{V}{r}\right] cos(\gamma) + 2\omega_{\mathrm{E}} + (\tfrac{1}{2}\rho V)\frac{S_{\mathrm{ref}}}{M(t)}\left[C_{\mathrm{Tn}}(Ma,\alpha) + C_{\mathrm{L}}(Ma,\alpha)\right] \tag{7.31a}$$

$$f_3(\underline{x},t) = +\,(\tfrac{1}{2}\rho V^2)\frac{S_{\mathrm{ref}}\ \bar{c}}{I_{\mathrm{y}}(t)}\left[C_{\mathrm{Ma}}(Ma,\alpha) + C_{\mathrm{Mp}}(Ma,\alpha)\right] \tag{7.31b}$$

$$g_3(\underline{x},t) = +\,(\tfrac{1}{2}\rho V^2)\frac{S_{\mathrm{ref}}\ \bar{c}}{I_{\mathrm{y}}(t)} \tag{7.31c}$$

$$f_4(\underline{x},t) = \omega(t) \tag{7.31d}$$

with, $\alpha(t) = \theta(t) - \gamma(t), \qquad Ma(t) = V/\sqrt{\gamma_{\mathrm{a}} RT}$

The nonlinear controllability matrix (7.28b) is now evaluated for these definitions of $f(\underline{x})$ and $g(\underline{x})$. [9]

$$\underline{C} = \begin{bmatrix} 0 & 0 & (\tfrac{1}{2}\rho V^2)\dfrac{2\ S_{\mathrm{ref}}^2\ \bar{c}}{M(t)\cdot I_{\mathrm{y}}(t)}\left[\dfrac{1}{V}\right]\left[\dfrac{\partial C_{\mathrm{Tn}}}{\partial\theta} + \dfrac{\partial C_{\mathrm{L}}}{\partial\theta}\right] \\ 0 & (\tfrac{1}{2}\rho V^2)\dfrac{S_{\mathrm{ref}}\ \bar{c}}{I_{\mathrm{y}}(t)} & 0 \\ (\tfrac{1}{2}\rho V^2)\dfrac{S_{\mathrm{ref}}\ \bar{c}}{I_{\mathrm{y}}(t)} & 0 & \left[(\tfrac{1}{2}\rho V^2)\dfrac{S_{\mathrm{ref}}\ \bar{c}}{I_{\mathrm{y}}(t)}\right]^2 \cdot \left[\dfrac{\partial C_{\mathrm{MP}}}{\partial\theta} + \dfrac{\partial C_{\mathrm{MA}}}{\partial\theta}\right] \end{bmatrix} \tag{7.32}$$

Since the elements $\underline{C}(3,1)$ and $\underline{C}(2,2)$ are nonzero for all time, to assure that $\underline{C}$ has full rank, it is required that,

[9] Actually, for the matrix $\underline{C}$ shown, the ordering used for the states in $f(\underline{x})$ was $[f_2(\underline{x},t)\ f_4(\underline{x},t)\ f_3(\underline{x},t)]^{\mathrm{T}}$.

$$\underline{C}(1,3) = (\tfrac{1}{2}\rho V^2)^2 \frac{S_{\mathrm{ref}}^2\ \bar{c}}{M(t)\cdot I_y(t)}\left[\frac{1}{V}\right]\left[\frac{\partial C_{\mathrm{Tn}}}{\partial\theta}+\frac{\partial C_{\mathrm{L}}}{\partial\theta}\right] \neq 0 \tag{7.33}$$

or,

$$\left[\frac{\partial C_{\mathrm{Tn}}}{\partial\theta}+\frac{\partial C_{\mathrm{L}}}{\partial\theta}\right] \neq 0 \tag{7.34}$$

Since $\theta = \gamma + \alpha$, and all derivatives with respect to γ are zero (i.e. $C_{\mathrm{Tn}} = C_{\mathrm{Tn}}(Ma,\alpha)$ and $C_{\mathrm{L}} = C_{\mathrm{L}}(Ma,\alpha)$ only), equation (7.34) becomes,

$$\left[\frac{\partial C_{\mathrm{Tn}}}{\partial\alpha}+\frac{\partial C_{\mathrm{L}}}{\partial\alpha}\right] \neq 0 \tag{7.35}$$

For hypersonic flight, these sensitivities are both nonzero and positive for all angles of attack except possibly at extreme values ($\alpha \longrightarrow 90^o$), or for large negative angles of attack where the bow shock becomes an expansion fan due to negative flow turning angles. For operational angles of attack, equation (7.35) will always hold and, therefore, the state equations (7.30a–c) are controllable from the input δ_{M}. In conclusion, the AHSV dynamic system is controllable from the equivalent controls δ_{F} and δ_{M}, which implies controllability of the original system (7.7a–h) with the assumptions given.

Qualitatively, one would have expected equations (7.30a–c) to be controllable, since from

$$\dot{\omega}(t) = f_3(\underline{x},t) + g_3(\underline{x},t)\ \delta_{\mathrm{M}} + d_3(t) \tag{7.36}$$

with,

$$g_3(\underline{x},t) = (\tfrac{1}{2}\rho V^2)\frac{S_{\mathrm{ref}}\ \bar{c}}{I_y(t)} \tag{7.37}$$

it is clear that $\omega(t)$ is controllable from δ_{M}. This is true, since g_3 is never zero, and from Assumption 2, see equations (7.22a–b), there is enough control authority in δ_{M} to achieve positive or negative pitch accelerations ($\dot{\omega}$) for all admissible states $\underline{x} \in \Omega_{\mathrm{x}}$, disturbances $|d_3(t)| \leq d_{3\mathrm{max}}(t)$, and any setting of the equivalent–control δ_{F}. Now, since $\dot{\theta}(t) = \omega(t)$, $\theta(t)$ is also controllable from δ_{M}. For the flight path equation, note that $\alpha = \theta - \gamma$, and, since $\dot{\gamma}(t)$ is a function of α, it is driven by this difference between θ and γ.

$$\dot{\gamma}(t) = f_2(\underline{x},t) + d_2(t) \tag{7.38}$$

with,

$$f_2(\underline{x},t) = \mathit{Inertial\ terms} + (\tfrac{1}{2}\rho V)\frac{S_{\mathrm{ref}}}{M(t)}\Big[C_{\mathrm{Tn}}(Ma,\underbrace{\theta\text{-}\gamma}) + C_{\mathrm{L}}(Ma,\underbrace{\theta\text{-}\gamma})\Big] \tag{7.39}$$

Differentiating $\gamma(t)$ three times with respect to time will recover the control input δ_M, so that δ_M does have an influence on all three states ω, θ, and γ. In short, pitch rate is directly controlled by δ_M, attitude (and thus angle of attack) is controllable through pitch rate, and flight path angle is controllable through the normal acceleration resulting from the vehicle's attitude.

Physical Controllability

From a theoretical point of view, controllability has been established for the AHSV dynamic equations given by (7.7a–k). This result has depended on some assumptions which made it possible to represent the control inputs in a simplified form, and which guaranteed the existence of adequate control authority for some admissible envelope of the state variables. Up to this point, however, the bounds of this controllable region have not been specified. The fact that all states cannot be controlled independently, the presence of rate limits on the actuators, and the aggressiveness of the desired trajectory, are all considerations that have not been included, and are related to the physical controllability of the system. Since the ability to not only track but to converge toward the desired trajectory is required, the objective is to establish controllability of the tracking error dynamics (rather than of the states themselves).

The tracking errors are defined by the difference between the states and their desired values.

$$\underline{e}(t) = [e_V(t)\ e_\gamma(t)\ e_\omega(t)\ e_\theta(t)]^{\mathrm{T}} \tag{7.40}$$

where,

$$e_V(t) = V(t) - V_d(t) \qquad e_\gamma(t) = \gamma(t) - \gamma_d(t) \tag{7.41a}$$

$$e_\omega(t) = \omega(t) - \omega_d(t) \qquad e_\theta(t) = \theta(t) - \theta_d(t) \tag{7.41b}$$

or,

$$\underline{e}(t) = \underline{x}(t) - \underline{x}_d(t) \tag{7.42}$$

The error dynamics are therefore defined by,

$$\underline{\dot{e}}(t) = \underline{\dot{x}}(t) - \underline{\dot{x}}_d(t) \tag{7.43}$$

which, referring to equations (7.19a–d), can be expressed as

$$\underline{\dot{e}}(t) = \underline{f}(\underline{x},t) + \underline{g}(\underline{x},t)\underline{u}(t) + \underline{d}(t) - \underline{\dot{x}}_d(t) \tag{7.44}$$

where $\underline{u}(t)$ represents the equivalent control inputs,

$$\underline{u}(t) = \begin{bmatrix} \delta_F \\ \delta_M \end{bmatrix} \tag{7.45}$$

Now, the margins[10] ($\dot{V}^{+}_{marg}$, $\dot{V}^{-}_{marg}$, $\dot{\gamma}^{+}_{marg}$, $\dot{\gamma}^{-}_{marg}$, $\dot{\omega}^{+}_{marg}$, $\dot{\omega}^{-}_{marg}$) that were defined in (7.20–7.25), represent limits on the range of achievable translational and rotational accelerations that can be guaranteed for a particular set of admissible states $\underline{x} \in \Omega_x$. These margins are needed for two reasons: tracking the desired trajectory $\underline{x}_d(t)$, and converging to it. If the desired trajectory involves accelerations that exceed the margins above, the vehicle's tracking ability cannot be guaranteed. Moreover, a second set of margins is needed to assure that control authority exists to converge to a desired trajectory that is "pulling away" at its maximum allowable rates. These new margins correspond to the excess control authority needed to reduce the tracking errors, and are essential for the controllability of the error dynamics. Conceptually, the acceleration margins are partitioned as follows:

$$\text{Acceleration Margins} = \left\{ \begin{matrix} \text{Accelerations Required By Trajectory} \\ + \\ \text{Error Rates For Tracking Convergence} \end{matrix} \right\} \tag{7.46}$$

[10] When it is not important to distinguish between positive and negative acceleration margins, the following shorthand notation will sometimes be used: $\dot{V}^{+/-}_{marg}$, $\dot{\gamma}^{+/-}_{marg}$, $\dot{\omega}^{+/-}_{marg}$.

More specifically,

$$\dot{V}^{+}_{\text{marg}} = Max[\dot{V}_{\text{d}}] + \dot{e}^{+}_{\text{Vmarg}} \qquad \dot{V}^{-}_{\text{marg}} = Min[\dot{V}_{\text{d}}] + \dot{e}^{-}_{\text{Vmarg}} \qquad (7.47a)$$

$$\dot{\gamma}^{+}_{\text{marg}} = Max[\dot{\gamma}_{\text{d}}] + \dot{e}^{+}_{\gamma_{\text{marg}}} \qquad \dot{\gamma}^{-}_{\text{marg}} = Min[\dot{\gamma}_{\text{d}}] + \dot{e}^{-}_{\gamma_{\text{marg}}} \qquad (7.47b)$$

$$\dot{\omega}^{+}_{\text{marg}} = Max[\dot{\omega}_{\text{d}}] + \dot{e}^{+}_{\omega_{\text{marg}}} \qquad \dot{\omega}^{-}_{\text{marg}} = Min[\dot{\omega}_{\text{d}}] + \dot{e}^{-}_{\omega_{\text{marg}}} \qquad (7.47c)$$

These equations indicate that there is a tradeoff between the accelerations of the desired trajectory and the ability of the system to recover from tracking errors.[11] The margins in $\dot{V}$, $\dot{\gamma}$, and $\dot{\omega}$, are dependent on the admissible state envelope $\underline{x} \in \underline{\Omega}_{\text{x}}$ (e.g. see 7.21a), and the maximum error rates depend on these margins and the properties of the desired trajectory. Clearly, equations (7.47a–c) define absolute upper bounds on the rates at which the desired states can change before it is physically impossible for the system to keep up. A *feasible* desired trajectory must, therefore, meet conditions consistent with the margins defined above,

$$Min[\dot{V}_{\text{d}}] < \dot{V}_{\text{d}}(t) < Max[\dot{V}_{\text{d}}] \qquad (7.48a)$$

$$Min[\dot{\gamma}_{\text{d}}] < \dot{\gamma}_{\text{d}}(t) < Max[\dot{\gamma}_{\text{d}}] \qquad (7.48b)$$

$$Min[\dot{\omega}_{\text{d}}] < \dot{\omega}_{\text{d}}(t) < Max[\dot{\omega}_{\text{d}}] \qquad (7.48c)$$

For controllability of the error dynamics, there must be nonzero margins for the error rates in each axis.

$$\dot{e}^{-}_{\text{Vmarg}} < 0 < \dot{e}^{+}_{\text{Vmarg}} \qquad (7.49a)$$

$$\dot{e}^{-}_{\gamma_{\text{marg}}} < 0 < \dot{e}^{+}_{\gamma_{\text{marg}}} \qquad (7.49b)$$

$$\dot{e}^{-}_{\omega_{\text{marg}}} < 0 < \dot{e}^{+}_{\omega_{\text{marg}}} \qquad (7.49c)$$

These equations do not imply that the tracking errors can all be reduced simultaneously, but that it is at least possible to influence each of the state errors in the correct direction.[12] This is achieved by defining an admissible tracking error

[11] For example, if the desired acceleration $\dot{V}_{\text{d}}$ is the same as the margin $\dot{V}^{+}_{\text{marg}}$, then a non-zero tracking error $e_{\text{V}} < 0$ $(V < V_{\text{d}})$ cannot be reduced.

[12] Attitude error-rate margins are not shown since $\theta(t)$ is controllable directly from $\omega(t)$.

envelope $\underline{x} \in \Omega_x$ in order to assure desired acceleration margins ($\dot{V}^{+}_{\mathrm{marg}}$, etc...), and then applying constraints to the desired trajectory (7.48a–c), such that the difference (7.47a–c) provides acceptable control margins for tracking performance (7.49a–c).

Recall from (7.13a–b) that $\underline{x} \in \Omega_x$ was defined as the set of states $\underline{x}(t) = [V(t)\ \gamma(t)\ \omega(t)\ \theta(t)]^{\mathrm{T}}$ such that,

$$|V(t)-V_{\mathrm{d}}(t)| \le \delta_V \qquad |\gamma(t)-\gamma_{\mathrm{d}}(t)| \le \delta_\gamma \qquad (7.50a)$$

$$|\omega(t)-\omega_{\mathrm{d}}(t)| \le \delta_\omega \qquad |\theta(t)-\theta_{\mathrm{d}}(t)| \le \delta_\theta \qquad (7.50b)$$

Now, referring back to the original statement of the problem, equations (7.1c) defined tracking tolerances on the states V, γ, ω, and θ. The first two of these constraints, $|V(t)-V_{\mathrm{d}}(t)| \le B_V(t)$ and $|\gamma(t)-\gamma_{\mathrm{d}}(t)| \le B_\gamma(t)$, limit the allowable translational tracking errors and are assumed to be given bounds.[13] The bound on pitch rate, $|\omega(t)-\omega_{\mathrm{d}}(t)| \le B_\omega(t)$, is envisioned as a given passenger comfort tolerance. The attitude constraint, $|\theta(t)-\theta_{\mathrm{d}}(t)| \le B_\theta(t)$, comes from a combination of the flight path constraint and a restricted angle of attack range due to requirements for the scramjet engine. The limits of the admissible state space $\underline{x} \in \Omega_x$ are therefore confined by,

$$\delta_V \le B_V(t) \qquad \delta_\gamma \le B_\gamma(t) \qquad (7.51a)$$

$$\delta_\omega \le B_\omega(t) \qquad \delta_\theta \le B_\theta(t) \qquad (7.51b)$$

based on *a priori* considerations. Otherwise, the bounds of the envelope $\underline{x} \in \Omega_x$ can be chosen (or constrained further) to attain desired acceleration margins. Since the primary control objective is to track a trajectory defined by the velocity and flight path angle, constraining these state errors (V, γ) any further would only make the problem more difficult. Adjusting the pitch rate constraints would have no effect on the achievable margins, since ω does not appear on the right hand side of equations (7.7a–h). Therefore, the only free parameter available which really has an effect on the acceleration margins is the attitude constraint. Furthermore, since the attitude

[13] For example, they may come from a sensitivity analysis of the results from an off-line Earth-to-orbit trajectory optimization.

constraint above is really an angle of attack constraint, and, since the aerodynamic, propulsion, and control coefficients (see 7.7a–h) are all direct functions of angle of attack (not pitch angle), it will be more convenient to work in terms of an angle of attack range, as defined below:

$$|\alpha(t)-\alpha_d(t)| \le \delta_\alpha \le B_\alpha(t), \ \ with\ \delta_\theta = \delta_\gamma + \delta_\alpha \tag{7.52}$$

From our original assumptions, there must be an admissible set of angles of attack for which acceleration margins do exist, and, clearly, given the constraints (7.51a–b), the choice of the range δ_α has a direct influence on these margins. For example, examine equation (7.21a) which is repeated below for convenience.

$$\dot{V}^{+}_{\mathrm{marg}} = \underset{\underline{x}\in\underline{\Omega}_x}{Min}\left[f_1(\underline{x},t)\right] + \underset{\underline{x}\in\underline{\Omega}_x}{Min}\left[g_1(\underline{x},t)\right]\cdot Max[\delta_{\mathrm{F}}] + Min[d_1(t)]$$

Increasing the admissible range of angle of attack, δ_α, enlarges the set $\underline{x} \in \underline{\Omega}_x$, and *decreases* the achievable marginal acceleration ($\dot{V}^{+}_{\mathrm{marg}}$) that can be guaranteed for a given equivalent control range (δ_{F}). The same property holds for the angular acceleration margins (see 7.23a–b). In sharp contrast, the normal acceleration margins are *increased* for a wider range of admissible angles of attack, as seen, for example, in the following equation (from 7.25a).

$$\dot{\gamma}^{+}_{\mathrm{marg}} = \underset{V}{Min}\ \underset{\gamma}{Min}\ \underset{\omega}{Min}\left\{\underset{\alpha}{Max}\left[f_2(\underline{x},t)\right]\right\} + Min[d_2(t)]$$

Finally, notice that the admissible range δ_α, and thus $\underline{x} \in \underline{\Omega}_x$, also limits the range of the equivalent control inputs.[14] Recall equation (7.18a),

$$F1 = Min\,[\delta_{\mathrm{F}}] = \underset{\underline{x}\in\underline{\Omega}_x}{Max}\left\{\underset{\delta_{\mathrm{T}}}{Min}\left[g'_V(\underline{x},\delta_{\mathrm{T}},t)\,/\,g_1(\underline{x},t)\right]\right\}$$

The choice of δ_α is clearly a compromise. A smaller admissible angle of attack range means more control authority (F1, F2, M1, M2) and better velocity and

[14] i. e. , the available force and torque that can be guaranteed to be independently achievable from the actuators for all $\underline{x} \in \underline{\Omega}_x$.

rotational margins ($\dot{V}^{+/-}_{\mathrm{marg}}$, $\dot{\omega}^{+/-}_{\mathrm{marg}}$). A larger admissible range means better normal acceleration margins ($\dot{\gamma}^{+/-}_{\mathrm{marg}}$) and more maneuverability for faster trajectory corrections.

The final equations which relate the tracking error–rate margins, the desired trajectory, and the admissible tracking error envelope are now presented. These equations are referred to as the physical controllability conditions, and are obtained by substituting equations (7.21a–b, 7.23a–b, 7.25a–b) into (7.47a,b,c) respectively.

Physical Controllability Conditions

$$\dot{e}^{+}_{V\mathrm{marg}} = \underset{\underline{x}\in\underline{\Omega}_x}{Min}\left[f_1(\underline{x},t)\right] + \underset{\underline{x}\in\underline{\Omega}_x}{Min}\left[g_1(\underline{x},t)\right]\cdot Max[\delta_F] + Min[d_1(t)] - Max[\dot{V}_d] \qquad (7.53a)$$

$$\dot{e}^{-}_{V\mathrm{marg}} = \underset{\underline{x}\in\underline{\Omega}_x}{Max}\left[f_1(\underline{x},t)\right] + \underset{\underline{x}\in\underline{\Omega}_x}{Min}\left[g_1(\underline{x},t)\right]\cdot Min[\delta_F] + Max[d_1(t)] - Min[\dot{V}_d] \qquad (7.53b)$$

$$\dot{e}^{+}_{\omega\mathrm{marg}} = \underset{\underline{x}\in\underline{\Omega}_x}{Min}\left[f_3(\underline{x},t)\right] + \underset{\underline{x}\in\underline{\Omega}_x}{Min}\left[g_3(\underline{x},t)\right]\cdot Max[\delta_M] + Min[d_3(t)] - Max[\dot{\omega}_d] \qquad (7.53c)$$

$$\dot{e}^{-}_{\omega\mathrm{marg}} = \underset{\underline{x}\in\underline{\Omega}_x}{Max}\left[f_3(\underline{x},t)\right] + \underset{\underline{x}\in\underline{\Omega}_x}{Min}\left[g_3(\underline{x},t)\right]\cdot Min[\delta_M] + Max[d_3(t)] - Min[\dot{\omega}_d] \qquad (7.53d)$$

$$\dot{e}^{+}_{\gamma\mathrm{marg}} = \underset{V}{Min}\;\underset{\gamma}{Min}\;\underset{\omega}{Min}\left\{\underset{\alpha}{Max}\left[f_2(\underline{x},t)\right]\right\} + Min[d_2(t)] - Max[\dot{\gamma}_d] \qquad (7.53e)$$

$$\dot{e}^{-}_{\gamma\mathrm{marg}} = \underset{V}{Max}\;\underset{\gamma}{Max}\;\underset{\omega}{Max}\left\{\underset{\alpha}{Min}\left[f_2(\underline{x},t)\right]\right\} + Max[d_2(t)] - Min[\dot{\gamma}_d] \qquad (7.53f)$$

where, $Max[\delta_F] = F2$, $Min[\delta_F] = F1$, $Max[\delta_M] = M1$, $Min[\delta_M] = M2$ *are defined in equations (7.18a-d)*

and, $\underline{x} \in \underline{\Omega}_x$ *is defined by the tolerances,*

$$|V(t) - V_d(t)| \le \delta_V = B_V(t) \qquad (7.53g)$$

$$|\gamma(t) - \gamma_d(t)| \le \delta_\gamma = B_\gamma(t) \qquad (7.53h)$$

$$|\omega(t) - \omega_d(t)| \le \delta_\omega = B_\omega(t) \qquad (7.53i)$$

$$|\theta(t) - \theta_d(t)| \le \delta_\theta \qquad (7.53j)$$

with, $\delta_\theta = \delta_\gamma + \delta_\alpha$, *for some choice of* δ_α

These equations balance the tradeoffs between the objectives of the desired trajectory and the size of tracking error envelope for which controllability can be assured. The selection of the actual values is a design problem, and is not addressed in this research. The nature of the overall desired trajectory, expected magnitude of the disturbances, sizing of the actuators, desired tracking convergence rates, and other tradeoffs would affect these choices for a particular design. In general, however, given the tolerances (7.53g–i), an admissible range of angles of attack could be chosen to guarantee control margins in terms of the achievable translational and rotational accelerations. The aggressiveness of the desired trajectory could then be balanced against the achievable tracking convergence rates (i.e. the error–rate margins). This process could be repeated using different values until a satisfactory combination is found.[15]

Given a set of tracking tolerances, an admissible angle of attack range with acceleration margins, and constraints on the desired trajectory that assure error–rate margins, most of the practical considerations that could prevent the system from being controllable have been taken into account. The only remaining concern is if the admissible tracking error envelope $\underline{x} \in \Omega_x$ can be maintained. If the bounds of this region are violated, the original assumptions may no longer be valid and the margins cannot be guaranteed.

Unachievable accelerations of the desired trajectory or physical limitations of the actuator rates may prevent the system from responding fast enough to remain inside the controllable envelope. For example, even with $\dot{V}_d < \dot{V}^+_{marg}$, if the desired acceleration changes too quickly (d^2V_d/dt^2), then a constraint of the form $d\delta_F/dt < \dot{\delta}_{F\max}$ might allow the tracking error to increase beyond the tolerance $|V(t)-V_d(t)| > \delta_V$. This problem can be avoided by limiting the accelerations of the desired trajectory, and constraining the second derivatives of the state errors in the admissible envelope. It will be assumed that actuator rate limits do not prevent the existence of a solution from admissible states (or that the controllable envelope accounts for these limits).

[15] Otherwise, the vehicle configuration or desired trajectory may require some modifications.

Another potential problem is the fact that the states (or the state tracking errors) are not independently controllable. For example, for an arbitrary (controllable) dynamic system, forcing one of the states to a desired value may require other states to temporarily diverge from their desired values. If the admissible state space is bounded, the solution might require the violation of these bounds. This does not present a problem for the dynamics of the AHSV. To demonstrate that the states can be controlled without violating the region $\underline{x} \in \underline{\Omega}_{\mathrm{x}}$, consider the following qualitative description of a control solution.

Given an initial condition with nonzero tracking errors for all states,

$$e_{\mathrm{V}}(t_0) = |\,V(t_0)\text{-}V_{\mathrm{d}}(t_0)| = e_{\mathrm{V}0}, \qquad e_{\gamma}(t_0) = |\,\gamma(t_0)\text{-}\gamma_{\mathrm{d}}(t_0)| = e_{\gamma_0} \tag{7.54a}$$

$$e_{\omega}(t_0) = |\,\omega(t_0)\text{-}\omega_{\mathrm{d}}(t_0)| = e_{\omega_0}, \qquad e_{\theta}(t_0) = |\,\theta(t_0)\text{-}\theta_{\mathrm{d}}(t_0)| = e_{\theta_0} \tag{7.54b}$$

and a feasible desired trajectory $\underline{x}_{\mathrm{d}}(t)$, for $t_0 \leq t \leq t_{\mathrm{f}}$.

- It has already been shown that the velocity state equation is independently controllable from the equivalent control δ_{F}. Therefore, the input $\delta_{\mathrm{F}}(t)$ is chosen to close the loop on the velocity error, so that $e_{\mathrm{V}}(t_0) > |\,V(t) - V_{\mathrm{d}}(t)| \longrightarrow 0.$ [16]

- For the remaining states, if the control of flight path angle (γ) is considered the priority, this can be achieved through the variation of attitude (actually angle of attack) to get desired normal accelerations ($V\dot{\gamma}$). Since attitude is dependent on the pitch rate ($\dot{\theta} = \omega$), and pitch rate is the state directly controllable from the equivalent input $\delta_{\mathrm{M}}(t)$ (7.19c), all three states are coupled, and the tracking errors $|\,\omega(t)\text{–}\omega_{\mathrm{d}}(t)|$ and $|\,\theta(t)\text{–}\theta_{\mathrm{d}}(t)|$ will not, in general, be monotonically decreasing as long as $|\,\gamma(t)\text{–}\gamma_{\mathrm{d}}(t)| \neq 0$. The tolerances from $\underline{x} \in \underline{\Omega}_{\mathrm{x}}$, however, can be easily observed. Since $\omega(t)$ is directly controlled from δ_{M}, the constraint boundary $|\,\omega(t)\text{–}\omega_{\mathrm{d}}(t)| < \delta_{\omega}$ is simply avoided. The

[16] Since $V_{\mathrm{d}}(t)$ is feasible, there is enough control authority from δ_{F} to track $V_{\mathrm{d}}(t)$, and to approach it with a converge rate of at least the error-rate margin.

only effect this has on $\theta(t)$ is that the rate at which it can change is limited. Provided that $\underline{x}_d(t)$ is feasible, and using $\omega(t)$ to control $\theta(t)$, the constraint $|\theta(t)-\theta_d(t)| < \delta_\theta$ can then also be observed. This, in turn, only affects $\gamma(t)$ in that its rate of change is limited ($\dot{\gamma}_{marg}^{+/-}$), and as long as $\underline{x}_d(t)$ is feasible, $|\gamma(t)-\gamma_d(t)| < \delta_\gamma$ can be maintained.

In summary, this section has demonstrated the controllability of the AHSV dynamics from both a theoretical and physical point of view. Some reasonable assumptions were made which made it possible to represent the inputs to the system in terms of an equivalent force and moment. Using this equivalent representation, theoretical controllability of the nonlinear system was established. A desired trajectory, constraints on the actuators, and tracking tolerances on the states, then motivated the need to examine the controllability issue from a practical (or physical) perspective. The objective was to find restrictions on the desired trajectory and the tracking–error envelope in order to assure feasibility and controllability. This was accomplished by deriving conditions for the existence of acceleration and error–rate margins for each of the state equations. The final relationships balance the aggressiveness of the desired trajectory against the tracking tolerances and the achievable tracking–error convergence rates.

This section has established conditions for the existence of control margins and a controllable tracking error envelope. The results, however, are based on some conservative bounds, and there may be better (less restrictive) conditions and larger controllable envelopes that could be found (albeit with substantially more effort).

Note, however, that these results do not restrict the controller itself in any way.[17] They are used only for the analysis and development in the following sections (where stability and convergence will depend on the existence of a solution) and are satisfactory for that purpose.

7.4 Stability

Given a controllable tracking error envelope and a feasible desired trajectory (from the previous section) it is now possible to approach the problem of finding a suitable control solution. Since stability is an essential property of the closed loop system, and since the RIFC approach searches over the control input space, it would be sensible (and advantageous) to limit the search to the set of candidate trajectories that exhibit a stabilizing behavior. In this section, a Lyapunov stability approach is used to find an achievable stability criterion that can be applied to the search as an additional constraint. The chosen Lyapunov function also turns out to be a useful measure of performance for candidate solutions in the A* optimization.

For the AHSV flight control problem, it is the stability of the tracking error dynamics that is of interest. Typically, for stability analysis, Lyapunov function candidates are tested for a particular system by examining the sufficient conditions for stability, which were presented in the theorems of Chapter 6. If the conditions are met, the system is stable. Otherwise, no determination can be made and alternative Lyapunov function candidates may be considered. Lyapunov functions can also be used to design stabilizing feedback control laws for open loop systems. This is often a trial and error process, but if a control law can be found, for which a

[17] The RIFC control algorithm searches directly in the actuator space for the best possible resulting trajectory. It does not use, or depend on, any information regarding the margins or restrictions discussed in this section. In the actual search, the physical constraints and actuator limitations must be observed. However, if the controllability conditions are not enforced, then it is still possible that the algorithm finds a solution. It would just not be guaranteed.

Lyapunov function of the closed loop system meets the stability conditions, then the design is successful.

The constrained nonlinear nature of the AHSV problem precludes the trial and error approach described above. Lyapunov stability theory can still be applied, however, to the determination of a stabilizing control solution. In Proposition 2, a general form for a Lyapunov function of the tracking errors is defined. A weighted quadratic norm of the errors is chosen for simplicity, and also because it has a physical meaning in terms of a scalar measure of the tracking error.[18] Note also that the criterion for tracking stability, in equation (7.56), is based on a discrete–time inequality. This accounts for a limited response time due to the effects of state and actuator constraints on the system.

For any choice of the weights Q_i ($Q_i > 0$) and the time interval Δt_L, if a control input function $\underline{u}(t)$ can be found that satisfies the criterion (7.56), then it is a stabilizing control law. Physically, however, such a solution may not be admissible due to violations of the constraints or tracking tolerances. Ideally, the stability criterion would be satisfied only by trajectories that converge to the desired trajectory and observe the controllability conditions of the previous section. The objective here, then, is to identify conditions on the weights Q_i and the time interval Δt_L, such that the existence of admissible solutions that meet the stability criterion is assured.

PROPOSITION 2: STABILITY

Given the following general form for a Lyapunov function of the state tracking error,

$$L(\underline{e}(t)) = \left\| \underline{e}(t) \right\|_Q = \left\| \underline{x}(t) - \underline{x}_d(t) \right\|_Q \qquad (7.55)$$

[18] This functional form is also chosen because it is suitable as a pointwise penalty function that can be accumulated over time to construct an optimality criterion.

there exists a weighting matrix $\underline{Q}$, a time interval Δt_L, and a control input function $\underline{u}(t) \in \Omega_u$ [19]*, such that the asymptotic stability criterion,*

$$\left\| \underline{x}(t+\Delta t_L)-\underline{x}_d(t+\Delta t_L) \right\|_Q - \left\| \underline{x}(t)-\underline{x}_d(t) \right\|_Q < 0 \qquad (7.56)$$

is satisfied, for all $\underline{x} \in \Omega_x$,
where,

$$\underline{x}(t+\Delta t) = \underline{x}(t) + \int_t^{t+\Delta t} \Big[f\,(\underline{x}(\tau),\tau) + \underline{g}(\underline{x}(\tau),\underline{u}(\tau),\tau) \Big]\, d\tau \qquad (7.57)$$

provided that the envelope $\underline{x} \in \Omega_x$ is controllable, and the desired trajectory is feasible, as prescribed in Proposition 1.

ARGUMENT 2:

The concepts of discrete Lyapunov stability and asymptotic stability are introduced in Chapter 6 as Definition 6.9 and 6.10. Sufficient conditions for stability are then given in Theorem 6.5, which presents Lyapunov's direct method for discrete–time autonomous systems. For convenience, this theorem is restated below in terms of the current variables:

THEOREM 7.2: Lyapunov's Stability of Discrete–Time Autonomous Systems (From Theorem 6.5)

Consider the discrete-time autonomous nonlinear system

$$\underline{e}(t+\Delta t) = \underline{f}_d(\underline{e}(t)), \quad \text{with } \underline{f}_d(\underline{0}) = \underline{0} \qquad (7.58a)$$

If, within a ball $\underline{e} \in \Omega_e$ there exists a scalar function $L(\underline{e})$ with continuous partial derivatives, and satisfying

(i) $L(\underline{e})$ *is locally positive definite* (7.58b)

(ii) $L(\underline{e}(t+\Delta t)) - L(\underline{e}(t)) \leq 0$ (7.58c)

then the equilibrium state $\underline{e} = \underline{0}$ is ***stable.***

[19] Admissible control inputs satisfy the expression $\underline{u}(t) \in \Omega_u$, where Ω_u defines the set of all achievable inputs given the control contraints from (7. 1b).

If, in addition,

(iii) $$L(\underline{e}(t+\Delta t)) - L(\underline{e}(t)) < 0 \qquad (7.58d)$$

then the equilibrium is ***asymptotically stable.***

Note that Theorem 7.2 provides sufficient conditions for the stability of an autonomous discrete–time dynamic system with an equilibrium at the origin, and no external inputs. Nevertheless, it is applicable to the control problem of tracking a desired trajectory for the AHSV for the following reasons:

(1) The time dependence of the AHSV dynamic equations is due to the variation in vehicle mass $M(t)$ and inertia $I_y(t)$ along a trajectory from Earth-to-orbit. On the short time scale of trajectory tracking control, this time variation is very small and will be neglected by the controller.

(2) The desired trajectory is assumed to be a solution to the state equations for some nominal values of the control inputs,

$$\dot{\underline{x}}_d(t) = \underline{f}(\underline{x}_d(t)) + \underline{g}(\underline{x}_d(t),\underline{u}_d(t)) \qquad (7.59)$$

therefore, the tracking error dynamics, defined by

$$\begin{aligned}\dot{\underline{e}}(t) &= \dot{\underline{x}}(t) - \dot{\underline{x}}_d(t)\\ &= [\underline{f}(\underline{x}(t)) + \underline{g}(\underline{x}(t),\underline{u}(t))] - [\underline{f}(\underline{x}_d(t)) + \underline{g}(\underline{x}_d(t),\underline{u}_d(t))] \qquad (7.60)\end{aligned}$$

has equilibrium states ($\dot{\underline{e}}(t) = \underline{0}$) wherever $\underline{x}(t) = \underline{x}_d(t)$ for an appropriate choice of the controls. Since $\underline{e}(t) = \underline{x}(t) - \underline{x}_d(t)$, this corresponds to the origin of the error space ($\underline{e}(t) = \underline{0}$).[20]

(3) The error dynamics for the tracking problem can be represented in discrete form by,

[20] The fact that the desired trajectory $\underline{x}_d(t)$ is time-varying does not preclude the use of the autonomous form of the Lyapunov stability theorem. The system itself is time-invariant according to item (1), and $\underline{x}_d(t)$ is just a known reference command input. It could, for example, be generated as an output of another time-invariant system.

$$\underline{e}(t+\Delta t) = \underline{x}(t+\Delta t) - \underline{x}_{\mathrm{d}}(t+\Delta t)$$

or, $\underline{e}(t+\Delta t) = \mathcal{F}(\underline{x}(t),\underline{u}(\tau),\underline{x}_{\mathrm{d}}(t+\Delta t))$, where $\tau \in [t,t+\Delta t]$, *(7.61)*

and the function $\mathcal{F}(\cdot)$ includes the integral (7.57).

Expressing $\mathcal{F}(\cdot)$ in terms of the tracking error,

$$\underline{e}(t+\Delta t) = \mathcal{F}(\underline{e}(t)+\underline{x}_{\mathrm{d}}(t),\underline{u}(\tau),\underline{x}_{\mathrm{d}}(t+\Delta t)) \qquad (7.62)$$

and since the desired trajectory is a given quantity, the discrete error dynamics can be written as in (7.58a),

$$\underline{e}(t+\Delta t) = f_{\mathrm{d}}(\underline{e}(t))$$

for any particular choice of the input function $\underline{u}(t)$. Theorem 7.2 can therefore be used as a sufficiency test to determine if a candidate control solution is asymptotically stabilizing for the system.[21]

The ball $\underline{e} \in \underline{\Omega}_{\mathrm{e}}$ in Theorem 7.2 is the same tracking error envelope defined earlier by $\underline{x} \in \underline{\Omega}_{\mathrm{x}}$: $\underline{\Omega}_{\mathrm{x}}$ is the set of admissible states (V, γ, ω, θ), and $\underline{\Omega}_{\mathrm{e}}$ is the set of admissible errors (e_V, e_γ, e_ω, e_θ), for which the conditions of (7.13a–b) are met (i.e. $|V(t)-V_{\mathrm{d}}(t)| \leq \delta_V$, etc.).

In order to verify Proposition 2, it is clear from Theorem 7.2 that it must be shown that the Lyapunov function can be chosen (Q, Δt_{L}), such that it is a locally positive definite function of the tracking error (7.58b), and that the inequality, $L(\underline{e}(t+\Delta t_{\mathrm{L}}))-L(\underline{e}(t)) < 0$, holds for some admissible control input function $\underline{u}(t) \in \underline{\Omega}_{\mathrm{u}}$ (7.58d).

The general form of the Lyapunov function (7.55) can be expressed equivalently as,

$$L(\underline{e}(t)) = [\underline{x}(t)-\underline{x}_{\mathrm{d}}(t)]^{\mathrm{T}}\, \underline{Q}\, [\underline{x}(t)-\underline{x}_{\mathrm{d}}(t)] \qquad (7.63)$$

[21] The system is well behaved between sampling times as long as the time-step Δt is small enough to capture the dominant modes of the system. For longitudinal hypersonic flight the time constant for the short period mode is typically of the order of several seconds or larger.

where, the matrix $\underline{Q}$ is taken to be a diagonal weighting matrix,

$$\underline{Q} = \begin{bmatrix} Q_1 & & & \\ & Q_2 & & \\ & & Q_3 & \\ & & & Q_4 \end{bmatrix}, \text{ with } Q_i > 0 \tag{7.64}$$

or alternatively,

$$\begin{aligned} L(\underline{e}(t)) = (V(t)-V_d(t))^2\, Q_1 + (\gamma(t)-\gamma_d(t))^2\, Q_2 \\ + (\omega(t)-\omega_d(t))^2\, Q_3 + (\theta(t)-\theta_d(t))^2\, Q_4 \end{aligned} \tag{7.65}$$

Referring to Definition 6.8, the function $L(\cdot)$ is positive definite for weighting matrices $\underline{Q}$ that are positive definite. A diagonal matrix of positive values clearly meets this condition.

The remainder of this section is devoted to showing that the weights Q_i, and the time interval Δt_L, can be chosen for the Lyapunov function L, such that the stability criterion $L(\underline{e}(t+\Delta t_L))-L(\underline{e}(t)) < 0$ can be met by a solution that observes the bounds of the controllability envelope $\underline{x} \in \underline{\Omega}_x$ and the constraints on the controls $\underline{u} \in \underline{\Omega}_u$.

Choosing The Weighting Matrix $\underline{Q}$

It is assumed that the desired trajectory is constrained to be feasible, and that a suitable tracking error envelope $\underline{x} \in \underline{\Omega}_x$ has been defined to provide adequate acceleration and error–rate margins according to the relationships derived in section 7.3.

Now, the stability criterion defined by equation (7.56) is valid for any choice of the weights Q_i, provided $Q_i > 0$. If the criterion can be met by a particular control input function, then the solution is asymptotically stable. The reason that the weights cannot be arbitrary positive values, is that a stable tracking solution must also be constrained to remain within the bounds of the controllability envelope $x \in \Omega_x$. A solution that meets this constraint, and also satisfies the stability criterion, cannot be guaranteed for all values of Q_i.

The choice of these weights determines the relative change in the Lyapunov function for a decrease in any of the state errors. Recall (from section 7.3), however, that the tracking errors cannot, in general, be reduced simultaneously, due to the coupling of the state equations. It is easy to see that a poor choice for the weights might cause a conflict. For instance, consider a case where Q_4 is much greater than Q_1, Q_2, and Q_3. In this example, the stability criterion would demand that the attitude error be reduced, even if the velocity or flight path error was diverging beyond acceptable limits.

The relationship between the stability criterion and the controllability conditions can be visualized as shown in Figure 7.2. This figure presents a two dimensional analog of the four dimensional situation. For any initial condition *e(t)* in the tracking error space, there is a region of accessible error–states R_A that can be reached after a time interval Δt_L. The controllability envelope $x \in \Omega_x$ defines another region denoted by R_C. Finally, there is a region R_Q that includes the set of error states, such that the transition from *e(t)* to this set in Δt_L would meet the stability criterion for a particular choice of the weights Q_i. The intersection of these regions, $R_S \equiv R_A \cap R_C \cap R_Q$, defines the largest possible set of admissible stabilizing trajectories that can be identified for a given Lyapunov function.

Since the constraints on the envelope $\underline{x} \in \underline{\Omega}_x$ and the desired trajectory have been chosen to assure controllability of the error dynamics, the regions R_A and R_C are known to overlap. Since the stability envelope, defined by the region R_Q, is a parameterized function of the weights, the size of the intersection of the three regions (R_S) depends on the values of the Q_i's.

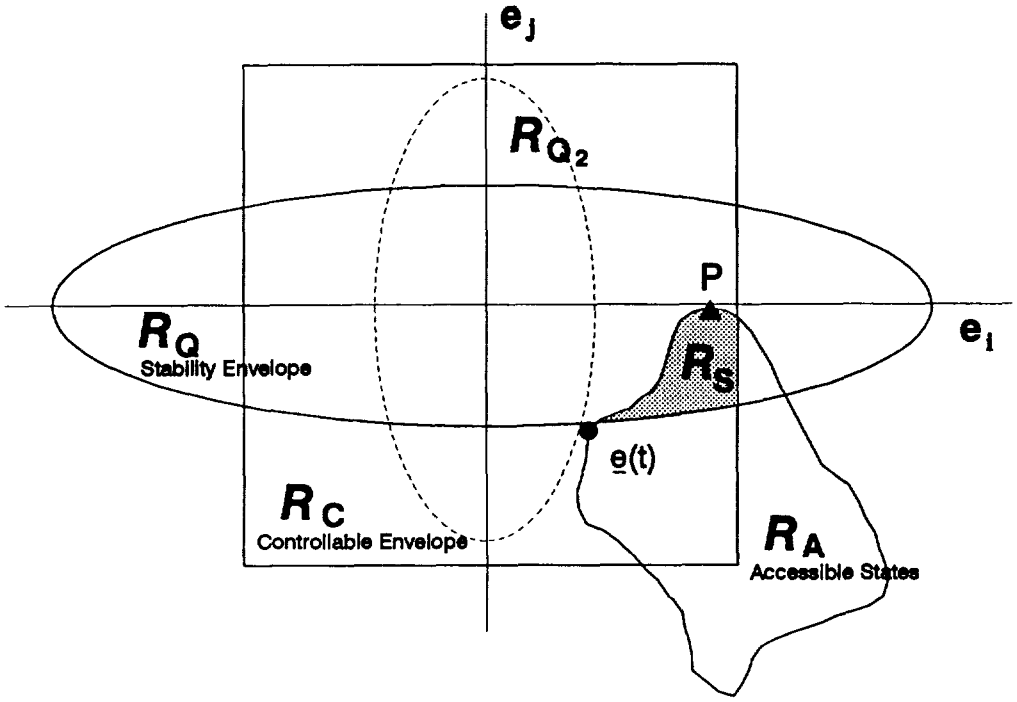


Figure 7.2: The Relationship Between Accessible States, the Controllability Envelope, and the Lyapunov Stability Criterion in the Error Space.

In the figure, the region R_{Q2} represents the stability envelope for a different set of weights Q_i that does not include any of the accessible states in Δt_L. The state marked P, for example, could be along a stabilizing trajectory with smaller tracking errors in the velocity vector. It is rejected by R_{Q2}, however, because the attitude errors (though admissible) are weighted too heavily by the Lyapunov function. Clearly, it would be desirable to choose the weights in such a way that the set R_S is as large as possible. It is enough, however, to constrain the Q_i's such that admissible solutions also satisfies the stability criterion.

In order to simplify the notation, let $\Delta L(\Delta t_L) = L(\underline{e}(t+\Delta t_L))-L(\underline{e}(t))$, so that the stability criterion can be written as, $\Delta L(\Delta t_L) < 0$. Also, from (7.65), define, $L_1(t) \equiv (V(t)-V_d(t))^2 Q_1 = e_V^2(t)Q_1$, and similarly, $L_2(t) \equiv e_\gamma^2(t)Q_2$, $L_3(t) \equiv e_\omega^2(t)Q_3$, $L_4(t) \equiv e_\omega^2(t)\ Q_4$. Finally, represent $\Delta L(\Delta t_L)$ as,

$$\Delta L(\Delta t_L) = \Delta L_1(\Delta t_L) + \Delta L_2(\Delta t_L) + \Delta L_3(\Delta t_L) + \Delta L_4(\Delta t_L) \qquad (7.66a)$$

where,

$$\Delta L_1(\Delta t_L) = [e_V^2(t+\Delta t_L) - e_V^2(t)]\ Q_1 \qquad (7.66b)$$

$$\Delta L_2(\Delta t_L) = [e_\gamma^2(t+\Delta t_L) - e_\gamma^2(t)]\ Q_2 \qquad (7.66c)$$

$$\Delta L_3(\Delta t_L) = [e_\omega^2(t+\Delta t_L) - e_\omega^2(t)]\ Q_3 \qquad (7.66d)$$

$$\Delta L_4(\Delta t_L) = [e_\theta^2(t+\Delta t_L) - e_\theta^2(t)]\ Q_4 \qquad (7.66e)$$

Now, the primary objective of the trajectory control system for the AHSV is to track the desired translational velocity vector (i.e. the velocity and flight path angle) from Earth to orbit. In fact, the rotational states, attitude and pitch rate, are expected to deviate from their nominal values in order to achieve normal accelerations required for corrections to the flight path.[22] Given a target envelope for the velocity and flight path tracking errors,

$$|\,V(t) - V_d(t)| < \lambda_V \qquad |\,\gamma(t) - \gamma_d(t)| < \lambda_\gamma \qquad (7.67)$$

and initial conditions outside this range $(|\,e_V(t_0)| > \lambda_V$ *and/or* $|\,e_\gamma(t_0)| > \lambda_\gamma)$, an admissible stabilizing solution can be characterized as a trajectory that converges in the translational tracking errors, while maintaining the attitude and pitch rate tolerances specified by the controllability conditions. Once the velocity vector is within the target range, then the tracking of a desired attitude is a secondary objective.

Constraints on the weightings in the Lyapunov function are obtained by requiring that the criterion for asymptotic stability, $\Delta L(\Delta t) < 0$, can be met in each of four cases of interest.

(1)	$\|\,e_V(t_0)\| > \lambda_V$, $\|\,e_\gamma(t_0)\| < \lambda_\gamma$	Velocity Error Off Target
(2)	$\|\,e_V(t_0)\| < \lambda_V$, $\|\,e_\gamma(t_0)\| > \lambda_\gamma$	Flight Path Error Off Target

[22] The use of attitude to control flight path was discussed in section 7. 3.

(3) $|e_V(t_0)| > \lambda_V, \quad |e_\gamma(t_0)| > \lambda_\gamma$ Both (e_V & e_γ) Off Target

(4) $|e_V(t_0)| < \lambda_V, \quad |e_\gamma(t_0)| < \lambda_\gamma$ Both Errors Within Target

The fourth case is not as important, since it is the condition where the vehicle's trajectory (velocity vector) is within the target range of the desired values.

Case 1: Velocity Error — $|e_V(t_0)| > \lambda_V$

In this case, flight path tracking error is within its target range, and the attitude errors are confined to the envelope $\underline{e} \in \underline{\Omega}_e$. That is,

$$|e_\gamma(t)| \le \lambda_\gamma, \quad |e_\omega(t)| \le \delta_\omega, \quad |e_\theta(t)| \le \delta_\theta \tag{7.68}$$

Beginning with,

$$\Delta L(\Delta t_L) = \Delta L_1(\Delta t_L) + \Delta L_2(\Delta t_L) + \Delta L_3(\Delta t_L) + \Delta L_4(\Delta t_L)$$

the terms $\Delta L_2(\Delta t_L)$, $\Delta L_3(\Delta t_L)$, $\Delta L_4(\Delta t_L)$ can be bounded as follows:

Since,

$$\Delta L_2(\Delta t_L) = [e_\gamma^2(t+\Delta t_L) - e_\gamma^2(t)]\ Q_2$$

the maximum difference $[e_\gamma^2(t+\Delta t_L) - e_\gamma^2(t)]$ would be obtained if $e_\gamma(t) = 0$, and $|e_\gamma(t+\Delta t_L)| = \lambda_\gamma$. Therefore,

$$\Delta L_2(\Delta t_L) \le \lambda_\gamma^2\ Q_2 \tag{7.69}$$

and likewise,

$$\Delta L_3(\Delta t_L) \le \delta_\omega^2\ Q_3 \tag{7.70}$$

$$\Delta L_4(\Delta t_L) \le \delta_\theta^2\ Q_4 \tag{7.71}$$

In order to meet the condition $\Delta L(\Delta t_L) < 0$, the following inequality is required:

$$\Delta L_1(\Delta t_L) < - Max[\Delta L_2(\Delta t_L) + \Delta L_3(\Delta t_L) + \Delta L_4(\Delta t_L)] \tag{7.72}$$

or,

$$\Delta L_1(\Delta t_L) < - [\lambda_\gamma^2\ Q_2 + \delta_\omega^2\ Q_3 + \delta_\theta^2\ Q_4] \tag{7.73}$$

Now,

$$\Delta L_1(\Delta t_L) = [e_V^2(t+\Delta t_L) - e_V^2(t)]\ Q_1$$
$$= [e_V(t+\Delta t_L) + e_V(t)][e_V(t+\Delta t_L) - e_V(t)]\ Q_1 \qquad (7.74)$$

and given achievable error-rate margins in velocity (see (7.53a-b)), $\dot{e}^+_{\mathrm{Vmarg}}$ and $\dot{e}^-_{\mathrm{Vmarg}}$, the *maximum* achievable decrease in velocity error that can be assured for the time interval Δt_L is given by,

$$|\, e_V(t+\Delta t_L) - e_V(t) | = (\dot{e}_{\mathrm{Vmarg}}) \cdot \Delta t_L \equiv \kappa_V, \qquad (7.75)$$

where, for simplicity the error-rate margins have been assumed to be symmetric ($\dot{e}_{\mathrm{Vmarg}} = \dot{e}^+_{\mathrm{Vmarg}} = |\, \dot{e}^-_{\mathrm{Vmarg}} |$). Equation (7.74) can therefore be written as follows:

$$\Delta L_1(\Delta t_L) = [e_V(t+\Delta t_L) + e_V(t)][-\ \kappa_V\ ]\ Q_1 \qquad \text{for } e_V(t) > 0 \qquad (7.76a)$$
$$\Delta L_1(\Delta t_L) = [e_V(t+\Delta t_L) + e_V(t)][+\ \kappa_V\ ]\ Q_1 \qquad \text{for } e_V(t) < 0 \qquad (7.76b)$$

Since, in either case $e_V(t)$ and $e_V(t+\Delta t_L)$ can be assumed to have the same sign,[23]

$$\Delta L_1(\Delta t_L) < -\ \kappa_V\ |\, e_V(t) |\ Q_1 \qquad (7.77)$$

Comparing (7.73) and (7.77), for $\Delta L(\Delta t_L) < 0$, it is required that,

$$\kappa_V\ |\ e_V(t)\ |\ Q_1\ >\ [\lambda_\gamma^2\ Q_2 + \delta_\omega^2\ Q_3 + \delta_\theta^2\ Q_4] \qquad (7.78)$$

and finally, since the stability condition is desired for values of $|\, e_V(t) | > \lambda_V$,

$$\kappa_V\ \lambda_V\ Q_1\ >\ [\lambda_\gamma^2\ Q_2 + \delta_\omega^2\ Q_3 + \delta_\theta^2\ Q_4] \qquad (7.79)$$

Equation (7.79) is the first constraining equation for the weights in the Lyapunov function. It assures that, under the conditions of case 1, the Lyapunov function can be made to decrease by decreasing the velocity tracking error. Since

[23] For $|\, e_V(t) | > \lambda_V$ there is no need to consider cases where $e_V(t+\Delta t_L)$ is of opposite sign to $e_V(t)$. Recall that the velocity state is directly controllable through the equivalent input δ_F, and also that the search space includes all admissible input functions. If $e_V(t)$ can be made to switch signs in Δt_L, then an input function must also exist which would bring it to zero, and this would be the preferred solution.

velocity is directly controllable from the equivalent control δ_F (section 7.3), a trajectory satisfying the stability criterion exists (for $\underline{x} \in \underline{\Omega}_x$).

Also, note that from (7.79) and (7.73)

$$\Delta L(\Delta t_L) < -\kappa_V \mid e_V(t) \mid Q_1 + [\lambda_\gamma^2 Q_2 + \delta_\omega^2 Q_3 + \delta_\theta^2 Q_4] \qquad (7.80)$$

The second term is dominated by the first, and the change in the Lyapunov function is negative $\Delta L(\Delta t_L) < 0$ as desired. The conditions for asymptotic stability are therefore satisfied for case 1.

Case 2: Flight Path Error – $|e_\gamma(t_0)| > \lambda_\gamma$

This case is treated in a similar manner to case 1. Here, the velocity tracking error is within its target range, and the attitude errors are confined to the envelope $\underline{e} \in \underline{\Omega}_e$. That is,

$$|e_V(t)| \leq \lambda_V, \quad |e_\omega(t)| \leq \delta_\omega, \quad |e_\theta(t)| \leq \delta_\theta \qquad (7.81)$$

From,

$$\Delta L(\Delta t_L) = \Delta L_1(\Delta t_L) + \Delta L_2(\Delta t_L) + \Delta L_3(\Delta t_L) + \Delta L_4(\Delta t_L)$$

this time, it is the terms $\Delta L_1(\Delta t_L)$, $\Delta L_3(\Delta t_L)$, $\Delta L_4(\Delta t_L)$ that can be bounded:

Since,

$$\Delta L_1(\Delta t_L) = [e_V^2(t+\Delta t_L) - e_V^2(t)]\, Q_1$$

the maximum difference $[e_V^2(t+\Delta t_L) - e_V^2(t)]$ would be obtained if $e_V(t) = 0$, and $|e_V(t+\Delta t_L)| = \lambda_V$. However, recall from section 7.3 that the velocity state can be independently controlled (from δ_F) for $\underline{x} \in \underline{\Omega}_x$. Therefore, the contribution from $\Delta L_1(\Delta t_L)$ can be assumed to be nonpositive, thus helping to reduce $\Delta L(\Delta t_L)$. Therefore,

$$\Delta L_1(\Delta t_L) \leq 0 \qquad (7.82)$$

and again,

$$\Delta L_3(\Delta t_L) \leq \delta_\omega^2 Q_3 \tag{7.83}$$

$$\Delta L_4(\Delta t_L) \leq \delta_\theta^2 Q_4 \tag{7.84}$$

For $\Delta L(\Delta t) < 0$, we require,

$$\Delta L_2(\Delta t_L) < -\, Max[\Delta L_1(\Delta t_L) + \Delta L_3(\Delta t_L) + \Delta L_4(\Delta t_L)] \tag{7.85}$$

or,

$$\Delta L_2(\Delta t_L) < -\, [\delta_\omega^2 Q_3 + \delta_\theta^2 Q_4] \tag{7.86}$$

Now,

$$\begin{aligned}\Delta L_2(\Delta t_L) &= [e_\gamma^2(t+\Delta t_L) - e_\gamma^2(t)]\; Q_2 \\ &= [e_\gamma(t+\Delta t_L) + e_\gamma(t)][e_\gamma(t+\Delta t_L) - e_\gamma(t)]\; Q_2\end{aligned} \tag{7.87}$$

At this point, there is a difference between case 2 and case 1, since the achievable error-rate margins for flight path (see (7.53e-f)), $\dot{e}^{+}_{\gamma_{\mathrm{marg}}}$ and $\dot{e}^{-}_{\gamma_{\mathrm{marg}}}$, cannot be guaranteed for the entire interval Δt_L. The constraints on the time interval (Δt_L) used for the stability criterion will be derived later in this section, but it must be a longer interval than the time required to reorient the vehicle to get maximum normal acceleration. It is assumed, here, that Δt_L is large enough so that the error-rate margins above are achievable for some nonzero time interval Δt_c.[24] Therefore, the maximum decrease in flight path error that can be assured for the time interval Δt_L is given by,

$$|\, e_\gamma(t+\Delta t_L) - e_\gamma(t)| = (\dot{e}_{\gamma_{\mathrm{marg}}}) \cdot \Delta t_c \equiv \kappa_\gamma \tag{7.88}$$

and again, the error-rate margins have been assumed to be symmetric. Equation (7.87) can therefore be written as follows:

$$\Delta L_2(\Delta t_L) = [e_\gamma(t+\Delta t_L) + e_\gamma(t)][-\,\kappa_\gamma\,]\; Q_2 \qquad for\; e_\gamma(t) > 0 \tag{7.89a}$$

$$\Delta L_2(\Delta t_L) = [e_\gamma(t+\Delta t_L) + e_\gamma(t)][+\,\kappa_\gamma\,]\; Q_2 \qquad for\; e_\gamma(t) < 0 \tag{7.89b}$$

[24] The time interval Δt_c is the last sub-interval of Δt_L, where Δt_L is long enough to account for actuator rate limits, and reorientation of the vehicle.

As in case 1, this can be reduced to,

$$\Delta L_2(\Delta t_L) < -\kappa_\gamma \, |e_\gamma(t)| \, Q_2 \tag{7.90}$$

Comparing (7.86) and (7.90), for $\Delta L(\Delta t_L) < 0$, it is required that,

$$\kappa_\gamma \, | \, e_\gamma(t) \, | \, Q_2 > [\delta_\omega^2 \, Q_3 + \delta_\theta^2 \, Q_4] \tag{7.91}$$

and finally, since the stability condition is desired for values of $|e_\gamma(t)| > \lambda_\gamma$,

$$\kappa_\gamma \lambda_\gamma \, Q_2 > [\, \delta_\omega^2 \, Q_3 + \delta_\theta^2 \, Q_4] \tag{7.92}$$

The inequality of equation (7.92) constrains the weights of the Lyapunov function. It assures that the Lyapunov function can be made to decrease by a trajectory which converges in the flight path error (when this error is outside its target range, and the velocity error is on target). Since controllability and the error-rate margins for flight path are guaranteed for the envelope $\underline{x} \in \underline{\Omega}_x$, the stability criterion can be satisfied.

Also, from (7.92) and (7.86)

$$\Delta L(\Delta t_L) < -\kappa_\gamma \, | \, e_\gamma(t) \, | \, Q_2 + [\delta_\omega^2 \, Q_3 + \delta_\theta^2 \, Q_4] \tag{7.93}$$

The second term is dominated by the first, and the change in the Lyapunov function is negative $\Delta L(\Delta t_L) < 0$ as desired. The conditions for asymptotic stability are therefore satisfied for case 2.

Case 3: Velocity & Flight Path Error – $|e_V(t_0)| > \lambda_V$, $|e_\gamma(t_0)| > \lambda_\gamma$

For this case, the analysis from the two previous cases can be combined using an analogous procedure. The results are given below without the details.

The constraining equation for the weights becomes,

$$\kappa_V \lambda_V \, Q_1 + \kappa_\gamma \lambda_\gamma \, Q_2 > [\delta_\omega^2 \, Q_3 + \delta_\theta^2 \, Q_4] \tag{7.94}$$

with,

$$\Delta L(\Delta t_L) < -\kappa_V \, |e_V(t)| \; Q_1 - \kappa_\gamma \, |e_\gamma(t)| \; Q_2 + [\delta_\omega^2 \, Q_3 + \delta_\theta^2 \, Q_4] \tag{7.95}$$

and the first two terms dominate the last, so that the change in the Lyapunov function is negative $\Delta L(\Delta t_L) < 0$, and the conditions for asymptotic stability are met.

Case 4: Velocity Vector On Target – $|e_V(t_0)| < \lambda_V, \; |e_\gamma(t_0)| < \lambda_\gamma$

In this case, the flight control objectives have been met, and the secondary objective is to stabilize the tracking error between the attitude and its nominal value. Here too, the stability criterion must account for the constraints on the controllability envelope $\underline{x} \in \underline{\Omega}_x$. Since attitude is affected through pitch rate, which is controlled directly by the equivalent input δ_M, the tracking of attitude should take precedence over pitch rate control (to avoid attitude constraint violations). Given a target range for the attitude error,

$$|\theta(t) - \theta_d(t)| < \lambda_\theta \tag{7.96}$$

it is desirable to choose the stability criterion such that it can be satisfied by trajectories which convergence in attitude error for the case $|e_\theta(t)| > \lambda_\theta$ (when $|e_V(t)| < \lambda_V$ and $|e_\gamma(t)| < \lambda_\gamma$).

Rigorous conditions for the weights are difficult to derive in this case since the velocity and flight path errors cannot be assumed to remain within their target ranges. However, since attitude tracking is of secondary importance, the following weaker condition is used to recognize the fact that attitude is a priority when compared to pitch rate:

$$\lambda_\theta^2 \, Q_4 > \delta_\omega^2 \, Q_3 \tag{7.97}$$

For sufficiently small velocity and flight path errors, we can consider only the θ and ω contributions to the Lyapunov stability criterion (i.e. $\Delta L_3(\Delta t_L)$ and

$\Delta L_4(\Delta t_L))$. In this case,

$$\Delta L(\Delta t_L) \cong [e_\omega^2(t+\Delta t_L) - e_\omega^2(t)]\; Q_3 + [e_\theta^2(t+\Delta t_L) - e_\theta^2(t)]\; Q_4 \qquad (7.98a)$$

To assure that $\Delta L(\Delta t_L) < 0$ for any admissible ω, this equation is maximized with respect to ω. This gives,

$$\Delta L(\Delta t_L) \cong \delta_\omega^2\; Q_3 + [e_\theta^2(t+\Delta t_L) - e_\theta^2(t)]\; Q_4 \qquad (7.98b)$$

and substituting from (7.97),

$$\Delta L(\Delta t_L) < [\lambda_\theta^2 + e_\theta^2(t+\Delta t_L) - e_\theta^2(t)]\; Q_4 \qquad (7.98c)$$

Condition (7.97) therefore implies that for asymptotic stability ($\Delta L(\Delta t_L) < 0$),

$$e_\theta^2(t+\Delta t_L) < e_\theta^2(t) - \lambda_\theta^2 \qquad (7.99)$$

which clearly indicates that, for any admissible changes in ω, the stability criterion can be satisfied by a decrease in attitude error.

The combined constraining relationships for the weights Q_i, from the results of each of the four cases, are presented below in Table 7.1.

$\kappa_V\lambda_V Q_1 \;>\; [\lambda_\gamma^2\, Q_2 + \delta_\omega^2\, Q_3 + \delta_\theta^2\, Q_4]$
$\kappa_\gamma\lambda_\gamma Q_2 \;>\; [\delta_\omega^2\, Q_3 + \delta_\theta^2\, Q_4]$
$\kappa_V\lambda_V Q_1 + \kappa_\gamma\lambda_\gamma Q_2 \;>\; [\delta_\omega^2\, Q_3 + \delta_\theta^2\, Q_4]$
$\lambda_\theta^2\, Q_4 > \delta_\omega^2\, Q_3$

Table 7.1: Constraining Inequalities For The Weights

These constraints guarantee that if a solution trajectory exists which converges toward the desired trajectory, while observing the bounds of the controllable envelope defined by $\underline{x} \in \underline{\Omega}_x$, then it will satisfy the stability criterion defined by equation (7.56). This result is limited to situations for which velocity or flight path tracking errors exceed given target ranges. In these cases, tracking convergence of the velocity vector toward its desired value implies a decrease in the

Lyapunov function $\Delta L(\Delta t_L) < 0$. For the case where translational tracking errors are within the target range, the additional (fourth) constraint is really just a performance tradeoff, since any choice for λ_θ is acceptable. In fact, aside from the constraints in Table 7.1, the selection of all the weightings Q_i is a design task. For the purposes of this research, Bryson's Rule will be used to make these choices,[25]

$$Q_1=\left[\frac{1}{\delta_V^2}\right] \qquad Q_2=\left[\frac{1}{\delta_\gamma^2}\right]\sigma_1 \qquad Q_3=\left[\frac{1}{\delta_\omega^2}\right]\sigma_2 \qquad Q_4=\left[\frac{1}{\delta_\theta^2}\right]\sigma_3 \qquad (7.100)$$

and the free parameters σ_1, σ_2, σ_3 will be used as scaling factors to adjust the weights to meet the conditions of Table 7.1.

Choosing The Time Interval Δt_L

Up to this point, it has been assumed that the time interval (Δt_L), used by the stability criterion, is long enough that the error–rate (and acceleration) margins for the controllability envelope can actually be achieved. A discrete form for the Lyapunov stability criterion was chosen precisely because these margins are **not** possible at all times, and a quadratic measure of the tracking error might have to increase before it decreases. Several factors are responsible for this time delay, including physical limitations on the actuator rates and the bounds of the controllability envelope $\underline{x} \in \underline{\Omega}_x$.

In terms of the equivalent force and moment controls, δ_F and δ_M (of section 7.3), the largest translational and rotational accelerations on the vehicle occur at the extreme values of the intervals $\delta_F \in [F1,F2]$, and $\delta_M \in [M1,M2]$ (see Figure 7.1). These intervals determine the largest magnitudes of the acceleration margins $\dot{V}^{+/-}_{marg}$ and $\dot{\omega}^{+/-}_{marg}$ that can be guaranteed by the controllability envelope $\underline{x} \in \underline{\Omega}_x$ (see 7.21a–b, 7.23a–b). Although larger accelerations may be possible, only these margins can be assured for the extreme values of the equivalent controls δ_F, δ_M.

[25] Bryson's Rule simply chooses the weighting matrix to be a diagonal matrix of the reciprocals of the squares of the maximum expected deviations for each state. In effect, the states are normalized by some characteristic values.

Now, since the physical control inputs are limited by maximum rates (7.1b), some time may be required (Δt_i) before these acceleration margins can be achieved.

For simplicity, suppose the physical actuator rate limits can be translated into rate limits on the equivalent controls.

$$\left\{\begin{array}{l} \left| \frac{d}{dt}(\delta_T(t)) \right| \leq \dot{\delta}_{T_{max}} \\ \left| \frac{d}{dt}(\delta e(t)) \right| \leq \dot{\delta} e_{max} \\ \left| \frac{d}{dt}(\delta v(t)) \right| \leq \dot{\delta} v_{max} \end{array}\right\} \Longrightarrow \left\{\begin{array}{l} \left| \frac{d}{dt}(\delta_F(t)) \right| \leq \dot{\delta}_{F_{max}} \\ \\ \left| \frac{d}{dt}(\delta_M(t)) \right| \leq \dot{\delta}_{M_{max}} \end{array}\right\} \qquad (7.101)$$

The upper bound on the time required to attain the desired control inputs, can therefore be expressed as,

$$Max[\Delta t_i] = Max\left\{\frac{(F2-F1)}{\dot{\delta}_{F\,max}}, \frac{(M2-M1)}{\dot{\delta}_{M\,max}}\right\} \qquad (7.102)$$

The magnitude of the other acceleration margin ($\dot{\gamma}^{+/-}_{marg}$), corresponding to the normal force on the vehicle, is not limited by the control inputs, but by the admissible range of attitudes within the controllability envelope $\underline{x} \in \underline{\Omega_x}$. Since the largest normal accelerations occur at extreme values of angle of attack, and the rate of change of attitude (pitch rate) is limited by the envelope $\underline{x} \in \underline{\Omega_x}$, some time may be required (Δt_a) to reorient the vehicle to the attitude required for a desired normal acceleration.

From (7.13b), the maximum pitch rate is governed by the constraint,

$$|\omega(t)-\omega_d(t)| \leq \delta_\omega$$

and it can be assumed that the desired value is typically zero, or negligible ($\omega_d(t) \approx 0$), so that,

$$|\omega(t)| \leq \delta_\omega$$

The maximum time required to attain a desired attitude (and therefore a desired normal acceleration) would then be given by,

$$Max[\Delta t_a] = \frac{1}{\delta_\omega}\left[\theta_{\max} - \theta_{\min}\right] + \left[\frac{2\delta_\omega}{\dot{\omega}^{+}_{\mathrm{marg}}}\right] \tag{7.103}$$

$$\textit{where } \theta_{\max} = \underset{\underset{\sim}{x} \in \underline{\Omega}_x}{Max}\left[\theta\right], \textit{ and } \theta_{\min} = \underset{\underset{\sim}{x} \in \underline{\Omega}_x}{Min}\left[\theta\right]$$

which accounts for the time required to achieve the maximum angular rate.

Finally, an extra time interval is defined (Δt_c) to assure that the system has some time to converge toward desired values once the acceleration margins (or error–rates) have been achieved. The time interval for the Lyapunov stability criterion is, therefore, given by,

$$\Delta t_L = \Delta t_i + \Delta t_a + \Delta t_c \tag{7.104}$$

where the value of Δt_c is a design choice. For larger values of Δt_c, the Lyapunov function can decrease further in the time Δt_L. However, since the overall time interval ($t_{\max}$) for a correction trajectory would include at least several intervals of time Δt_L, ($t_{\max} = K\Delta t_L$), the choice of Δt_c is immaterial. Since the Lyapunov criterion will be used to eliminate destabilizing trajectories in the search process, the smallest possible value for Δt_L would be desirable.[26]

In summary, this section has established a criterion for asymptotic stability of the tracking error dynamics for the AHSV. The general form of a candidate Lyapunov function was defined as a weighted quadratic measure of the state tracking errors. Since the admissible domain of solution trajectories was limited by the controllability conditions defined in section 7.3, it was necessary to restrict the values of the weights in the Lyapunov function. This was accomplished by defining target ranges for the translational tracking errors, and assuring that solutions which converge toward these targets also satisfy the stability criterion. Constraints on the weights came from examining four possible cases, in which the errors in velocity and/or flight path were inside or outside their target ranges, while the attitude states remained within the bounds of the controllability envelope. The stability criterion was based on a discrete time interval, since constraints on the states and

[26] The time-step Δt_L should also be large enough to include any nonminimum phase characteristics of the vehicle dynamics. In Chapter 8 this issue is addressed in detail.

actuators prevented the system from instantaneously achieving desired accelerations (or error–rates). An upper bound on the necessary time interval was obtained by observing the maximum time required to attain any set of control inputs and to reorient the vehicle to achieve desired normal accelerations. The results of this section assure that, if an admissible control solution exists for a given tracking error envelope $\underline{x} \in \Omega_x$, then this solution will meet the stability criterion defined by Proposition 2.[27]

7.5 Convergence

The AHSV trajectory optimization problem addressed by the RIFC algorithm was defined in section 7.1 in terms of the state equations, constraints, desired trajectory, parametric uncertainty[28], and disturbances. The objectives of satisfying a stability criterion (developed in the previous section), and minimizing a cost function of the tracking errors, completed the description of the problem. The purpose of this section is to define the cost function, revise the constraints, and demonstrate that, if a solution trajectory exists, the A^* search algorithm will converge.

PROPOSITION 3: CONVERGENCE

Given the optimization problem defined by,

(1) the nonlinear dynamic equations of the system (7.6a–d, 7.7a–k),
(2) the state, control, and dynamic constraints (7.1b–d),
(3) the constraints of a controllability envelope, and a feasible desired trajectory from Proposition 1 (7.53a–j),
(4) the stability criterion defined in Proposition 2 (7.56),

[27] And if the controllability conditions of section 7.3 are met, then the existence of an admissible control solution is guaranteed.

[28] Parametric uncertainty will not be included in the problem until Proposition 4.

(5) *and the cost function chosen as a cumulative sum of tracking errors measured by the Lyapunov function (error norm),*

$$J = \sum_{k=1}^{N} \left\| \underline{x}(k\Delta t_s) - \underline{x}_d(k\Delta t_s) \right\|_Q \qquad (7.105)$$

then,

if a solution exists,
the A^* search algorithm using an optimistic cost function (to evaluate candidate trajectories) will converge to the optimal solution.

ARGUMENT 3:

The cost function defined above represents a cumulative measure of the tracking error at discrete intervals of time (Δt_s), from an initial time ($t = t_0$) up through some final time ($t = t_{max} = t_0 + N\cdot\Delta t_s$). Note that Δt_s is not the same time step as the interval Δt_L in the previous section. The interval Δt_s is the control cycle time and, in general, $\Delta t_s < \Delta t_L$. The time t_{max} is the horizon time required for a solution to converge to the desired trajectory. This "look–ahead" time will depend on the size of the admissible envelope $\underline{x} \in \underline{\Omega}_x$ and the achievable error–rate margins for a particular desired trajectory. An upper bound for the required t_{max} is derived later in this section. In general, however, $t_{max} > \Delta t_L > \Delta t_s$.[29]

The search for an optimal (or suboptimal) solution is performed by an A^* optimization algorithm. The RIFC trajectory control system actually uses an enhanced A^* approach which takes advantage of several devices to improve its efficiency in view of practical considerations, such as computation time and memory limitations. These issues are addressed in Chapter 8. In this section, only the fundamental A^* algorithm is considered.

The search begins with an initial state that is disturbed from the desired trajectory. A quantized control input space is then used to generate the admissible

[29] The Lyapunov stability criterion $\Delta L(\Delta t_L) < 0$ can be applied at each time-step $t = n\cdot\Delta t_s$ by computing $\Delta L(\Delta t_L) = L(\underline{e}(n\cdot\Delta t_s)) - L(\underline{e}(n\cdot\Delta t_s - \Delta t_L))$ for $n \geq \Delta t_L/\Delta t_s$.

input combinations $(\delta e, \delta_T, \delta v)$ for the first control cycle time Δt_s. These candidate partial trajectories are integrated forward Δt_s seconds[30] and the resulting predicted states become possible nodes for further exploration. The search direction is guided by estimates of the cost function for complete trajectories passing through candidate nodes and terminating on the desired trajectory. Branches of the search graph are truncated when a constraint (from items (2) or (3) of Proposition 3) or the stability criterion (4) is violated. The search continues expanding nodes and predicting forward at intervals of Δt_s until a solution is found, or the search fails. The control solution is implemented for one time step Δt_s, and the procedure is repeated with a new initial condition. It will be shown below that, for the conditions of Proposition 3, the A* search succeeds in finding a solution and, in fact, converges to the optimal solution.[31]

An important element of the A* algorithm is the definition of terminal conditions for a solution. These conditions identify the set of goal states. That is, any trajectory which observes the constraints and terminates in a state that meets the terminal conditions is considered a solution to the problem. Of these trajectories, the one which minimizes the cost function is the optimal solution.[32] For the AHSV flight control problem the goal states will be defined as a target envelope in the tracking error space. This envelope can be represented by requiring the Lyapunov function of the errors to be less than some value. Since translational tracking is the principal objective, the target envelope is defined by,

$$L_T = [\, \lambda_V^2\, Q_1 + \lambda_\gamma^2\, Q_2 + \delta_\omega^2\, Q_3 + \delta_\theta^2\, Q_4 \,] \tag{7.106}$$

and the terminal conditions are,

$$L(\underline{x}(t) - \underline{x}_d(t)) < L_T \tag{7.107}$$

The magnitude of L_T corresponds to the case where velocity and flight path errors are within their respective target ranges, while pitch rate and attitude errors are admissible for controllability. The condition (7.107), therefore, is designed to

[30] The integration time step can be considerably smaller than the control cycle-time Δt_s.

[31] A detailed description of the A* algorithm is presented in section 6. 2.

[32] There may be a set of trajectories that minimize the cost function.

classify a trajectory as a solution once its velocity vector has converged to the desired vector. Note that this condition is less restrictive than the requirement that each individual state error be within its respective range. It has the advantage, however, that the target conditions can be met by convergence of the errors in velocity and flight path alone.

Solution Trajectories

Referring back to section 7.4, there were three cases in which the velocity or flight path tracking errors exceeded the bounds of their target ranges. In each case, it was shown that the conditions for asymptotic stability $\Delta L(\Delta t_{\mathrm{L}}) < 0$ could be met by choosing the weights in the Lyapunov function according to the inequality constraints of Table 7.1. In each case, an expression was given for the change in the Lyapunov function in the time Δt_{L}. Repeating equations (7.80, 7.93, and 7.95),

Case 1: $(|e_{\mathrm{V}}| > \lambda_{\mathrm{V}})$ $\Delta L(\Delta t_{\mathrm{L}})_{\mathrm{case1}} < -\kappa_{\mathrm{V}} Q_1 |e_{\mathrm{V}}(t)| + [\lambda_{\gamma}^2 Q_2 + \delta_{\omega}^2 Q_3 + \delta_{\theta}^2 Q_4]$

Case 2: $(|e_{\gamma}| > \lambda_{\gamma})$ $\Delta L(\Delta t_{\mathrm{L}})_{\mathrm{case2}} < -\kappa_{\gamma} Q_2 |e_{\gamma}(t)| + [\delta_{\omega}^2 Q_3 + \delta_{\theta}^2 Q_4]$

Case 3: *(Both 1,2)* $\Delta L(\Delta t_{\mathrm{L}})_{\mathrm{case3}} < -\kappa_{\mathrm{V}} Q_1 |e_{\mathrm{V}}(t)| - \kappa_{\gamma} Q_2 |e_{\gamma}(t)| + [\delta_{\omega}^2 Q_3 + \delta_{\theta}^2 Q_4]$

In order for a trajectory to be considered a solution to the optimization problem, it is desirable to have,

$$\Delta L(\Delta t_{\mathrm{L}}) \leq -\delta_{\mathrm{L}} < 0 \qquad (7.108)$$

where δ_{L} is a positive constant. The achievable value of δ_{L}, which is a function of the control margins and weights, can be obtained from the following:

$$Max\left\{ \underset{|e_{\mathrm{V}}| > \lambda_{\mathrm{V}}}{Max}\left[\Delta L(\Delta t_{\mathrm{L}})_{\mathrm{case1}}\right], \underset{|e_{\gamma}| > \lambda_{\gamma}}{Max}\left[\Delta L(\Delta t_{\mathrm{L}})_{\mathrm{case2}}\right], \underset{|e_{\mathrm{V}}| > \lambda_{\mathrm{V}}}{Max}\ \underset{|e_{\gamma}| > \lambda_{\gamma}}{Max}\left[\Delta L(\Delta t_{\mathrm{L}})_{\mathrm{case3}}\right]\right\} \leq -\delta_{\mathrm{L}} \qquad (7.109)$$

Therefore, given the controllability envelope $\underline{x} \in \underline{\Omega}_{\mathrm{x}}$, a feasible desired trajectory, and an appropriate choice for the weights of the stability criterion, a solution trajectory would decrease according to $\Delta L(\Delta t_L) \leq -\delta_L$. [33]

This reduction is applicable to all cases for which errors in velocity and/or flight path angle exceed their target ranges. The concept is illustrated in Figure 7.3, where a value of the Lyapunov function is represented by an ellipse in the four dimensional error–space (only 3D in figure).

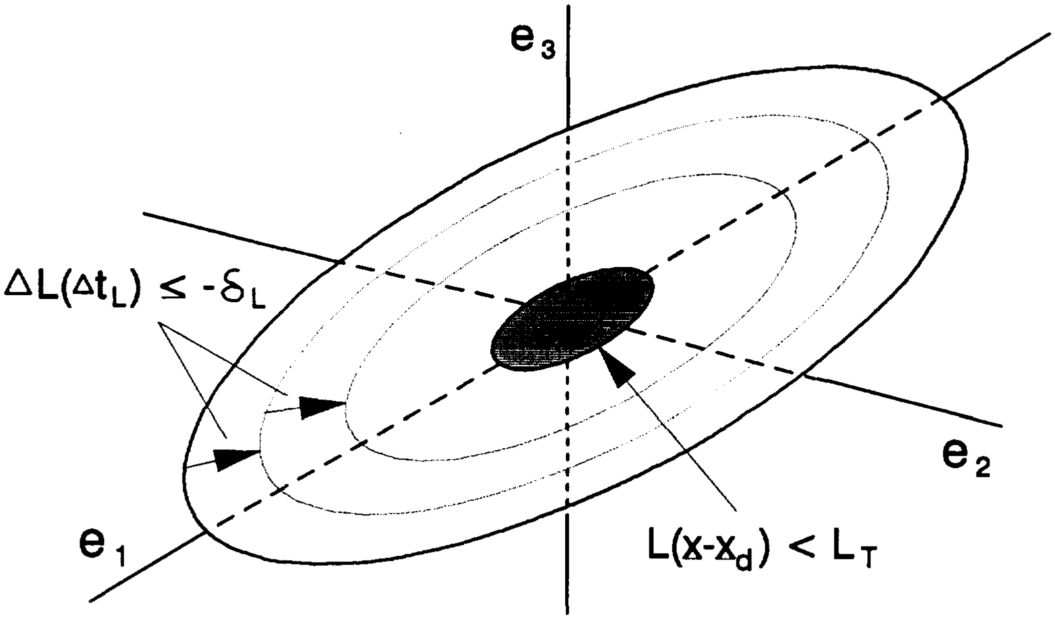


Figure 7.3: Existence of a Solution in the Error–Space

The boundaries of this error–space are defined by the limits of the controllability envelope $\underline{x} \in \Omega_x$ (or $\underline{e} \in \Omega_e$), and the equation $\Delta L(\Delta t_L) \leq -\delta_L$ corresponds to a series of shrinking ellipses, at intervals of time Δt_L.

Since, by definition (7.106), the target envelope is achieved for $|e_V| \leq \lambda_V$ and $|e_\gamma| \leq \lambda_\gamma$ (with admissible ω,θ), a solution trajectory converges to the target

[33] Note that $\Delta L(\Delta t_L)$ is a relative value. It would take larger values of Q_1 and Q_2 to obtain a larger reduction δ_L, but then the value of the function $L(\underline{e}(t))$ would also be larger for the same tracking error.

envelope after K intervals of Δt_L, where

$$K = \frac{L(\underline{e}(t_0))-L_T}{\delta_L} \tag{7.110}$$

This also gives an upper bound on the horizon time required (t_{max}) to assure that it is possible for a solution to converge to the target envelope. Since the maximum value of $L(\underline{e}(t_0))$ occurs at the limits of the controllability envelope,

$$K_{max} = \left[\frac{1}{\delta_L}\right]\left[\delta_V^2 Q_1+\delta_\gamma^2 Q_2+\delta_\omega^2 Q_3+\delta_\theta^2 Q_4\right] \tag{7.111}$$

and therefore,

$$t_{max} = K_{max}\,\Delta t_L = \left[\frac{1}{\delta_L}\right]\left[\delta_V^2 Q_1+\delta_\gamma^2 Q_2+\delta_\omega^2 Q_3+\delta_\theta^2 Q_4\right]\Delta t_L \tag{7.112}$$

Convergence Of The A* Search

One advantage of combining a Lyapunov stability criterion with the A* optimization is that the solution space over which the algorithm must search is drastically reduced. Moreover, by using the Lyapunov function as the stepwise measure of the cost, the A* search is biased to proceed in directions that explore the portions of the graph which meet the stability criterion first (before searching elsewhere). These properties are extremely useful from the viewpoint of limiting the required search effort. Figure 7.4 illustrates the relationship between the entire search space and the subspace of solutions that meet the stability criterion.

Before discussing the convergence properties of the A* search with respect to the optimization problem of Proposition 3, it is necessary to define the heuristic evaluation function $f(n)$ used to estimate the complete cost of partially explored trajectories. The actual cost of a complete trajectory is given by,

$$J = \sum_{k=1}^{N} J_k[\underline{x}(k\Delta t_s),\, \underline{x}_d(k\Delta t_s)] \tag{7.113}$$

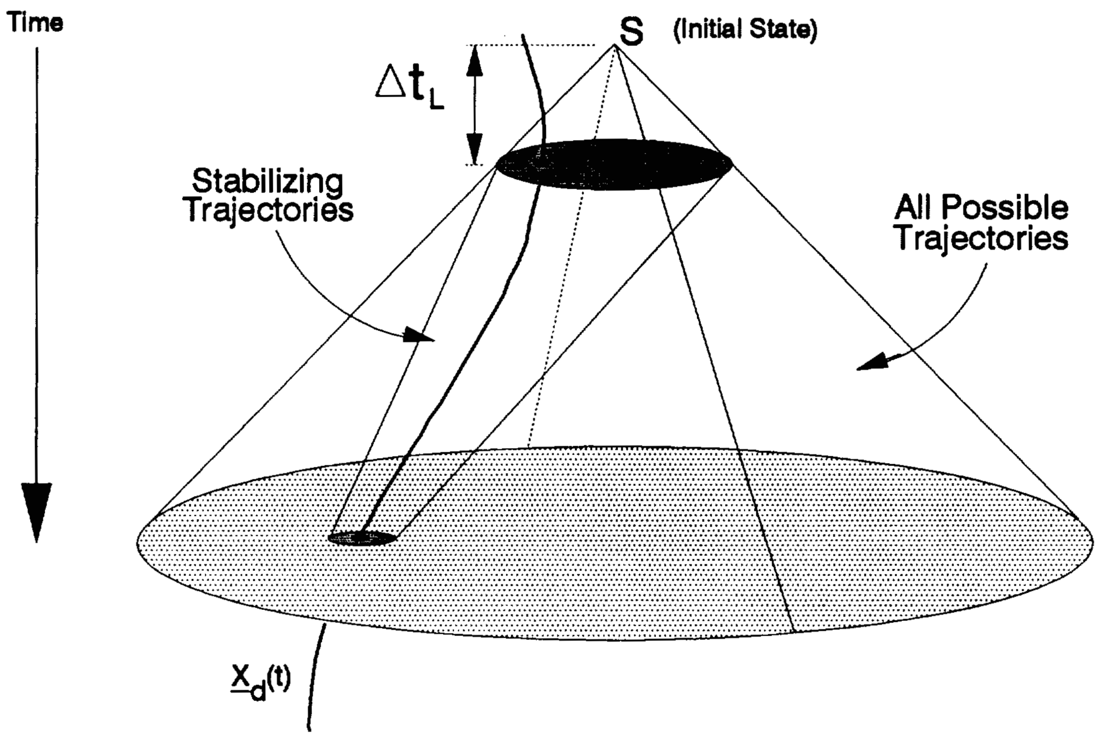


Figure 7.4: Subspace of Trajectories Meeting the Stability Criterion

where, from (7.105)

$$J_k[\underline{x}(k\Delta t_s), \underline{x}_d(k\Delta t_s)] = \left\| \underline{x}(k\Delta t_s) - \underline{x}_d(k\Delta t_s) \right\|_Q \tag{7.114}$$

Now, recall from Chapter 6 (equation 6.7) that,

$$f(n) = g(n) + h(n)$$

and, in this case,

$$g(n) = \sum_{k=1}^{k_n} \left\| \underline{x}(k\Delta t_s) - \underline{x}_d(k\Delta t_s) \right\|_Q \tag{7.115}$$

where $g(n)$ is the tracking error cost from the initial condition (Start node) to the state represented by the node n.

The value of k_n indicates the depth of the node n in the graph[34], and the partial trajectory is given by the set of states $\{\underline{x}(k\Delta t_s)\colon k \in [1, k_n]\}$ along the path to the node n. Therefore, for a partially explored path, the function $g(n)$ is the known part of the cost function. The cost of the remaining portion of a partial trajectory is unknown, and the A* algorithm uses the heuristic function $h(n)$ to estimate this cost:

$$h(n) = E\left\{ \sum_{k=k_n+1}^{N} \left\| \underline{x}(k\Delta t_s) - \underline{x}_d(k\Delta t_s) \right\|_Q \right\} \tag{7.116}$$

where $E\{\cdot\}$ represents an estimated value.

For reasons explained below (and in Chapter 6), it is desirable to have a heuristic function $h(n)$ that approximates the actual terminating cost as closely as possible, while being an optimistic estimate. If $h^*(n)$ represents the actual optimal cost of a trajectory from the state n to a state in the target envelope, then the estimate $h(n)$ is defined to be an *admissible heuristic function* (according to Definition 6.12) if,

$$h(n) \leq h^*(n) \qquad \forall\, n \tag{7.117}$$

An admissible cost estimate $h(n)$ can be easily obtained for the AHSV trajectory control problem, by assuming that each state variable can be independently controlled using the full control authority of each input. That is, $h(n)$ would be the cumulative cost of a path from state n to a state in the target envelope, where the path is a fictitious trajectory with maximum possible convergence rates on all axes. Since the coupling of the states and controls would prevent such a trajectory from actually being possible, this cost estimate would be optimistic. This idea is clarified in Figure 7.5, which shows the difference between the terminal portion of an optimal solution (from a state n) and the corresponding trajectories for the heuristic estimate.

[34] It also represents the time at state n, $t = k_n\Delta t_s$.

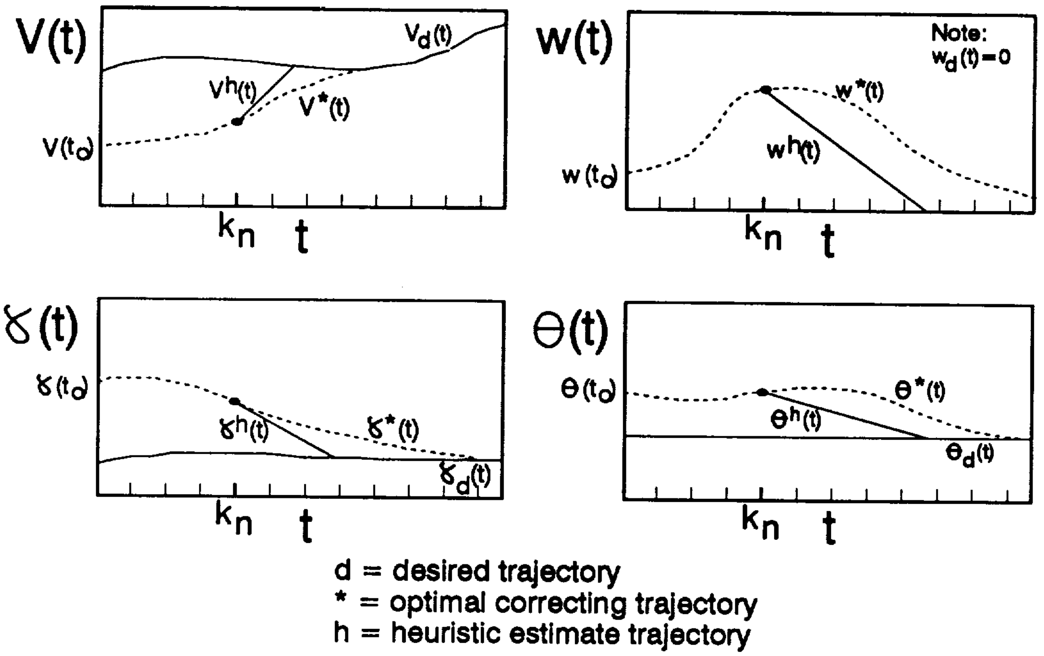


Figure 7.5: Optimal and Heuristic Terminal Trajectories

The heuristic trajectory $\underline{x}^{\mathrm{h}}(t) \equiv [V^{\mathrm{h}}(t)\ \ \gamma^{\mathrm{h}}(t)\ \ \omega^{\mathrm{h}}(t)\ \ \theta^{\mathrm{h}}(t)]^{\mathrm{T}}$ is obtained by minimizing the difference $\underline{x}(k\Delta t_s)-\underline{x}_{\mathrm{d}}(k\Delta t_s)$ for $k \in [k_{\mathrm{n}}+1,\ N]$ with respect to the admissible inputs $\underline{u} \in \underline{\Omega}_{\mathrm{u}}$ and states $\underline{x} \in \underline{\Omega}_{\mathrm{x}}$ for each state separately.

$$\underline{x}^{\mathrm{h}}(k\Delta t_s) = \underline{x}(k_{\mathrm{n}}\Delta t_s) + \sum_{j=k_{\mathrm{n}}+1}^{k} \dot{\underline{x}}^{\mathrm{h}}(j\Delta t_s)\,\Delta t_s \qquad (7.118)$$

Referring to equations (7.6a–d), the best possible rates for each state variable in the error correcting directions are given by,

$$\dot{V}^{\mathrm{h}}(t) = \left\{ \begin{array}{ll} \underset{\underline{u}\in\underline{\Omega}_{\mathrm{u}}}{Max}\ \underset{\underline{x}\in\underline{\Omega}_{\mathrm{x}}}{Max}\left[\dot{V}(t)\right] = \underset{\underline{u}\in\underline{\Omega}_{\mathrm{u}}}{Max}\ \underset{\underline{x}\in\underline{\Omega}_{\mathrm{x}}}{Max}\left[f_{\mathrm{V}}(\underline{x},t)+g_{\mathrm{V}}(\underline{x},\underline{u},t)\right] & \text{for } V < V_{\mathrm{d}} \\ \underset{\underline{u}\in\underline{\Omega}_{\mathrm{u}}}{Min}\ \underset{\underline{x}\in\underline{\Omega}_{\mathrm{x}}}{Min}\left[\dot{V}(t)\right] = \underset{\underline{u}\in\underline{\Omega}_{\mathrm{u}}}{Min}\ \underset{\underline{x}\in\underline{\Omega}_{\mathrm{x}}}{Min}\left[f_{\mathrm{V}}(\underline{x},t)+g_{\mathrm{V}}(\underline{x},\underline{u},t)\right] & \text{for } V > V_{\mathrm{d}} \end{array} \right\} \qquad (7.119a)$$

$$\dot{\gamma}^{h}(t)=\left\{\begin{array}{ll} \underset{\underline{u}\in\underline{\Omega}_{u}}{Max}\ \underset{\underline{x}\in\underline{\Omega}_{x}}{Max}\left[\dot{\gamma}(t)\right] = \underset{\underline{u}\in\underline{\Omega}_{u}}{Max}\ \underset{\underline{x}\in\underline{\Omega}_{x}}{Max}\left[f_{\gamma}(\underline{x},t)+g_{\gamma}(\underline{x},\underline{u},t)\right] & \text{for } \gamma < \gamma_{d} \\ \\ \underset{\underline{u}\in\underline{\Omega}_{u}}{Min}\ \underset{\underline{x}\in\underline{\Omega}_{x}}{Min}\left[\dot{\gamma}(t)\right] = \underset{\underline{u}\in\underline{\Omega}_{u}}{Min}\ \underset{\underline{x}\in\underline{\Omega}_{x}}{Min}\left[f_{\gamma}(\underline{x},t)+g_{\gamma}(\underline{x},\underline{u},t)\right] & \text{for } \gamma > \gamma_{d} \end{array}\right] \qquad (7.119b)$$

$$\dot{\omega}^{h}(t)=\left\{\begin{array}{ll} \underset{\underline{u}\in\underline{\Omega}_{u}}{Max}\ \underset{\underline{x}\in\underline{\Omega}_{x}}{Max}\left[\dot{\omega}(t)\right] = \underset{\underline{u}\in\underline{\Omega}_{u}}{Max}\ \underset{\underline{x}\in\underline{\Omega}_{x}}{Max}\left[f_{\omega}(\underline{x},t)+g_{\omega}(\underline{x},\underline{u},t)\right] & \text{for } \omega < \omega_{d} \\ \\ \underset{\underline{u}\in\underline{\Omega}_{u}}{Min}\ \underset{\underline{x}\in\underline{\Omega}_{x}}{Min}\left[\dot{\omega}(t)\right] = \underset{\underline{u}\in\underline{\Omega}_{u}}{Min}\ \underset{\underline{x}\in\underline{\Omega}_{x}}{Min}\left[f_{\omega}(\underline{x},t)+g_{\omega}(\underline{x},\underline{u},t)\right] & \text{for } \omega > \omega_{d} \end{array}\right] \qquad (7.119c)$$

$$\dot{\theta}^{h}(t)=\left\{\begin{array}{ll} \underset{\underline{u}\in\underline{\Omega}_{u}}{Max}\ \underset{\underline{x}\in\underline{\Omega}_{x}}{Max}\left[\dot{\theta}(t)\right] = \delta_{\omega} & \text{for } \theta < \theta_{d} \\ \\ \underset{\underline{u}\in\underline{\Omega}_{u}}{Min}\ \underset{\underline{x}\in\underline{\Omega}_{x}}{Min}\left[\dot{\theta}(t)\right] = -\delta_{\omega} & \text{for } \theta > \theta_{d} \end{array}\right] \left[\begin{array}{l} \textit{Since } \delta_{\omega} \textit{ is the} \\ \textit{maximum allowable} \\ \textit{pitch rate.} \end{array}\right] \qquad (7.119d)$$

Since it is assumed that the dynamics are autonomous for the duration of the horizon time in which the trajectory is being optimized (constant mass and inertias), the rates above can be written as $\dot{\underline{x}}^{h} \equiv [\dot{V}^{h}\ \dot{\gamma}^{h}\ \dot{\omega}^{h}\ \dot{\theta}^{h}]^{T}$. The cost estimate $h(n)$ can then be represented by,

$$h(n) = \sum_{k=k_{n}+1}^{N'} \left\| \left[\underline{x}(k_{n}\Delta t_{s}) + \dot{\underline{x}}^{h}\cdot(k-k_{n})\Delta t_{s}\right] - \underline{x}_{d}(k\Delta t_{s}) \right\|_{Q} \qquad (7.120)$$

where N' is the index for which $\underline{x}(N'\Delta t_{s})$ is within the target envelope, (i.e. $L(\underline{x}(N'\Delta t_{s})-\underline{x}_{d}(N'\Delta t_{s})) \leq L_{T}$).

Now that an admissible heuristic cost estimate has been defined, the convergence properties of the A* search follow directly from the Theorems in Chapter 6.

<u>From Theorem 6.7:</u>

The A* algorithm always terminates with a solution if one exists. (The proof is given in section 6.2).

> *Since we have established a feasible desired trajectory, a controllable envelope, and a stability criterion that can be satisfied by admissible converging trajectories, the A^* search cannot terminate without finding a solution.*

From Theorem 6.9:

> *The A^* algorithm using an admissible heuristic function $h(n)$ converges to the optimal solution (if one exists). (The proof is also given in section 6.2).*
>
> *Since a tracking solution exists and the heuristic function $h(n)$ optimistically estimates the termination cost of partial trajectories (and is therefore admissible), the A^* search will converge to the optimal solution.*

Based on Theorems 6.7 and 6.9 (and the results of the last three sections), the theoretical convergence properties of the A^* algorithm have been established for the AHSV tracking problem. The only remaining obstacles for convergence are related to practical considerations, such as computation time and memory requirements. Although these issues will be addressed in Chapter 8, it is worth noting that the addition of a stability constraint to the optimization problem (in this section – item (4) of Proposition 3) improves the efficiency of the A^* search, since a majority of the solution space can be immediately truncated. Another important efficiency factor is the conservativeness of the heuristic function $h(n)$. For $V(t)$ and $\gamma(t)$ outside the target envelope, the magnitude of $L(\underline{e}(t))$ is governed primarily by the errors $e_V(t)$ and $e_\gamma(t)$. Now, it is clear that all of the rates $\underline{\dot{x}}^{h}(t)$ used by the heuristic cost estimate $h(n)$ cannot be achieved independently. However, since velocity can be controlled directly from the equivalent input δ_F, and flight path angle can be controlled from δ_M, the rates $\dot{V}(t)$ and $\dot{\gamma}(t)$ can be independently changed. Therefore, the optimal trajectory may not differ too significantly from the heuristic solution in terms of the translational state variables. This means that (outside the target envelope) the heuristic function $h(n)$ is reasonably well informed (i.e. $h(n) \approx h^*(n)$), and (from Chapter 6) this implies an efficient search. Recall that, if the function $h(n)$ were known precisely $(h(n) = h^*(n))$, then the search would only explore nodes along optimal paths.

In summary, this section has combined the original AHSV trajectory control problem (of section 7.1), with the constraints of a controllability envelope (section 7.3), and the Lyapunov stability criterion (section 7.4). The cost function has been defined as a cumulative measure of the tracking error based on the Lyapunov error norm, and a target envelope or terminating condition for tracking solutions has been defined. An upper bound on the time to reach the target envelope ($t_{\max}$) was also derived. Finally, an admissible cost estimate function was developed for the A^* search, and convergence of the algorithm to the optimal solution was confirmed.

7.6 Robustness

Up to this point, the analysis of this chapter has not included the effects of parametric uncertainty in the aerodynamic, propulsion, and control coefficients. In order to account for this uncertainty, the results of the previous sections must be augmented to include variations in the parameter space.

For controllability, the additional robustness requirement translates into reduced acceleration margins and/or a more restrictive controllability envelope. The assumptions of Proposition 1, and the final controllability conditions (7.53a–j) must be modified. For the purposes of this section, it is assumed that uncertainties in the parameters are small enough that there exists some nonempty controllable region in the state space. If this were not the case, then the RIFC controller would still perform a search for the best solution trajectory; the result, however, could not be guaranteed to exhibit the desired tracking properties. Flight conditions for which the above assumption does not hold indicate the need for an improved coefficient database model or a change in the vehicle configuration that compensates for the uncertainty with additional control authority.

The Lyapunov criterion for stability of the tracking error dynamics, equation (7.56), can be made into a robust criterion by including a maximization over the parameter space. A trajectory that satisfies the resulting stability criterion must, therefore, exhibit a diminishing tracking error (measured by the Q–norm) for any set of parameters within the parameter space. The inequality constraints on the weights Q_i, and the time interval Δt_L, must also be recomputed to reflect changes in the controllability conditions due to parametric uncertainties.

The issue of convergence for the A^* algorithm also requires reexamination in the presence of uncertainty because the cost function and the terminal conditions for a solution must change. The cost function is modified to minimize the worst case cumulative tracking error rather than the predicted error for trajectories with nominal parameter values. Since the prediction error increases with time, however, it is not sensible to search specifically for solutions that terminate on some target envelope. At some point along each trajectory, the magnitude of the uncertainty will have grown to be as large as the predicted tracking errors. Searching for optimal solutions beyond this depth is pointless, since the results would be completely unreliable. As such, the definition of a solution is modified to include a combination of the original target envelope and an error–uncertainty matching condition. Otherwise, the arguments for the existence of a solution to the optimization problem, and convergence of the A^* search procedure, are essentially the same as that presented in section 7.5.

PROPOSITION 4: ROBUSTNESS

Given the following conditions:

(1) The assumptions of Proposition 1 remain valid in the presence of interval bounded uncertainty on the aerodynamic, propulsion, and control coefficients.

(2) There exists a nonempty region in the state space, such that the controllability conditions of Proposition 1 (7.53a-j) can be satisfied for the case with parametric uncertainty included.

(3) The Lyapunov function from Proposition 2 is augmented as follows,

$$L(\underline{e}(t),\underline{\phi}) = \underset{\underline{\phi}\,\in\underline{\Omega}_{\phi}}{Max}\left\| \underline{x}(\underline{\phi},t)-\underline{x}_{\mathrm{d}}(t) \right\|_Q \tag{7.121}$$

with the stability criterion given by,

$$\underset{\underline{\phi}\,\in\underline{\Omega}_{\phi}}{Max}\left\| \underline{x}(\underline{\phi},t+\Delta t_{\mathrm{L}})-\underline{x}_{\mathrm{d}}(t+\Delta t_{\mathrm{L}}) \right\|_Q - \underset{\underline{\phi}\,\in\underline{\Omega}_{\phi}}{Max}\left\| \underline{x}(\underline{\phi},t)-\underline{x}_{\mathrm{d}}(t) \right\|_Q < 0 \tag{7.122}$$

where $\underline{x}(\cdot,t)$ is a measured quantity (independent of the parametric uncertainty) and $\underline{x}(\underline{\phi},t+\Delta t_{\mathrm{L}})$ is a predicted value.

(4) The cost function from Proposition 3 is redefined as,

$$J = \sum_{k=1}^{N} \underset{\underline{\phi}\,\in\underline{\Omega}_{\phi}}{Max}\left\| \underline{x}(\underline{\phi},k\Delta t_{\mathrm{s}})-\underline{x}_{\mathrm{d}}(k\Delta t_{\mathrm{s}}) \right\|_Q \tag{7.123}$$

then,

if a solution to the optimization problem exists that is robust to the uncertainty in parameters,
the A^* search converges to the optimal solution.

ARGUMENT 4:

Robust Controllability

In the analysis of section 7.3 the issues of theoretical and physical controllability were addressed. In the presence of uncertainty in the values of the aerodynamic, propulsion, and control coefficients, the analysis for theoretical controllability is unchanged. Some revisions are required, however, in order to establish physical controllability for this case. The assumptions from Proposition 1 are presented below with modifications that account for parametric uncertainty.

Assumption 1

$$\underset{\phi \in \Omega_\phi}{Min}\left\{\underset{\delta_F}{Max}\left[f_1(\underline{x},t) + g_1(\underline{x},t)\,\delta_F + d_1(t)\right]\right\} \geq \dot{V}^+_{marg} > 0 \qquad (7.124a)$$

$$\underset{\phi \in \Omega_\phi}{Max}\left\{\underset{\delta_F}{Min}\left[f_1(\underline{x},t) + g_1(\underline{x},t)\,\delta_F + d_1(t)\right]\right\} \leq \dot{V}^-_{marg} < 0 \qquad (7.124b)$$

Assumption 2

$$\underset{\phi \in \Omega_\phi}{Min}\left\{\underset{\delta_M}{Max}\left[f_3(\underline{x},t) + g_3(\underline{x},t)\,\delta_M + d_3(t)\right]\right\} \geq \dot{\omega}^+_{marg} > 0 \qquad (7.125a)$$

$$\underset{\phi \in \Omega_\phi}{Max}\left\{\underset{\delta_M}{Min}\left[f_3(\underline{x},t) + g_3(\underline{x},t)\,\delta_M + d_3(t)\right]\right\} \leq \dot{\omega}^-_{marg} < 0 \qquad (7.125b)$$

Assumption 3

$$\underset{\phi \in \Omega_\phi}{Min}\left\{\underset{V}{Min}\,\underset{\gamma}{Min}\,\underset{\omega}{Min}\left\{\underset{\alpha}{Max}\left[f_2(\underline{x},t)\right] + d_2(t)\right\}\right\} \geq \dot{\gamma}^+_{marg} > 0 \qquad (7.126a)$$

$$\underset{\phi \in \Omega_\phi}{Max}\left\{\underset{V}{Max}\,\underset{\gamma}{Max}\,\underset{\omega}{Max}\left\{\underset{\alpha}{Min}\left[f_2(x,t)\right] + d_2(t)\right\}\right\} \leq \dot{\gamma}^-_{marg} < 0 \qquad (7.126b)$$

where $\phi^T \equiv [C_L\ C_D\ C_{Ma}\ C_{Tn}\ C_{Tv}\ C_{Mp}\ C_{X_{\delta e}}\ ...]$, and Ω_ϕ represents the set of all possible parameter vectors.

These assumptions are essentially the same as given in section 7.3. Assumption 1 means that it is always possible to achieve a positive or negative acceleration in the velocity direction (of at least $\dot{V}^{+/-}_{marg}$) for some choice of δ_F provided the states remain within the controllable envelope $x \in \Omega_x$. In this context, however, the margins $\dot{V}^{+/-}_{marg}$ must be recalculated to reflect the achievable accelerations despite a range of uncertainty in the parameters. Assumption 2 means that it is always possible to achieve a positive or negative angular acceleration (of at least $\dot{\omega}^{+/-}_{marg}$) for some choice of δ_M provided the states remain within the controllable envelope $x \in \Omega_x$. Here again, the margins $\dot{\omega}^{+/-}_{marg}$ must be recalculated to account for uncertainty. Finally, Assumption 3 means that, even in the presence of parametric uncertainty, the maximum normal accelerations are experienced at the extreme values of the admissible angle of attack range. Furthermore, the normal acceleration is positive for the maximum angle of attack, and negative for the minimum angle of attack. The margins corresponding to each of the above assumptions are given below:

$$\dot{V}^{+}_{\mathrm{marg}} = \underset{\phi \in \Omega_{\Phi}}{Min}\ \underset{x \in \Omega_{X}}{Min}\left[f_1(x,t)\right] + \underset{\phi \in \Omega_{\Phi}}{Min}\ \underset{x \in \Omega_{X}}{Min}\left[g_1(x,t)\right] \cdot F2 + Min[d_1(t)] \qquad (7.127a)$$

$$\dot{V}^{-}_{\mathrm{marg}} = \underset{\phi \in \Omega_{\Phi}}{Max}\ \underset{x \in \Omega_{X}}{Max}\left[f_1(x,t)\right] + \underset{\phi \in \Omega_{\Phi}}{Min}\ \underset{x \in \Omega_{X}}{Min}\left[g_1(x,t)\right] \cdot F1 + Max[d_1(t)] \qquad (7.127b)$$

$$\dot{\omega}^{+}_{\mathrm{marg}} = \underset{\phi \in \Omega_{\Phi}}{Min}\ \underset{x \in \Omega_{X}}{Min}\left[f_3(x,t)\right] + \underset{\phi \in \Omega_{\Phi}}{Min}\ \underset{x \in \Omega_{X}}{Min}\left[g_3(x,t)\right] \cdot M2 + Min[d_3(t)] \qquad (7.127c)$$

$$\dot{\omega}^{-}_{\mathrm{marg}} = \underset{\phi \in \Omega_{\Phi}}{Max}\ \underset{x \in \Omega_{X}}{Max}\left[f_3(x,t)\right] + \underset{\phi \in \Omega_{\Phi}}{Min}\ \underset{x \in \Omega_{X}}{Min}\left[g_3(x,t)\right] \cdot M1 + Max[d_3(t)] \qquad (7.127d)$$

$$\dot{\gamma}^{+}_{\mathrm{marg}} = \underset{\phi \in \Omega_{\Phi}}{Min}\left\{\underset{V}{Min}\ \underset{\gamma}{Min}\ \underset{\omega}{Min}\left\{\underset{\alpha}{Max}\left[f_2(x,t)\right]\right\}\right\} + Min[d_2(t)] \qquad (7.127e)$$

$$\dot{\gamma}^{-}_{\mathrm{marg}} = \underset{\phi \in \Omega_{\Phi}}{Max}\left\{\underset{V}{Max}\ \underset{\gamma}{Max}\ \underset{\omega}{Max}\left\{\underset{\alpha}{Min}\left[f_2(x,t)\right]\right\}\right\} + Max[d_2(t)] \qquad (7.127f)$$

where the limits *F1, F2, M1, M2* are given by,

$$F1 = Min\left[\delta_{\mathrm{F}}\right] = \underset{\phi \in \Omega_{\Phi}}{Max}\left\{\underset{x \in \Omega_{X}}{Max}\left\{\underset{\delta_{\mathrm{T}}}{Min}\left[g'_V(x,\delta_{\mathrm{T}},t) \,/\, g_1(x,t)\right]\right\}\right\} \qquad (7.128a)$$

$$F2 = Max\left[\delta_{\mathrm{F}}\right] = \underset{\phi \in \Omega_{\Phi}}{Min}\left\{\underset{x \in \Omega_{X}}{Min}\left\{\underset{\delta_{\mathrm{T}}}{Max}\left[g'_V(x,\delta_{\mathrm{T}},t) \,/\, g_1(x,t)\right]\right\}\right\} \qquad (7.128b)$$

$$M1 = Min\left[\delta_{\mathrm{M}}\right] = \underset{\phi \in \Omega_{\Phi}}{Max}\left\{\underset{x \in \Omega_{X}}{Max}\left\{\underset{\delta e}{Min}\ \underset{\delta v}{Min}\left[g'_\omega(x,\delta e,\delta v,t) \,/\, g_3(x,t)\right]\right\}\right\} \qquad (7.128c)$$

$$M2 = Max\left[\delta_{\mathrm{M}}\right] = \underset{\phi \in \Omega_{\Phi}}{Min}\left\{\underset{x \in \Omega_{X}}{Min}\left\{\underset{\delta e}{Max}\ \underset{\delta v}{Max}\left[g'_\omega(x,\delta e,\delta v,t) \,/\, g_3(x,t)\right]\right\}\right\} \qquad (7.128d)$$

(Refer back to equations 7.12, 7.13 for definitions of g'_V, g_1, g'_ω, g_3*)*

In equations (7.127a–f) the margins are obtained by calculating the largest possible accelerations that can be achieved in each direction while the values of the states, parameters, and disturbances oppose the effort as much as possible. Likewise, the maximum and minimum equivalent control inputs (7.128a–d) represent the largest possible force and moment coefficients that can be assured for all possible parameter values and all admissible states.

For the purposes of this section, it is assumed that the uncertainty in the coefficients is small enough and the control authority is large enough, so that the margins specified above actually exist. That is, Assumptions 1, 2, and 3 hold and, therefore, there exists a range of admissible states $x \in \Omega_x$ such that the margins $\dot{V}^+_{\text{marg}}$, $\dot{\omega}^+_{\text{marg}}$, $\dot{\gamma}^+_{\text{marg}}$ are, in fact, positive quantities (and their counterparts $\dot{V}^-_{\text{marg}}$, $\dot{\omega}^-_{\text{marg}}$, $\dot{\gamma}^-_{\text{marg}}$ are negative). These assumptions are reasonable, since what they imply qualitatively is that for some admissible range of states, there is enough control authority to assure that accelerations in desired directions are possible, despite the uncertainty in the state equations, by applying enough control effort.[35] Also note that these assumptions do not limit the ability of the RIFC controller, since the search for a trajectory that minimizes the worst–case tracking error does not depend on the margins above. However, if a controllability envelope cannot be defined that is robust to uncertainty in the state equations, then tracking performance cannot be guaranteed.

The controllability conditions of section 7.3 are now augmented to account for parametric uncertainty and presented below:

<u>*Robust Controllability Conditions*</u>

$$\dot{e}^+_{\text{Vmarg}} = \underset{\phi\in\Omega_\phi}{Min}\ \underset{x\in\Omega_x}{Min}\left[f_1(x,t)\right] + \underset{\phi\in\Omega_\phi}{Min}\ \underset{x\in\Omega_x}{Min}\left[g_1(x,t)\right]\cdot F2 + Min[d_1(t)] - Max[\dot{V}_d] \qquad (7.129a)$$

$$\dot{e}^-_{\text{Vmarg}} = \underset{\phi\in\Omega_\phi}{Max}\ \underset{x\in\Omega_x}{Max}\left[f_1(x,t)\right] + \underset{\phi\in\Omega_\phi}{Min}\ \underset{x\in\Omega_x}{Min}\left[g_1(x,t)\right]\cdot F1 + Max[d_1(t)] - Min[\dot{V}_d] \qquad (7.129b)$$

[35] Or by attaining an extreme angle-of-attack for achieving normal accelerations.

$$\dot{e}^{+}_{\omega_{\text{marg}}} = \underset{\phi \in \Omega_{\phi}}{Min}\ \underset{x \in \Omega_{x}}{Min}\left[f_{3}(x,t)\right] + \underset{\phi \in \Omega_{\phi}}{Min}\ \underset{x \in \Omega_{x}}{Min}\left[g_{3}(x,t)\right] \cdot M2 + Min[d_{3}(t)] - Max[\dot{\omega}_{d}] \qquad (7.129c)$$

$$\dot{e}^{-}_{\omega_{\text{marg}}} = \underset{\phi \in \Omega_{\phi}}{Max}\ \underset{x \in \Omega_{x}}{Max}\left[f_{3}(x,t)\right] + \underset{\phi \in \Omega_{\phi}}{Min}\ \underset{x \in \Omega_{x}}{Min}\left[g_{3}(x,t)\right] \cdot M1 + Max[d_{3}(t)] - Min[\dot{\omega}_{d}] \qquad (7.129d)$$

$$\dot{e}^{+}_{\gamma_{\text{marg}}} = \underset{\phi \in \Omega_{\phi}}{Min}\left\{\underset{V}{Min}\ \underset{\gamma}{Min}\ \underset{\omega}{Min}\left\{\underset{\alpha}{Max}\left[f_{2}(x,t)\right]\right\}\right\} + Min[d_{2}(t)] - Max[\dot{\gamma}_{d}] \qquad (7.129e)$$

$$\dot{e}^{-}_{\gamma_{\text{marg}}} = \underset{\phi \in \Omega_{\phi}}{Max}\left\{\underset{V}{Max}\ \underset{\gamma}{Max}\ \underset{\omega}{Max}\left\{\underset{\alpha}{Min}\left[f_{2}(x,t)\right]\right\}\right\} + Max[d_{2}(t)] - Min[\dot{\gamma}_{d}] \qquad (7.129f)$$

As in section 7.3 these equations express the relationship between the objectives of the desired trajectory, the size of the tracking error envelope for which controllability can be assured, and the achievable error–rate margins. All quantities here, however, include the additional element of parametric uncertainty. In general, this uncertainty will decrease the magnitude of the equivalent control limits *F1*, *F2*, *M1*, *M2*, making it necessary to further restrict the envelope $x \in \Omega_x$ in order to attain desired margins. In effect, uncertainty in the coefficients will limit the controllability envelope, degrade the aggressiveness of feasible desired trajectories, and/or reduce the attainable error–rate margins. Nevertheless, equations (7.129a–f) can be used to select an admissible tracking error envelope $x \in \Omega_x$, constrain the desired trajectory, and allow for acceptable error–rate margins, in order to guarantee the existence of a robust tracking solution (given Assumptions 1, 2, & 3).

Robust Stability

The concept of robust stability for discrete systems was introduced in Chapter 6 as Definition 6.11. Sufficient conditions for robust stability are then given in Theorem 6.6, which presents Lyapunov's direct method for robust stability of discrete–time autonomous systems. This theorem is restated below in terms of the current variables:

THEOREM 7.3: Lyapunov's Theorem for robust stability of discrete–time autonomous systems (From Theorem 6.6)

Consider the discrete-time autonomous nonlinear system

$$\underline{e}(t+\Delta t_L) = \underline{f}_d(\underline{e}(t),\underline{\phi}), \quad \text{with } \underline{f}_d(\underline{0},\underline{\phi}) = \underline{0} \qquad (7.130a)$$

If, within a ball $\underline{e} \in \underline{\Omega}_e$ there exists a scalar function $L(\underline{e},\underline{\phi})$ with continuous partial derivatives, and satisfying

(i) $L(\underline{e}(t),\underline{\phi})$ *is locally positive definite in* $\underline{e}$ *(7.130b)*

(ii) $L(\underline{e}(t+\Delta t_L),\underline{\phi}) - L(\underline{e}(t),\underline{\phi}) \leq 0$ *(7.130c)*

for all $\underline{\phi} \in \underline{\Omega}_\phi$,

then the equilibrium state $\underline{e} = \underline{0}$ *is* ***stable.***

If, in addition,

(iii) $L(\underline{e}(t+\Delta t_L),\underline{\phi}) - L(\underline{e}(t),\underline{\phi}) < 0$ *(7.130d)*

for all $\underline{\phi} \in \underline{\Omega}_\phi$, *then the equilibrium* is ***asymptotically stable.***

The Lyapunov function $L(\underline{e}(t),\underline{\phi})$ is chosen to be a simple variation of the tracking error norm defined in section 7.4. A maximization over the parameter space is performed in order to make this norm a measure of the worst–case tracking error for all possible values of the uncertain coefficients.

$$L(\underline{e}(t),\underline{\phi}) = \underset{\underline{\phi}\in\underline{\Omega}_\phi}{Max}\left\| \underline{x}(\underline{\phi},t)-\underline{x}_d(t)\right\|_Q \qquad (7.131)$$

For this choice of the Lyapunov function, the criterion for robust asymptotic stability becomes,

$$\underset{\underline{\phi} \in \underline{\Omega}_{\phi}}{Max} \left\| \underline{x}(\underline{\phi},t+\Delta t_L)-\underline{x}_d(t+\Delta t_L) \right\|_Q - \underset{\underline{\phi} \in \underline{\Omega}_{\phi}}{Max} \left\| \underline{x}(\underline{\phi},t)-\underline{x}_d(t) \right\|_Q < 0 \qquad (7.132)$$

As in Proposition 2, the Lyapunov function $L(\underline{e}(t),\underline{\phi})$ is a positive definite function since $\underline{Q}$ is a positive definite matrix. It only remains to be shown that for an appropriate choice of the weights Q_i and the time interval Δt_L, there exists an admissible control input function $\underline{u}(t) \in \underline{\Omega}_u$ and a corresponding trajectory $\underline{x}(t) \in \underline{\Omega}_x$, such that the criterion above can be satisfied.

It is assumed that the desired trajectory is constrained to be feasible, and that a suitable tracking error envelope $\underline{x} \in \underline{\Omega}_x$ has been defined to provide adequate acceleration and error–rate margins according to the controllability conditions from section 7.3 which have been modified in equations (7.129a–f) to include the effects of parametric uncertainty. Under these conditions, robust controllability of the error dynamics assures the existence of an admissible robust tracking control solution. As before, the choice of Q_1, Q_2, Q_3, Q_4 and Δt_L must be constrained to assure that such a solution can also meet the robust stability criterion. The situation is analogous to that represented by Figure 7.2; the only difference being that the controllable region (R_C) is now smaller due to uncertainty.

The development of the constraining equations for the weights Q_i remains essentially unchanged from that presented in section 7.4. The only effect of the uncertainty in the coefficients is to change the achievable reduction of tracking errors in the time Δt_L. This results from the change in the error–rate margins for the case where uncertainty is included. Table 7.1, repeated below, appears the same as before, but the values of κ_V and κ_γ are computed using robust margins.

$\kappa_V\lambda_V Q_1 \;>\; [\lambda_\gamma^2 Q_2 + \delta_\omega^2 Q_3 + \delta_\theta^2 Q_4]$
$\kappa_\gamma\lambda_\gamma Q_2 \;>\; [\delta_\omega^2 Q_3 + \delta_\theta^2 Q_4]$
$\kappa_V\lambda_V Q_1 + \kappa_\gamma\lambda_\gamma Q_2 \;>\; [\delta_\omega^2 Q_3 + \delta_\theta^2 Q_4]$
$\lambda_\theta^2 Q_4 > \delta_\omega^2 Q_3$

where $\kappa_V = (\dot{e}_{V\text{marg}})\cdot\Delta t_L$, and $\kappa_\gamma = (\dot{e}_{\gamma\text{marg}})\cdot\Delta t_c$ (see 7.75 and 7.88).

These constraints guarantee that, if a tracking control solution exists and observes the robust controllability conditions (7.129a–f), then it will also satisfy the robust stability criterion (7.132), in the cases for which velocity or flight path errors exceed their targets.

The time interval used for the stability criterion (Δt_L) was given in section 7.4 (equation 7.104) as $\Delta t_L = \Delta t_i + \Delta t_a + \Delta t_c$, where Δt_i is the required time to attain desired control inputs, Δt_a is the time to achieve a desired attitude, and Δt_c is an additional convergence time (once desired error–rates have been achieved). Since these time intervals depend upon the controllability envelope and the acceleration margins, the calculation of Δt_L must be revised to account for parametric uncertainty. From (7.102) and (7.103),

$$Max[\Delta t_i] = Max\left\{\frac{(F2-F1)}{\dot{\delta}_{F\max}}, \frac{(M2-M1)}{\dot{\delta}_{M\max}}\right\} \tag{7.133}$$

$$Max[\Delta t_a] = \frac{1}{\delta_\omega}\left[\theta_{\max} - \theta_{\min}\right] + \left[\frac{2\delta_\omega}{\dot{\omega}^{+}_{\text{marg}}}\right] \tag{7.134}$$

where $F1$, $F2$, $M1$, $M2$, $\dot{\omega}^{+}_{\text{marg}}$, $\theta_{\max}$, and $\theta_{\min}$ are now values consistent with the robust controllability conditions (7.129a–f) presented in this section.

In short, equation (7.132) defines a robust stability criterion that requires acceptable trajectories to exhibit an overall decrease in tracking error (as measured by the Q–norm) over a specified time interval for any set of parameters within the parameter space. For an appropriate choice of the weights Q_i and the time interval Δt_L, it is also guaranteed that a trajectory which observes the robust controllability conditions and converges to the desired trajectory, will also satisfy the robust stability criterion.

Robust Convergence of the A* Algorithm

In order to account for uncertainty in the dynamics, the cost function for the AHSV trajectory optimization problem is redefined as shown below:

$$J = \sum_{k=1}^{N} \underset{\underline{\phi} \in \underline{\Omega}_{\phi}}{Max} \left\| \underline{x}(\underline{\phi}, k\Delta t_s) - \underline{x}_d(k\Delta t_s) \right\|_Q \tag{7.135}$$

As in Proposition 3, this cost function represents a cumulative sum of tracking errors measured by the Lyapunov function (Q–norm). The maximization over the parameter space makes this a worst–case cost. That is, for the case with uncertain parameters, each candidate input history $\underline{u}(t)$ (for $t \in [t_0, t_0+t_{max}]$) represents **not** only one trajectory, but a set of possible trajectories depending on the actual values of the parameters. The stepwise cost in equation (7.135) is the cost of the worst trajectory in this set (i.e. the one with the largest measure of tracking error). Therefore, if an optimal (admissible) trajectory for the overall cost function exists, then it has the property that its performance is robust to parametric uncertainty.

The accuracy of the predicted system response to a candidate input function $\underline{u}(t)$ naturally depends upon the uncertainty in the parameters. Moreover, since the prediction error due to the integration of uncertain dynamics is cumulative, it is clear that the reliability of a predicted trajectory degrades with the "look ahead" time. In fact, for any solution that converges to the desired trajectory (for nominal values of the parameters), there must be some time $t > t_0$ at which the state prediction uncertainty dominates the predicted tracking error. As such, it is possible that the worst–case predicted trajectories never reach any target envelope L_T as defined in section 7.5.

This concept is illustrated in Figure 7.6. The curve *e(t)* represents the nominal predicted tracking error (norm) for a particular candidate input function $\underline{u}(t)$. The prediction uncertainty $\Delta e(t)$ is shown by dashed curves which bound the range of possible tracking errors when parametric uncertainty is included. The figure shows a nominal trajectory which has converged to within the target zone by time $t = t_1$. At $t = t_2$, however, the uncertainty has already grown to a magnitude equivalent to the target error. Beyond $t = t_2$ the prediction uncertainty dominates, and the tracking error for the worst–case trajectory never reaches the target.

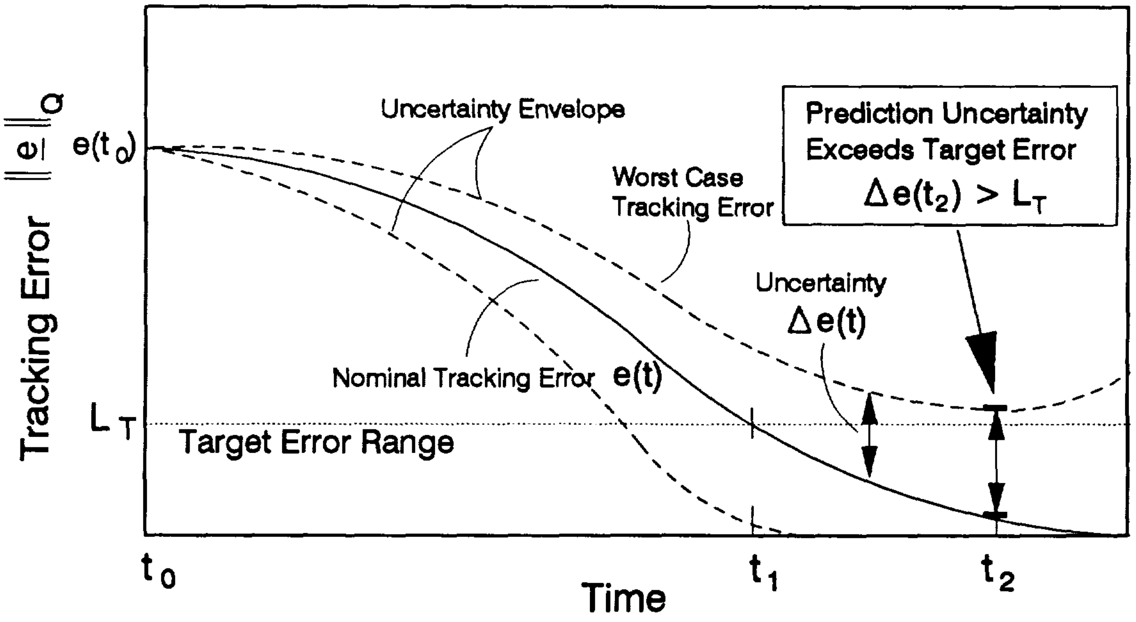

Tracking Error $\|\underline{e}\|_Q$
e(t₀)
L_T
Uncertainty Envelope
Worst Case Tracking Error
Prediction Uncertainty Exceeds Target Error
$\Delta e(t_2) > L_T$
Uncertainty
$\Delta e(t)$
Nominal Tracking Error e(t)
Target Error Range
t₀
t₁
t₂
Time

Figure 7.6: The Effect of Parametric Uncertainty on the Prediction Error

Based on the discussion above, it should be apparent that the definition of a target error envelope (as in section 7.5) does not constitute sufficient terminal conditions (or goal states) for solution trajectories when parametric uncertainty is considered. To rectify this problem the terminal conditions are modified as follows:

Robust Trajectory Terminal Conditions

The terminal state of a solution trajectory to the optimization problem must meet one of the following two conditions:

$$(1) \qquad \left\| \underline{x}(\hat{\underline{\phi}}, k\Delta t_s) - \underline{x}_d(k\Delta t_s) \right\|_Q < L_T \text{ for some } k > 0 \qquad (7.136a)$$

where,

$$L_T = [\, \lambda_V^2\, Q_1 + \lambda_\gamma^2\, Q_2 + \delta_\omega^2\, Q_3 + \delta_\theta^2\, Q_4 \,] \qquad (7.136b)$$

or,

$$(2a) \qquad \underset{\underline{\phi} \in \underline{\Omega}_\phi}{Max} \left\| \underline{x}(\underline{\phi}, k\Delta t_s) - \underline{x}_d(k\Delta t_s) \right\|_Q < \left\| \underline{e}(t_0) \right\|_Q \text{ for some } k \qquad (7.137a)$$

and,

$$(2b) \qquad \underset{\underline{\phi} \in \underline{\Omega}_\phi}{Max} \left\| \underline{x}(\underline{\phi}, k\Delta t_s) - \underline{x}_d(k\Delta t_s) \right\|_Q - \left\| \underline{x}(\hat{\underline{\phi}}, k\Delta t_s) - \underline{x}_d(k\Delta t_s) \right\|_Q \geq L_T \qquad (7.137b)$$

Condition (1) is the same target envelope terminal condition defined in section 7.5. Note that the measure of tracking error applies only to the nominal trajectory specified by the parameter estimates $\hat{\underline{\phi}}$. Condition (2) is a new terminal condition in which the worst–case predicted tracking error (at some future time $k\Delta t_s$) is less than the initial tracking error (2a), but the uncertainty in the predictions has become as large as the target error $\Delta e(t) \geq L_T$ (2b). This new condition is a weaker criterion that captures any candidate solutions that converge to the desired trajectory (in a robust sense) for a limited time, but are not explored deeply enough to meet condition (1) due to excessive prediction uncertainty.[36]

In general, for arbitrary parametric uncertainty, it is impossible to guarantee the existence of a robust control solution. This is the reason for leaving condition

[36] Note that there is a difference between the terminal conditions for a solution to the optimization problem and the terminal conditions for a trajectory. The former, given here, determine when the search considers a trajectory to be a viable solution. The latter defines the conditions for a complete trajectory. This distinction becomes vague in the uncertain case because it may not be possible to accurately predict far enough ahead to properly terminate candidate trajectories. All partial trajectories, however, are still extrapolated to the target envelope using the heuristic terminal functions (Figure 7. 5) before their cost is evaluated.

(1) in terms of the nominal trajectory. In other words, the algorithm will always consider nominally converging trajectories as (at least suboptimal) solutions. If the uncertainty is so great that all worst–case trajectories diverge, then the search will return with its best nominal solution. The addition of condition (2), then, can only add to the number of viable trajectories, and the convergence of the A* search to some solution is guaranteed.

If the uncertainty is small enough, the robust controllability conditions can be met, and the resulting error–rate margins (7.129a–f) assure the existence of a robust (admissible) tracking solution. Since the robust stability criterion is chosen (Q_{i},Δt_{L}) such that it is satisfied by such a trajectory, the first part of condition (2), equation (7.137a), can be met trivially for $k\Delta t_{\mathrm{s}} = \Delta t_{\mathrm{L}}$ (see 7.132). Now, depending on the degree of uncertainty, converging partial trajectories (meeting 2a) may reach a level of prediction error that satisfies condition (2b) well before condition (1) is reached. For the robust case, this is considered an acceptable solution since any predictions beyond this point are unreliable.

Convergence of the A* search for the robust optimization problem can now be established. First, the definitions of the terms in the heuristic evaluation function $f(n) = g(n) + h(n)$ are given for this case. That is,

$$g(n) = \sum_{k=1}^{k_{\mathrm{n}}} \underset{\underline{\phi} \in \underline{\Omega}_{\phi}}{Max} \left\| \underline{x}(\underline{\phi}, k\Delta t_{\mathrm{s}}) - \underline{x}_{\mathrm{d}}(k\Delta t_{\mathrm{s}}) \right\|_{Q} \tag{7.138a}$$

$$h(n) = E\left\{ \sum_{k=k_{\mathrm{n}}+1}^{N} \underset{\underline{\phi} \in \underline{\Omega}_{\phi}}{Max} \left\| \underline{x}(\underline{\phi}, k\Delta t_{\mathrm{s}}) - \underline{x}_{\mathrm{d}}(k\Delta t_{\mathrm{s}}) \right\|_{Q} \right\} \tag{7.138b}$$

where k_{n} indicates the depth of the node n in the graph, $g(n)$ is the worst–case predicted tracking error cost from the initial condition (Start node) to the state represented by the node n, and $h(n)$ is the estimated cost of completing the trajectory from the node n to a terminal node within the target envelope.

The cost estimate function $h(n)$ given in section 7.5 can still be used as an optimistic measure of the cost to complete a trajectory.

$$h(n) = \sum_{k=k_{\mathrm{n}}+1}^{N'} \left\| \left[\underline{x}(k_{\mathrm{n}}\Delta t_{\mathrm{s}}) + \underline{\dot{x}}^{\mathrm{h}} \cdot (k-k_{\mathrm{n}})\Delta t_{\mathrm{s}}\right] - \underline{x}_{\mathrm{d}}(k\Delta t_{\mathrm{s}}) \right\|_{Q} \qquad (7.139)$$

Here, N' is the index for which $\underline{x}(N'\Delta t_{\mathrm{s}})$ reaches the target envelope, and the vector $\underline{\dot{x}}^{\mathrm{h}}$ represents optimistic rates for each of the state variables. For this to remain an admissible heuristic function in the presence of parametric uncertainty, however, the calculation of $\underline{\dot{x}}^{\mathrm{h}}$ (see equations 7.119a–d) must be modified to include the uncertainty by maximizing or minimizing over the parameter space $\underline{\phi} \in \underline{\Omega}_{\phi}$ to obtain the best possible rates.

Finally, as in section 7.5, Theorem 6.7 guarantees that the A* algorithm will terminate with a solution (if one exists). Moreover, the search will converge to the optimal solution, according to Theorem 6.9, since the heuristic function $h(n)$ is admissible.

In summary, this section has augmented the results of the previous three sections to include the effects of parametric uncertainty in the analysis of the RIFC trajectory control system. The purpose has been to demonstrate that, if enough control authority is available so that constraints on the desired trajectory and a controllable tracking error envelope can be defined for all possible values of the uncertain coefficients, then, if a robust tracking solution exists to the optimization problem defined in Proposition 4, the A* algorithm will converge to this solution. Toward this end, robust versions of the controllability conditions, the stability criterion, and the tracking error cost function have been developed. It has been assumed for this analysis that the uncertainty in the parameters is small enough that the robust acceleration margins can actually be achieved. To account for uncertainty, some changes were required in the constraining equations for the weighting matrix, the time interval for the discrete Lyapunov function, and for the terminal conditions of solution trajectories.

In brief, the robust controllability conditions are used to constrain the operational envelope to assure the existence of control margins; the robust stability criterion identifies the admissible trajectories that converge toward the desired trajectory; and the robust cost function, associated terminal conditions, and cost estimate function *h(n)*, assure that the A^* search converges to a solution that minimizes the worst–case cumulative tracking error.

Finally, note that the analysis in this chapter is based on some conservative assumptions and bounds, and it may be possible (with significant additional effort) to find less restrictive controllability conditions. The purpose of this chapter, however, has been to show that the performance of the RIFC algorithm can be guaranteed if certain assumptions and conditions can be satisfied. Otherwise, the applicability of this approach to the AHSV trajectory control problem is not limited by any of the conditions in this chapter.

7.7 The Fundamental RIFC Algorithm

Conditions for which the important properties of controllability, stability, convergence, and robustness can be guaranteed for the RIFC control approach have been established in the previous sections of this chapter. As a consequence of this development certain details of the algorithm have been determined. These specifics include the choice of the stability criterion, cost function, maximum look–ahead time, and the relationship between the desired trajectory, tracking tolerances, and convergence margins.

The purpose of this section is to present the fundamental Robust Intelligent Flight Control algorithm which can be applied directly to the solution of the problem specified in section 7.1. The performance of this algorithm is guaranteed with the provision that the assumptions and conditions of the previous sections are satisfied. These conditions can be assured through a design process of selecting and balancing several constraints, margins, limits, and specifications for a particular vehicle configuration and desired trajectory. Note that the fundamental algorithm described here does not yet include any enhancements which will be required to deal with practical implementation issues such as time and memory limitations. These issues will be addressed in Chapter 8.

In order to apply the RIFC algorithm, the dimensions of the control input space, state space, and elapsed time, need to be divided into discretized intervals. The search graph must consist of a finite number of possible trajectories, and the level of discretization is clearly a trade off between computational resources and desired precision.[37] The branching degree of the graph is determined by the number of possible input combinations at each control cycle time–step Δt_s, and the depth of the tree is limited by the maximum look–ahead time t_{max} which is given by equation (7.112). The cost function is given by equation (7.123), and partial trajectories are verified with respect to all constraints, including the stability criterion (7.122), before they are considered candidates for further exploration. The full nonlinear equations (7.6a–d, 7.7a–k) are integrated forward to predict the vehicle response to control input histories generated by the search algorithm. Decisions regarding search direction are based on the A^* procedure described in section 6.2 (see flow chart – Figure 6.2). The cost estimate function used for the search is given by $f(n) = g(n) + h(n)$, where $g(n)$ and $h(n)$ are found in equations (7.138a) and (7.139).

The logical flow diagram for the fundamental RIFC algorithm is shown in Figure 7.7. At intervals of the control cycle–time Δt_s, a measurement of the current state is taken, the search procedure identifies an optimal short–term trajectory (of length $t_{max} = K\Delta t_s$), and the control solution is implemented. Each solution is implemented for only one interval of Δt_s seconds before the process repeats.

Each trajectory optimization begins with an initialization procedure which resets the initial conditions as well as other internal search variables.[38] A node expansion is then performed to generate all possible control inputs beginning with the start node (current vehicle state). These options are placed on a stack and removed one at a time for evaluation. For each possibility, the predictor is first initialized to the current state (node n) and the aerodynamic, propulsive, and

[37] An analysis of computational effort is also reserved until Chapter 8.

[38] In the enhanced RIFC algorithm the search begins with an initial guess for the optimal trajectory based on the previous solution.

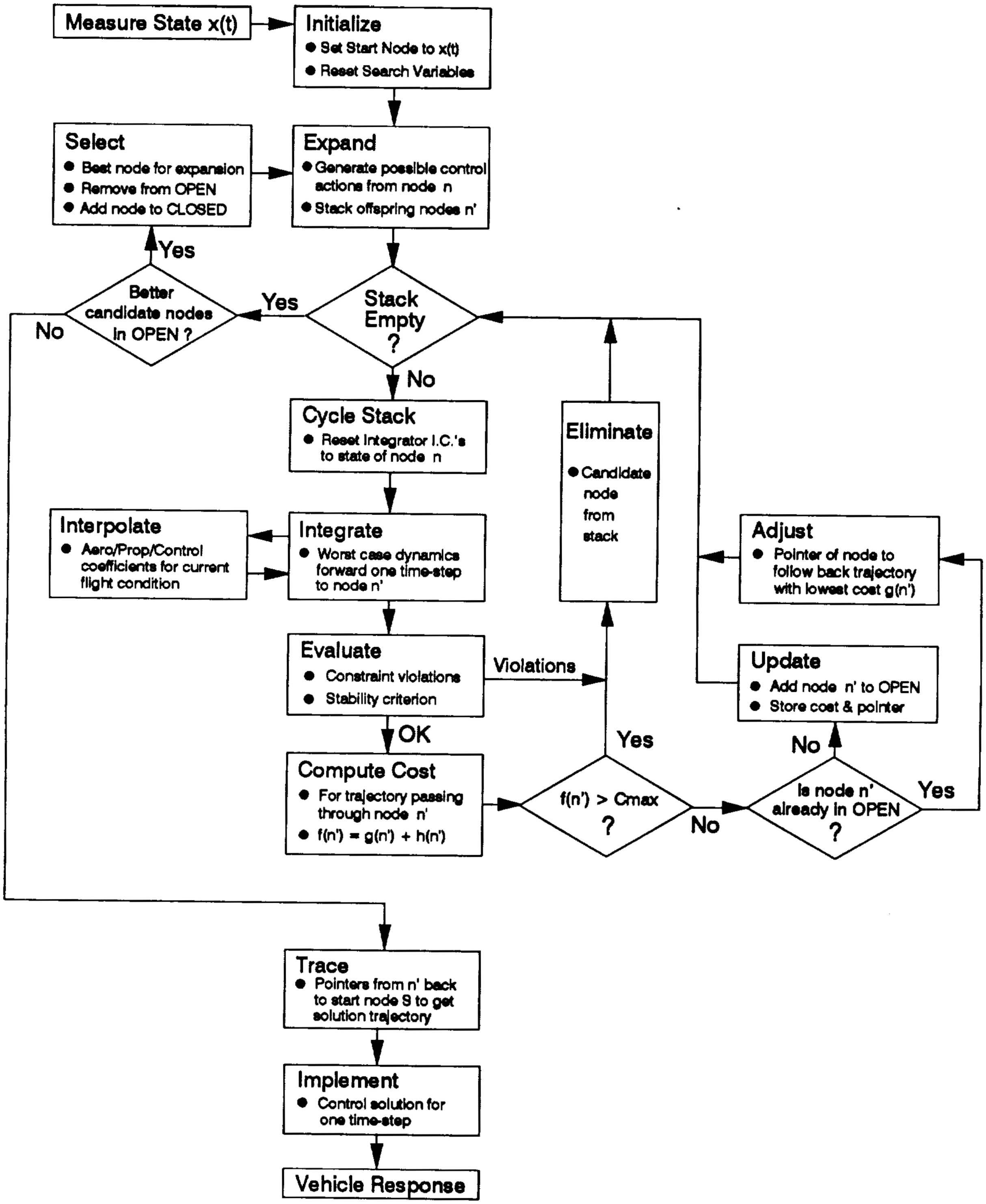

Measure State x(t)
Initialize
● Set Start Node to x(t)
● Reset Search Variables
Select
● Best node for expansion
● Remove from OPEN
● Add node to CLOSED
Expand
● Generate possible control actions from node n
● Stack offspring nodes n'
Yes
Better candidate nodes in OPEN ?
No
Yes
Stack Empty ?
No
Cycle Stack
● Reset Integrator I.C.'s to state of node n
Eliminate
● Candidate node from stack
Interpolate
● Aero/Prop/Control coefficients for current flight condition
Integrate
● Worst case dynamics forward one time-step to node n'
Adjust
● Pointer of node to follow back trajectory with lowest cost g(n')
Evaluate
● Constraint violations
● Stability criterion
Violations
Update
● Add node n' to OPEN
● Store cost & pointer
OK
Yes
No
Compute Cost
● For trajectory passing through node n'
● f(n') = g(n') + h(n')
f(n') > Cmax ?
No
Is node n' already in OPEN ?
Yes
Trace
● Pointers from n' back to start node S to get solution trajectory
Implement
● Control solution for one time-step
Vehicle Response

Figure 7.7: The Fundamental Robust Intelligent Flight Control Algorithm

control coefficients are interpolated from the database model for the current flight condition. The dynamics are then integrated forward one control cycle time–step (Δt_s),[39] and the resulting partial trajectory is evaluated for constraint violations. If the trajectory does not satisfy all the constraints, the corresponding offspring node n' is eliminated and the next control input is considered. Otherwise, the partial trajectory is completed using an optimistic/heuristic path that converges to the desired trajectory, and the full trajectory cost is computed. If this cost is greater than the cost of a solution which is already known, the trajectory is again eliminated. Otherwise, the state corresponding to node n' is checked to see if it has been reached before by another trajectory (node n' in set OPEN – see section 6.2). If it has, the trajectory with the lower cost is retained and the other is eliminated. Otherwise, the trajectory is stored as a candidate partial solution by adding the node n' to the set OPEN. This process is repeated for each possible control input from the current state until the stack of options has been exhausted.

Once all the one–step possibilities have been evaluated, the best one is compared to all other viable partial trajectories (i.e. other nodes in OPEN). If another node with a lower cost estimate exists then it is selected for expansion. The current state is set to the state corresponding to this "best" node, and all the control input options are generated. Each of these possible trajectories is then evaluated as described in the preceding paragraph. The search continues to expand, evaluate, and explore nodes until a trajectory is found with a cost that is lower than any other cost estimates. Since all cost estimates for partial trajectories are known to be optimistic, this implies that the optimal solution has been found. The solution is utilized for one time–step by implementing the control inputs for the first step of the trajectory. The entire process is then repeated with the next measurement of the vehicle state.

Based on the results in this chapter, the fundamental RIFC algorithm as described above is guaranteed to converge to the optimal trajectory provided that the assumptions and conditions of Propositions 1 through 4 are satisfied.

[39] The integration procedure can use any desired time-step $\Delta t < \Delta t_s$ to achieve the desired precision.

Chapter 8

Implementation Issues

The purpose of this chapter is to address a number of issues that arise in the context of the actual implementation of the RIFC controller. In Chapter 7, it was shown that the fundamental RIFC algorithm can be guaranteed to exhibit desirable convergence and robustness properties provided that certain controllability conditions are met. These conditions are not necessary[1]; but if they are satisfied, the algorithm will theoretically converge to a solution when one exists. From a practical viewpoint, however, theoretical convergence is not satisfactory. This is because other considerations, such as memory and time limitations, may prevent the search process from reaching its logical conclusion.

In this chapter, several enhancements are made to the fundamental RIFC algorithm (see section 7.7) in order to improve the search efficiency and guarantee convergence within a prescribed memory limit. Alternative methods for attaining the best possible solution with limited computation time are also suggested. The logical structure of the complete RIFC controller is then described, and its suitability for a parallel implementation is discussed. This chapter also examines some issues related to the quantization of the states and controls, and the controller bandwidth. Finally, the computational requirements of the RIFC approach are discussed.

[1] Less conservative conditions may be possible for which solutions still exist.

8.1 Practical Considerations: Memory & Time

8.1.1 GUARANTEED CONVERGENCE WITH LIMITED MEMORY

In section 6.2.3, the A^* procedure was described as a graph searching algorithm that utilizes two sets of nodes to keep track of candidate solutions: the OPEN set is a list of nodes that have been generated; while the CLOSED set is a list of nodes that have been explored (i.e. parent nodes). As the search progresses, nodes are selected on the basis of their estimated total cost $f(n)$, moved from OPEN to CLOSED, and then explored by generating all successor nodes.

The total number of nodes in the graph (N_T) is a simple function of the branching degree for each expansion (N_B), and the number of levels required (N) to reach terminal nodes. That is,

$$N_T = \sum_{i=1}^{N} N_B{}^i \tag{8.1}$$

The A^* algorithm, however, is effectively an optimal combination between a breadth– and depth–first search; therefore, only a fraction of the entire graph would actually be explored. This fraction depends upon the nature of the problem and the discriminating power of the heuristic cost estimate $h(n)$, where $f(n) = g(n)+h(n)$. As explained in section 6.2.4, the closer $h(n)$ comes to approximating $h^*(n)$, the more efficient the search.[2] Unfortunately, the specific number of nodes that must be explored cannot be computed in advance. It is still easy to see, however, that even for reasonably well informed cost estimates the actual memory required to store the lists (OPEN and CLOSED) can become enormous very quickly.

For the fundamental RIFC algorithm, the memory required is a function of the control cycle–time, maximum look–ahead time, quantization of the control inputs, cost estimate accuracy, as well as the constraints and the stability criterion.

[2] Recall that if $h(n) = h^*(n)$ the A^* algorithm would only explore the nodes along the optimal path.

The admissibility conditions on the states, the limitations of the control actuators, and the stability criterion, all contribute to a reduction in the possible number of nodes that must be explored and therefore stored in memory. Given a limited amount of computer memory to work with, however, it may still be necessary to truncate the list of OPEN nodes well before a solution has been found.[3] In this case, some fraction of the least promising (highest cost) nodes would simply be removed (pruned) from the graph in order to make room for new, hopefully, better candidate solutions. Clearly, if the only solutions happen to be in the set of truncated nodes, then the search will fail. Regardless of how unlikely it is that the optimal solution lies in the set of highest–cost partially–explored trajectories, any guarantees of convergence to the optimal solution are forfeit once nodes are truncated.

Since the dimensionality of the AHSV flight control problem is quite high, an alternative node storage scheme has been developed in order to circumvent the limitations otherwise imposed by computer memory constraints. Conceptually, this new approach takes advantage of several characteristics of the problem in order to compact the space of all possible trajectories into a time–invariant hypercube in the dimensions of the state tracking errors. The result is a drastically reduced memory requirement and, more importantly, the property that no candidate solutions are ever truncated unnecessarily (thus retaining the convergence guarantees inherent in the fundamental RIFC algorithm).

The main difference between the list storage scheme and the new scheme is in the dimensions of the storage array. In the original list structure, candidate nodes are enumerated on the basis of where they are in the solution graph. The sequence of arcs from the start node to a node n uniquely identifies that node. For the AHSV problem, it is, therefore, the sequence of control inputs (at each time–step) that uniquely identifies each candidate trajectory. This structure does have the advantage that node storage is dynamically allocated only for nodes that are actually generated. It suffers, however, from the list truncation problem when too many nodes are explored.

[3] If the available memory is large enough to store the entire tree, then this is not a problem.

In the new scheme, the nodes are stored in the output (state) space rather than the input (control) space. A grid is formed by quantizing the admissible state tracking errors in each dimension[4], discretizing time, and constructing an array with the tracking errors and time as the independent variables (i.e. indices). The grid storage location for any partial trajectory is then determined by the difference between the state associated with its deepest node and the desired state at that time. As will be shown below, it turns out that the time dimension of this grid can also be collapsed. Therefore the entire search space is reduced to a spacially quantized grid, which has only four dimensions for the longitudinal AHSV tracking control problem. A three–dimensional analog of this four–dimensional grid is illustrated in Figure 8.1. Note that since the axes of the grid are defined as the tracking *errors*, the center of the grid effectively follows the desired trajectory and always represents the zero error state. The limits on each axis represent the admissible state errors, and the target envelope is always some smaller region encompassing the origin.

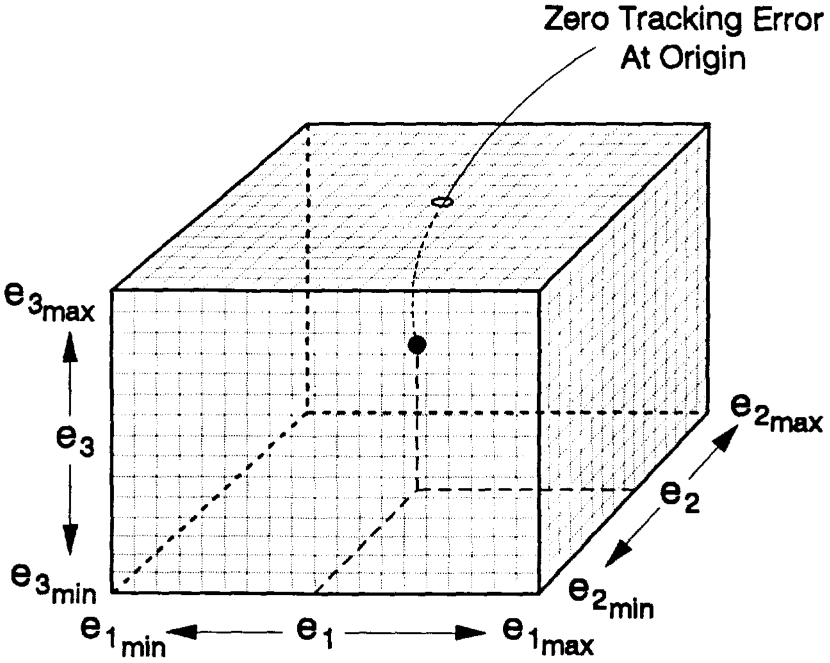


Figure 8.1: Node Storage Grid In State Tracking Error Space

[4] i. e. velocity, flight path, pitch rate, and attitude errors.

The greatest advantage of this new grid structure is that it recognizes overlapping trajectories in the state space, while the list structure enumerates all such trajectories separately. When two trajectories result in the same tracking error (in all axes) at the same time, it is clear that the path with the lowest cost leading to that state is preferable, and the more expensive trajectory can be eliminated. A similar argument holds for trajectories that reach the same error state at different times (as will be explained later). With the grid storage scheme, overlapping trajectories are handled quite easily, since it is not necessary to search through a long list of nodes to determine if an overlap has occurred. The node storage element is directly specified by the (quantized) state errors corresponding to the deepest node on the trajectory. That is, the memory location is a function of the state tracking error itself.[5] Therefore, overlapping trajectories automatically access the same memory location, and any contentions can be resolved directly as new nodes are stored. Since all admissible trajectories are representable within the grid structure, and any duplication is handled by retaining the lower cost paths, there is never any need to truncate candidate solutions using this new node storage scheme.

Another important advantage of this grid structure is that the quantization of the state space greatly reduces the complexity of the search. As explained in Chapter 6, the unenhanced A^* algorithm tends to spend much of its time deciding between trajectories with similar costs.[6] Since quantizing the states has the effect of grouping together closely adjacent trajectories (usually with similar costs), the total number of nodes is considerably reduced, and the search only has to compare trajectories that are significantly different.[7]

The use of a time–invariant grid can be justified in terms of the nature of the AHSV tracking control problem. Recall from Chapter 7 that the target envelope L_T

[5] Conveniently, this also means that the state information does not have to be stored, since it is coded into the memory address.

[6] In fact, this was one of the reasons for introducing the A_ϵ^* approach (section 6. 2. 5).

[7] What is "significant" is defined by the choice of quantization intervals for each state error (see section 8. 3).

is defined as an ellipsoid within the tracking error space. At any given time, the objective is to reduce the tracking errors in the sense that the Lyapunov function decreases until $L(\underline{x}(t)-\underline{x}_d(t)) < L_T$ (see equations 7.106, 7.107). Since the state and dynamic constraints, actuator capabilities, and uncertainties are assumed to be time–invariant (on the scale of the short–term correcting trajectory), the time at which a specific tracking error is reached is not as important as the cost. Therefore, if two trajectories reach the same error state at different times, the lower cost path can be saved and the other eliminated. This property holds for time–invariant tracking problems, and has the advantage that it eliminates the need for a time dimension in the node storage grid.[8]

There is one notable situation in which it would not necessarily be desirable to retain the lowest cost path to a particular node in the grid. This occurs if the state (represented by the node) is close enough to a constraint, and is moving in such a direction, that a violation can only be prevented by certain combinations of the control inputs. Although the required control authority to reduce tracking errors is assured by the controllability conditions, it may not be available quickly enough due to input rate limitations. In this case, the state of the controls becomes important, and the lowest cost path to the node could easily leave the inputs in the worst possible state. An obvious solution would be to add the dimensions of the control state to the storage grid. This is not a desirable option, however, since the required memory would tend to defeat the other advantages of this structure. Another

[8] Collapsing the time dimension of the grid does eliminate certain types of solutions from the tree. One can imagine situations where the smoothest trajectory actually "cuts a corner" by anticipating a future change in the direction of the desired trajectory. On the time-invariant grid this type of solution might be superseded by another trajectory that reaches the same error state at an earlier time. In the context of the RIFC controller, however, this is not seen as a disadvantage. The controllability conditions would assure that both trajectories can be made to terminate within the target envelope provided that the state errors remained admissible to that point. Furthermore, the smooth trajectory may not satisfy the stability criterion, while the second trajectory would probably be the one with the lower cumulative cost. Since prediction uncertainty increases with look-ahead time, the faster converging trajectory would also be preferable even if the costs were identical. For these reasons, it is concluded that no desirable solutions will be unduly truncated by the policy of keeping only the lowest cost paths to any particular error state.

option is to simply let the search eliminate nodes that encounter these situations, because they represent "dangerous" flight conditions for the vehicle. Adopting this philosophy, a candidate trajectory is considered undesirable if it passes through a state that is sensitive enough to the actual control values that it may lead to constraint violations. With this qualification, retaining the lowest cost path to each node will not unduly truncate desirable solutions.

Another consideration related to the use of the node storage grid is the possibility of a trajectory overlap between a newly generated node and a node that has already been explored. Should this situation occur, all offspring of that node would have to be reevaluated. The situation is illustrated in Figure 8.2 where new paths (shown by dashed lines) intersect other paths (solid lines) that have already been explored to deeper levels.

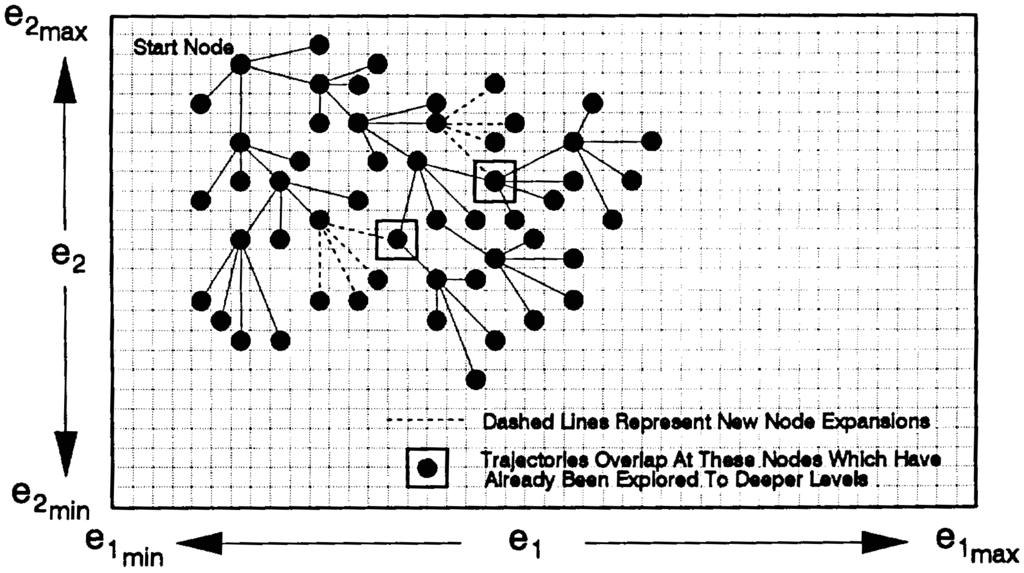

e_{2max}
Start Node
e_2
Dashed Lines Represent New Node Expansions
Trajectories Overlap At These Nodes Which Have Already Been Explored To Deeper Levels
e_{2min}
e_{1min}
e_1
e_{1max}

Figure 8.2: Overlapping Trajectories in the Error Space (2–D)

For the RIFC controller, this possibility is avoided by choosing the heuristic cost estimate to be a monotonic function. From Definition 6.13, a *heuristic function* $h(\cdot)$ *is monotonic if it satisfies* $h(n) \leq c(n,n') + h(n')$ *for all* n,n' *such that* n' *is a successor to node* n. Intuitively, this means that the cost estimate improves as the path gets closer to a solution. According to Theorem 6.11, *an* A^* *algorithm using a*

monotonic heuristic function finds optimal paths to each expanded node, that is, *$g(n) = g^*(n)$.* In other words, a node n will not be expanded until the optimal path from the start node to n has been found. Therefore, for an A* search with monotonic $h(n)$, partial trajectories can only overlap at unexplored nodes, and the problem of reevaluating offspring nodes is nonexistent. Figure 8.3 illustrates the situation for a monotonic heuristic function.

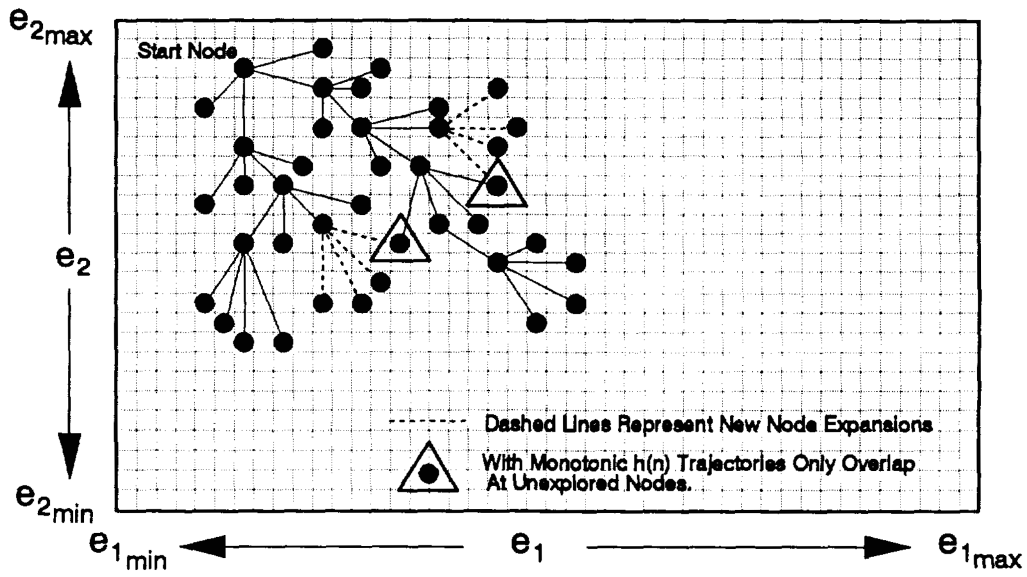

e_{2max}
Start Node
e_2
e_{2min}
Dashed Lines Represent New Node Expansions
With Monotonic h(n) Trajectories Only Overlap At Unexplored Nodes.
e_{1min}
e_1
e_{1max}

Figure 8.3: Overlapping Trajectories with a Monotonic Heuristic Function

It only remains to verify that the heuristic cost estimate function $h(n)$, as defined in Chapter 7 for the RIFC algorithm, is monotonic. Recall from equation (7.139) that,

$$h(n) = \sum_{k=k_n+1}^{N} \left\| \left[\underline{x}(k_n \Delta t_s) + \dot{\underline{x}}^{h} \cdot (k-k_n) \Delta t_s \right] - \underline{x}_d(k \Delta t_s) \right\|_Q \tag{8.2}$$

In order to simplify the notation, note that the term in the brackets represents the optimistic state history beginning with the next state. At the k*th* time–step this term will be abbreviated by $\hat{\underline{x}}^{h}(k)$, and the desired state will be written as $\underline{x}_d(k)$. Therefore $h(n)$ can be expressed as,

$$h(n) = \sum_{k=k_{\mathrm{n}}+1}^{N} \left\| \hat{\underline{x}}^{\mathrm{h}}(k)-\underline{x}_{\mathrm{d}}(k) \right\|_{Q} \tag{8.3}$$

and expanding this out for one term gives,

$$h(n) = \left\| \hat{\underline{x}}^{\mathrm{h}}(k_{\mathrm{n}}+1)-\underline{x}_{\mathrm{d}}(k_{\mathrm{n}}+1) \right\|_{Q} + \sum_{k=k_{\mathrm{n}}+2}^{N} \left\| \hat{\underline{x}}^{\mathrm{h}}(k)-\underline{x}_{\mathrm{d}}(k) \right\|_{Q} \tag{8.4}$$

Now, the first term of equation (8.4) is an optimistic cost for a single step of the trajectory starting from node *n*. If node *n′* is a successor to node *n*, then the actual cost from *n* to *n′* must exceed this term. That is,

$$\left\| \hat{\underline{x}}^{\mathrm{h}}(k_{\mathrm{n}}+1)-\underline{x}_{\mathrm{d}}(k_{\mathrm{n}}+1) \right\|_{Q} \leq c(n,n') \tag{8.5}$$

Also since $\hat{\underline{x}}^{\mathrm{h}}(k_{\mathrm{n}}+1)$ is optimistic, it must be closer to $\underline{x}_{\mathrm{d}}(k_{\mathrm{n}}+1)$ than the state corresponding to node *n′*. Therefore the second term from equation (8.4) would underestimate the cost *h(n′)*.

$$\sum_{k=k_{\mathrm{n}}+2}^{N} \left\| \hat{\underline{x}}^{\mathrm{h}}(k)-\underline{x}_{\mathrm{d}}(k) \right\|_{Q} \leq \sum_{k=k_{\mathrm{n}}'}^{N} \left\| \hat{\underline{x}}^{\mathrm{h}}(k)-\underline{x}_{\mathrm{d}}(k) \right\|_{Q} = h(n') \tag{8.6}$$

Combining equations (8.4), (8.5), and (8.6) gives,

$$h(n) \leq c(n,n')+h(n') \tag{8.7}$$

and, therefore, the heuristic cost estimate for the RIFC controller meets the conditions for monotonicity defined above.

To summarize, this section has motivated the use of an alternative node storage scheme which avoids the problems associated with the truncation of possible solutions due to memory limitations. The amount of memory[9] required is

[9] For each node storage element in the grid, only 14 bytes are required to represent an entire partial trajectory- one integer (2 bytes) is required to store a code which indicates the control inputs used at the last time-step; two floating point values (8 bytes) are needed to store the partial trajectory cost $g(n)$ and the estimated total cost $f(n)$; and a pointer (4 bytes) is used to refer back to the grid element of the parent node. By tracing back the pointers from any node to the start node the entire trajectory can be reconstructed.

drastically reduced[10], and is determined by the quantization of the admissible state tracking errors.[11] This quantization, in turn, reduces the complexity of the search by grouping adjacent trajectories together and reducing the number of possible nodes that may require exploration. The compactness of this scheme, its efficiency in handling overlapping trajectories, and its ability to maintain the convergence guarantees of the A^* algorithm, make it the preferable storage structure.

8.1.2 NONMINIMUM PHASE INSIGHTS

As a tail–controlled aircraft the AHSV exhibits a characteristic nonminimum phase behavior in the transfer function from elevon deflection to flight path angle (and altitude). Physically, this means that when elevons are used to achieve a desired normal acceleration ($V\dot{\gamma}$) through a reorientation of the vehicle attitude, the initial change in the flight path angle (γ) is in the wrong direction due to the force on the elevons. For example, a downward force on the elevons accelerates the vehicle down, as it applies a moment that will eventually pitch the nose up and increase the total lift due to the higher angle of attack. Once a certain attitude has been reached, the change in lift for the vehicle exceeds the force on the elevons, and the normal acceleration changes to the desired direction. Until this time, however, changes in flight path and altitude will be in a direction opposite to the desired effect.[12]

[10] The grid storage scheme typically requires far less memory because the maximum amount required is determined by the state quantization intervals, rather than the branching degree and depth of the search graph. For example, with the original list structure, assuming 1000 possible control actions at each state, and a depth of 10 time steps, equation 8.1 gives a maximum of approximately 10^{30} nodes. For the grid structure, even if the state errors were divided into 100 values on each axis, a maximum of 10^{8} nodes are possible. The experiments described in Chapter 9, in fact, successfully used a grid of only 160,000 nodes, which was easily stored in a conventional personal computer.

[11] The selection of these quantization intervals is addressed in section 8.4.

[12] For linear systems the nonminimum phase behavior corresponds to a zero in the right half complex plane. The frequency of this zero is only a function of the force and moment stability derivatives with respect to elevon deflection and angle of attack. These derivatives, in turn, only depend on the vehicle mass, pitching inertia, elevon moment arm to the center of mass, and the normal force derivatives with respect to angle of attack.

For the RIFC controller the nonminimum phase behavior is an issue because it means that trajectories which initially appear to diverge from the desired trajectory (in at least the γ state variable) may actually lead to optimal solutions. This is perfectly acceptable, and, in fact, the stability criterion of Chapter 7 is specifically chosen (Q_{i} and Δt_{L}) to account for this behavior.[13] The only problem is that for time t < Δt_{L} the stability criterion cannot be applied, and the search is left to explore all possibilities.[14] Since the A* algorithm explores the lowest cost partial trajectories first, it would exhaust all minimum phase possibilities before looking for nonminimum phase solutions. The result is that the algorithm wastes time in the initial stages of the search. Depending upon the number of control cycles in the interval $[0,\Delta t_{\mathrm{L}}]$, and the branching degree of each node (i.e. the number of input combinations), this could translate into a significant performance penalty.

In the interest of making the RIFC controller more efficient, it would be helpful to somehow inform the algorithm that it should be looking for nonminimum phase solutions in cases where the flight path error is outside its target range. This is not a simple objective, however, since the dynamics are very nonlinear and highly coupled in the controls. In addition, there may very well be satisfactory solutions using thrust vectoring and throttle controls that converge to the desired trajectory in a minimum phase fashion.

One possible approach, or heuristic, would be to integrate forward the effects of any control actions for t < Δt_{L} and evaluate the cost function as if each partial trajectory was explored to a depth of at least Δt_{L}. This could be done by considering constant controls, constant control rates, or pulsed control inputs for time steps less that Δt_{L}. In effect, this would limit the fidelity of the controller (for t < Δt_{L}) in order to save search time.

[13] The weights Q_{i} are biased to assure that a decrease in the Lyapunov function is possible with trajectories that allow deviations in any of the state errors as long as the controllability limits are observed. The time interval Δt_{L} is chosen to allow enough time to attain the desired acceleration margins for each state (see section 7.4).

[14] Recall that the control cycle time-step is Δt_{S}, which can be smaller than Δt_{L}.

A far better approach comes from examining the problem more carefully and realizing that the nonminimum phase behavior arises because of our choice of state variables. The AHSV dynamics in Chapters 4 and 7 are written in terms of the motions of the center of mass. This is very convenient because the translational and rotational dynamics can be decoupled. In Chapter 7 this is critical to the arguments for controllability because this decoupling is used to decouple the effects of the controls as well (into δ_F and δ_M). It turns out, however, that if the trajectory of a different point on the vehicle were considered, the nonminimum phase behavior could be eliminated altogether.

It can be shown that, for a tail–controlled vehicle, the frequency of the nonminimum phase zero increases as the point of reference moves forward from the center of mass (*CM*) toward the nose [McR1]. At a distance $X_c - X_{cm} = I_y/(M\, l_e)$ a center of rotation is reached, where the rotation of the vehicle exactly balances the translation from the elevon force.[15] For points forward of X_c the behavior is actually minimum phase; that is, a positive elevon deflection gives an immediate positive change in flight path angle. It is easily demonstrated that the center of rotation must be located on the vehicle somewhere between the center of mass and the nose. Consider the simple geometry shown in Figure 8.4.

The downward force on the elevons can be represented by an equivalent force and torque (of magnitude of $T_e = l_e F_e$) at the center of mass. The initial acceleration of the *CM* and the point *P* are found to be,

$$A_{cm} = -F_e/M \tag{8.8a}$$

$$A_P = -F_e/M + l_p l_e F_e/I_y \tag{8.8b}$$

which accounts for the acceleration of point *P* relative to the *CM*. Therefore, in order to have an upward acceleration at *P*, $A_P \geq 0$, it is required that,

$$-F_e/M + l_p l_e F_e/I_y \geq 0 \tag{8.9}$$

or

$$l_p \geq I_y/(M\, l_e) \tag{8.10}$$

[15] I_y is the pitching moment of inertia, M is the mass, and l_e is the elevon moment arm.

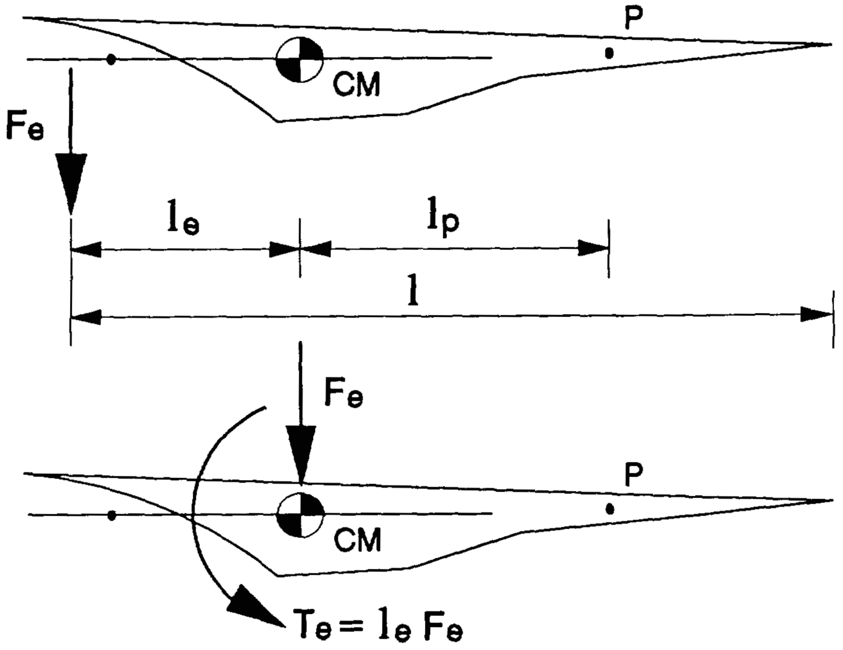


Figure 8.4: Center of Rotation for Elevon Moment

Now, assuming that the elevons act at the tail of the vehicle, it can be shown that the maximum possible value for the inertia is given by,[16]

$$I_y \le l_e \, (l-l_e) \, M \tag{8.11}$$

Therefore, in the worst case, equation (8.10) requires that $l_p = (l-l_e)$, which means that the point P is at the vehicle nose (with an acceleration $A_p = 0$). Since any realistic configuration would have a mass distribution with $I_y < l_e \, (l-l_e) \, M$, the nose of the vehicle accelerates upward, and the center of rotation must be somewhere between the *CM* and the nose.

[16] This is obtained by maximizing I_y with respect to the mass distribution, and using the defining equation for the center of mass as a constraint. Since the inertia will be largest if all the mass is located as far away from the CM as possible, the problem is simplified by considering two masses M_1 and M_2 at the tail and nose. Then, $M_1 + M_2 = M$, $l_e \, M_1 = (l-l_e) \, M_2$ since $X_{cm} = l_e$, and $I_y = l_e^2 \, M_1 + (l-l_e)^2 \, M_2$.

The discussion above suggests that an appropriate change of variables in the cost function could eliminate the nonminimum phase nature of the solutions and, therefore, greatly improve the search efficiency. For simplicity, and to assure a minimum phase behavior, the nose of the vehicle is chosen as the reference point P. The new states $\underline{x}_{\mathrm{P}} = [\,V_{\mathrm{P}}\ \gamma_{\mathrm{P}}\ \omega_{\mathrm{P}}\ \theta_{\mathrm{P}}]^{\mathrm{T}}$ can then be expressed as,

$$V_{\mathrm{P}} = [\,V^2 + l_{\mathrm{P}}^2\omega^2 + 2Vl_{\mathrm{P}}\omega sin(\gamma-\theta)]^{\frac{1}{2}} \tag{8.12a}$$

$$\gamma_{\mathrm{P}} = tan^{-1}\left[\frac{Vsin(\gamma)\ +\ l_{\mathrm{P}}\omega cos(\theta)}{Vcos(\gamma)\ -\ l_{\mathrm{P}}\omega sin(\theta)}\right] \tag{8.12b}$$

$$\omega_{\mathrm{P}} = \omega \tag{8.12c}$$

$$\theta_{\mathrm{P}} = \theta \tag{8.12d}$$

and the cost function from equation (7.135) becomes,

$$J = \sum_{k=1}^{N} \underset{\underline{\phi}\,\in\underline{\Omega}_{\Phi}}{Max} \left\| \underline{x}_{\mathrm{P}}(\underline{\phi},k\Delta t_{\mathrm{s}})-\underline{x}_{\mathrm{d}}(k\Delta t_{\mathrm{s}}) \right\|_{Q} \tag{8.13}$$

It is important to note that a change in the cost function does not change anything about the admissible trajectories. All that is affected is the ranking of the candidate trajectories according to a different measure. In fact, the controllability conditions from Chapter 7 can still be applied to the original dynamics. All bounds, margins, and constraints can be left in terms of the original state variables. The RIFC algorithm would even predict the vehicle response to candidate inputs using the original dynamics model. Therefore, if tracking solutions exist for the original problem, they are still present with the new cost function. The only assumption is that if the nose of the vehicle tracks the desired trajectory within target tolerances, then so does the rest of the vehicle.[17] To use the cost function above, all that is needed is the calculation of $\underline{x}_{\mathrm{P}}(t)$ once $\underline{x}(t)$ is known for a candidate trajectory (using 8.12). Then, based on this new cost, the order in which trajectories are searched will be different. The nonminimum phase behavior will no longer be able to adversely affect the A* search.

[17] That is, the tolerances on the overall trajectory from Earth to orbit are not so strict that it matters which part of the vehicle is on track.

Since the modified cost function is essentially a cumulative measure of tracking errors (using the Lyapunov function of Chapter 7 as a scalar indication of the state error at any point), it would also be advantageous if the stability criterion could be applied to the new states $\underline{x}_{\mathrm{p}}$ as well. This would eliminate the need to explicitly include a nonminimum phase time delay in the interval Δt_{L}[18], and the stability criterion could constrain the solution space at earlier levels of the search.

From equation (7.65) the Lyapunov function is given by,

$$L(\underline{e}(t)) = (V(t)-V_{\mathrm{d}}(t))^2\, Q_1 + (\gamma(t)-\gamma_{\mathrm{d}}(t))^2\, Q_2 \\ + (\omega(t)-\omega_{\mathrm{d}}(t))^2\, Q_3 + (\theta(t)-\theta_{\mathrm{d}}(t))^2\, Q_4 \qquad (8.14)$$

and the stability criterion is $L(\underline{e}(t+\Delta t_{\mathrm{L}}))-L(\underline{e}(t)) < 0$. Referring back to equations (8.12c–d), since $\omega_{\mathrm{p}} = \omega$ and $\theta_{\mathrm{p}} = \theta$ it is clear that these substitutions would not change the Lyapunov function. Furthermore, equation (8.12a) can be expressed as,

$$V_{\mathrm{p}} = V\left[1 + \left[\frac{l_{\mathrm{p}}^2\omega^2 + 2Vl_{\mathrm{p}}\omega \sin(\gamma-\theta)}{V^2}\right]\right]^{\frac{1}{2}} \qquad (8.15)$$

which, for hypersonic speeds, becomes $V_{\mathrm{p}} \approx V$ since the second term on the right hand side is negligible compared to 1. Therefore, V_{p}, ω_{p}, and θ_{p} can replace V, ω, and θ in the original Lyapunov function without affecting any of its properties. Of course, the main difference between the reference point P and the CM is in the flight path angle, and the substitution $\gamma_{\mathrm{p}} \approx \gamma$ cannot be made arbitrarily.

This problem can be solved, however, by observing that the weights Q_{i} for the Lyapunov function were chosen (section 7.4) in order to assure that the criterion $\Delta L(\Delta t_{\mathrm{L}}) < 0$ could be met by trajectories that satisfied the admissibility conditions for the states ($\underline{x} \in \underline{\Omega}_{\mathrm{x}}$). As such, the constraints derived for the weights are primarily a function of the bounds of this region. More specifically, these constraining equations depend on the bounds of the admissible state tracking errors $(\delta_V,\delta_\gamma,\delta_\omega,\delta_\theta)$, the limits of the target envelope $(\lambda_V,\lambda_\gamma,\lambda_\omega,\lambda_\theta)$, and indirectly on the

[18] Recall that Δt_{L} already includes a delay Δt_{i} for the rate limits on the control inputs, and another interval Δt_{a} to assure that all acceleration margins can be attained.

acceleration margins ($\dot{V}^{+/-}_{\mathrm{marg}}$, $\dot{\gamma}^{+/-}_{\mathrm{marg}}$) through the constants κ_V and κ_γ (see Table 7.1). Since there is effectively no difference between ($V_{\mathrm{p}},\omega_{\mathrm{p}},\theta_{\mathrm{p}}$) and ($V,\omega,\theta$), only the γ–related bounds and margins are of any concern here. It can also be argued that the normal acceleration margins at point P are, for all intents and purposes, approximately the same as those for the center of mass, that is $\dot{\gamma}^{+/-}_{\mathrm{pmarg}} \approx \dot{\gamma}^{+/-}_{\mathrm{marg}}$, and therefore $\kappa_{\gamma_{\mathrm{p}}} \approx \kappa_\gamma$.[19]

Now, since the target envelope is a design choice (the λ's), and the admissible tracking error limits (the δ's) are chosen solely on the basis of assuring controllability, the weights used in the Lyapunov function do not have to change in order to apply the stability criterion to the trajectory of the vehicle nose. The Lyapunov function for point P becomes,

$$L(\underline{e}_{\mathrm{p}}(t)) = (V_{\mathrm{p}}(t)-V_{\mathrm{d}}(t))^2\, Q_1 + (\gamma_{\mathrm{p}}(t)-\gamma_{\mathrm{d}}(t))^2\, Q_2 \\ + (\omega_{\mathrm{p}}(t)-\omega_{\mathrm{d}}(t))^2\, Q_3 + (\theta_{\mathrm{p}}(t)-\theta_{\mathrm{d}}(t))^2\, Q_4 \qquad (8.16)$$

and the criterion can be applied for $t > \Delta t_{\mathrm{L}}$, where $\Delta t_{\mathrm{L}} > \Delta t_{\mathrm{i}} + \Delta t_{\mathrm{a}}$ (i.e. no nonminimum phase time delay is necessary). Intuitively, the reason why the same stability criterion can be applied to a different point on the vehicle is as follows: the velocity, pitch rate, and attitude variables are effectively the same; the normal acceleration is different (as is the flight path), but the achievable margins are effectively the same since the additional contribution due to $\dot{\omega}$ is negligible at the angle of attack limits; finally, the weightings in the Lyapunov function only depend on these margins and the fixed bounds of the controllability and target envelopes.

[19] To first order, equation (8. 12b) can be written as $\gamma_{\mathrm{p}} \approx \gamma + l_{\mathrm{p}}\omega/V$, with first derivative $\dot{\gamma}_{\mathrm{p}}$ given approximately by,

$$\dot{\gamma}_{\mathrm{p}} \approx \dot{\gamma} + l_{\mathrm{p}}\dot{\omega}/V$$

Now, assuming (as in Chapter 7) that the largest normal accelerations are obtained at the maximum and minimum admissible angles of attack, then at these limits the acceleration caused by the term $l_{\mathrm{p}}\dot{\omega}$ is negligible compared to the normal acceleration due to lift. Therefore, the achievable acceleration margin is $\dot{\gamma}^{+/-}_{\mathrm{pmarg}} \approx \dot{\gamma}^{+/-}_{\mathrm{marg}}$, since the margins $\dot{\gamma}^{+/-}_{\mathrm{marg}}$ were derived under the same assumptions.

In conclusion, the efficiency of the RIFC control algorithm can be significantly improved by making a modification to the cost function such that the tracking performance is optimized for the nose of the vehicle rather than the center of mass. This change of output variables eliminates the nonminimum phase behavior and prevents the A^* procedure from exploring in inappropriate directions during the initial stages of the search.

8.1.3 ENHANCED SEARCH STRATEGIES

In this section, several possible enhancements to the RIFC algorithm are proposed for improving the performance of the search. Since the number of candidate trajectories that can be explored in a given time is a fixed quantity (that depends only upon the capabilities of the flight computer), the strategies presented here are aimed at finding the best possible solution in the least amount of time. In the following, it is assumed that for the real flight control problem there are actually many possible solutions that are acceptable.[20] Therefore, it may be advantageous to sacrifice some degree of optimality in order to find viable solutions more quickly.

Suboptimal Search A_ϵ^*

The A_ϵ^* algorithm is a variation of the A^* procedure in which optimality can be traded for search performance in a controllable fashion. It is based on the fact that the A^* algorithm typically wastes much of its effort deciding among nearly equivalent solutions. This property is overcome in the A_ϵ^* approach by requiring a minimum cost improvement (ϵ) before the search is forced to jump to alternative branches of the graph. Otherwise it probes to deeper levels of the tree by choosing one of the locally generated nodes for expansion. A detailed description is given in section 6.2.5.

[20] That is, there exists a reasonably large set of control input histories corresponding to trajectories that converge to the target envelope while observing the constraints.

To use this method, the node selection rule is modified, and a value for ϵ must be chosen. Instead of selecting nodes for expansion from the set of all OPEN nodes, the algorithm first compares the cost of the best LOCAL node[21] (C_L^*) to the best OPEN node (C^*). If the difference is significant,

$$C_L^* > (1+\epsilon)\ C^* \tag{8.17}$$

then the best OPEN node is explored. Otherwise, the best LOCAL node is used for the next expansion.[22]

The value of ϵ determines the fractional difference between the final solution found using A_ϵ^* and the cost of the optimal solution. This property is guaranteed by Theorem 6.13, which states that *the A_ϵ^* algorithm will converge to a suboptimal path costing no more than $(1+\epsilon)$ times the optimal cost, $f(T) \leq (1+\epsilon)\ C^*$.*

Unfortunately, there is no way to compute, *a priori*, the exact value of ϵ required in order to limit the search to a given fraction of the total number of nodes. Therefore, an acceptable balance between search performance and the optimality of the solutions may have to be determined empirically through repetitive simulation with different values of ϵ.

Hybrid Search

Another method of guiding the search to reach goal level depths more quickly is to combine best– and depth–first search decisions in stages throughout the solution graph. This approach forces the search to explore nodes from the LOCAL group for several intervals in a row, before another best–first cycle is executed using the best node from OPEN. The advantage of this approach is that the bias of the search towards breadth and depth can be easily controlled without affecting the

[21] The LOCAL group is defined in section 6.2.5 as the set of all nodes n' generated from the last expansion of a node n^*.

[22] Note that the terminology of OPEN and CLOSED sets of nodes can still be used in conjuction with the new node storage grid defined in section 8.1.1. OPEN is still the set of expanded (offspring) nodes, and CLOSED is the set of explored (parent) nodes.

convergence properties of the A^* algorithm. Depending on the nature of the problem, and the choice of a best/depth interval parameter (*BD*), the search could potentially find viable solutions almost immediately.

The search strategy in the RIFC controller can also perform *BD* depth–first expansions in sequence, and then select the next node for exploration based on the A_ϵ^* method. If the parameter $BD = 1$ this combination reduces to the A_ϵ^* algorithm, and if $BD = t_{max}/\Delta t_s$ the search explores every node it selects to deep enough levels that possible solution trajectories might emerge after only *BD* expansions.[23] This is at the risk, however, of spending too much time in local regions of the graph. The best choice for the *BD* parameter, as with the ϵ in A_ϵ^*, would have to be determined through simulation experience.

The fact that this hybrid search strategy can be used without affecting the convergence properties of the A^* algorithm can be understood as follows. If for each depth–first expansion, all of the LOCAL nodes are added to the set OPEN, then the effect of the depth–first excursions is only to add more nodes to OPEN between each best–first pass. The depth–first component of the search acts only as an interruption to the normal A^* procedure, except that it increases the number of candidate trajectories.

Cost Bounded Paths

A final technique for improving the efficiency of the A^* search is to upper bound the allowable cost of solution trajectories in cases when such a bound is known. For the AHSV tracking problem, estimating a useful upper bound to the cost function would be difficult *a priori*. However, during the search process, as soon as any trajectory that meets the constraints and terminal conditions is found, it is automatically an upper bound to the optimal solution cost (C^*). Therefore, in the RIFC algorithm, the cost of the best solution (C_{max}) is stored and updated as

[23] Recall that t_{max} is the maximum required look-ahead time to assure the existence of a solution provided the controllability conditions are satisfied (otherwise it is just the maximum look-ahead time). The interval Δt_s is the control cycle time-step.

the search progresses. It is then used as an additional constraint to eliminate other candidate trajectories that do not have an estimated total cost lower than this value ($f(n) < C_{\max}$). Since the function $f(n)$ is optimistic, any trajectories that fail to meet this condition cannot possibly improve on the solution that is already known. Using this rule, the algorithm can substantially reduce the search space as it works to find better trajectories.

8.1.4 STARTING SOLUTIONS

The final enhancement to the RIFC algorithm to be discussed is the addition of initial solution "guesses" to the search procedure. The idea is to start the algorithm off with a database of promising OPEN nodes, and perhaps even a solution trajectory.

The first initial guess is an **Open Loop Trajectory**. The control inputs are held constant at their present values, and the trajectory is predicted forward to $t_{\max}$ or until the target envelope is reached. Clearly if the state errors are within their tolerances, or they are already converging toward these values, then this trajectory may already be a solution. Even if this trajectory fails, however, it does provide some useful information. By simulating the entire open–loop response (to $t_{\max}$ or the target state), this trajectory determines an immediate upper bound to the acceptable cost of a solution trajectory.[24] The value of $C_{\max}$ (from the previous section) can be initialized with the cost of this trajectory, and the cost–bound constraint can be used in the search even though no solutions have yet been found.

The next initial guess is a **Local Gradient Trajectory**. This path is generated by performing a sequence of depth–first expansions all the way to $t_{\max}$ or until the target envelope has been reached. In other words, the path with the lowest cost from each node expansion is explored to deeper levels. Here too, it is possible that a solution is found immediately, and the trajectory provides another (probably better) upper bound to the complete trajectory cost $C_{\max}$.

[24] This is based on the reasonable assumption that, since the cost function is an integrated measure of tracking error, most admissible solutions would have a lower cost than trajectories that do not converge to the target envelope.

Finally, the most important initial guess is the **Previous Optimal Trajectory**. This path is based on the solution found by the last pass of the RIFC controller. Since the algorithm implements only the first time–step of each solution trajectory, it is beneficial to preserve the remainder of the solution, and to test its performance at the next time–step. Although the system state may not be exactly as predicted from the previous control cycle (due to uncertainties and disturbances), the previous solution may provide an excellent first guess. Since all the nodes along this path are added to the set of OPEN nodes, the search algorithm starts with a candidate trajectory that may only require minor modifications.

All of the above initial guess trajectories are implemented as part of the RIFC control algorithm. Each one is used to supplement the search with a number of promising candidate nodes, and the best of the three is used to initialize the upper bound cost ($C_{\max}$) for solution trajectories.

8.2 The Complete RIFC Autopilot

With the enhancements to the fundamental RIFC algorithm described in the previous section, it is now possible to present an overview of the complete RIFC autopilot. Since the controller software alone involves over *120* subroutines, for everything from storing nodes to interpolating coefficients, a detailed discussion of the software implementation will be omitted.[25] Instead, this section gives a functional description of the autopilot subsystems, and then reviews the higher level logic of the controller.

[25] This is not to say that the software implementation is without its own difficulties. In fact, a significant amount of effort was invested to make the code as efficient as possible. For example, many of the required routines are devoted to memory management, including some for coding and decoding trajectory information in order to store it with fewer bytes. Also, since this work was performed on a serial machine, the speed of many of the core subroutines was of major concern. For example, to speed up the node selection algorithm it is assisted by another routine which keeps up-to-date information on the location of the best nodes in the tree.

A functional partition of the RIFC autopilot is shown in Figure 8.5. At intervals of the control cycle time–step, the algorithm starts with the actual **Vehicle State** $\underline{x}(t_0)$, and upon completion of one pass of the autopilot it returns a **Control Solution** $\underline{u}(t)$ for $t \in [t_0, t_0+t_{\max}]$. The **Desired Trajectory** $\underline{x}_d(t)$ is also specified for $t \in [t_0, t_0+t_{\max}]$ at the beginning of each control cycle. A **Controller Configuration** block is used to initialize all search variables to the values specified by the user (see Chapter 5). Some examples of configuration parameters include the following: the admissible trajectory bounds and target ranges, the control actuator rate limits and bounds, the constraints, the search control values ϵ and *BD*, the maximum look–ahead time $t_{\max}$, and the weights for the cost function. The **Previous Solution** block is a time–shifted reproduction of the control input history from the last time–step's optimization. One time–step of this trajectory has been used and the remainder is now tested as a first guess for the current state.

As the search explores candidate trajectories, nodes which satisfy all of the required conditions are saved in the **Node Storage Grid**. As explained in section 8.1.1, the storage location for any node is a function of the state errors of the deepest node of any partial trajectory. A separate **Node Manager** is used to maintain an array of pointers to the best nodes in the main storage grid. This list is also updated as nodes are stored or eliminated, so that it can always be used by the **Node Selection** block in the decision of which node to explore next. This decision includes the considerations of the best– and depth–first strategies as well as the A_ϵ^* algorithm.

The **Node Expansion** block generates all possible options from any state that is being explored. This is done by enumerating all combinations of the quantized control inputs. Actually, it is the control rates, and not the values, that are quantized and used to generate possible trajectories. This results in continuous input histories that are more realistic, and the control state as well as the vehicle state can be updated by the **Trajectory Integration** block. This integration is used to predict only one control cycle time–step (Δt_s) forward for each offspring node at each expansion. Several integration steps (Δt) may be used, however, and for each

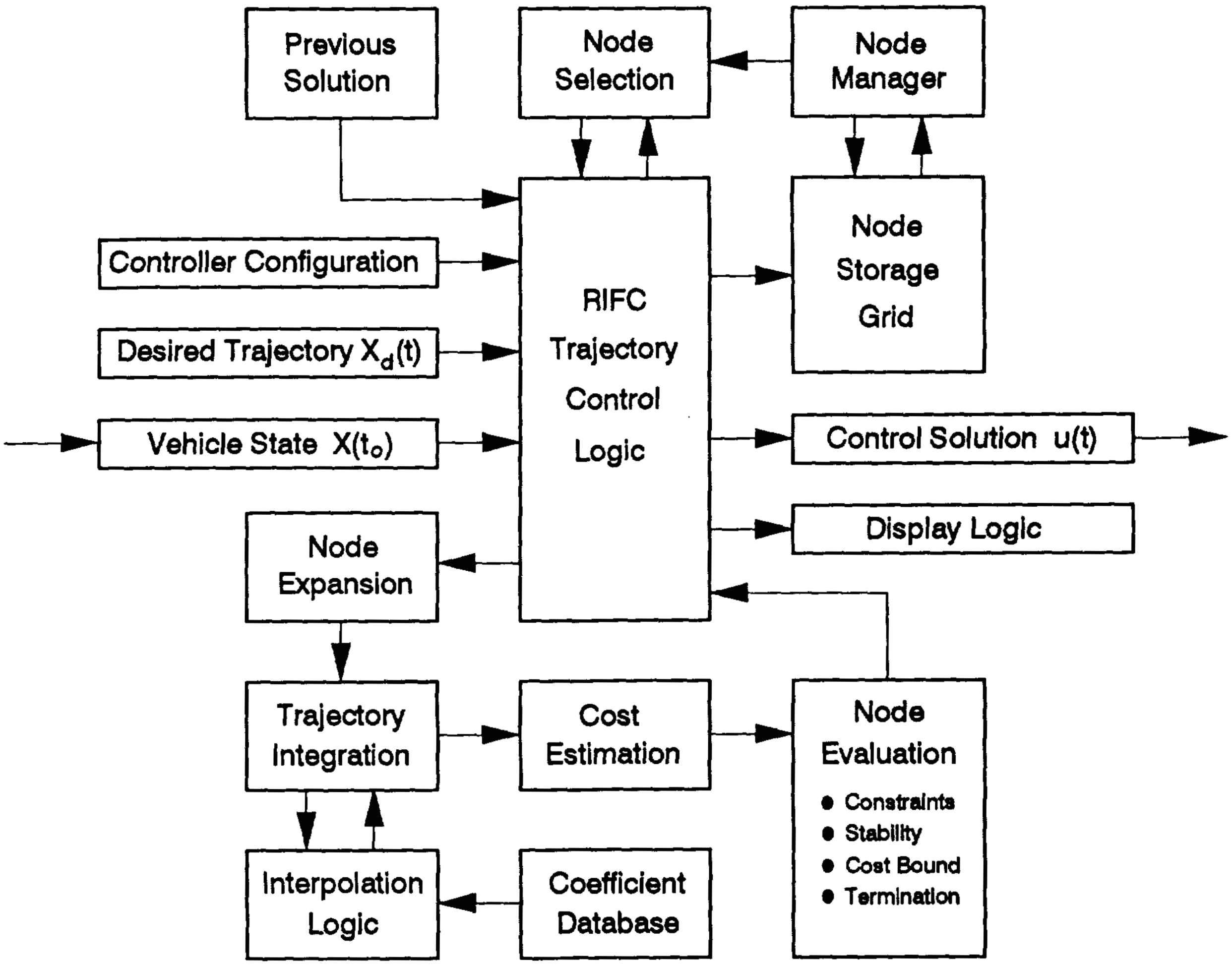


Figure 8.5: Functional Partition for the Complete RIFC Algorithm

of these, the **Interpolation Logic** must determine the values of all aerodynamic, propulsion, and control coefficients from the **Coefficient Database**.

Once each candidate offspring trajectory is integrated forward, the **Cost Estimation** block computes $g(n)$ by adding the one–step cost to the value of $g(n)$ for the parent node. The predicted cost to target $h(n)$ and the total estimated trajectory cost *f(n)* are also computed. Note that these costs are determined for the tracking errors of the vehicle nose, as discussed in section 8.1.2, in order to circumvent the problems associated with the nonminimum phase behavior. Finally, each candidate offspring goes through a **Node Evaluation**. If it does not verify all constraints and the stability criterion, then it is eliminated. This block also checks if the trajectory satisfies the terminal conditions and, therefore, qualifies as a solution.

The logical flow diagram for the complete RIFC algorithm is shown in Figure 8.6, which includes all of the enhancements introduced in this chapter. Since the essence of this figure is the same as that shown in Figure 7.7 (and described in section 7.7), only the differences will be mentioned here.

The first addition is the **Starting Solutions** block, which generates three full trajectories before the search process even begins. The **Open Loop**, **Local Gradient**, and **Previous Optimal** trajectories, described in the last section, are integrated forward from the initial state until the target envelope or the time t_{max} has been reached. All the nodes generated in the process of evaluating these trajectories are added to the node storage grid and become candidate partial trajectories for the search to explore further. In addition, the cost bound C_{max} is initialized to the cost of the best of the three trajectories.

Another change is the selection of the next node for expansion based on the best/depth parameter BD and the A_ϵ^* search. If the current pass is to be a depth–first cycle the next node is chosen from the last LOCAL group. Otherwise, if a significantly better node exists elsewhere (better by ϵ) then the search selects that node for expansion. If not, then the best node in the LOCAL group is used again.

Since the new node storage grid only affects the way in which the search graph is managed in memory, no changes to the flow chart are necessary in this regard. The sets OPEN and CLOSED are still used to distinguish between unexplored and parent nodes. Finally, the change of variables in the stability criterion and cost function is imbedded in the **Evaluate** and **Compute Cost** blocks.

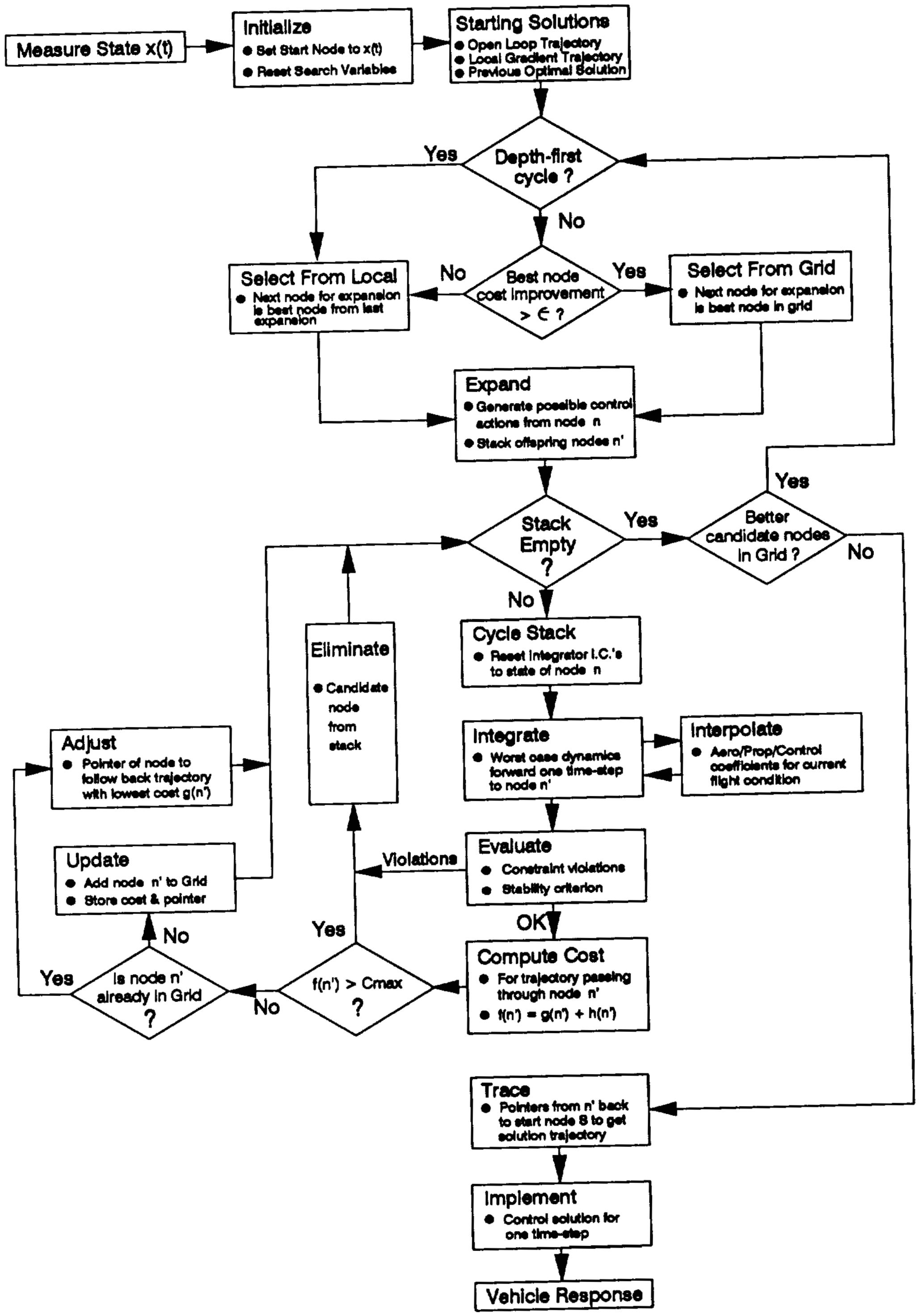

Measure State x(t)
Initialize
● Set Start Node to x(t)
● Reset Search Variables
Starting Solutions
● Open Loop Trajectory
● Local Gradient Trajectory
● Previous Optimal Solution
Yes
Depth-first cycle ?
No
Select From Local
● Next node for expansion is best node from last expansion
No
Best node cost improvement > ∈ ?
Yes
Select From Grid
● Next node for expansion is best node in grid
Expand
● Generate possible control actions from node n
● Stack offspring nodes n'
Yes
Stack Empty ?
Yes
Better candidate nodes in Grid ?
No
No
Cycle Stack
● Reset integrator I.C.'s to state of node n
Eliminate
● Candidate node from stack
Adjust
● Pointer of node to follow back trajectory with lowest cost g(n')
Integrate
● Worst case dynamics forward one time-step to node n'
Interpolate
● Aero/Prop/Control coefficients for current flight condition
Update
● Add node n' to Grid
● Store cost & pointer
Violations
Evaluate
● Constraint violations
● Stability criterion
OK
No
Yes
Yes
Is node n' already in Grid ?
No
f(n') > Cmax ?
Compute Cost
● For trajectory passing through node n'
● f(n') = g(n') + h(n')
Trace
● Pointers from n' back to start node S to get solution trajectory
Implement
● Control solution for one time-step
Vehicle Response

Figure 8.6: Logical Flow Diagram for the Complete RIFC Algorithm

8.3 Quantization, Bandwidth, and Feedback

The RIFC autopilot determines its control solution through a search over the space of input histories. For a fixed value of $t_{\max}$, the dimensions of the input space are a function of the control cycle time–interval (Δt_s) and the number of input combinations possible at any time (N_B). It has also been shown that the partial trajectories generated by the search can be stored in a grid with dimensions related to the quantization of the state tracking errors. It has not yet been explained, however, how the time–step Δt_s, and the quantization of the states ($\Delta \underline{x}$) and controls ($\Delta \underline{u}$), should be selected. The choice is affected by several factors, including the maximum look–ahead time, the properties of the disturbances, the desired bandwidth, and the limitations of the flight computer.

One important consideration, which indicates an interrelationship between the time–step and quantizations, is associated with the detectability of different inputs in the changes of the state. In other words, if two choices of the quantized inputs do not differ enough in their effect on the resulting state (after Δt_s seconds), then the quantized states may not register the difference. In this case, the control actions are effectively the same, and one of them is redundant. Qualitatively, it is clear that this situation can be avoided by making the $\Delta \underline{u}$'s larger, the $\Delta \underline{x}$'s smaller, or the time–step Δt_s longer. An approximate relationship between $\Delta \underline{x}$, $\Delta \underline{u}$, and Δt_s can be obtained from the original dynamic equations:

$$\dot{\underline{x}}(t) = \underline{f}(\underline{x},t) + \underline{g}(\underline{x},\underline{u},t) + \underline{d}(t) \tag{8.20}$$

Assuming the dynamics are time–invariant over the interval Δt_s, a first order approximation for the change in the state for any input $\underline{u}$ is given by,

$$\delta\underline{x}(\underline{u},\Delta t_s) \equiv \underline{x}(t+\Delta t_s)-\underline{x}(t) \approx [\underline{f}(\underline{x}) + \underline{g}(\underline{x},\underline{u}) + \underline{d}]\cdot\Delta t_s \tag{8.21}$$

Now, the change in the state due to a change in the control can be defined as,

$$\Delta\underline{x}_u = \delta\underline{x}(\underline{u}+\Delta\underline{u},\Delta t_s)-\delta\underline{x}(\underline{u},\Delta t_s) \tag{8.22}$$

which gives (to first order in $\underline{u}$),

$$\Delta\underline{x}_{u} \approx \left[\frac{\partial \underline{g}}{\partial \underline{u}}\right] \Delta\underline{u}\, \Delta t_s \tag{8.23}$$

This equation locally relates a change in the control input to its effect on the resulting states. Since it is desirable to have each quantized control increment cause a noticeable effect on the states, this relationship can be used to choose $\Delta\underline{x}$ given $\Delta\underline{u}$, or vice–versa (for a fixed Δt_s). For the AHSV problem, including uncertainty and the variation of $\partial\underline{g}/\partial\underline{u}$ over the admissible state space, equation (8.23) leads to the following approximate inequalities:

$$\Delta x_V \lesssim \underset{\underline{\phi}\in\underline{\Omega}_\phi}{Min}\ \underset{\underline{x}\in\underline{\Omega}_x}{Min}\ \underset{\underline{u}\in\underline{\Omega}_u}{Min} \left|\frac{\partial g_V}{\partial \delta_T}(\underline{x},\underline{u})\right| \Delta\delta_T\, \Delta t_s \tag{8.24a}$$

$$\Delta x_\omega \lesssim \underset{\underline{\phi}\in\underline{\Omega}_\phi}{Min}\ \underset{\underline{x}\in\underline{\Omega}_x}{Min}\ \underset{\underline{u}\in\underline{\Omega}_u}{Min} \left|\frac{\partial g_w}{\partial \delta e}(\underline{x},\underline{u})\right| \Delta\delta e\, \Delta t_s \tag{8.24b}$$

$$\Delta x_\omega \lesssim \underset{\underline{\phi}\in\underline{\Omega}_\phi}{Min}\ \underset{\underline{x}\in\underline{\Omega}_x}{Min}\ \underset{\underline{u}\in\underline{\Omega}_u}{Min} \left|\frac{\partial g_w}{\partial \delta v}(\underline{x},\underline{u})\right| \Delta\delta v\, \Delta t_s \tag{8.24c}$$

where, δ_T, δe, δv are the throttle, elevon, and thrust vector controls,
Δx_V, Δx_ω are the velocity and pitch rate state quantizations,
and g_V, g_w are the velocity and pitch rate components of $\underline{g}(\underline{x},\underline{u})$ defined in equations (7.7b) and (7.7f),

Equations (8.24a–c) assume, as in Chapter 7, that the throttle input is used primarily for its control coefficient in the velocity equation, and similarly for the elevon and thrust vector controls and the pitch rate equation. Since the controls do not directly enter the attitude and flight path equations, some additional inequalities are needed to guide the choice of quantization intervals for these states. Omitting the details, it is desired to have small enough attitude intervals to register discrete changes in $\omega(t)$, and likewise for flight path angle with attitude changes.[26]

$$\Delta x_\theta \lesssim \underset{\underline{\phi}\in\underline{\Omega}_\phi}{Min}\ \underset{\underline{x}\in\underline{\Omega}_x}{Min} \left|\frac{\partial f_\theta}{\partial \omega}\right| \Delta\underline{x}_\omega\, \Delta t_s \tag{8.25a}$$

[26] In equations (8. 25a-b), note that $g_\theta = g_\gamma = 0$ (see equations 7. 19a-d).

$$\Delta x_{\gamma} \lesssim \underset{\underset{\sim}{\phi} \in \underline{\Omega}_{\phi}}{Min} \; \underset{\underset{\sim}{x} \in \underline{\Omega}_{x}}{Min} \left| \frac{\partial f}{\partial \theta}\gamma \right| \Delta \underset{\sim}{x}_{\theta} \, \Delta t_s \tag{8.25b}$$

Assuming that the control cycle time Δt_s is known, equations (8.24a–c) and (8.25a–b) can be used iteratively to determine acceptable quantization intervals for the states and the controls. Another constraint that has not been mentioned above is that the product $(2\delta_V/\Delta x_V)\cdot(2\delta_\gamma/\Delta x_\gamma)\cdot(2\delta_\omega/\Delta x_\omega)\cdot(2\delta_\theta/\Delta x_\theta)$ must not exceed the available computer memory for node storage. In addition, it is required that these quantizations be small enough so that it is possible to meet the tolerances of the target envelope (i.e. $\Delta x_V \le \lambda_V$, $\Delta x_\gamma \le \lambda_\gamma$, $\Delta x_\omega \le \lambda_\omega$, and $\Delta x_\theta \le \lambda_\theta$).[27]

The choice for the interval Δt_s is a design tradeoff between the bandwidth of the RIFC controller and the computational requirements. If numerical effort was not a concern, then it would be desirable to recompute an optimal trajectory at small enough time intervals to correct for any significant deviations caused by the disturbances. This would be ideal, since the RIFC algorithm could act as both a feedback controller as well as a feed–forward trajectory planner. This may not be possible, however, depending upon the nature of the disturbances and the available computational resources. If good disturbance rejection requires a high bandwidth for the controller, then choosing a small enough Δt_s might cause the dimensions of the search space to become too large for the capabilities of the flight computer. In this case, a separate feedback controller would be needed to track a lower bandwidth feed–forward trajectory from the RIFC controller.

The smallest possible time–step Δt_s is therefore limited by computer, both in terms of memory and speed. The quantization of the state tracking error space is limited by available memory; and from equations (8.24a–c) and (8.25a–b) this, in turn, places a lower bound on Δt_s, since the maximum size of the $\Delta \underline{u}$ intervals are bounded by constraints. Even if memory was not a factor, then smaller values of

[27] These tolerances also indirectly limit the maximum values for the $\Delta \underline{u}$'s, since the controls should have enough fidelity to actually reach the target envelope. In practice, this is not a problem since the states change more slowly than the controls, and the final state is a function of the entire control sequence, which has many possible combinations even for large $\Delta \underline{u}$.

Δt_s would increase the number of nodes that would have to be explored. Since the size of the graph that can be searched in a given amount of time is limited by the speed of the computer, this also places a lower bound on Δt_s.

Another consideration for choosing Δt_s comes from the time interval Δt_L in the stability criterion. Recall that $\Delta t_L > \Delta t_i + \Delta t_a$, which is enough time to achieve any desired input (given actuator rate limits), and to attain a desired attitude. Since the stability criterion applies at intervals of Δt_L, it is required that $\Delta t_s \leq \Delta t_L$. Smaller values of Δt_s would allow higher fidelity in the controls between Δt_L intervals, but would also increase the number of levels required to reach a depth of t_{max}.

An upper bound for the Δt_s that would allow the RIFC algorithm to operate without a separate feedback controller can be obtained from the state equations $\dot{\underline{x}}(t) = \underline{f}(\underline{x},t) + \underline{g}(\underline{x},\underline{u},t) + \underline{d}(t)$. The maximum change in the states caused by the disturbances (in the time interval Δt_s) can be expressed as,

$$\Delta \underline{x}_d \leq \underline{d}_{max} \, \Delta t_s \tag{8.26}$$

where $\underline{d}_{max}$ represents the magnitude of the maximum possible disturbances. Now, if $\Delta \underline{x}_d < \Delta \underline{x}$, then the disturbances cannot affect the states enough, within the interval Δt_s, to require additional feedback. This requires,

$$\Delta t_s \leq \underset{i}{Min} \left[\frac{\Delta x_i}{d_{i_{max}}} \right] \tag{8.27}$$

which would assure that the control cycle time is fast enough to compensate for the disturbances.

If Δt_s cannot be chosen to satisfy (8.27), (8.24a–c), and (8.25a–b) without exceeding the limitations of the computer, then the RIFC algorithm should be used as a feed–forward trajectory planner with a lower bandwidth, and a separate feedback controller should be incorporated. The objective of the feedback would be to track the predicted RIFC correction trajectory in the presence of the disturbances. Since the RIFC trajectory is designed to observe all of the constraints, the feedback mechanism can be operated strictly as a local controller. Several control methods would be applicable, such as robust linear control techniques, or the nonlinear Sliding Mode approach. With these methods, however, the feedback controller design could be a major independent task.

A far simpler approach would be to use the RIFC algorithm in a single–step mode to provide its own feedback controller. That is, while the trajectory planner is working on the problem at intervals of Δt_s, it could also compute single–step optimal feedback controls at much faster intervals ($\Delta t'$). This is equivalent to doing one extra node expansion per time–step $\Delta t'$. The only difference is that the possible inputs would be quantized fractions of the $\Delta \underline{u}$ intervals used for the feed–forward trajectory. The main advantage of this approach is that very little additional software is required. Furthermore, the nonlinearities and uncertainty can be handled in exactly the same manner as in the feed–forward controller.

In summary, the selection of Δt_s, $\Delta \underline{x}$, and $\Delta \underline{u}$ is a design choice dependent on the properties of the disturbances and the limitations of the flight computer. Since the modelling of the disturbances is beyond the scope of this research, and the limitations of the computer would be specific to the vehicle, further analysis of the tradeoffs between Δt_s, $\Delta \underline{x}$, and $\Delta \underline{u}$ are left to the designer. Using the relationships derived above, however, the following guidelines can be used to choose appropriate values:

(1) Choose Δt_s based on upper bounds for the disturbances (8.27) and the criterion that $\Delta t_s \leq \Delta t_L$.

(2) Choose the quantizations $\Delta \underline{x}$ according to the available memory.

(3) Choose the quantizations $\Delta \underline{u}$ to satisfy equations (8.24a–c).

(4) Iterate between equations (8.24a–c) and (8.25a–b) searching for acceptable values for $\Delta \underline{x}$ and $\Delta \underline{u}$.

(5) If the required $\Delta \underline{x}$'s are too small, or similarly the $\Delta \underline{u}$'s too large, then increase Δt_s and go back to (2).

(6) If the final value of Δt_s does not satisfy (8.27) then a feedback controller is required to track the RIFC trajectory.

C-4

8.4 Parallel Implementation

Since the state–of–the–art in massively parallel computers is advancing at a rapid pace, the feasibility of applying numerically intensive algorithms to real–time flight control problems is slowly becoming a reality. Current parallel systems include as many as *16,000* processors, and are capable of speeds of up to *27,000* MFLOPS (million floating point operations per second). A variety of computer architectures are also available, including, for example, the connectionist type of machine which is designed to perform calculations on banks of data in parallel. How soon these capabilities will be available for flight qualified hardware is uncertain; at the current rate of progress, however, significantly advanced systems may be available well before the NASP or other hypersonic vehicles are ready.

An important advantage of the RIFC trajectory control algorithm is that it is highly suitable for parallel implementation. Many of the main functions for conducting the search can be performed on multiple nodes simultaneously. In addition, there are many subtasks within each function that can be parallelized.

Figure 8.7 illustrates the concept of a parallel structure for the RIFC algorithm. For each node expanded, a bank of node processing units would simultaneously explore each of the candidate offspring nodes. Each processor (or group of processors) would have parallel access to all of the blocks shown in the figure. For each possible control input, the dynamics would be integrated one step forward using interpolated values from the coefficient database. Each of the state equations can be computed simultaneously, and the coefficients themselves can be interpolated in parallel. Each node processor would then evaluate the constraints, cost function, and terminal conditions for each partial trajectory. Since there are several constraints to evaluate, these tasks can be parallelized as well. The nodes that correspond to admissible trajectories could then be saved in the node storage grid simultaneously.[28]

[28] Multiple attempts to address the same grid location could be handled by some priority scheme.

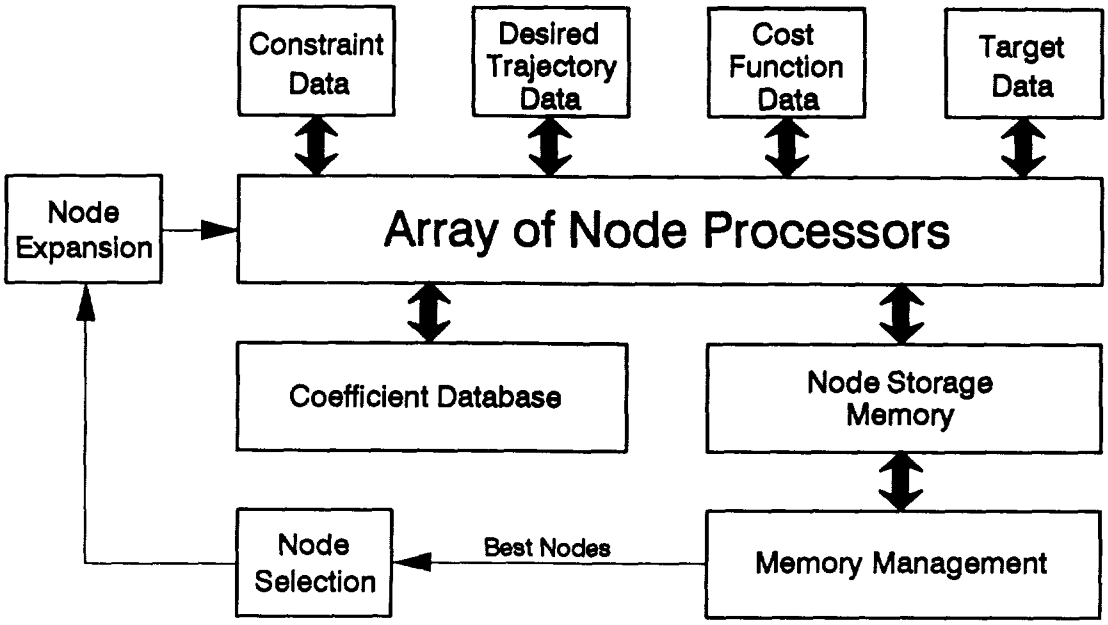


Figure 8.7: Parallel Implementation of the RIFC Algorithm

Also shown in Figure 8.7 is a separate node memory management block which could maintain an up–to–date list of the locations of the most promising nodes for further exploration. All operations related to memory management could be performed separately from, and in parallel with, the rest of the search procedure. Finally, if enough processors are available, it would also be possible to perform low level arithmetic tasks in a parallel fashion (as would be true of any algorithm).

8.5 Computational Requirements

It has been shown [Peal], that the A^* algorithm is the optimal search technique for solving minimization problems involving an additive cost function (in the sense that it expands the least number of nodes when compared to all other procedures given the same amount of information).[29] In general, however, there is no *a priori* means of determining how many nodes must be explored before a solution is found. From Theorem 6.12, it is known that the A^* algorithm will expand every node which satisfies the condition,

$$f(n) = g(n) + h(n) < C^* \tag{8.28}$$

Without knowing C^* in advance, and without specific information that makes it possible to determine which or how many nodes fail to meet (8.28), the only upper bound on the number of explored nodes is that which would correspond to a breadth–first search.

Let Z represent the number of explored nodes. Since A^* with any function $h(n)$, such that $0 < h(n) < h^*(n)$, would perform better than a breadth–first search (i.e. $h(n) = 0$), the expected value of Z is upper bounded by,

$$E\{ Z \} < \frac{1}{2} \left[\frac{N_B+1}{N_B-1}\right] N_B{}^N \tag{8.29}$$

where N_B is the branching degree, and N is the depth required to reach solution nodes. The right hand side of equation (8.29) is easily derived from equation (8.1) for a breadth–first search.

The minimum number of node explorations using A^* occurs when the heuristic cost estimate exactly equals the optimal terminating cost ($h(n) = h^*(n)$, for all n). In this case only nodes along the optimal path are explored and $Z = N$. Therefore,

$$N < E\{ Z \} < \frac{1}{2} \left[\frac{N_B+1}{N_B-1}\right] N_B{}^N \tag{8.30}$$

and the actual number of nodes explored will depend on the specific characteristics of the function $h(n)$ and the problem domain.

[29] This applies to the class of procedures which use a terminal cost estimation function $h(n)$ and are guaranteed to find a solution that is as good as A^*'s. The A^* algorithm is also the optimal best-first algorithm that is admissible when $h < h^*$.

Although equation (8.30) does very little to narrow the range of the expected complexity of A*, experience suggests that the A* algorithm typically explores only a small fraction of the total number of possible nodes. One approach for characterizing the search complexity is to use a probabilistic model of the likelihood that any particular node survives to be expanded by the A* search. In this model the values of $h(n)$ are assumed to be independent random variables having some probability distribution over the range $h(n) \in [0,\ h^*(n)]$. A survival rate (q) is defined to represent the probability that a particular node satisfies equation (8.28), in which case it would be expanded.[30] Based on these assumptions, a formula for the expected number of node expansions by A* is given in reference [Peal] as,

$$E\{ Z \} = N + \left[\frac{N_B - 1}{N_B}\right] \sum_{j=1}^{N} \sum_{d=1}^{j} {N_B}^{d} \prod_{k=1}^{d} q_{j,k} \tag{8.31}$$

This expression is derived by simply examining the expected number of offspring nodes that are generated by each off–course subtree of the overall graph. The first term (N) is the optimal path. The second term sums over the N non–optimal branches at each node along the optimal path, and counts up the expected number of surviving nodes for each subsequent expansion. If the survival rates are assumed to be the same for all nodes in the entire graph, equation (8.31) can be reduced to the following expression:

$$E\{ Z \} = \frac{N_B q \left({N_B}^{N} q^{N} - 1 \right)}{\left(N_B q - 1 \right)} \tag{8.32}$$

As an example consider the case for which the survival rate is $1/2$, that is each node has an even probability of satisfying $f(n) < C^*$. Comparing equation (8.32) to equation (8.29) a savings of a factor of $1/512$ (in general q^{1-N}) is realized using A* for a problem with only 10 levels ($N = 10$). For the AHSV flight control problem, this savings is further augmented by the state and control constraints, as well as the Lyapunov stability criterion, since these factors also reduce the probability that a

[30] The survival rate would of course be lower once constraints are included in the problem.

node survives for later exploration. The A_ϵ^* search has a similar effect, since the condition for node survival becomes $f(n)\cdot(1+\epsilon) < C^*$, which is more difficult to meet.

Unfortunately, although it suggests the potential for good performance, the probabilistic model above cannot guarantee an improved upper bound to the required number of node expansions by A^* without knowledge of the actual survival rates; and these will depend on the accuracy of the heuristic function $h(n)$ and the characteristics of the problem domain.

Fortunately, a significantly reduced upper bound is possible as a result of the state space quantization and node storage grid defined in this Chapter. In section 8.1.1 it was shown that re–expansion of nodes in the grid is never required due to the fact that the heuristic cost estimate $h(n)$ is monotonic. Therefore, the maximum number of nodes expanded by the search cannot exceed the number of nodes in the grid. Since this number is determined by the quantization of the state space, and not the branching degree of the input space, it is typically much smaller than the limits obtained using the A^* complexity formulas already given.

For example, consider the case $N = 10$, $N_B = 100$, $q = 1/2$, and a node storage grid with *20* divisions in each state error. Equation (8.32) gives $E\{ Z \} = 9.76\times10^{16}$ nodes, but it is known that no more than *160,000* nodes can be expanded since this is the maximum number in the grid. Clearly, for the approach taken here, the dimensions of the grid will usually determine the lowest upper bound. Experience shows, however, that even this number is very conservative. In Chapter 9 it is seen that most solution trajectories are found with only a few hundred node explorations. Even for the most difficult cases, optimal solutions were found in less than *2000* node expansions.

An estimate of the computational requirements for the RIFC algorithm can be obtained by decomposing the search procedure into its lower level functions and counting the number of operations required to evaluate each node. Since the actual calculations used to process any given node depend on the outcome of many tests (e.g. whether or not the constraints are satisfied), in each case the most (computationally) expensive possibility is always assumed, and the results below are therefore conservative. In addition, since there are many auxilliary variables and support routines that are not specifically involved in the evaluation of nodes (e.g. keeping track of the location and order of best nodes), a conservative computational overhead will also be assumed.

Referring back to Figure 8.6, the RIFC algorithm can be divided into *4* levels of computation according to the number of loops in which a particular section of code is imbedded. At the highest level, the software related to initialization and solution implementation is only executed once per control cycle. These routines (which include the blocks labelled **Initialize**, **Starting Solutions**, **Trace**, and **Implement**) comprise only a very small fraction of the required calculations. At the next level is the logic used for node selection and expansion. These routines are called for each cycle of the A^* search. Included in this category is the regeneration of the state and control history leading to the node that is being further explored. Functions at the third level are repeated for each offspring node. This is the main body of the search and it includes the blocks labelled **Cycle Stack**, **Evaluate**, **Compute Cost**, **Eliminate**, **Update**, and **Adjust**. The **Integrate** and **Interpolate** blocks are considered to be at a fourth level, since these routines are repeated for each of the multiple integration steps used to predict forward along each partial trajectory.

Using N_I to represent the number of integration steps per control step ($N_I = \Delta t_s / \Delta t$), Table 8.1 summarizes the computational requirements (in FLOPS) for each level described above.[31]

[31] Recall that N = Maximum search depth, N_B = Branching degree, and Z = Number of nodes explored.

Level 1:	Initialize	1058
	Starting Solutions	$3\cdot(238+N\cdot(50+2950\cdot N_I))$
	Trace	$N\cdot(56+2950\cdot N_I)$
	Implement	48
Level 2:	Node Selection	$220\cdot Z$
	Regenerating State	$N\cdot(50+2950\ N_I)\cdot Z$
	Node Expansion	$(326+4\cdot N_B)\cdot Z$
Level 3:	Cycle Stack	$4\cdot N_B\cdot Z$
	Evaluate	$238\cdot N_B\cdot Z$
	Compute Cost	$(80+136\cdot N)\cdot N_B\cdot Z$
	Store/Eliminate	$370\cdot N_B\cdot Z$
Level 4:	Integration	$814\cdot N_I\cdot N_B\cdot Z$
	Interpolation	$2136\cdot N_I\cdot N_B\cdot Z$

Table 8.1: Computational Requirements of the RIFC Algorithm

Combining these results, and allowing for an additional *20* percent of overhead software, gives the following expression for the maximum number of operations required:

$$FLOPS = 1.2 \times \left\{ Z\cdot\left[N_B\cdot\{2950\cdot N_I+136\cdot N+696\} + 2950\cdot N_I\cdot N + 50\cdot N + 546\right] + N\cdot\{11{,}800\cdot N_I+206\} + 1820 \right\} \tag{8.33}$$

For example, consider the case with *100* possible control actions at each state (N_B), *10* integration steps for each control step (N_I), and a maximum depth of *20* steps ahead (N). For this problem, approximately *4.5* MFLOPS are required for each node expansion. A parallel computer with *100* (N_B) processors running at *100* MFLOPS would be capable of performing over *2000* node expansions per second. Based on the results in the next Chapter, this would typically be adequate performance to obtain optimal solutions on–line.

Chapter 9

Results

The purpose of this chapter is to demonstrate the performance of the RIFC algorithm through a variety of flight experiments. First, however, the severity of the system nonlinearities is illustrated by examining a linearization of the vehicle dynamics. In addition, the limitations of a linear controller design approach are demonstrated. The RIFC controller is then evaluated in terms of the number of node expansions required for the search to find solutions. The benefits gained from each of the enhancements described in Chapter 8 are demonstrated, and the effect of including uncertainty in the model is explored. Finally, the performance of the RIFC controller is tested in simulated flight, and the results are compared to those for the SSOC controller.

9.1 Vehicle Design

In order to examine the RIFC algorithm within a consistent framework, a representative AHSV configuration is first selected and then used to obtain the results presented in this chapter. For this purpose, a design flight condition of Mach *10* at *30,000* meters altitude is chosen. This corresponds to one point along a trajectory from Earth to orbit which maintains a dynamic pressure of approximately

2000 psf.[1] For simplicity, a cruise configuration is designed such that it is possible to maintain a constant attitude and speed at this altitude. An equilibrium flight condition is desirable since it will be convenient to examine the optimization characteristics of the A* search in the context of a regulation problem.

The vehicle design (shown graphically in Figure 4.1) is specified by the parameter values shown in Table 9.1. This configuration was obtained by iterating on the design until the net forces and moments on the vehicle were approximately balanced (i.e. zero) for nominal values of the control inputs.

Vehicle Length	l_v	= 50.9 m	Ramp Angle 2	γ_2	= 10.0 °
Ving Span	b	= 35.3 m	Ramp Length 2	l_{f2}	= 8.01 m
Sweep Angle	δ	= 61.3 °	Nozzle Angle	β	= 25.8 °
Body Aspect Ratio	$\mathcal{AR}_B$	= 0.17	Nozzle Length	l_N	= 12.5 m
Combustor Vidth	ω_C	= 15.3 m	Nozzle Lip Length	l_l	= 2.50 m
Combustor Length	l_c	= 12.0 m	Inlet Height	h_i	= 0.29 m
Capture Height	h_c	= 5.00 m	Inlet Angle	γ_3	= 50.0 °
Ramp Angle 1	γ_1	= 7.00 °	Elevon Length	l_e	= 5.00 m
Ramp Length 1	l_{f1}	= 18.4 m	C.M. Bias	CM_b	= 0.33
Design Conditions: Mach 10, H = 30 Km, Angle Of Attack = 2.50 °					

Table 9.1: Representative AHSV Design Specifications

Once the design is selected, the performance of the scramjet engine (thrust, thrust angle, fuel mass flow rate, specific impulse, etc.) is tabulated for off–design conditions. This involves cycling through the propulsion model for a sequence of Mach numbers, angles of attack, altitudes, throttle settings, and thrust vector angles.[2] Interpolated values from the tabulated data are used to represent the scramjet engine during AHSV flight simulations.

Next, the controller's model of the dynamics (the coefficient database) is constructed by sampling the outputs of the full simulation at another set of state sequences. The coefficients (CD, CL, etc., listed in section 4.7) are determined and

[1] This is expected to be reasonable for AHSV's.

[2] The thrust vector angle is modulated using a variable nozzle lip length.

stored in a table. In order to develop uncertain models for the controller, the coefficients of this table are modified by multiplying each value by a random number as follows: $C^{i}_{\mathrm{estimate}} = (1+\delta_i)\, C^{i}_{\mathrm{actual}}$, where δ_i is a fraction ($|\delta_i| < \phi_i \leq 1$), and the ϕ_i's are the maximum uncertainty for each parameter. The ϕ_i's, which can be functions of state, are stored along with the coefficients to indicate the accuracy of each value.

Using the vehicle model described above, the nonlinear dynamics have been linearized numerically at a number of angles of attack within a few degrees of the design flight condition. The results, shown in Figure 9.1, dramatically illustrate the sensitivity of the linearized system to changes in attitude. The curves shown represent the movement of the phugoid and short period poles as a function of angle of attack. Note that only a two degree change in attitude significantly alters the dynamic properties of the longitudinal modes.[3] This variation indicates that the original system is strongly nonlinear in attitude and cannot be modelled accurately by a linear system even for small attitude changes.

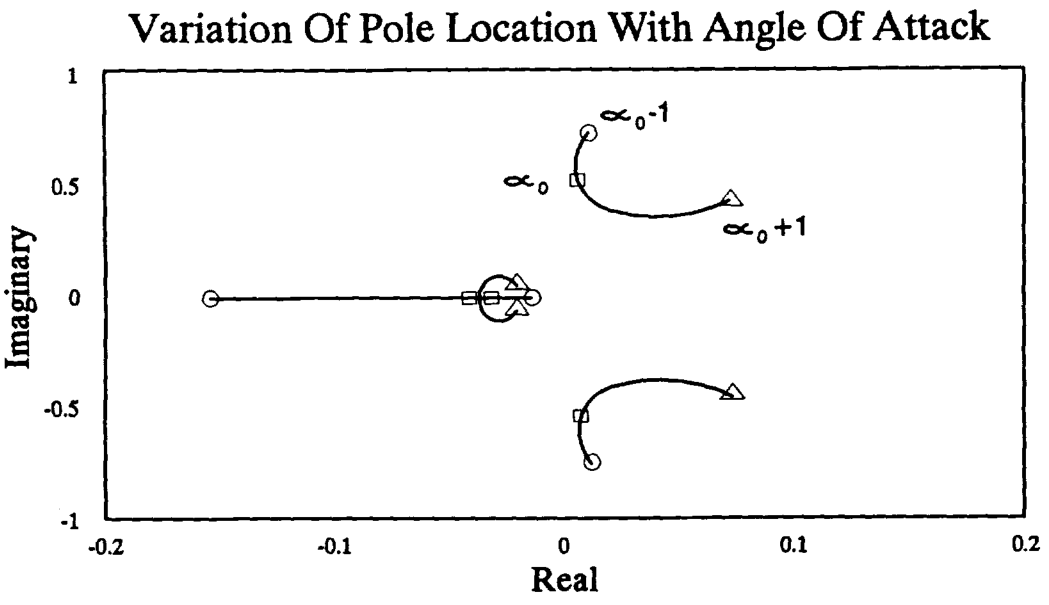


Figure 9.1: Sensitivity of the Linearized System to Attitude Variations

[3] Also note that the short period mode is unstable.

Despite the warning evident in Figure 9.1, a linear controller was designed and tested for perturbations about the equilibrium flight condition using the linear quadratic regulator (LQR) approach [Kwal]. In the LQR design, the state and control weighting matrices were carefully chosen such that simulations of the linearized system resulted in admissible trajectories without control constraint violations for reasonably large initial attitude errors. The LQR feedback gain matrix was then tested in the full nonlinear system with an initial attitude error. All other states were initially set to the equilibrium values and no disturbances were included.

Preliminary tests with the LQR controller failed, with the vehicle attitude rapidly diverging. One problem was that the linearization ($\dot{\underline{x}} = A\underline{x} + B\underline{u}$) is a poor representation for the effects of large control inputs. In particular the overall behavior of the elevon coefficients is not well approximated by a linearization near zero deflection angle (as shown in Figure 9.2). Since elevons are the primary actuators for attitude control, the controller may require significant elevon deflections to stabilize the system (even with small state errors). Taking this into consideration, a revised linear model was obtained by using larger control perturbations in the calculation of the B matrix. The state and control histories which result from a *1* degree initial attitude error are shown in Figure 9.3. This is compared with the response to a *1.2* degree initial error shown in Figure 9.4.

Figures 9.3 and 9.4 serve to demonstrate the difficulty of applying linear control methods to the flight control problem of air–breathing hypersonic vehicles. For such vehicles, the sensitivity of the dynamics to attitude invalidates the linear model for even small perturbations; and the linear controller fails.[4] In this example, the modelling error in the controls, and the control rate limits, were the initial cause for the attitude divergence. Then, once the attitude error became significant, these problems were compounded by extensive errors in the A matrix as well. Clearly, the addition of disturbances, uncertainty in the nonlinear parameters, and a larger (useful) operational envelope make the situation worse.

[4] Gain scheduling is the standard approach for dealing with changing dynamics. However, it is not really applicable to this situation, since attitude is a fast changing state. A controller scheduled with attitude would have no guarantees of stability.

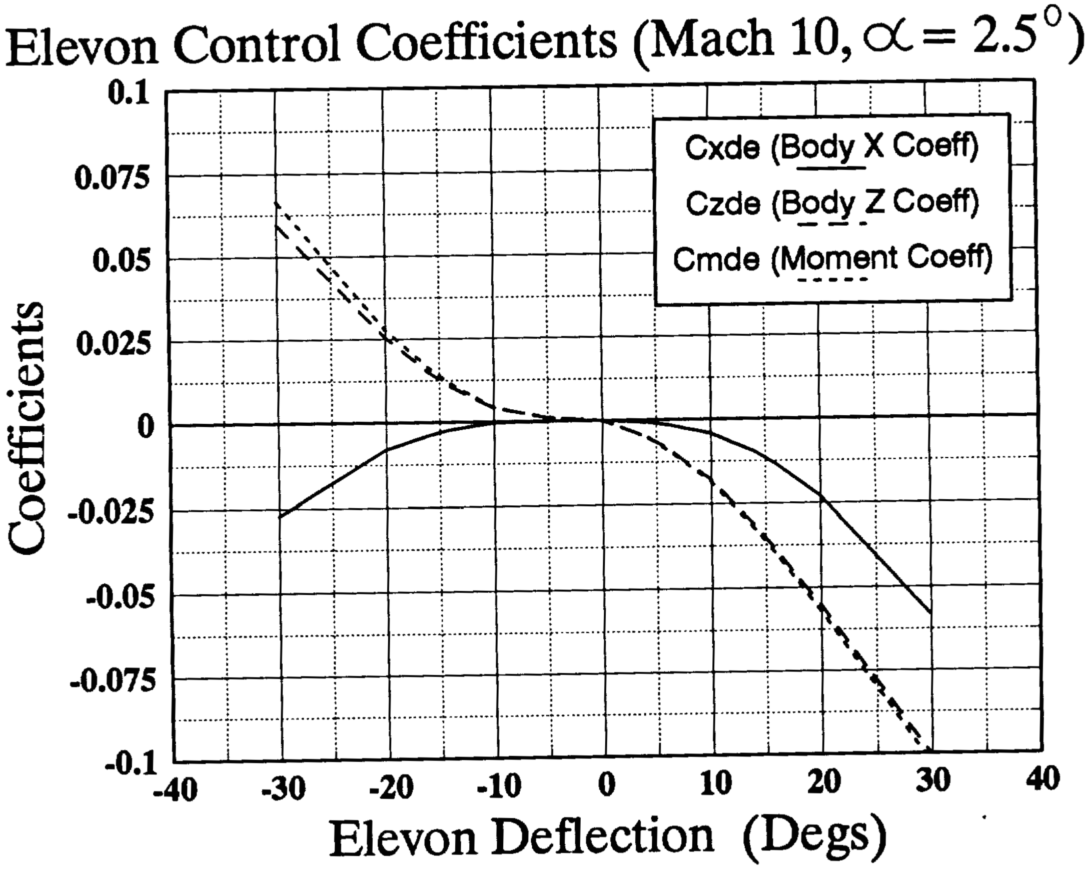
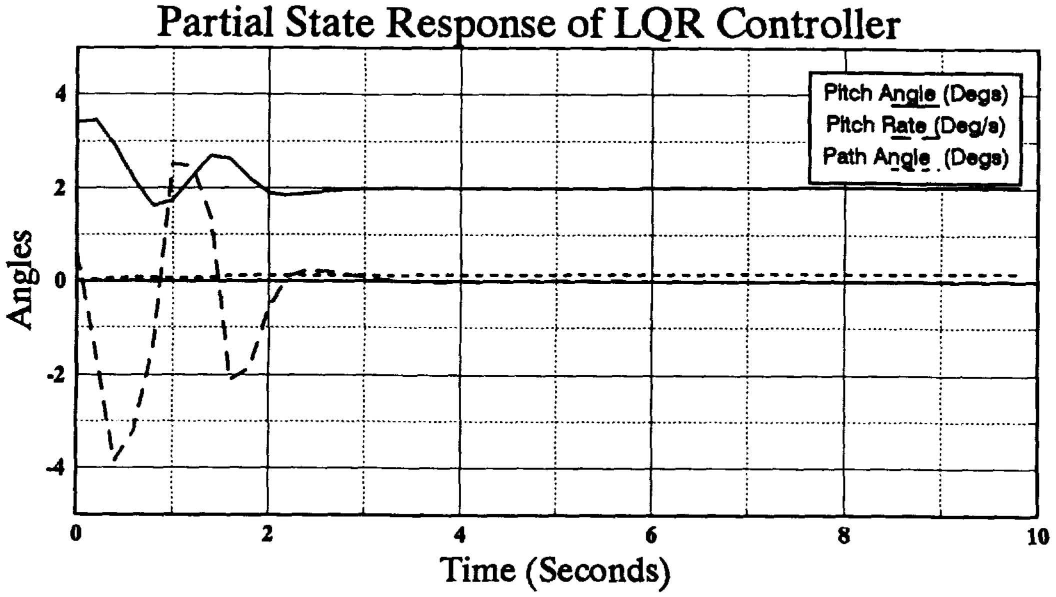


Figure 9.2: Elevon Control Coefficients vs Deflection Angle



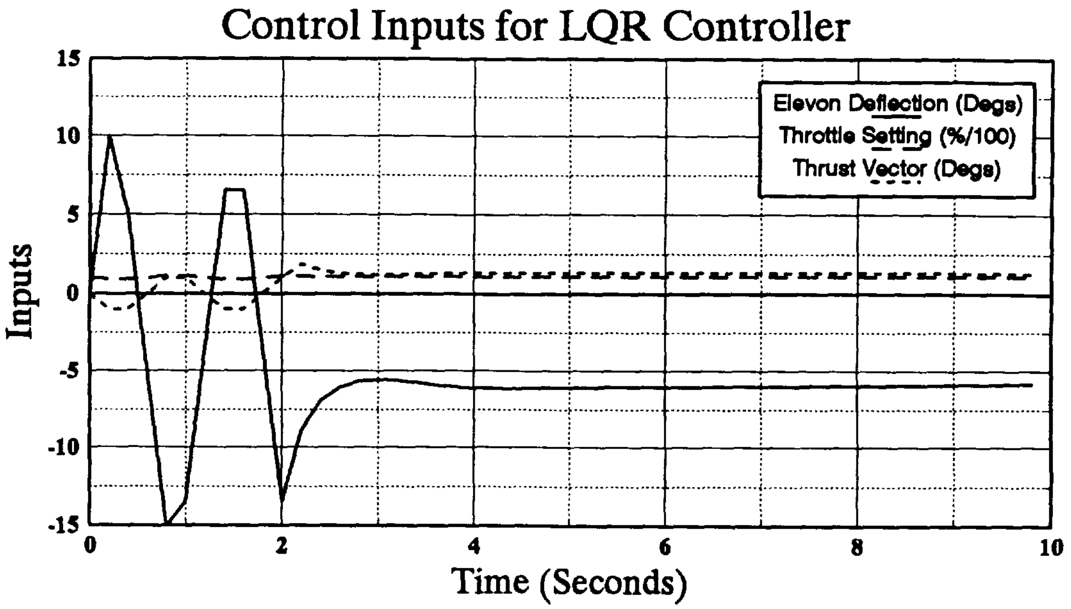
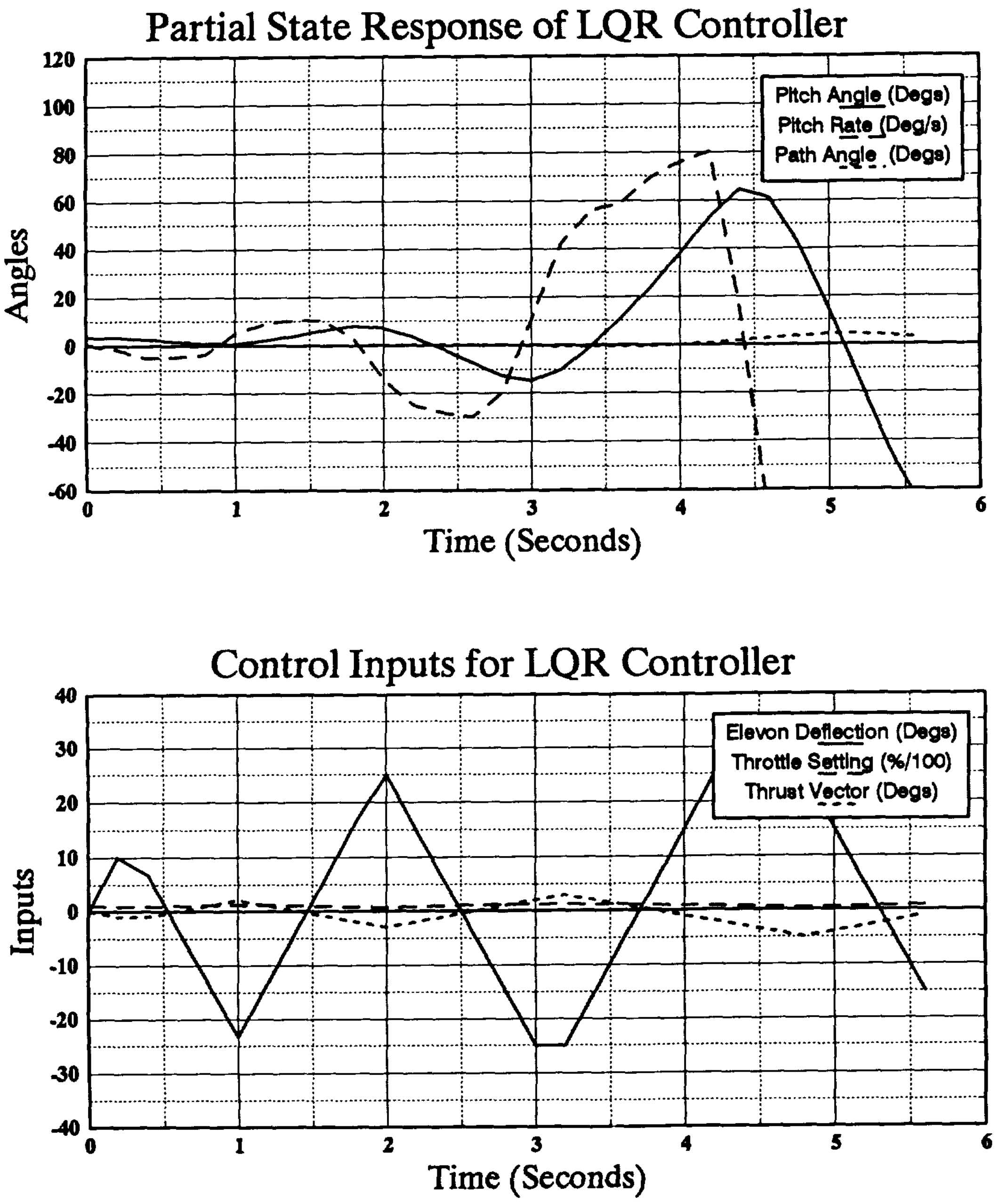


Figure 9.3: LQR State and Control Response to a 1.5 Degree Attitude Error


Partial State Response of LQR Controller

Control Inputs for LQR Controller

Figure 9.4: LQR State and Control Response to a 1.7 Degree Attitude Error

9.2 Optimization Characteristics

In this section the performance of the RIFC controller is analyzed in terms of the search effort required to find short–term solution trajectories. Toward this end, *15* series of *20* or more optimizations were performed in order to characterize the algorithm's performance under different conditions.

In each test case, the state and control constraints, weights for the cost function, control cycle time, quantization of the state space, and target envelope remained the same. The cost function weights were chosen, based on the analysis in Chapter 7, to assure that solutions which meet the controllability conditions could also satisfy the stability criterion. The state/control quantizations, and control cycle time, were obtained using the guidelines presented in Chapter 8, which balance the memory available with the precision required to identify changes of state between time steps and to reach the tolerances specified by the target. The initial conditions and fidelity of the controls were allowed to vary between tests.[5] More difficult problems were obtained by choosing initial conditions with larger tracking errors. Typically this implies a search to deeper levels in order to reach states within the target envelope.

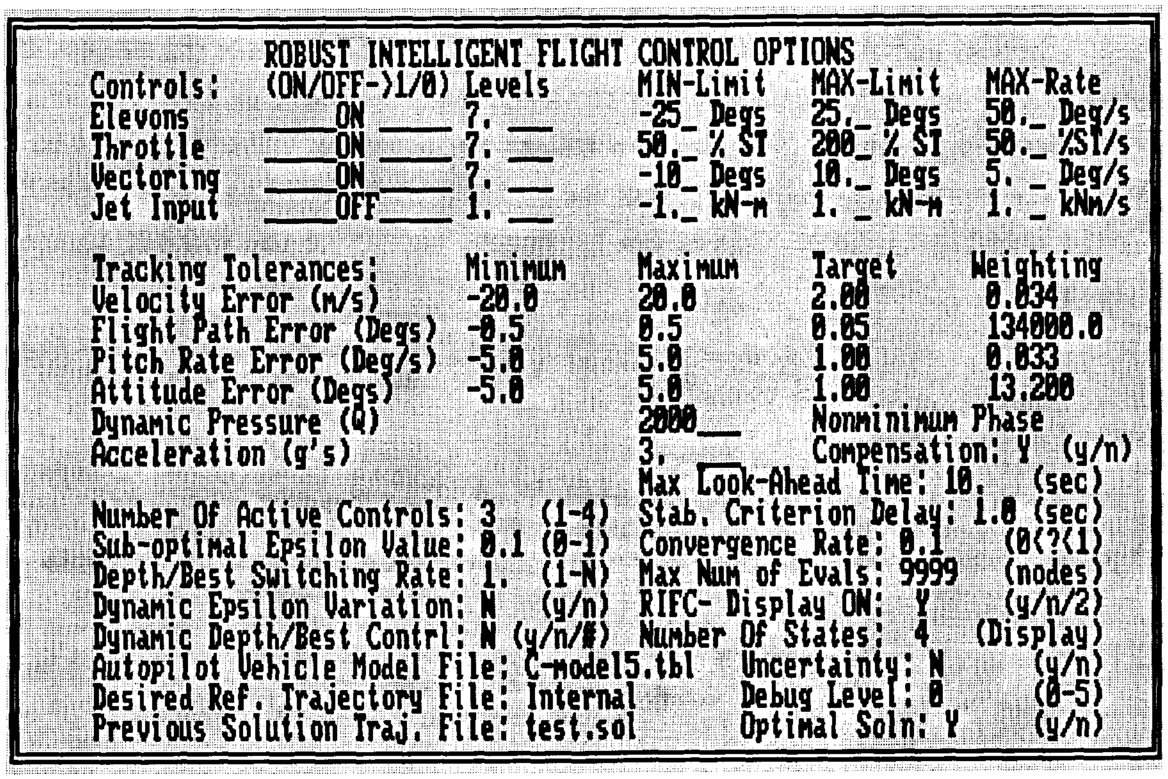
Given the available resources for this research, the state space was quantized into *160,000* nodes and the control cycle time was *0.5* seconds. The controls were quantized according to available input *rates*, and a minimum of *5* and up to *10* possible rates were used for each control actuator.

One of the objectives of running several series of tests was to measure the effect of using different enhancements to the A^* search. The benefits gained through use of the stability criterion, nonminimum phase change of variables, A^*_ϵ

[5] Some cases used only two controls, elevons and throttle. Other cases used a higher level of control quantization than required by the conditions in Chapter 8.

search, and best/depth interval parameter (*BD*) were explored. The effect of uncertainty on the search performance was also examined.

The results of a RIFC trajectory optimization are now presented as an example case. The initial condition is [V, γ, ω, θ] = [*3020* m/s,*–0.4°*,*0.0°*,*2.0°*], and the desired states change from [*3000,0°,0°,2°*] to [*3020,0.5°,0°,2.5°*] over the course of 10 seconds. The configuration of the controller is shown in Figure 9.5. Here, the control limits and rates, admissible states, targets, and weights are specified. For the example, nonminimum phase compensation (i.e. the change of variables) is used, the stability criterion is applied at intervals of *1.0* second with a required convergence of *10* percent, ϵ = *0.1*, and BD = *1*. No parametric uncertainty is included in this example, and the search is configured to continue until the optimal solution is found.



```
               ROBUST INTELLIGENT FLIGHT CONTROL OPTIONS
Controls:      (ON/OFF->1/0) Levels     MIN-Limit    MAX-Limit    MAX-Rate
Elevons        _____ON _____ 7. ___     -25_ Degs    25._ Degs    50._ Deg/s
Throttle       _____ON _____ 7. ___     50._ % ST    200_ % ST    50._ %ST/s
Vectoring      _____ON _____ 7. ___     -10_ Degs    10._ Degs    5. _ Deg/s
Jet Input      _____OFF_____ 1. ___     -1._ kN-m    1. _ kN-m    1. _ kNm/s

Tracking Tolerances:        Minimum     Maximum      Target       Weighting
Velocity Error (m/s)        -20.0       20.0         2.00         0.034
Flight Path Error (Degs)    -0.5        0.5          0.05         134000.0
Pitch Rate Error (Deg/s)    -5.0        5.0          1.00         0.033
Attitude Error (Degs)       -5.0        5.0          1.00         13.200
Dynamic Pressure (Q)                    2000___      Nonminimum Phase
Acceleration (g's)                      3.           Compensation: Y  (y/n)
                                        Max Look-Ahead Time: 10.   (sec)
Number Of Active Controls: 3   (1-4)    Stab. Criterion Delay: 1.0 (sec)
Sub-optimal Epsilon Value: 0.1 (0-1)    Convergence Rate: 0.1     (0<?<1)
Depth/Best Switching Rate: 1.  (1-N)    Max Num of Evals: 9999    (nodes)
Dynamic Epsilon Variation: N   (y/n)    RIFC- Display ON:  Y      (y/n/2)
Dynamic Depth/Best Contrl: N (y/n/#)    Number Of States:  4    (Display)
Autopilot Vehicle Model File: C-model5.tbl    Uncertainty: N        (y/n)
Desired Ref. Trajectory File: Internal        Debug Level: 0        (0-5)
Previous Solution Traj. File: test.sol        Optimal Soln: Y       (y/n)
```

Figure 9.5: RIFC Configuration Panel – Settings for Example Case

A view of the four dimensional search in the state space grid is shown in Figure 9.6. These displays identify nodes which have been expanded by the search. The first display shows the velocity and flight path errors, while the second shows attitude and pitch rate errors. There is not a one–to–one correspondence between points on each display, since, for example, several nodes could have the same velocity but different attitudes. Therefore the total number of active nodes at any time is some "multiplication" of the points on both displays. As the search progresses nodes are added or deleted from this grid depending on the results of exploring different paths. The view in Figure 9.6 represents the final state of the search graph when the optimal solution was found. The nodes displayed are either states along partial trajectories, or the terminal states of these trajectories. The optimal solution passes through a subset of the nodes shown.

Finally, Figures 9.7 and 9.8 illustrate the optimization results. The control input histories and resulting states are shown in Figure 9.7, and the constraints and costs are shown in Figure 9.8. The states are seen to converge to their desired values after 4.0 seconds, while all constraints are observed. The costs plotted in Figure 9.8 represent useful measures of the best trajectory found so far (at depth *k*), as the search progresses: first is the Lyapunov function of the nominal terminal state error,

$$J_1 = \left\| \underline{x}(k\Delta t_s) - \underline{x}_{\mathrm{d}}(k\Delta t_s) \right\|_Q \qquad (9.1a)$$

second is,

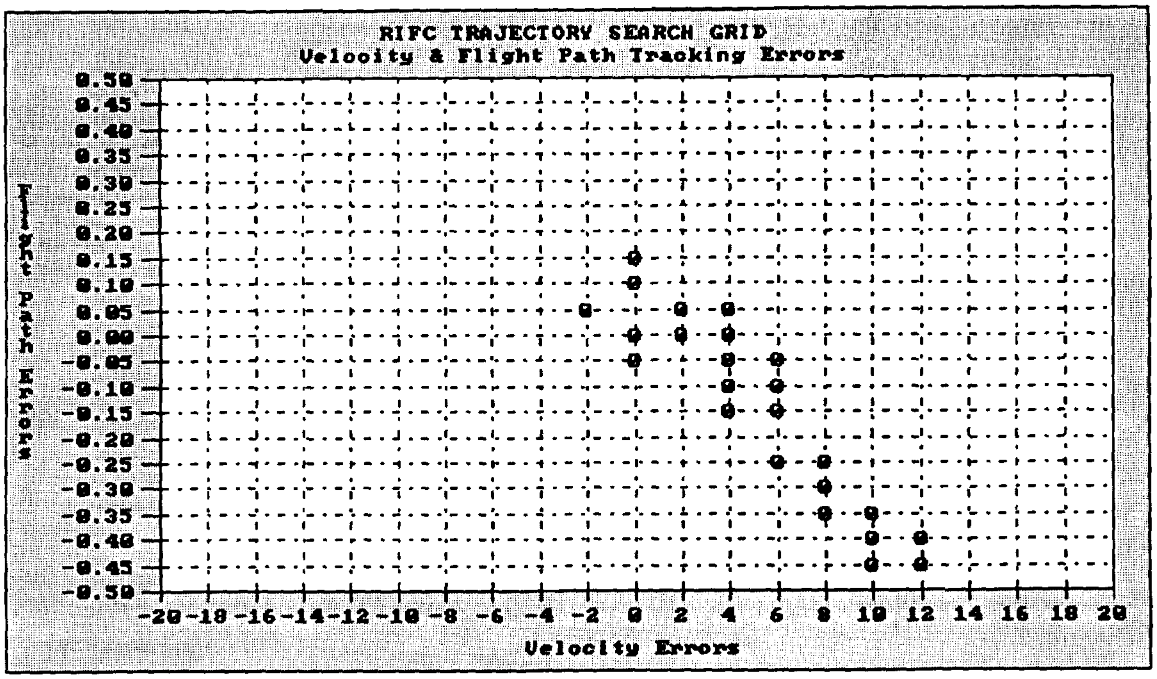
$$J_2 = \underset{\underline{\phi} \in \underline{\Omega}_\phi}{Max} \left\| \underline{x}(\underline{\phi}, k\Delta t_s) - \underline{x}_{\mathrm{d}}(k\Delta t_s) \right\|_Q \qquad (9.1b)$$

for the uncertain case[6], and third is the cumulative cost normalized by the depth,

$$J_3 = \frac{1}{k} \sum_{j=1}^{k} \underset{\underline{\phi} \in \underline{\Omega}_\phi}{Max} \left\| \underline{x}(\underline{\phi}, j\Delta t_s) - \underline{x}_{\mathrm{d}}(j\Delta t_s) \right\|_Q \qquad (9.1c)$$

Also note that only *177* node expansions were required to find the optimal solution. In fact, this was typical, with no case found which exceeded *2000* expansions.

[6] Note that $J_1 = J_2$ in this example, since no uncertainty is included.



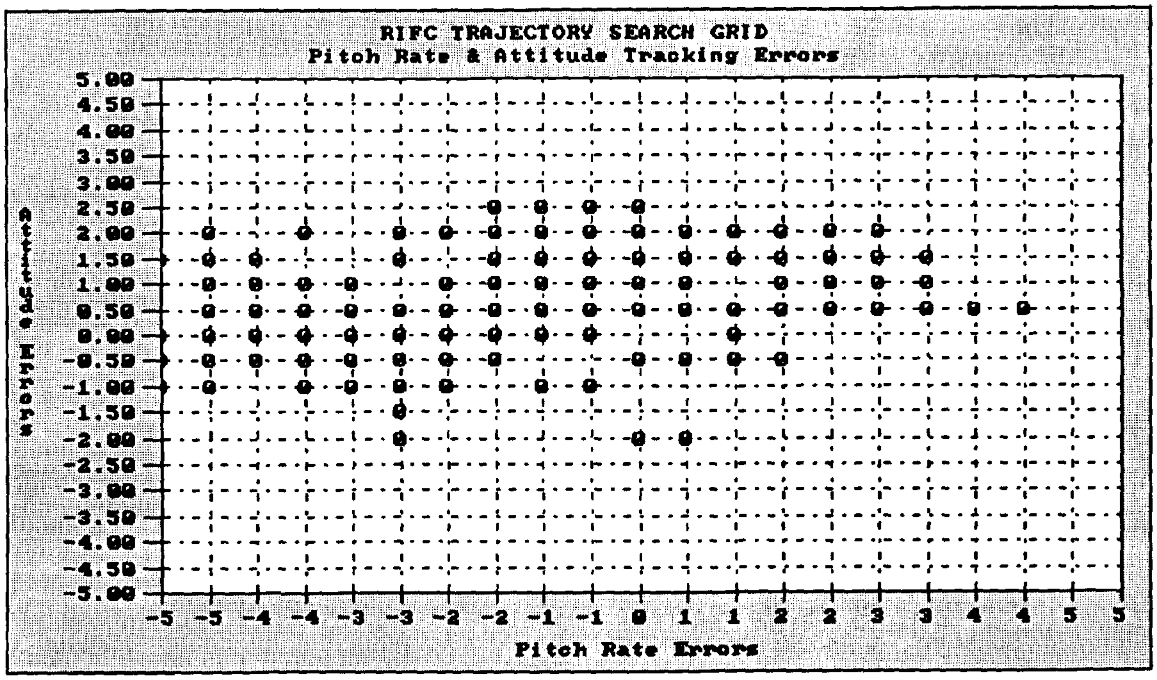


Figure 9.6: Expanded Nodes Viewed in the State Error Space

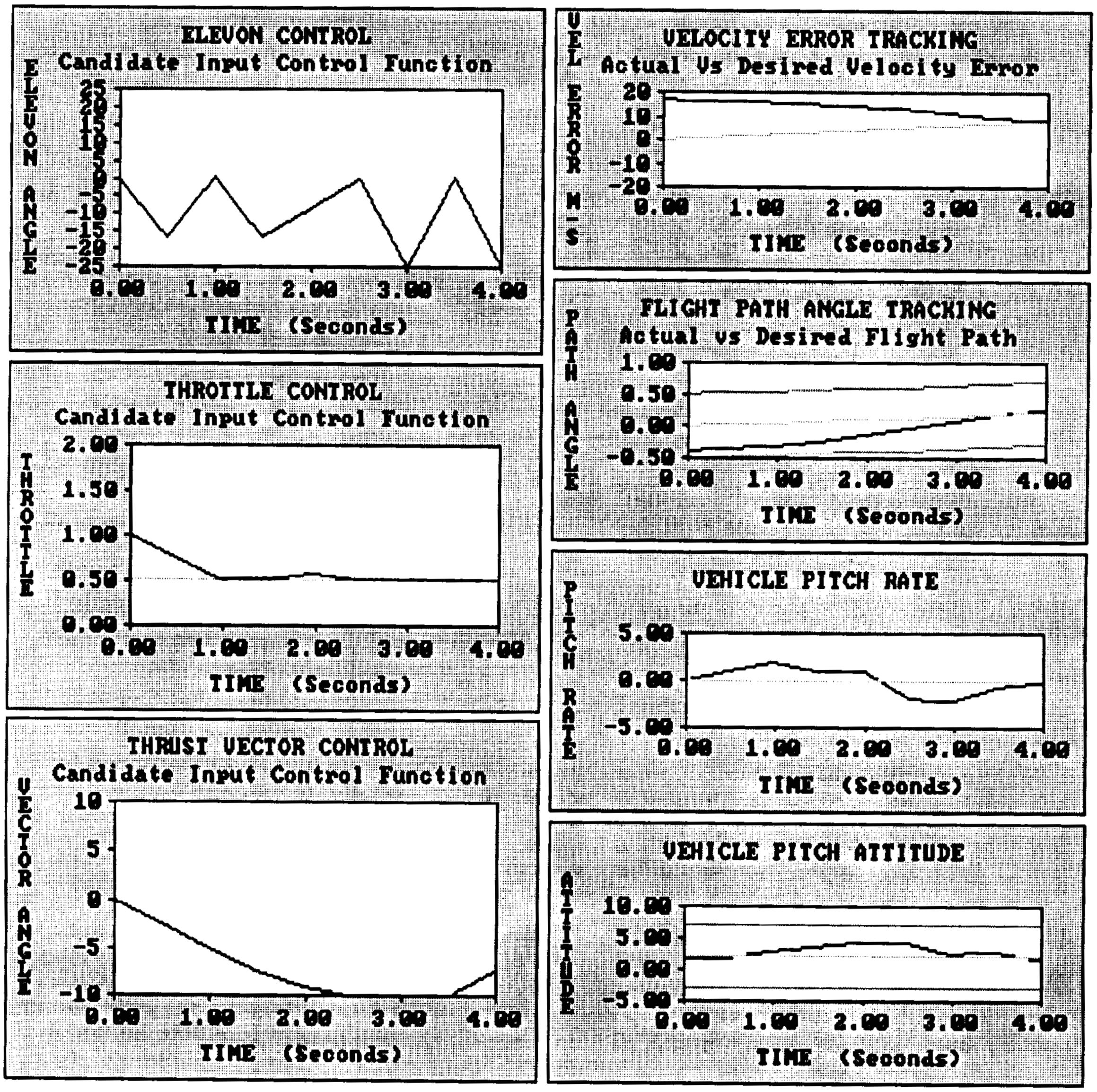


Figure 9.7: Control and State Histories for Example Solution Trajectory

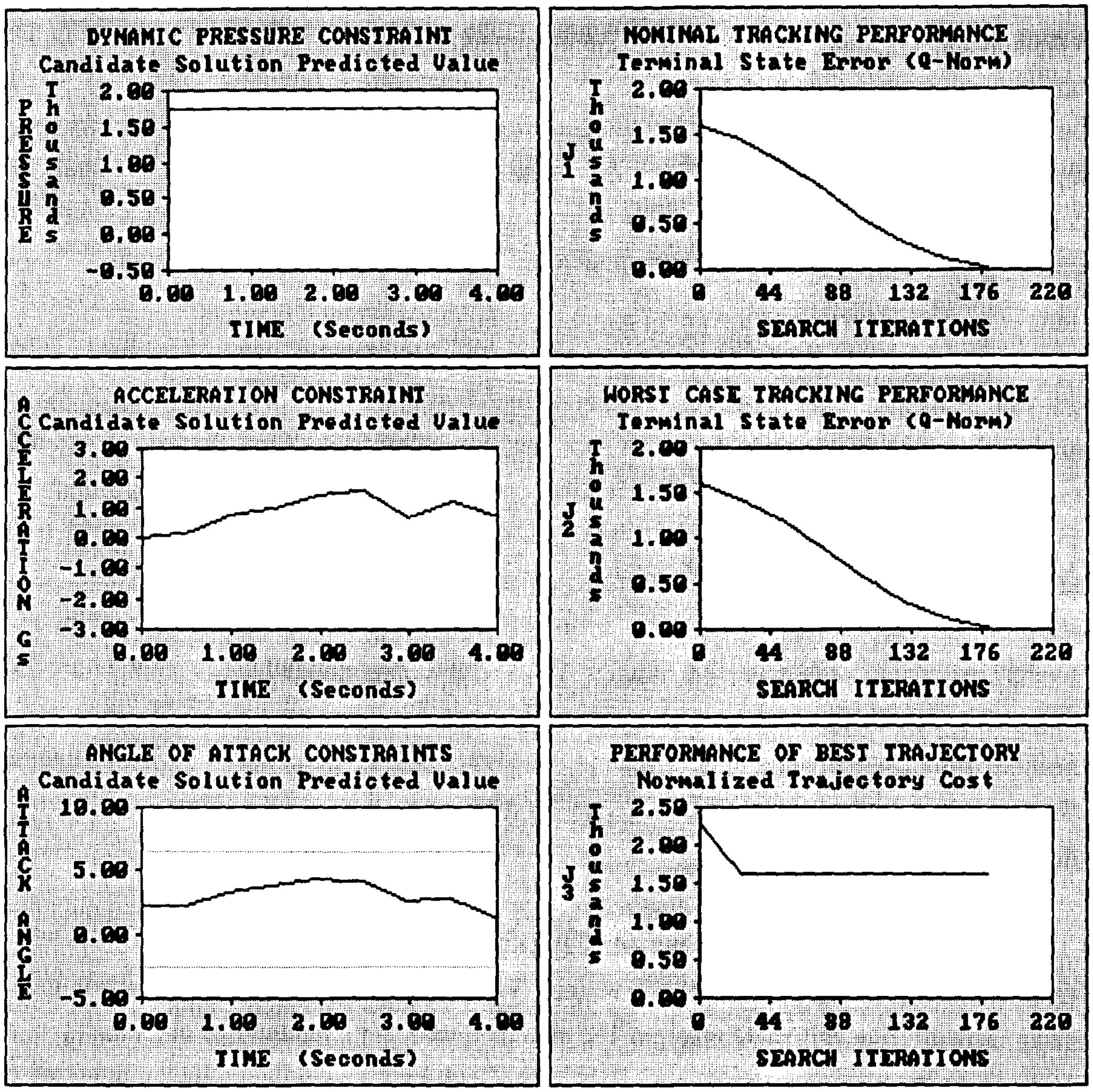


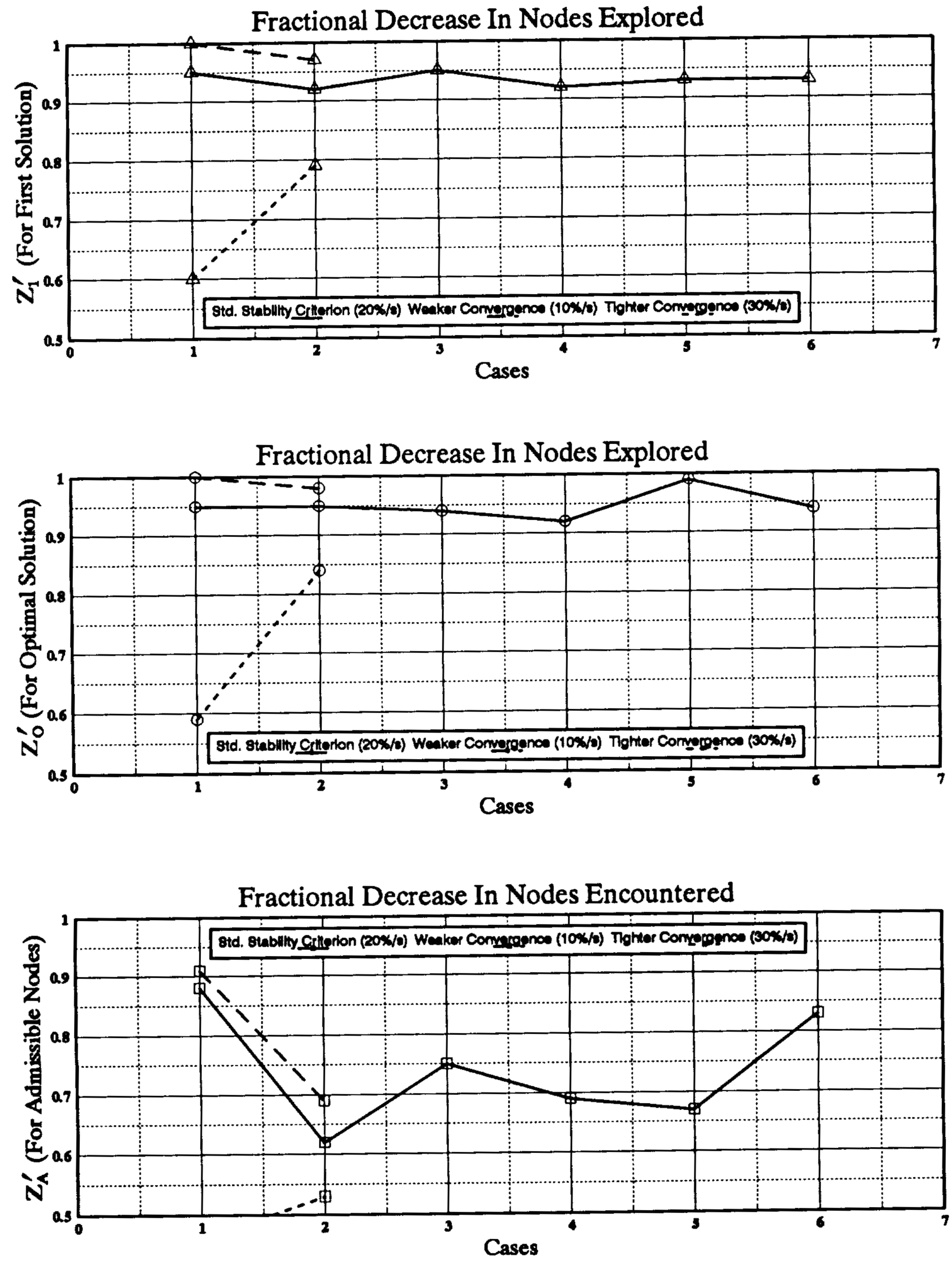
Figure 9.8: Constraints and Costs for Example Solution Trajectory

The remainder of this section is devoted to the examination of search performance of the RIFC algorithm. The first property to be examined is the effect of the Lyapunov stability criterion on the search. Figure 9.9 presents three graphs which show the number of nodes explored before finding the first solution (Z_1), the optimal solution (Z_O), and then the number of admissible nodes encountered before termination of the search (Z_A). The values have been normalized (Z'_1, Z'_O, Z'_A), however, so that the values plotted represent the fraction of nodes with the stability criterion to nodes without it. Therefore, any value less that *1.0* means that the stability criterion reduced the search effort.

The results indicate that the stability criterion always has a beneficial effect on the search effort required. However, the fraction by which the number of explored nodes is reduced is often not very substantial. This is not completely unexpected since the A* algorithm is already biased to search in the most promising directions first. Trajectories that satisfy the stability criterion will also tend to be the ones that have the lowest cumulative costs, and are therefore most likely to be selected for expansion by the A* search. Applying the stability criterion as a constraint only prevents the search from exploring candidate nodes that look promising due to the fact that they have not been explored as deeply as other nodes, and because the heuristic cost estimate is optimistic. In fact, the benefit of applying the stability criterion would increase as the accuracy of the heuristic estimate decreases.[7]

The more significant effect of the stability criterion (seen in Figure 9.9) is to decrease the number of admissible nodes. This property is important since it means that the stability criterion reduces the search space, independent of the ability of A* to explore in the right directions. It is this effect that guarantees convergence of the algorithm to a solution which is a stabilizing trajectory. The Lyapunov analysis of Chapter 7, and the choice of the weights Q_i and time step Δt_L (to assure that a certain decrease of the Lyapunov function can be achieved), also assure the existence of a solution that converges to the target within a known time t_{max}. Since

[7] This behavior was observed for a few test cases. However, since the A* algorithm is biased in the same manner even for $h = 0$, the difference is not significant enough to illustrate it here.



Fractional Decrease In Nodes Explored

Fractional Decrease In Nodes Explored

Fractional Decrease In Nodes Encountered

Figure 9.9: The Effect of the Stability Criterion on Search Effort

the convergence rate obtained from Chapter 7 was based on conservative control margins, additional tests were performed to assess the effect of using a tighter stability criterion (i.e. a faster decrease in $L(\underline{x}(t)-\underline{x}_{d}(t))$). These results are also shown in Figure 9.9. An improvement in search efficiency is achieved, at the risk, however, of not finding any solution if the demanded convergence is too strict. Through simulated experiments, such as these, an appropriate convergence rate could be determined in order to benefit the most from the stability criterion.

The next property examined is the effect of redefining the variables used in the cost function in order to eliminate the nonminimum phase behavior from the system outputs. Recall from Chapter 8 that by tracking the motions of some point forward of the center of rotation, the nonminimum phase character of the transfer function from elevon deflection to flight path angle no longer exists. It was conjectured that this change of variables would improve the search efficiency, since it would decrease the search effort wasted looking for minimum phase trajectories (with respect to the center of mass). The results are shown in Figure 9.10, which compares the search effort in terms of nodes explored and admissible nodes as a function of problem difficulty. Larger initial tracking errors typically correspond to more difficult problems, and the independent axis is simply the number of admissible nodes encountered divided by *1000.*[8] As in the previous figure Z_1', Z_0', Z_A' are the ratios of nodes explored for the first solution, optimal solution, and admissible nodes, to their corresponding values in the case where no change of variables was used.

Clearly, the results indicate a significant improvement in search efficiency using the change of variables described above. Gains of *10* to *80* percent were realized in most cases. Moreover, this improvement reduced the effort required to find first solutions as well as optimal solutions. Another interesting observation is that the benefits also seem to be greater for more difficult problems. With a few exceptions the data tend to exhibit a larger improvement in convergence time when the search space was larger. More experience with the algorithm would be needed to confirm this behavior.

[8] For the case where the search did not use the change of variables.

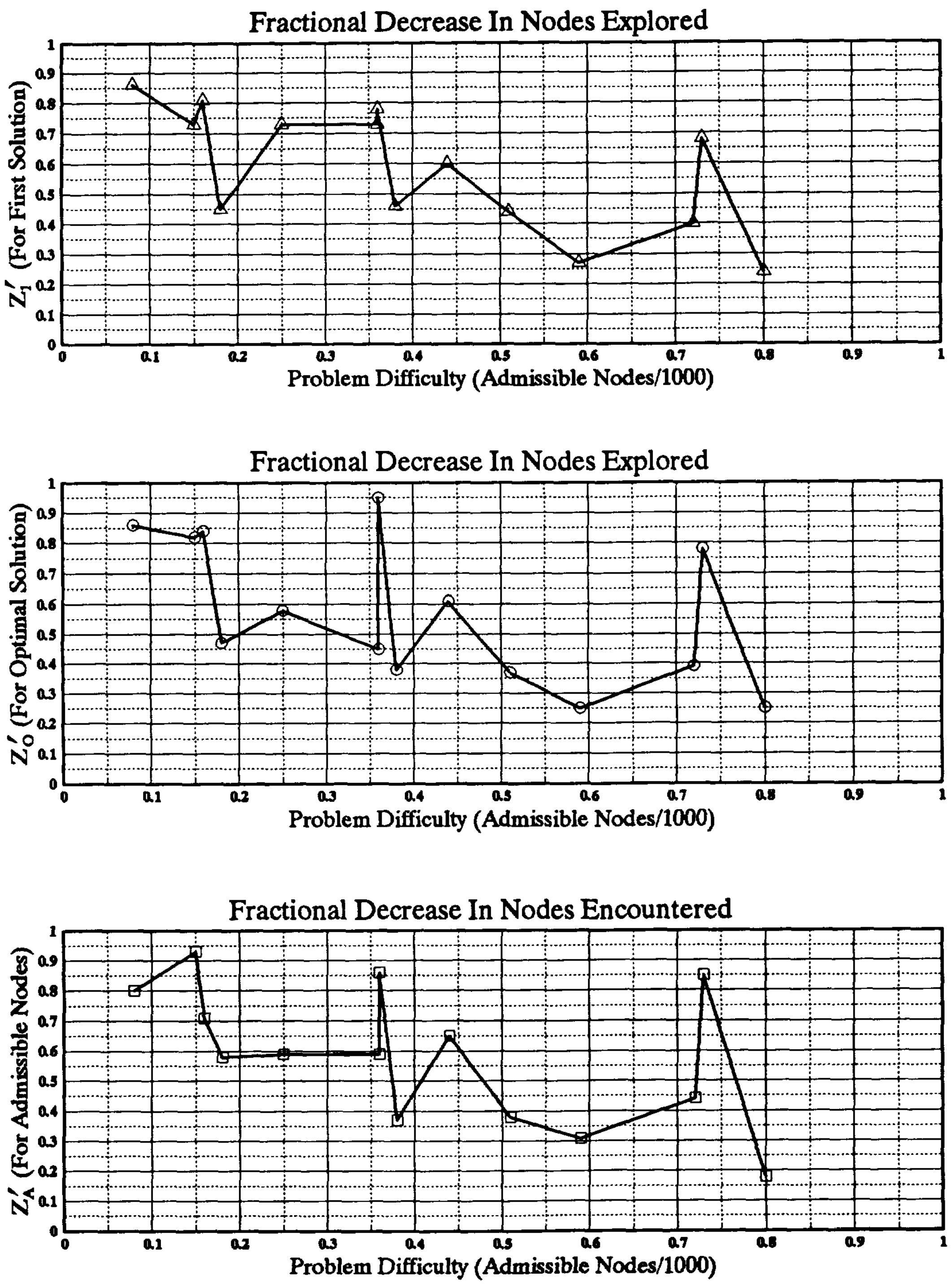


Figure 9.10: The Effect of Changing Variables in the Cost Function

In Figure 9.11 the performance of the A_ϵ^* search is compared to A^* for a range of values for ϵ. In this case several problems were solved using a series of ϵ values in order to determine the effect of accepting greater and greater degrees of suboptimality. It was suggested in Chapter 8 that larger values of ϵ would guide the search to explore deeper levels of the tree sooner, with the result that admissible solutions could be discovered more quickly.[9]

Although the results demonstrate a significant improvement in search performance for a range of ϵ values, note that increasing ϵ does not always translate into faster convergence to the first solution. The explanation for this behavior can be attributed, in part, to the node storage scheme used by the RIFC algorithm. Normally, the utility of a suboptimal search is to avoid wasted effort distinguishing between trajectories with similar costs. The quantization of the state space, however, has the positive effect that similar trajectories (with similar costs) are already grouped together. In this case, the additional benefits of using the suboptimal search are diminished, and increasing the value of ϵ is of limited advantage. Then, as ϵ continues to increase, it becomes more likely that the search will waste time exploring in misguided directions to larger depths. The advantages of using A_ϵ^* would be more apparent (for smaller ϵ) if the state space were not already quantized.

Another interesting result is that the effort required to reach optimal solutions did not appear to increase as ϵ became large.[10] This might simply have occurred because the particular cases chosen happened to find the optimal solution along depth excursions encouraged by the ϵ search. More difficult problems with less favorable initial conditions could very well contradict this behavior.

[9] Note: For convenience, the ϵ parameter used here is not exactly the same as described in Chapter 8. Instead of using the condition $C_L^* > (1+\epsilon)\ C^*$ to decide if the best local or global node should be expanded (see equation 8.17), the condition $C_L^*(1-\epsilon) > C^*$ is used instead. In this case ϵ is in the range $\epsilon \in [0,\ 1]$ rather than $\epsilon \in [0,\ \infty]$.

[10] One might expect Z_0 to increase with increasing ϵ, since more search effort would be wasted exploring deeper in wrong directions.

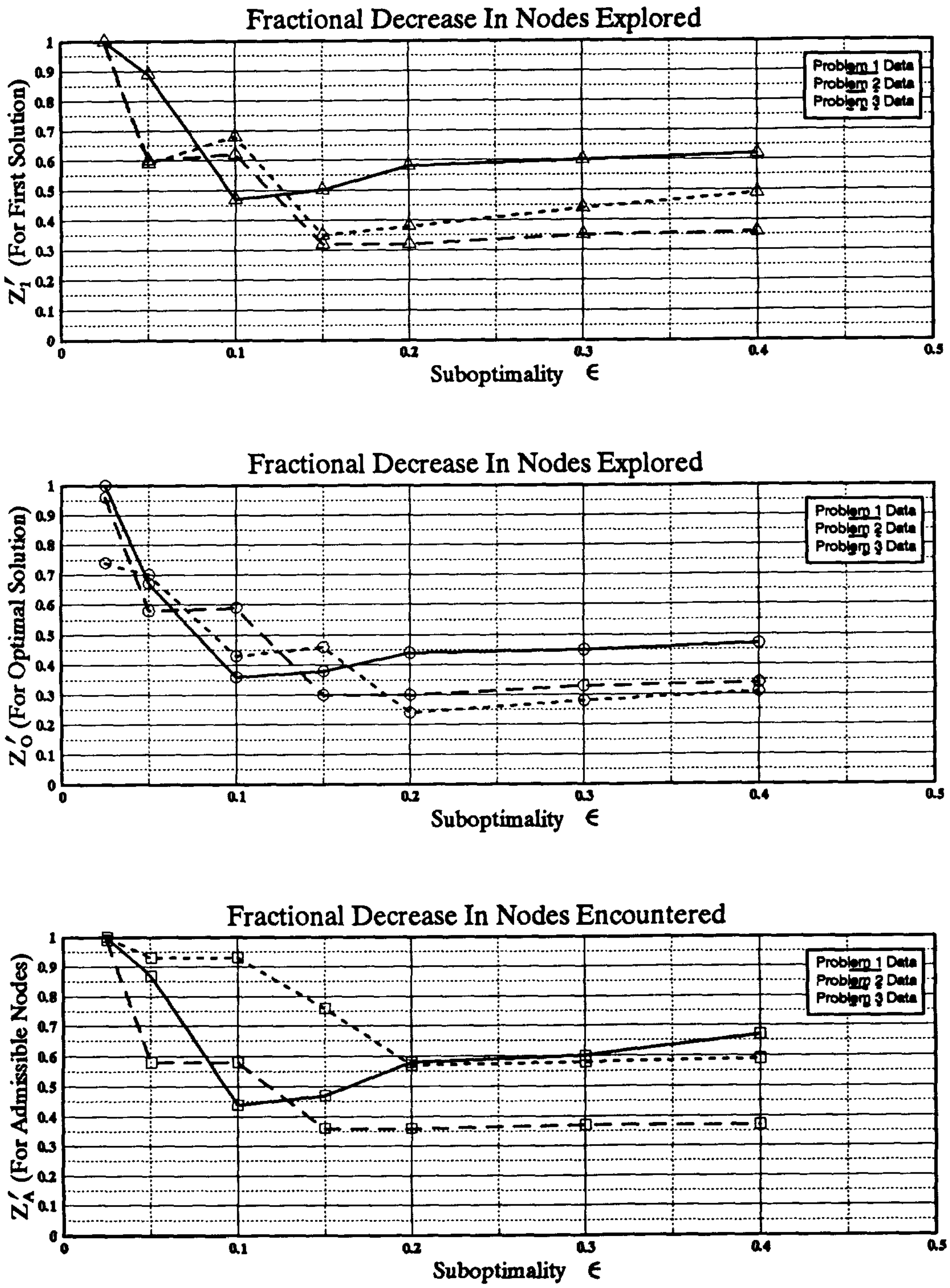


Figure 9.11: Gains in Search Efficiency Using A_ϵ^*

The final enhancement to the RIFC algorithm is the hybrid best/depth search which requires each node to be explored locally to a predetermined depth before performing the next best–first expansion. Several cases were tested with a series of values for the *BD* parameter (see section 8.1.3) ranging from *1* to *5* levels. The results are presented in Figure 9.12. As in the previous cases Z_1', Z_0', Z_A' represent ratios of the node counts with and without the *BD* search option.

Based on the data in these plots, it is evident that the *BD* parameter can either increase or decrease the search efficiency. In the examples tested it was more likely to help. It seems, however, that the benefit depends on whether or not the search happens to encounter solutions early. If local gradients are likely to lead to admissible solution trajectories then the search performs well. Obviously, cases were found which took longer because the local gradients did not lead to solutions. The properties of the *BD* search would probably also have been more favorable if the nodes were not stored in a quantized state space. In that case, the A^* search would have wasted more time deciding between similar paths, and the *BD* search would force it to look deeper.

The effect of model uncertainty on the performance of the RIFC algorithm is now examined. With uncertainty in the coefficients, the search attempts to minimize the worst–case integrated tracking error. Admissible trajectories are required to satisfy the constraints and the stability criterion both nominally and in the worst–case sense. For simplicity the uncertainty in each of *15* coefficients was taken to be the same fraction[11], and this fraction was increased until no trajectory could guarantee robust convergence in the tracking error. The results are shown in Figure 9.13 for the same parameters that have been used in the previous figures (Z_1', Z_0', Z_A'). The second graph in the figure depicts the same results with the robust stability criterion removed. That is, the worst–case trajectories are required to be admissible, but they are not constrained to satisfy the Lyapunov convergence rate.

[11] Actually, the drag and horizontal thrust coefficients (C_D and C_{Tv}) are treated differently because they appear in the dynamic equations as a difference between two large quantities. The fractional uncertainty is applied to the difference $C_D - C_{Tv}$, rather than to each separately, since otherwise the velocity error constraints are violated by all worst-case trajectories for relatively small uncertainties ($\sim 2\%$).

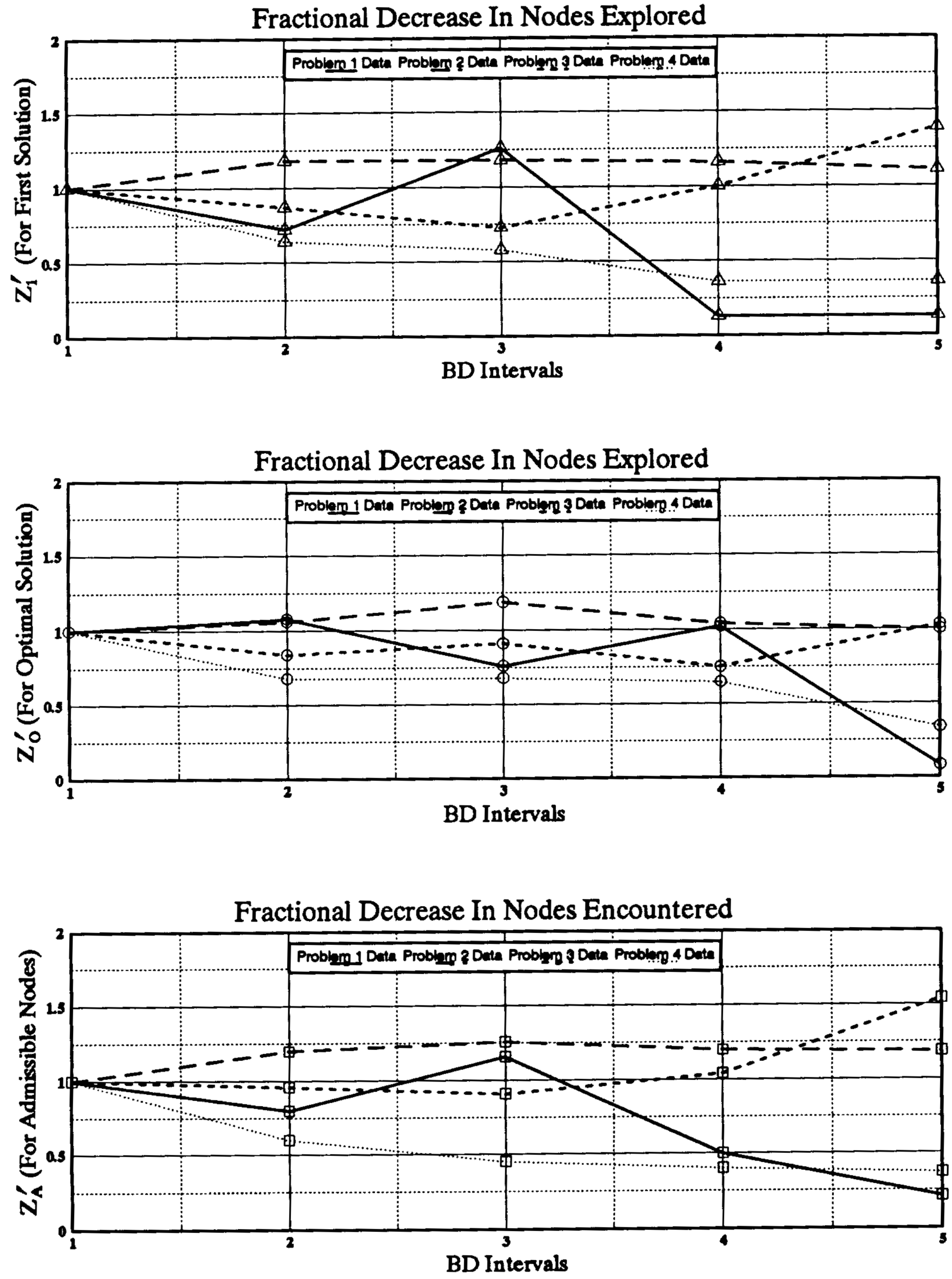


Figure 9.12: The Effect of the BD Search on Performance

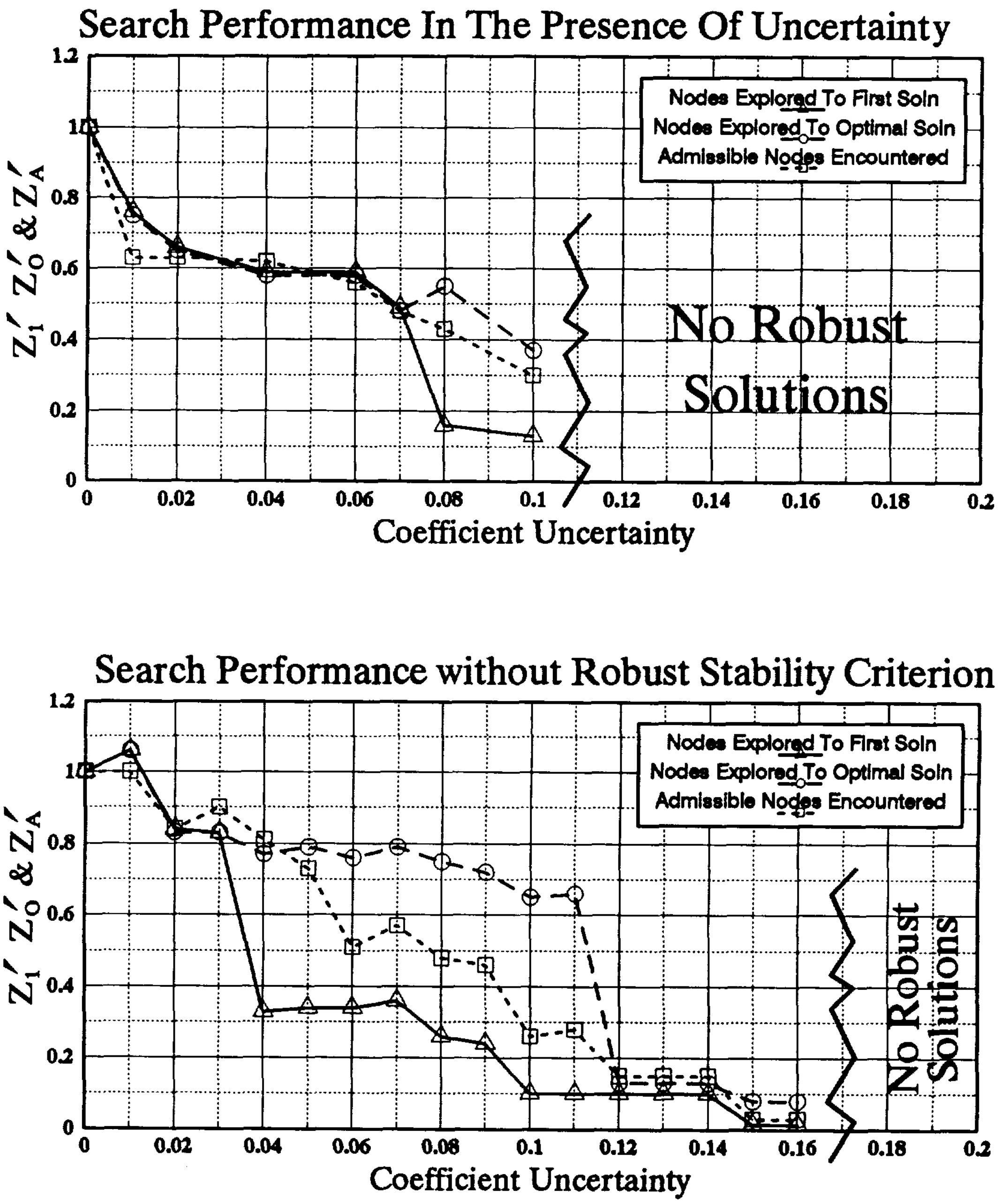


Figure 9.13: The Effect of Uncertainty on Search Performance

Clearly indicated by Figure 9.13 is that uncertainty decreases the number of admissible nodes in the search space. Greater degrees of uncertainty make it more difficult for candidate trajectories to satisfy all conditions for both nominal and worst–case histories. This, in fact, improves the search efficiency, since fewer nodes are available for exploration. Once the uncertainty is large enough, however, all trajectories lead to violations in the worst–case and the search fails. This occurred at an uncertainty of approximately *11 %* for all coefficients in the first graph.

Since it is desirable for the search to return its best solution, even if it is not robust to the current levels of uncertainty, the actual RIFC controller does not actually truncate nodes which fail the robust stability criterion. If a robust solution is found that satisfies this criterion, then it can only be superceded by another such solution with a lower worst–case cost. Otherwise, the best solution that does not meet this condition is returned.[12]

The second graph in Figure 9.13 illustrates the search performance under the relaxed stability criterion. Solutions were found for uncertainties as high as *17 %* on all coefficients. These solutions meet all constraints nominally and in the worst–case, but only satisfy the stability criterion for nominal trajectories.

One noteworthy characteristic, not shown in these figures, is that as the uncertainty increases, the depth to which the search pursues candidate trajectories tends to decrease. Whereas solutions found for *1 %* uncertainty may have been *5* second trajectories, those for *10 %* uncertainty were only *0.5* second solutions. This property was discussed in Chapter 7 and was the reason for introducing the second set of terminal conditions (see equations 7.137). As the uncertainty increases, the ability of the search to look further ahead in time decreases, and the RIFC algorithm approaches the single–step optimal controller in behavior.

As a final illustration of the effect of uncertainty on the algorithm, the ability of the controller to bias its solution towards less uncertain control actions is

[12] The same hierarchy is used for the constraints. A solution will be allowed to violate the constraints in the worst-case, if no other solution can satisfy these constraints.

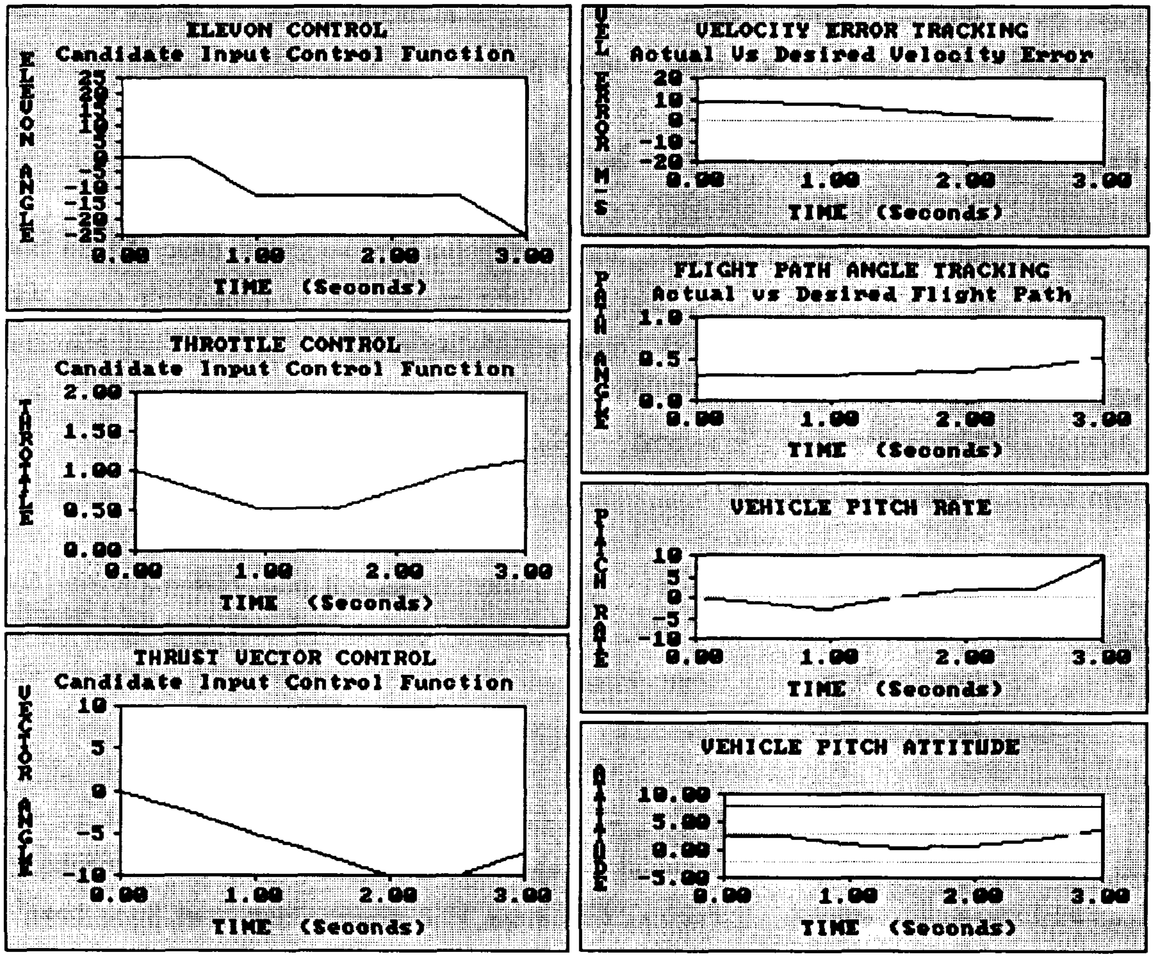


Figure 9.14: Search Bias Towards Less Uncertain Controls – Case 1

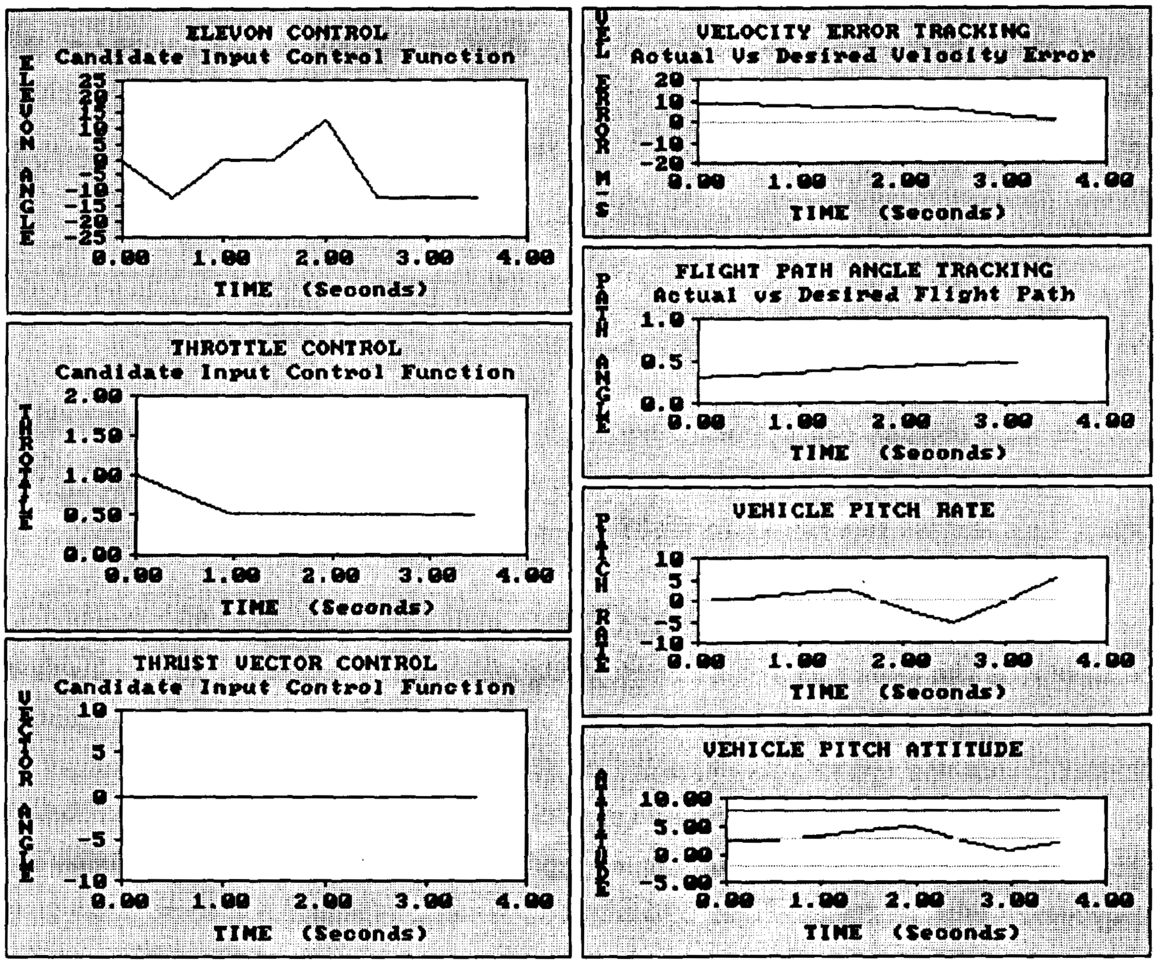


Figure 9.15: Search Bias Towards Less Uncertain Controls – Case 2

demonstrated. Figure 9.14 shows an example case with no uncertainty in the control coefficients. Figure 9.15 shows exactly the same case with an exaggerated degree of uncertainty in the thrust vector control (*100* %). Notice that in the second case the controller chooses a solution that completely avoids the use of the thrust vector input. In the same manner, the minimax nature of the RIFC optimization can account for uncertainty which changes with the state or control values.

9.3 Flight Test Results

In this section the performance of the RIFC controller is demonstrated in simulated flight. Four example cases are presented which illustrate the algorithm's capabilities and its superiority to the single–step optimal controller.

As explained in Chapter 3, the SSOC approach was used for comparison because it is the most compatible existing algorithm applicable to the same kinds of problems. The SSOC algorithm is based on the idea that, for optimal control problems that are too difficult to solve in real–time, the optimal solution might be well approximated by a step–wise optimal trajectory. In effect, the terminal time is assumed to be only one time–step ahead, and the inputs are chosen to minimize the single–step cost function. This optimization is then repeated for each time–step. Clearly, the RIFC controller can be thought of as an extension of the SSOC method with the additional ability to look further ahead in time to verify its single–step decision. Both algorithms can account for nonlinearities, constraints, and uncertainties in the same manner. For a detailed discussion of the SSOC algorithm see reference [Flo1].

In the first example, the SSOC and RIFC controllers are given the task of correcting an initial tracking error. In this case, each controller has a perfect model of the vehicle, and no external disturbances are included. Figures 9.16 and 9.17 show the results for the SSOC and RIFC controllers, respectively. Both controllers are seen to smoothly reduce the velocity and flight path angle errors. Notice,

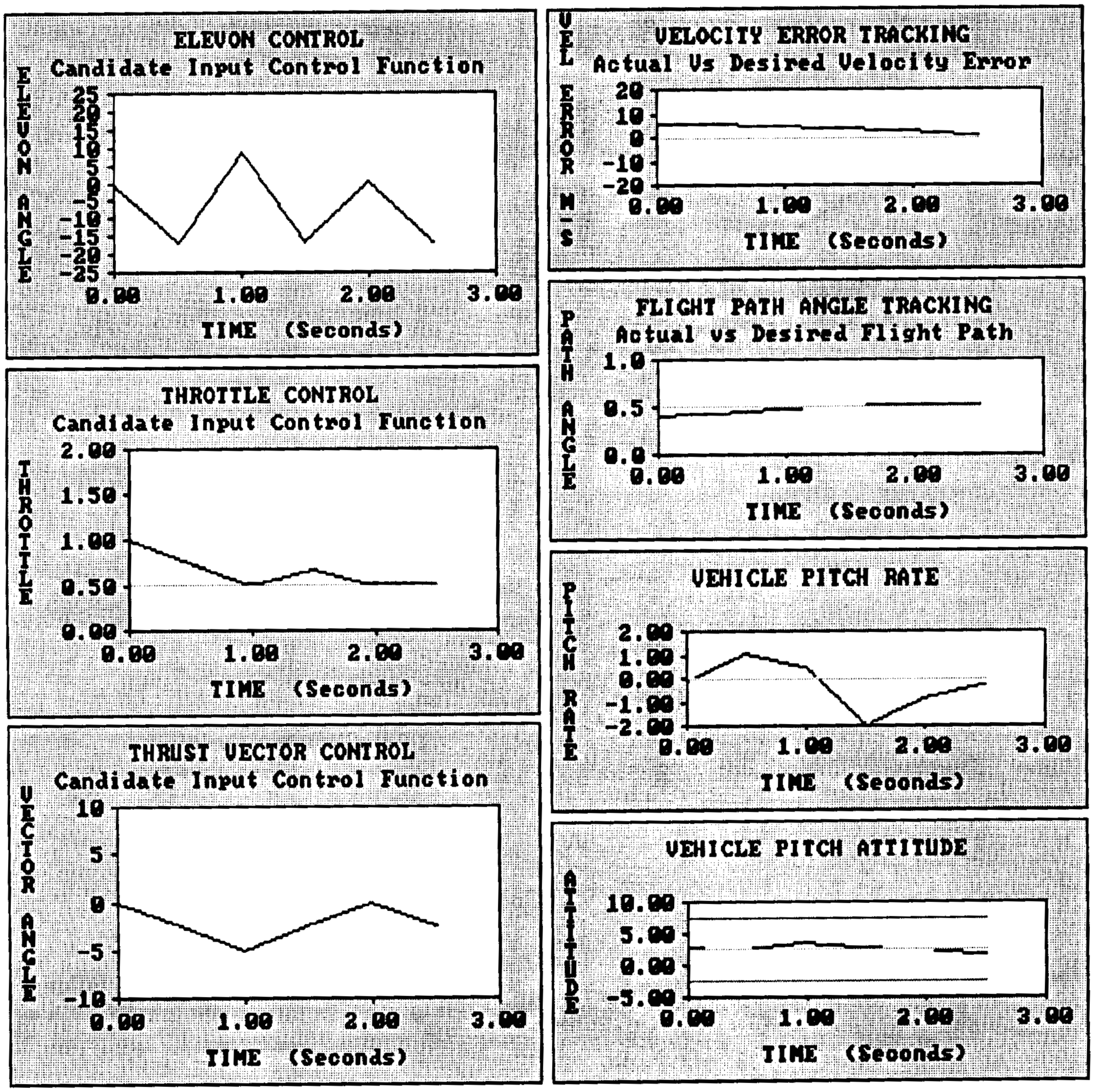


Figure 9.16: Successful Initial Condition Response of SSOC Controller – Case 1

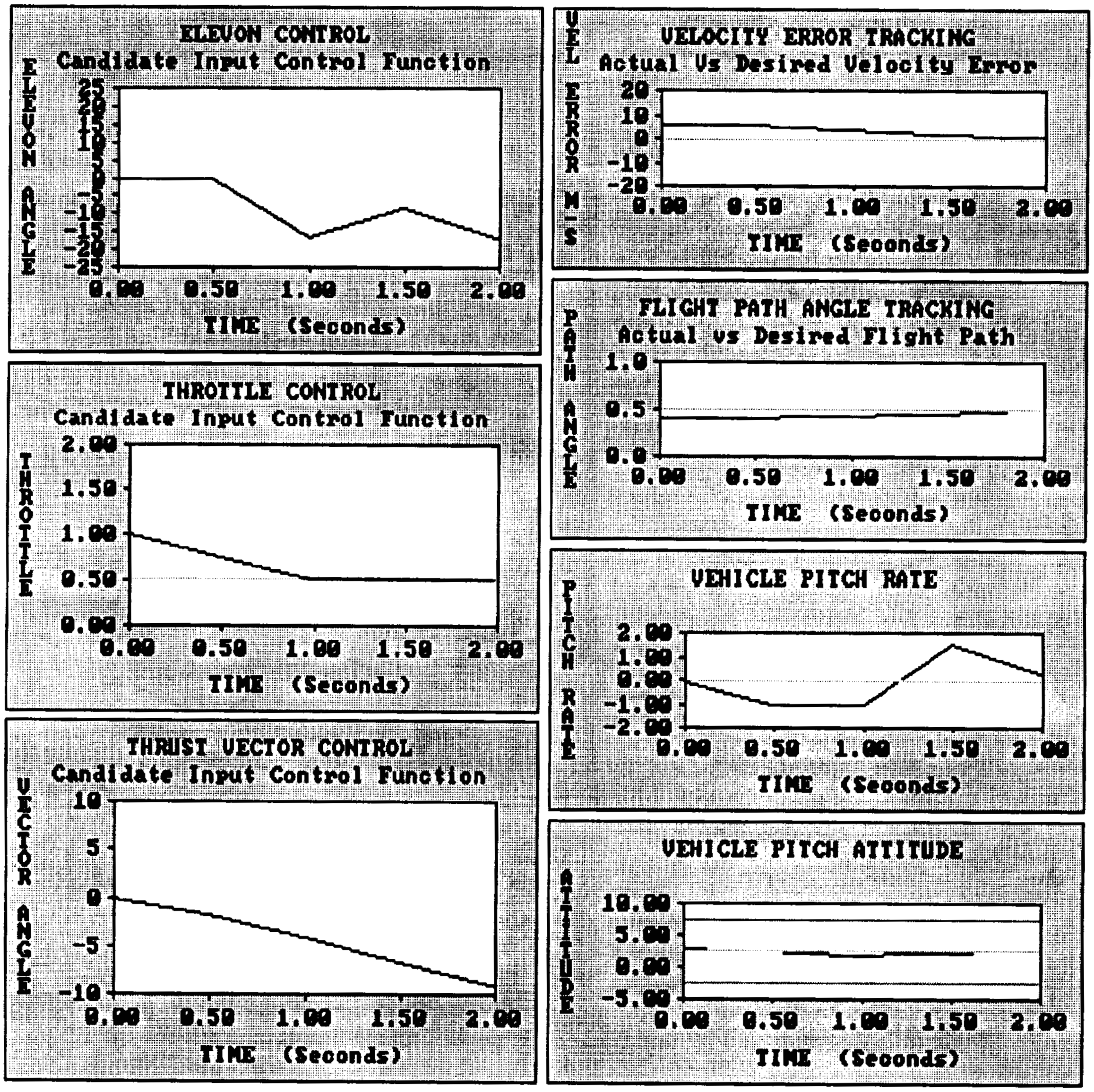


Figure 9.17: Successful Initial Condition Response of RIFC Controller – Case 1

however, that the RIFC controller found a solution which converged more quickly (*2* seconds compared to *2.5* seconds). It is also interesting to note that the overall optimal (RIFC) solution was quite different from the SSOC solution in character. The pitch rate (and attitude) histories for the two figures are opposite: the SSOC trajectory used attitude to obtain positive normal acceleration, while the RIFC controller found a solution using the resultant normal force from some combination of the controls.

As in the example above, it was generally found that the SSOC controller can perform well for small perturbations about the desired trajectory. It should be pointed out that, for the results shown in this section, the SSOC controller was given the benefit of the nonminimum phase compensation (change of variables) used by the RIFC algorithm. In addition, it was aided by an additional constraint intended to prevent attitude violations resulting from high pitch rates. The admissible pitch rate was varied from zero at the attitude error limits to its maximum value at the desired attitude. This constraint helps to avoid certain unrecoverable situations. For consistency, it was also given to the RIFC controller for the results in this section.

To obtain good performance with the SSOC algorithm it was also necessary to make some adjustments to the cost function weights. Although the weights chosen for the RIFC controller account for the relative importance of errors in each axis, the avoidance of constraints is left to the A^* search. Using the same weights for the SSOC controller was found to result in excessive pitch rates or attitude violations in many cases. In contrast, large weights on pitch rate or attitude errors were found to allow velocity or flight path errors to diverge. The weights used in the examples of this section have been tuned to give good overall tracking performance without attitude or rate constraint violations. However, it was only possible to achieve this locally. As seen in the next example, once errors in the velocity vector become large the SSOC controller runs into difficulties.

Figure 9.18 shows the SSOC control and state histories for the same configuration as in Case 1, but with a larger initial tracking error. With limited ability to look ahead (one step is *0.5* seconds), the controller tries too hard to reduce

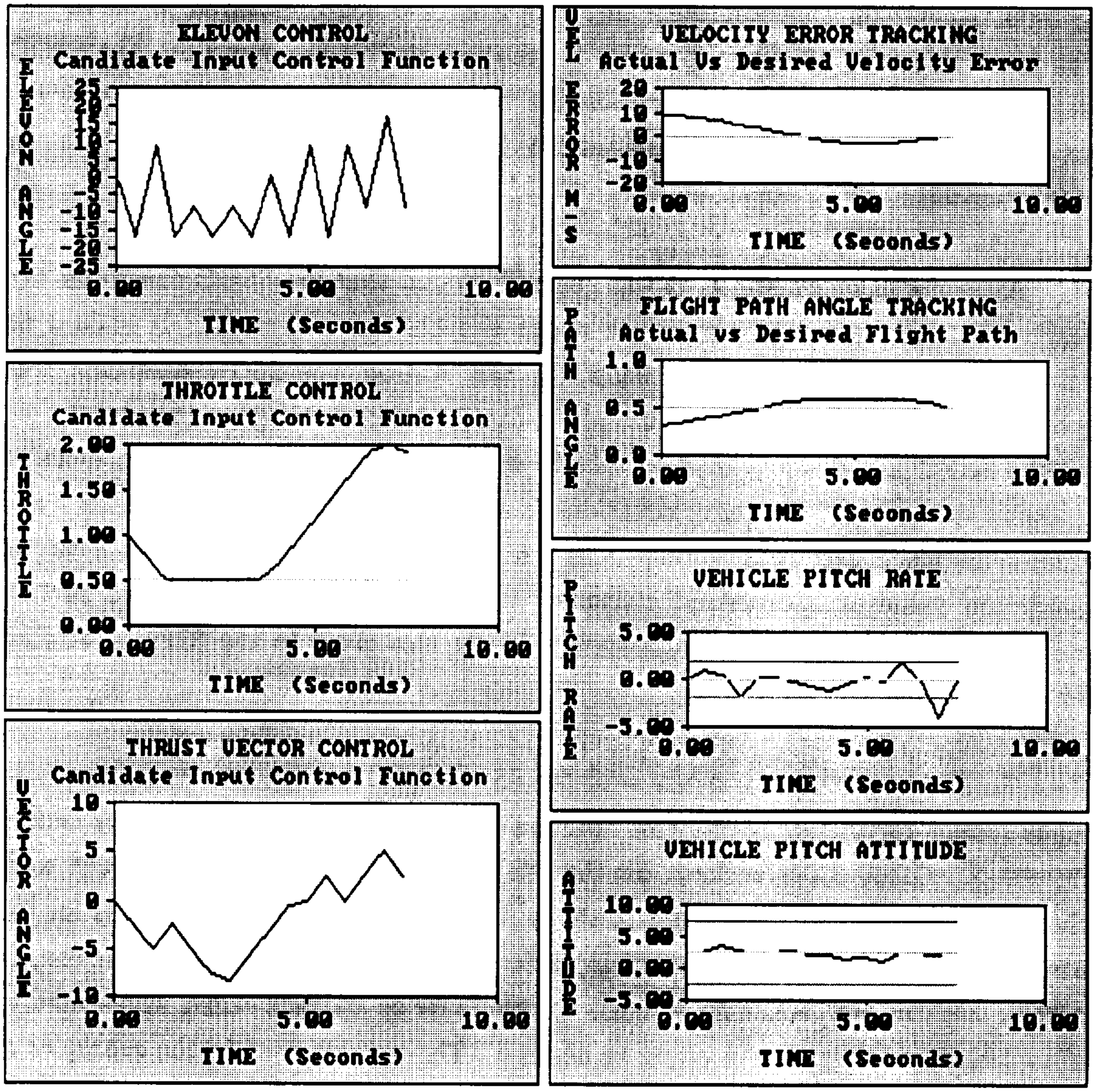


Figure 9.18: Unsuccessful Initial Condition Response of SSOC Controller — Case 2

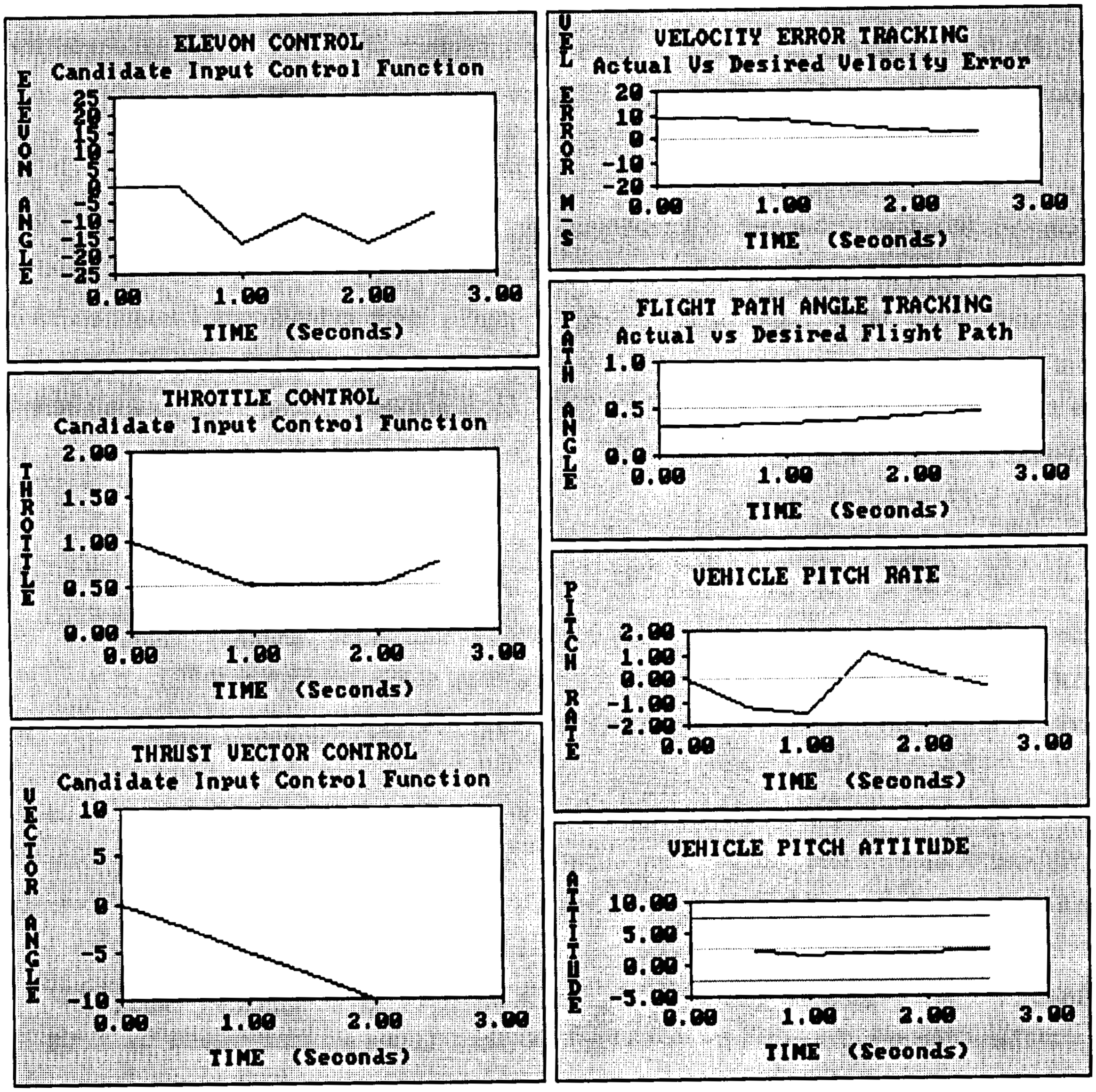


Figure 9.19: Successful Initial Condition Response of RIFC Controller – Case 2

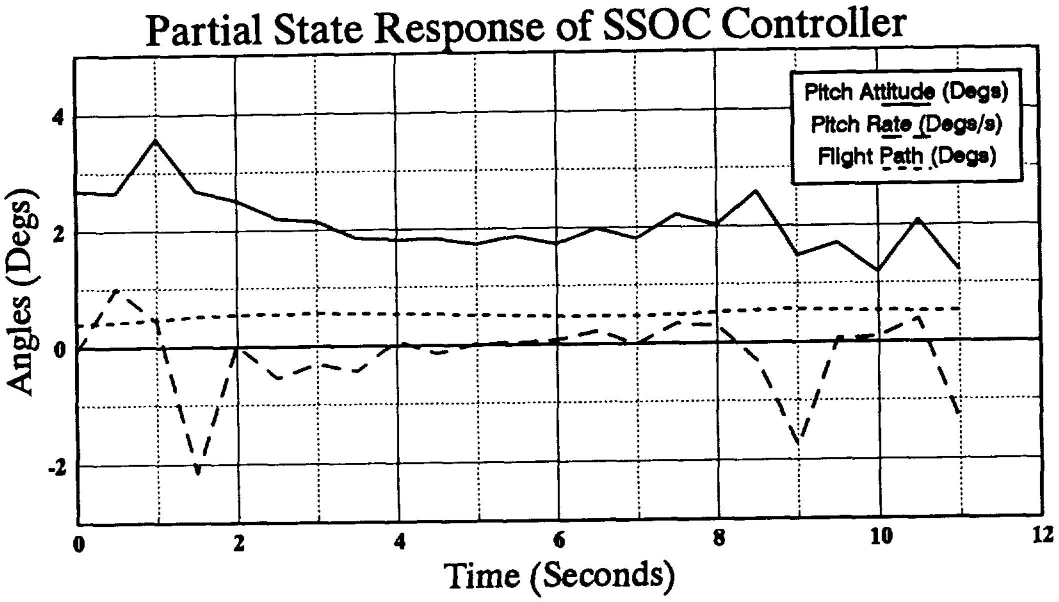
velocity and flight path errors resulting in an overshoot in these values. The chattering of the elevon input is the result of large pitch rates, which eventually cause the constraint violation in $\omega(t)$ at $t = 7.2$ seconds. Figure 9.19 shows the results for the RIFC controller in the same situation (Case 2). A smooth solution trajectory is found which differs very little from the Case 1 solution.

Case 3 examines the performance of the SSOC and RIFC algorithms in the presence of uncertainty and disturbances. An uncertain model for the controller was obtained by randomly modifying each of the coefficients by a maximum of *5* percent,[13] and then angle of attack disturbances were introduced (modelled as a first–order markov process, $\sigma = 0.25$ degrees). In Figure 9.20, the SSOC controller is seen to roughly maintain a desired flight path angle of *0.5* degrees, and a desired attitude of *2.0* degrees. The same is true of the RIFC controller shown in Figure 9.21.

In Case 4, the uncertainty is increased to *7.5* % and the angle of attack disturbances are increased to $\sigma = 0.5$ degrees. Under these circumstances the SSOC controller eventually loses control and diverges in attitude (Figure 9.22). The RIFC controller is still able to maintain stability (Figure 9.23).

Although many possible example simulations could be shown, the main purpose of these cases was to demonstrate that the RIFC algorithm has a significant advantage over the SSOC approach by virtue of its ability to look further ahead. Of course, it is more computationally expensive than the SSOC approach. However, based on the results in this chapter, the RIFC algorithm is extremely efficient at searching the solution space. With optimal trajectories typically found in a matter of a few hundred node expansions, and admissible solutions often much faster, this method could be used successfully on not–too–distant–future parallel flight computers.

[13] Except for the thrust and drag coefficients, in which case the difference $C_{Tv}-C_D$ was given a maximum error of 5 percent.



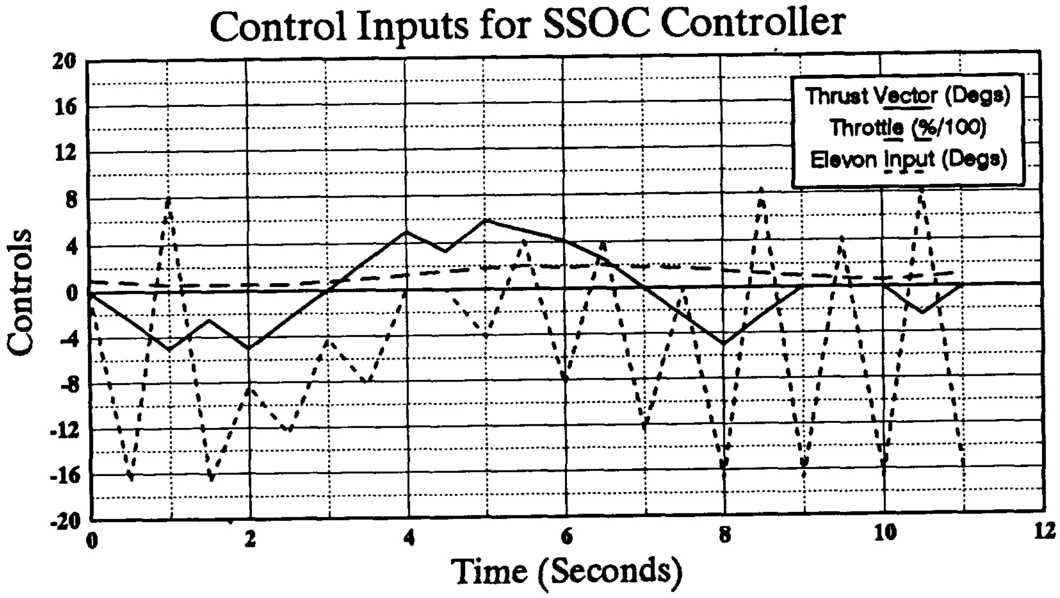
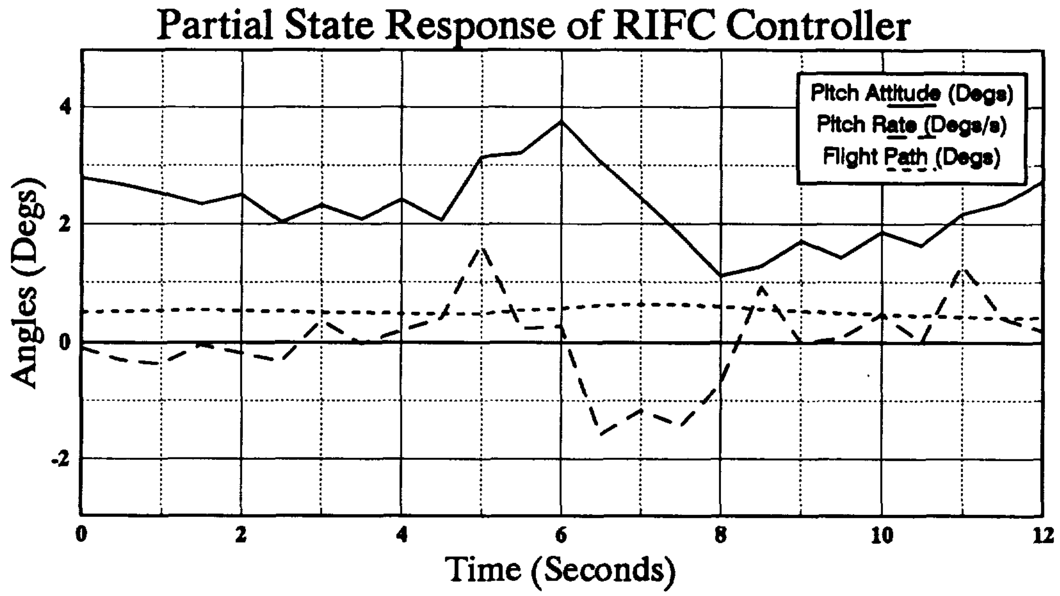


Figure 9.20: Successful SSOC Flight with Uncertainty and Disturbances — Case 3



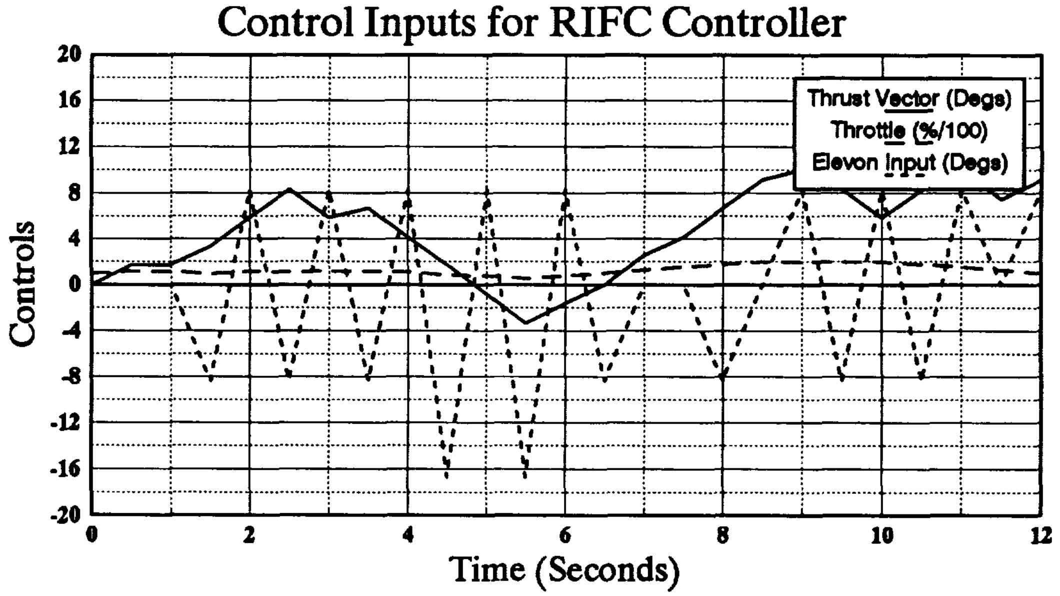


Figure 9.21: Successful RIFC Flight with Uncertainty and Disturbances – Case 3

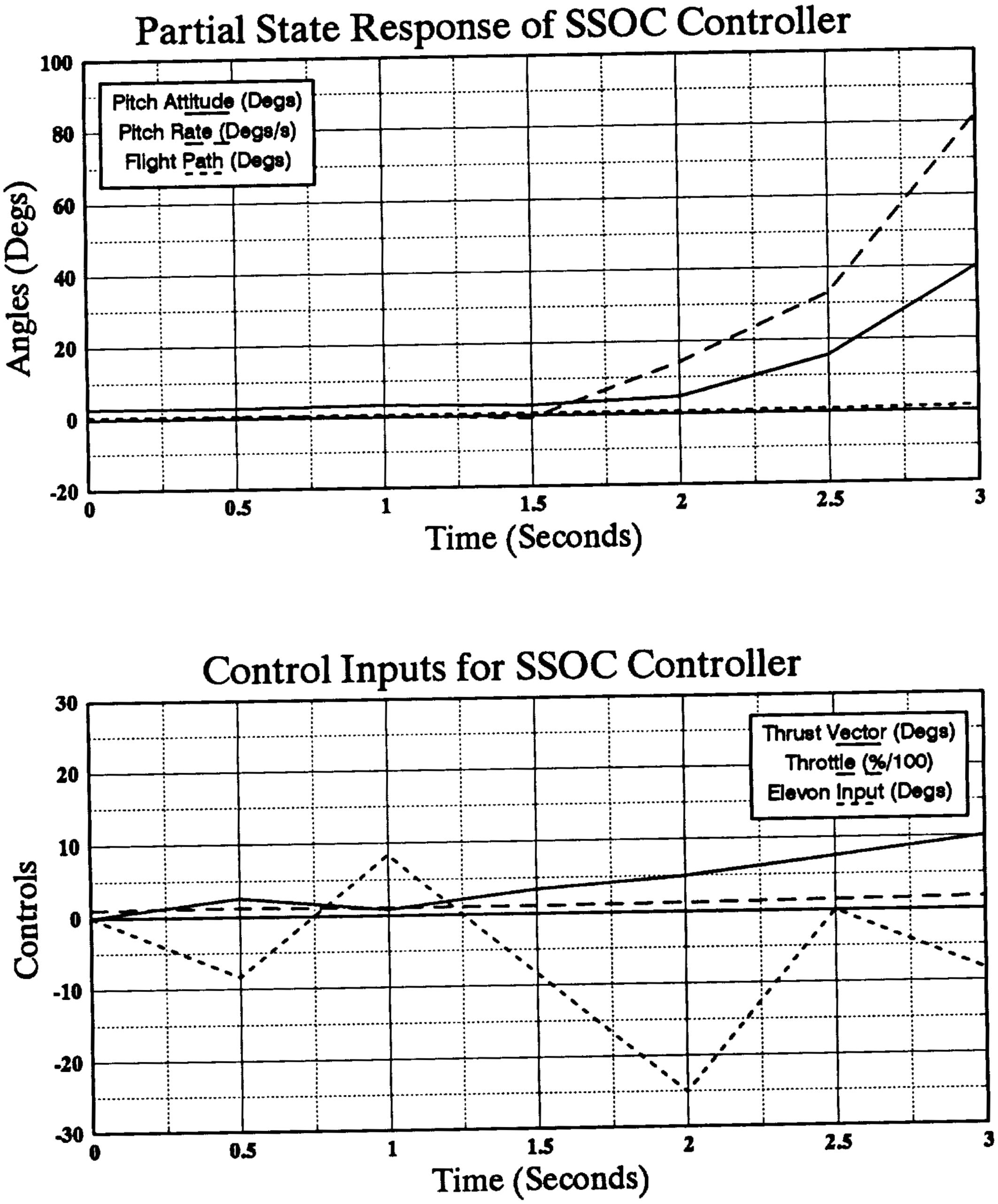
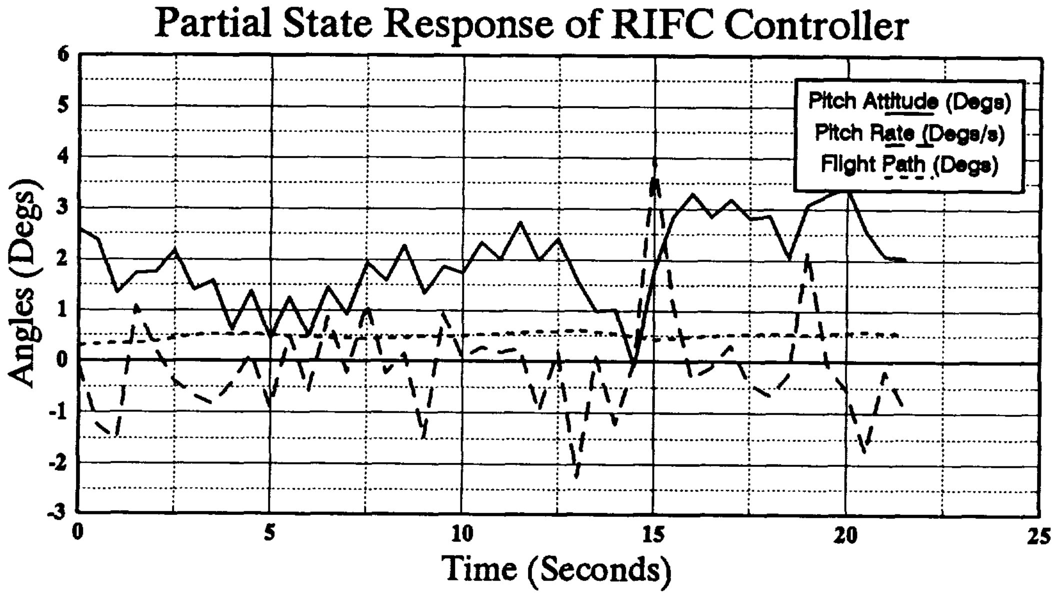


Figure 9.22: Unsuccessful SSOC Flight with Uncertainty and Disturbances — Case 4



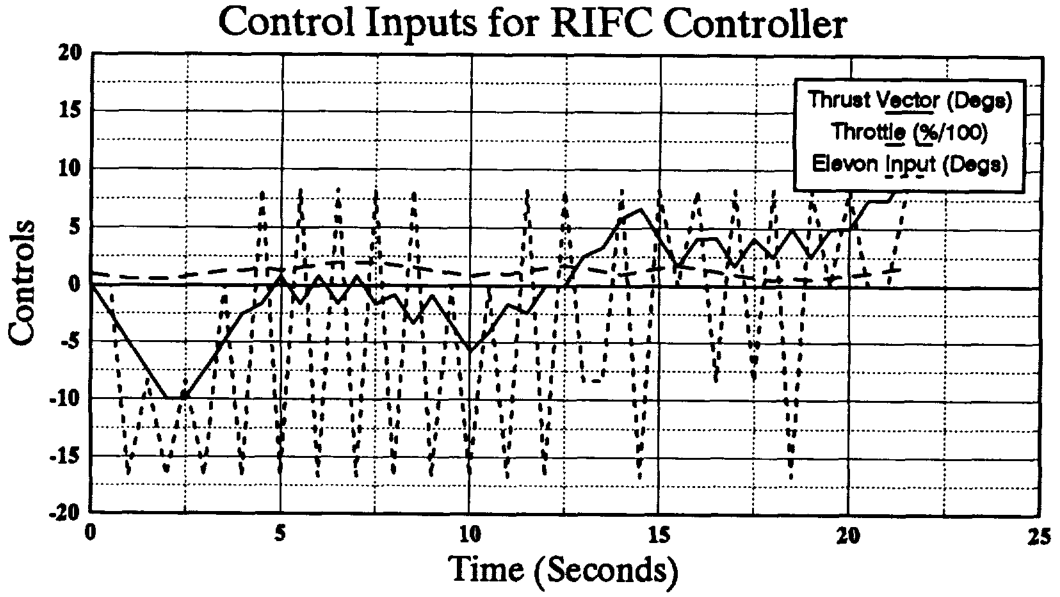


Figure 9.23: Successful RIFC Flight with Uncertainty and Disturbances — Case 4

Chapter 10

Conclusion

10.1 Summary

The main objective of this thesis has been to address some of the flight control problems associated with air–breathing hypersonic vehicles (AHSV). The nonlinear nature of the vehicle dynamics, complex constraints, and coefficient uncertainties have motivated the development of a new control approach capable of robust real–time short–term trajectory planning. This new method, called Robust Intelligent Flight Control (RIFC), achieves stable tracking of a desired trajectory through the repetitive solution of a receding–horizon nonlinear optimal control problem which includes all constraints and uncertainties. A viable correction trajectory is generated, followed for a short interval of time, and then recomputed. The flight control approach consists of an enhanced A^* optimizatïon technique that incorporates a Lyapunov stability criterion in a highly parallelizable algorithm. The efficiency of the A^* search, and the theoretical guarantees of a Lyapunov approach, are both achieved. The analysis and development of the RIFC controller, the construction of a realistic hypersonic vehicle simulation, and the evaluation of the controller's performance using this simulation test bed, were the primary efforts in this research.

Many of the challenging characteristics of AHSV's were discussed in Chapter 2. Extreme surface temperatures, a chemically reacting ionized flow, intense structural loading, and aerodynamic/propulsion interactions are some of the dominant obstacles associated with hypersonic flight. The single–stage–to–orbit objective creates additional difficulties, such as conflicting design objectives for different flight regimes, stringent tracking tolerances, complex constraints, and the requirement for a multi–mode propulsion system (or multiple engines). In addition, since the extreme flight conditions of hypersonic flight cannot be reproduced in currently available wind tunnels, much of the vehicle design and evaluation will have to be done using numerical flow analysis codes. Although much progress has been made, many hypersonic flow phenomena are still not fully understood, such as hypersonic boundary–layer transition, turbulence, and combustion flow chemistry. To some extent, therefore, it is likely that the validation of these codes will occur in actual flight; and the models available to the control system will include some degree of uncertainty.

In Chapter 3, the characteristics and structure of the RIFC controller were discussed. One advantage of this approach is that it can directly include all nonlinearities, constraints, and uncertainties, in its determination of the best control solution. No approximations are required, and it is applicable to the situation where a tabulated set of coefficients represent the vehicle model.[1] It is also a highly parallelizable algorithm, suitable for combining a variety of actuator types, and easily reconfigurable (for example, in the presence of actuator failures). Finally, the RIFC controller has the additional advantage of being able to choose its control solution based on the viability (and optimality) of a full multi–step correcting trajectory, thus avoiding future as well as present constraint violations.

An overview of A^* optimization techniques, as well as the fundamentals of Lyapunov stability theory, were presented in Chapter 6. The specific theorems and properties that are required for the development and analysis of the RIFC controller were also presented. The flight control problem was then formally stated in

[1] Since this model is not easily inverted, it presents a problem for most conventional control methodologies.

Chapter 7 as a receding horizon optimal control problem. The specific form of the dynamics, constraints, and uncertainties, as well as the desired trajectory and performance objective function were given. This chapter then proceeds to establish conditions for which controllability of the vehicle, convergence of the optimization algorithm, and stability of the correction trajectory are assured. Robustness of the solution to interval bounded parametric uncertainty was achieved through a minimax optimization in which the worst–case cumulative tracking error is minimized by the solution trajectory. The analysis of Chapter 7 served to demonstrate that the RIFC algorithm can be guaranteed to provide robust tracking performance in the presence of uncertainty and constraints, when such a solution exists.

Implementation issues and enhancements to the A^* algorithm were addressed in Chapter 8. An alternative node storage scheme was developed for which convergence of the A^* search could be guaranteed within a prescribed memory limit. In this structure, nodes are stored in a grid of quantized state tracking errors. Since all admissible trajectories are representable on the finite grid, there is never any need to unduly truncate viable candidate solutions due to memory limitations. Other advantages of this structure include its ability to store partial trajectories in a very compact format, and its natural tendency to group together similar trajectories (by quantizing the states), thus reducing the required search effort. Other enhancements to the A^* search include a change of variables to compensate for the nonminimum phase system behavior, a suboptimal A^*_ϵ search, and a hybrid best/depth search procedure. Several initial guess trajectories are also tested by the RIFC algorithm before a full search is begun. These trajectories may provide admissible solutions immediately, depending upon the level of disturbances between control cycles. Finally, this chapter assessed the computational requirements of the RIFC algorithm, and the feasibility of a parallel implementation was discussed. It was shown that a reasonably fast parallel flight computer would be adequate to obtain optimal solutions on–line.

The atmospheric, mass properties, airframe, aerodynamic, and propulsion models for simulating an AHSV in hypersonic flight were developed in Chapter 4.

These models were designed to work with a parameterized vehicle geometry and engine (*41* variables and *50* polygonal surfaces), thus enabling the representation of a wide range of possible vehicle configurations. The hypersonic aerodynamics model is based on a Newtonian flow approximation to obtain pressure coefficients for each panel. Oblique shock and expansion fan calculations are used to determine local flow properties, which are then used to estimate skin friction coefficients using a flat plate analysis. The scramjet propulsion model, which was adapted from reference [Ren1], performs a complete inlet, combustor, and nozzle analysis to determine engine performance for any flight condition.

In Chapter 9, the performance of the RIFC controller was evaluated using a typical AHSV configuration in simulated hypersonic flight. First, the sensitivity of a linearization of the system to small attitude variations served to demonstrate the severity of nonlinearities in the dynamics; and the limitations of a linear optimal control scheme were illustrated. An examination of the optimization characteristics of the RIFC algorithm then found that the Lyapunov stability criterion and nonminimum phase compensation (change of variables) had a positive effect on search performance. The A_ϵ^* procedure was also shown to reduce the number of node expansions required to find admissible solutions, although its benefits were limited due to the natural grouping of similar trajectories by the quantization of the grid storage structure. The advantages of using a hybrid best/depth search procedure were found to depend on the likelihood that local gradients encountered admissible solutions, and any improvements in search performance were not found to be consistent. Parametric uncertainty was seen to improve search efficiency due to a reduction in the admissible search space. The requirement that solution trajectories satisfy the constraints and stability criterion, both nominally and in the worst–case sense, was shown to eventually disqualify all possible solutions once the uncertainty reached a high enough level (*11%* in the example). A relaxation of the robust stability criterion made it possible to find more solutions, with even larger uncertainties (17% in the example), but without the guarantee of (worst–case) convergence. Finally, the RIFC controller was shown to successfully track a desired trajectory in the presence of uncertainty and disturbances. A comparison to the performance of a single–step–optimal–controller (SSOC) demonstrated that situations could arise in which the SSOC algorithm fails where the RIFC algorithm succeeds.

10.2 Suggestions for Further Research

The application of the RIFC control approach has been limited in this research to the problem of longitudinal control for hypersonic flight. Obvious extensions that would be useful include a six degree–of–freedom controller and the possibility of using many more actuators. Conceptually, there is no reason that the current control scheme cannot be applied directly to the more general problem. Practically, however, the required computational effort is exponentially related to the dimensions of the input space, and the memory required is exponentially related to the dimensions of the state space. Therefore, some means of simplifying the problem would be necessary in order to apply this approach more generally. This may not be as formidable a problem as it seems, however, since motions in the yaw axis direction can be assumed to be negligible (and intolerable) for air–breathing hypersonic vehicles.[2] Therefore, the only required additional states are roll and roll–rate.

A suggestion for handling many actuators might be to devise a method for constructing a map between the true actuators and some pseudo–controls (similar to those used in the analysis of Chapter 7), such that the available forces and torques (and their rate limits) are computable for any given flight condition (including the actuator states). This would limit the number of controls to two pairs of force and torque inputs (4 controls), which can then be mapped back into actual control values. This mapping could also be used to make the controllability conditions derived in Chapter 7 less conservative. This would make it possible to guarantee the existence of solutions for a larger tracking error envelope through increased acceleration and error–rate margins.

[2] Assuming a separate control loop to maintain a zero sideslip angle.

An even more efficient node storage scheme would greatly benefit the algorithm, since this would reduce the difficulty of dealing with the higher dimensionality of more general problems. Although the structure developed in this research is very compact, experience has shown that quite often most of the grid is empty. Since the grid coordinates were used as a code to determine the node number for indirect addressing, it seems difficult to condense this structure any further. However, there may be yet another computer programming "trick" that can be used.

In Chapter 8, it was shown that the RIFC algorithm can act as both a feed–forward and feedback controller if it is executed at a high enough rate to compensate for external disturbances. Otherwise an inner–loop controller to track the RIFC trajectory is required, and this loop is not completely addressed by this research. The best design for this controller would depend on the nature of the expected disturbances, and since the modelling of disturbances in the hypersonic environment is beyond the scope of this research, the existence of an inner–loop controller was assumed. It was suggested, however, that the SSOC algorithm was a sensible choice for accomplishing this task, since it could account for the nonlinearities and constraints in the same manner as in the RIFC controller.

References

[Ada1] Adams, N.J. and Hattis, P.D., ***An Integrated Configuration and Control Analysis Technique for Hypersonic Vehicles,*** Technical Report CSDL–C–6099, C.S. Draper Labs, Cambridge, M.A., Aug. 1989.

[And2] Anderson, J.D. Jr., "***A Survey of Modern Research in Hypersonic Aerodynamics,***" *AIAA 17th Fluid Dynamics, Plasma Dynamics, and Lasers Conference*, Paper AIAA–84–1578, Snowmass, C.O., June 25–27, 1984.

[Asa1] Asada, H., "***Intelligent Control and Sensing of Physical Systems, Course Lecture Notes,***" Massachusetts Institute of Technology, 1989.

[Ast3] Astrom, K.J., Anton, J.J., and Arzen, K.E., "***Expert Control,***" *Automatica*, Vol. 22, No. 3, pp. 277–286, 1986.

[Ast4] Astrom, K.J., "***Adaptive Feedback Control,***" *IEEE Proceedings*, Vol. 75, pp.185–217, Feb. 1987.

[Bak1] Baker, W., America In Space, pp. 152–153, Crescent Books, New York, N.Y., 1986.

[Bar2] Barnes, Daniel S., ***Design Study of a Self–Optimizing Autopilot,*** Master's Thesis, Massachusetts Institute of Technology, 1957.

[Bar3] Barto, Andrew J., ***Connectionist Learning for Control: An Overview,*** COINS Technical Report, Department of Computer and Information Science, Univ. of Massachusetts, Amherst, M.A., Sept. 1989.

[Bek1] Bekey, Ivan, "***Potential Directions for a Second Generation Space Shuttle,***" *37th International Astronautical Congress*, Innsbruck, Austria, IAF–86–106, Oct 4–11, 1986.

[Bit1] Bittanti, S. and Scattolini, R., "***A Self–Tuning Algorithm for Multi–Variable Control,***" *Proceedings of the 7th Symposium On Identification and System Parameter Estimation*, Vol.1, pp. 323–328, York, England, July 3–7, 1985.

[Bow2] Bowcutt, K.G., Anderson, J.D., and Capriotti, D., "***Viscous Optimized Hypersonic Waveriders,***" *AIAA 25th Aerospace Sciences Meeting*, Reno, N.V., Jan 12–15, 1987.

[Bry1] Bryson, A.E., Ho, Y.C., Applied Optimal Control, Hemisphere Publishing Co., Washington, D.C., 1975.

[Bus1] Busemann, A. and Culp, R., ***Theory of Hypersonic Flight***, Final Technical Report, NASA Grant NSG 1056, Colorado University, Boulder, C.O., Sept. 1976.

[Cha1] Chambers, E.A. et al., "***Solid Rocket Technologies for the 21st Century,***" *Aerospace America*, pp. 44–47, July 1990.

[Cla2] Clary, J.P. and Franklin, G.F., "***Self–Tuning Control With A Priori Plant Knowledge,***" *Proceedings of 23rd Conference on Decision and Control*, pp. 369–374, Las Vegas, N.V., 1984.

[Col1] Cole, J.D., "***Newtonian Flow Theory for Slender Bodies,***" California Institute of Technology, Sept 3, 1956.

[Cor1] Corda, S., and Anderson, J.D. Jr., "***Viscous Optimized Hypersonic Waveriders Designed from Axisymmetric Flow Fields,***" *AIAA 26th Aerospace Sciences Meeting*, Reno, N.V., Jan. 11–14, 1988.

[Cra1] Crabb, T.M. et al., "***Designing for Operations and Support,***" *Aerospace America*, pp. 18–30, Nov. 1988.

[Dav1] Davis, J.G. Jr., Dixon, S.C., "***Beyond Simulation,***" *Aerospace America*, pp. 38–42, July 1988.

[Dec1] Dechter, R. and Pearl, J., ***Generalized Best–First Strategies and the Optimality of A$*$***, Technical Report UCLA–ENG–83–19, Cognitive Systems Laboratory, UCLA, Los Angeles, C.A., 1983.

[Dem1] DeMeis, R., "***Sweetening the Orbital Bottom Line,***" *Aerospace America*, pp. 26–30, Aug. 1988.

[Dem2] DeMeis, R., "***An Orient Express to Capture the Market,***" *Aerospace America*, pp. 44–47, Sept. 1987.

[Dri1] Driver, Cornelius, "***Impact of Emerging Technology on Supersonic Transport,***" *Proceedings of ICAS 15th Congress*, Vol. 1, pp. 213–220, London, Sept 7–12, 1986.

[Dwo1] Dwoyer, D., Kutler, P., and Povinelli, L., "***Retooling CFD for Hypersonic Aircraft,***" *Aerospace America*, pp. 32–41, Oct. 1987.

[Edw1] Edwards, C.L.W. et al., "***Studies of Scramjet/Airframe Integration Techniques for Hypersonic Aircraft,***" *NASA Langley, 13th AIAA Sciences Meeting*, Pasadena, Jan. 20–22, 1975.

[Etk1] Etkin, B., Dynamics of Atmospheric Flight, John Wiley & Sons, Toronto, Ontario, 1972.

[Flo1] Floyd, M.A., ***Single Step Optimal Control Of Large Space Structures***, Ph.D. Thesis, Massachusetts Institute of Technology, 1984.

[Fre1] Freiherr, Greg, ***"Shuttle Pit Stop,"*** *Smithsonian Air & Space*, pp. 71–80, Oct. 1990.

[Gen1] Gentry, Arvel, ***"Aerodynamic Characteristics of Arbitrary Three–Dimensional Shapes at Hypersonic Speeds,"*** *Fifth ICAS Congress*, London, England, Sept. 1966.

[Goo1] Goodwin, G.C., Ramadge, P.J., and Caines, P.E., ***"Discrete–Time Stochastic Adaptive Control,"*** *SIAM Journal of Control and Optimization*, Vol. 19, No. 6, Nov. 1981.

[Goo2] Goodwin, G.C., Ramadge, P.J., and Caines, P.E., ***"Discrete–Time Multivariable Adaptive Control,"*** *IEEE Transactions on Automatic Control*, Vol. AC–25, No. 3, June 1980.

[Gre1] Gregory, T.J., ***"Credibility of NASP,"*** *Aerospace America*, pp. 42–46, Sept. 1989.

[Gue1] Guez, Allon, ***"Heuristically Enhanced Optimal Control,"*** *Proceedings of the 25th Conference on Decision and Control*, pp. 633–637, Athens, Greece, Dec. 1986.

[Hal1] Hallion, R.P., ***On The Frontier – Flight Research at Dryden 1946–1981***, NASA SP–4303, 1984.

[Han1] Hansen, James R., ***"Transition to Space – A history of 'spaceplane' concepts at Langley Aeronautical Laboratory,"*** *Journal Of Pioneering Space*, British Interplanetary Society, Vol. 40, pp. 67–80, Feb. 1987.

[Har2] Hart, P.E., Nilsson, N.J., and Raphael, B., ***"A Formal Basis for the Heuristic Determination of Minimum Cost Paths,"*** *IEEE Transactions on System Science and Cybernetics*, Vol. 4, No. 2, 1968.

[Har3] Hart, P.E., Nilsson, N.J., and Raphael, B., ***"Correction to a Formal Basis for the Heuristic Determination of Minimum Cost Paths,"*** *SIGART Newsletter*, Vol. 37, 1972.

[Hat2] Hattis, P.D., ***Useful Modeling Data Provided by Dryden***, C.S. Draper Laboratory, Cambridge, M.A., Internal Memo #TAV–86–9, Aug. 1986.

[Hat3] Hattis, P.D., ***A First Order Aerospaceplane Mass Property and Volume Requirements Model***, C.S. Draper Laboratory, Cambridge, M.A., Internal Memo #TAV–86–10, Sept. 1986.

[Hat4] Hattis, P.D., ***Aerospaceplane First Order Propulsion System Mass Property and Performance Models***, C.S. Draper Laboratory, Cambridge, M.A., Internal Memo #TAV–86–11, Sept. 1986.

[Hay1] Hayes, W.D. and Probstein,R.F., "***Newtonian Flow Theory in Hypersonic Aerodynamics,***" Paper based on Text – Hypersonic Flow Theory, Academic Press, 1959.

[Hed1] Hedrick, J.K., and Paynter, H.M., Nonlinear System Analysis And Synthesis: Volume 1 – Fundamental Principles, American Society of Mechanical Engineers, New York, N.Y., 1978.

[Hel1] Held, M. and Karp, R.M., "***The Travelling Salesman Problem and Minimum Spanning Trees,***" *Operations Research*, Vol. 18, 1970.

[Her1] Hermann, R. and Krener, A.J.,"***Nonlinear Controllability and Observability,***" *IEEE Transactions on Automatic Control*, Vol. AC–22, No. 5, Oct. 1977.

[Hon1] Honeywell Systems and Research Center, ***Proposed System Functional Requirements for a NASP Type Vehicle Air Data System***, NASP Contractor Report 1044, Minneapolis, M.N., 1989.

[Hug1] Hughes, P.C., Spacecraft Attitude Dynamics, John Wiley & Sons, Toronto, Ontario, 1986.

[Hun1] Hunt, R.L., Su, R., and Meyer, G., "***Global Transformations of Nonlinear Systems,***" *IEEE Transactions on Automatic Control*, Vol. AC–28, No. 1, pp. 24–31, 1983.

[Ise1] Isermann, Rolf, "***Parameter Adaptive Control Algorithms – A Tutorial,***" *Automatica*, Vol. 18, pp. 513–528, Sept. 1982.

[Jac1] Jackson, L.R. et al., "***Hypersonic Structures and Materials: a Progress Report,***" *Aerospace America*, pp.24–30, Oct. 1987.

[Jas1] Jaslow, Howard, "***Aerodynamic Relationships Inherent in Newtonian Impact Theory,***" *AIAA Journal*, Vol. 6, No. 4, pp. 608–612, 1968.

[Joh1] Johnston, P.J., Whitehead, A.H. Jr., and Chapman, G.T., "***Fitting Aerodynamics and Propulsion into the Puzzle,***" *Aerospace America*, pp. 32–37, Sept. 1987.

[Kan1] Kang, B., ***Integrated Data Processing System for Air Breathing Hypersonic Vehicles***, Internal Memo EGD–91–030, Charles Stark Draper Laboratory, June 1991.

[Kap1] KappenWallner, G., "***Fundamentals of Hypersonics: Aerodynamics and Heat Transfer,***" In Von Karman Institute for Fluid Dynamics, Lecture Series, Feb. 6–10, 1984.

[Kau1] Kaufman, H., et al., ***Digital Adaptive Flight Controller Development***, Technical Report CR–2466, Rensselaer Polytechnic Institute, NASA Contract, Dec. 1974.

[Kau2] Kaufman, H., ***Research in Digital Adaptive Flight Controllers***, Technical Report CR–2684, Rensselaer Polytechnic Institute, NASA Contract, May 1976.

[Key1] Keyworth, G.A., and Abell, B.R., "***How To Make Space Launch Routine***," *Technology Review*, pp. 26–31, Oct. 1990.

[Kir1] Kirk, D.E., Optimal Control Theory: An Introduction, Prentice Hall, N.J., 1970.

[Klu1] Kluger, J., " ***Space Plane***," *Discover*, pp. 81–84, Nov. 1989.

[Kor1] Kors, D.L., and Kissinger, R.D., "***The Challenge of Demonstrating the X–30***," *Aerospace America*, pp. 22–24, July 1990.

[Kre1] Kreisselmeier, Gerhard, "***Perspectives on the Application of Adaptive Control to Aircraft Systems***," article in Applications of Adaptive Control, Naraendra, K.S. and Monopoli, R.V., Editors, pp. 269–290, Academic Press, New York, N.Y., 1980.

[Kwa1] Kwakernaak, H. and Sivan, R., Linear Optimal Control Systems, Wiley–Iterscience, New York, N.Y., 1972.

[Lan1] Landau, I.D. and Courtiol, B., "***Adaptive Model Following Systems for Flight Control and Simulation***," AIAA Paper No. 72–95, AIAA 10th Aerospace Sciences Meeting, San Diego, C.A., Jan. 1972.

[Lea1] Learmount, D., "***U.S. Air Transport Technology – Where Next*** ," *Flight International*, Vol 130, pp. 120–128, Aug 30, 1986.

[Lew1] Lewis, M.J. and Kothari, A.P., "***The Use of Hypersonic Waveriders for Planetary Exploration***," *AIAA/JPL 2nd International Conference on Solar System Exploration*, Pasadena, C.A., Aug 22–24, 1989.

[Lew3] Lewis, Mark J., ***The Prediction of Inlet Flow Stratification and its Influence on the Performance of Air–Breathing Hypersonic Propulsion Systems***, Ph.D. Thesis, Massachusetts Institute of Technology, 1988.

[Lie1] Liepmann, Hans W., and Roshko, A., Elements of Gasdynamics, California Institute of Technology, John Wiley & Sons, Inc., 1957.

[Loz1] Lozano–Leal, R. and Goodwin, Graham C., "***A Globally Convergent Adaptive Pole Placement Algorithm Without a Persistency of Excitation Requirement***," *IEEE Transactions on Automatic Control*, Vol. AC–30, No. 8, August 1985.

[Mac1] MacConochie, I.O., and Klich, P.J., ***Techniques for the Determination of Mass Properties of Earth–to–Orbit Transportation Systems***, NASA Technical Report N78–30164, NASA Langley Research Center, V.A., June 1978.

[Mar1] Martinez–Sanchez, M., "***Fundamentals of Hypersonic Airbreathing Propulsion,***" *AIAA Professional Study Series, Preceding the 24th Joint Propulsion Conference,* Boston, M.A., July 1988.

[Mat1] Mattson, Erik A., ***An Implicit Algorithm for Analysis of an Airframe–Integrated Scramjet Propulsion Cycle***, Master's Thesis, Massachusetts Institute of Technology, 1988.

[May1] Mayne, D.Q., and Michalska, H., "***Receding Horizon Control Of Non–Linear Systems,***" *Proceedings of the 27th Conference on Decision and Control*, pp. 464–465, Austin, T.X., Dec. 1988.

[Mcc1] McCarty, J.P. et al., "***U.S. Liquid Propulsion Technology Stands Still,***" *Aerospace America*, pp. 40–43, July 1990.

[McR1] McRuer, D. et al., Aircraft Dynamics and Automatic Control, Princeton University Press, Princeton, N.J., 1973.

[Mil2] Miller, W.T. and Sutton, R.S., Neural Networks for Control, MIT Press, Cambridge, M.A., 1990.

[Mil3] Miles, R.B., Santavicca, A., and Zimmermann, M., ***Evaluation of Non–Intrusive Flow Measurement Techniques for a Re–Entry Flight Experiment***, NASA Contractor Report 172142, Princeton University, Princeton, N.J., Nov. 1983.

[Mor1] Morse, A.S., "***Global Stability of Parameter–Adaptive Control Systems,***" *IEEE Transactions on Automatic Control*, Vol. 25, No. 3, pp. 433–439, June 1980.

[Nar1] Narendra, K.S., and Valavani, L.S., "***Stable Adaptive Controller Design – Direct Control,***" *IEEE Transactions on Automatic Control*, Vol. 23, No. 4, pp. 570–582, Aug. 1978.

[Nar2] Narendra, K.S., Lin, Y.H., Valavani, L.S., "***Stable Adaptive Controller Design Part II: Proof of Stability,***" *IEEE Transactions on Automatic Control*, Vol. 25, pp. 440–448, June 1980.

[Nar3] Narendra, Kumpati S. and Annaswamy, A.M., "***A New Adaptive Law For Robust Adaptation Without Persistent Excitiation,***" *IEEE Transactions on Automatic Control*, Vol. AC–32, No. 2, pp. 134–145, Feb. 1987.

[New1] Newman, N. and Pinckert, R., "***Materials for the NASP,***" *Aerospace America*, pp. 24–26,31, May 1989.

[Nii1] Niiya, Craig K., ***An Application of the A^* Search Technique To Trajectory Optimization***, Master's Thesis, Massachusetts Institute of Technology, June 1990.

[Oss1] Ossman, Kathleen A. and Kamen, Edward W., "***Adaptive Regulation of MIMO Linear Discrete–Time Systems Without Requiring A Persistent Excitation,***" *IEEE Transactions on Automatic Control*, Vol. AC–32, No. 5, May 1987.

[Par1] Paradiso, Joe, ***Application of Linear Programming to Coordinated Management of Jets and Aerosurfaces for Aerospace Vehicle Control***, Technical Report CSDL–R–2065, C.S. Draper Laboratory, Nov. 1988.

[Par2] Paradiso, J.A., ***A Search–Based Approach to Steering Single Gimballed CMGs***, Technical Report, CSDL–R–2261, C.S. Draper Laboratory, Cambridge, M.A., Aug. 1991.

[Par3] Parks, P.C., "***Lyapunov Redesign of Model Reference Adaptive Control Systems,***" *IEEE Transactions on Automatic Control*, Vol. 11, pp. 362–367, 1966.

[Pea1] Pearl, J., Heuristics – Intelligent Search Strategies For Computer Problem Solving, Addison–Wesley, Menlo Park, C.A., 1985.

[Pik1] Pike, Jack, "***Moments and Forces on General Convex Bodies in Hypersonic Flow,***" *AIAA Journal*, Vol. 12, No. 9, pp. 1241–1247, 1974.

[Pov1] Povinelli, L.A., Dwoyer, D.L., and Green, M.J., "***CFD Propels NASP Propulsion Progress,***" *Aerospace America*, pp. 26–27, July 1990.

[Ren1] Ren, Chiang–Hwa, ***A Computer Based Model for the Performance Analysis of a Scramjet Propulsion System***, Master's Thesis, Massachusetts Institute of Technology, Jan. 1989.

[Roh1] Rohrs, C.E., Athans, M., Valavani, L., and Stein, G., "***Some Design Guidelines For Discrete–Time Adaptive Controllers,***" *Automatica*, Vol. 20, No. 5, pp. 653–660, 1984.

[Roh2] Rohrs, C.E., Valavani, L.S., Athans, M., and Stein, G., "***Robustness of Continuous–Time Adaptive Control Algorithms in the Presence of Unmodelled Dynamics,***" *IEEE Transactions on Automatic Control*, Vol. AC–30, No. 9, Sept. 1985.

[Roh3] Rohrs, C.E., "***Rethinking Adaptive Control for the 90's,***" *Proceedings of the 29th Conference on Decision and Control*, pp. 3143–3145, Honolulu, Hawaii, Dec. 1990.

[Ryn1] Rynaski, E.G., "***Adaptive Control: Application to Aircraft,***" article in Applications of Adaptive Control, Narendra, K.S. and Monopoli, R.V., Editors, pp. 245–267, Academic Press, New York, N.Y., 1980.

[San1] Sansoy, Sabri A., ***Design Optimization of A Transatmospheric Launch Vehicle***, Master's Thesis, Massachusetts Institute of Technology, 1987.

[She1] Sheetz, N.W., "***Ballistics Range Boundary Layer Transition Measurements on Cones at Hypersonic Speeds,***" in C.S. Wells, editor, Viscous Drag Reduction, pp. 53–84, Plenum Press, New York, N.Y., 1980.

[Slo1] Slotine, J.J., and Li, W., Applied Nonlinear Control, Prentice Hall, N.J., 1991.

[Ste1] Stein, Gunter, "***Adaptive Flight Control: A Pragmatic View,***" article in Applications of Adaptive Control, Narendra, K.S. and Monopoli, R.V.Editors, pp. 291–312, Academic Press, New York, N.Y., 1980.

[Ten1] Tenney,D.R., Lisagor, W.B., and Dixon, S.C., "***Materials and Structures for Hypersonic Vehicles,***" *Journal of Aircraft*, Vol. 26, No. 11, pp. 953–968, Nov. 1989.

[Thr1] Throckmorton, D., "***Enabling Technologies Research and Development for the National Aero–Space Plane,***" *AIAA First National Aero–Space Plane Conference*, AIAA–89–5009, Dayton, O.H., July 20–21, 1989.

[Vid1] Vidyasagar, M., Nonlinear Systems Analysis, Prentice Hall, N.J., 1978.

[Voe1] Voelcker, J., "***The iffy Orient Express,***" *Spectrum*, pp. 31–33, Aug. 1988.

[Wal1] Waltrup, P.J. et al., "***A Procedure for Optimizing the Design of Scramjet Engines,***" *Journal of Spacecraft*, Article 78–1110, Vol 16, No. 3, pp. 163–171, May–June 1979.

[Wei1] Weidner, J.P. et al., "***Scramjet Integration on Hypersonic Research Airplane Concepts,***" *Journal of Aircraft*, Vol. 14, No. 5, pp. 460–466, May 1977.

[Whi1] Whitaker, H.P., ***Design of Model Reference Adaptive Control Systems for Aircraft***, Technical Report R–0164, MIT Instrumentation Laboratory, Sept. 1958.

[Whi2] Whitaker, H.P., ***Design Capabilities of Model Reference Adaptive Systems***, Technical Report R–374, MIT Instrumentation Laboratory, July 1962.

[Wil1] Williams, R.M., "***National Aerospace Plane: Technology for America's Future,***" *Aerospace America*, pp. 18–22, Nov. 1986.