Trajectory Optimization for a Fixed Trim Re-entry Vehicle Using Direct Collocation with Nonlinear Programming

by

Robert T. Bibeau, ENS, USNR

B.S. Aerospace and Systems Engineering
United States Naval Academy
(1997)

SUBMITTED TO THE DEPARTMENT OF AERONAUTICS AND ASTRONAUTICS
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

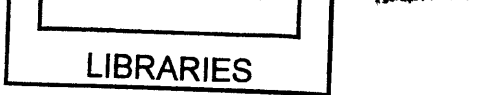
MASTER OF SCIENCE IN
AERONAUTICS AND ASTRONAUTICS
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
MAY, 1999
[June 2001]

Signature of Author _____
Department of Aeronautics and Astronautics
May, 1999

Approved by _____
David Rubenstein
The Charles Stark Draper Laboratory, Inc.
Technical Supervisor

Certified by _____
Rudrapatna Ramnath
Senior Lecturer, Department of Aeronautics and Astronautics
Thesis Supervisor

Accepted by _____
Jaime Peraire
Associate Professor of Aeronautics and Astronautics
Chairman, Department Graduate Committee



Trajectory Optimization for a Fixed Trim Re-entry Vehicle Using Direct Collocation with Nonlinear Programming

by

R.T. Bibeau

Submitted to the Department of Aeronautics and Astronautics on May 7, 1999 in partial fulfillment of the requirements for the Degree of Master of Science in Aeronautics and Astronautics

Abstract

A procedure is developed to calculate locally optimal trajectories for a class of fixed-trim atmospheric re-entry vehicles. A four degree-of-freedom vehicle model is introduced and appropriate environmental models are chosen and implemented. Software is developed to discretize the optimal control problem using a direct collocation method. The resulting parameter optimization problem is solved using the MINOS non-linear programming software package. The resulting collocation guidance software is tested using data for the Kistler K-1 vehicle system and an existing vehicle simulation. Mass, wind, density, and entry angle dispersions are considered, as are various strategies for updating the trajectory during flight. The results demonstrate that the collocation method is a viable approach to the re-entry vehicle guidance problem. The collocation method integrates the vehicle equations of motion to a useful degree of accuracy using as few as 10 nodes, and the resulting control histories yield acceptably small final position errors.

Thesis Supervisor: Rudrapatna Ramnath
Senior Lecturer, Department of Aeronautics and Astronautics
Massachusetts Institute of Technology

Technical Supervisor: David Rubenstein
Senior Member of the Technical Staff
Charles Stark Draper Laboratory

Acknowledgments

In no particular order, I would like to thank:

My family.

Dave Rubenstein for his constant patience and encouragement. His guidance was instrumental in producing this thesis. Without his help this thesis would have been all but impossible to complete (and it would have been left-justified).

The numerous other Draper employees and MIT faculty members who helped along the way, including Professor Ramnath, Phil Hattis, Dave Carter, Mike Ricard, Doug Fuhry, and others. It amazes me that in my two years at Draper every person I asked for assistance was willing to help.

The Aerospace Engineering faculty at the United States Naval Academy, especially Professors Karpouzian and Rogers, without whose instruction I wouldn't be here.

All of the other people who have made my stay in New England more enjoyable (or in some cases just more colorful), including Meg, Nick, Geoff, Mike, Nate and the rest of CIVLANT '97, Lee, the Banks family, and even Oliver.

My friends from the USNA class of '97. I'll be joining you soon.

This thesis was prepared at the Charles Stark Draper Laboratory, Inc., under Contract #191-4000. 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.

Robert T. Bibeau

3 MAY 99
(Date)

Table of Contents

List of Figures

List of Tables

List of Symbols, Acronyms, Abbreviations, Notation, and Constants

List of Symbols

α Re-entry vehicle angle of attack
$\Delta\alpha$ Angle of attack deviation from trim
β Re-entry vehicle side-slip angle
$\Delta\beta$ Side-slip angle deviation from trim
γ_i Re-entry vehicle inertial flight path angle
μ Gravitational constant of the Earth
ρ Atmospheric density
ϕ Re-entry vehicle bank angle
ϕ_i Re-entry vehicle inertial bank angle
ϕ_r Position component of re-entry vehicle inertial bank angle
$\dot{\phi}$ Re-entry vehicle bank rate
ψ_i Re-entry vehicle inertial azimuth angle
ω_e Magnitude of the angular velocity of the Earth about its rotational axis
a Local speed of sound
$\vec{a}$ Coefficient vector for Hermite polynomial estimate of vehicle state
$\vec{b}$ Coefficient vector for Hermite polynomial estimate of vehicle state
$\vec{c}$ Coefficient vector for Hermite polynomial estimate of vehicle state
C_d Re-entry vehicle drag coefficient
C_l Re-entry vehicle lift coefficient
D Magnitude of re-entry vehicle drag
$\vec{d}_i = \begin{Bmatrix} d_{i_1} \\ d_{i_2} \\ \vdots \\ d_{i_N} \end{Bmatrix}$ Defect vector associated with the i^{th} collocation point
e_p Vehicle final position error
$\vec{e}$ Coefficient vector for Hermite polynomial estimate of vehicle state
$f(\vec{x},u)$ Re-entry vehicle equations of motion function
f_{total} Total ACS fuel used during entire vehicle trajectory
$\vec{g}_a = \begin{Bmatrix} g_{a_x} \\ g_{a_y} \\ g_{a_z} \end{Bmatrix}$ Gravitational acceleration vector expressed in the "a" reference frame
$\bar{I}$ Identity matrix
I_{bank} Re-entry vehicle moment of inertia about bank axis
i Index variable

J_n Coefficients of Earth's gravitational potential
j Index variable
k Index variable
L Magnitude of re-entry vehicle lift
M Local free-stream Mach number
$M_{control}$ Control moment about the re-entry vehicle bank axis
m Total mass of re-entry vehicle
m_{defect} Number of nonlinear constraints due to the defect vector
N Number of nodes used in the collocation scheme
O Objective value to be minimized in the collocation scheme
p number of non-zero elements of the constraint Jacobian
R Specific gas constant for air
r_{axis} Local distance from the rotational axis of the Earth
r_e Equatorial radius of the Earth
S_{ref} Vehicle reference area used to calculate aerodynamic coefficients
t Time
t_f Time at end of trajectory
Δt Time step between nodes
u Non-linear programming problem control variable
u_c Non-linear programming problem control variable at the center of a trajectory segment
W Weighting factor used in the objective function to be minimized
$\vec{x}$ Re-entry vehicle state
$\vec{x}_H$ Hermite estimate of re-entry vehicle state
$\vec{x}_{H_c}$ Hermite estimate of re-entry vehicle state at center of trajectory segment
$\dot{\vec{x}}_{EOM}$ Re-entry vehicle state derivative at center of trajectory segment found using vehicle equations of motion
$\dot{\vec{x}}_H$ Hermite estimate of re-entry vehicle state derivative
$\dot{\vec{x}}_{H_c}$ Hermite estimate of re-entry vehicle state derivative at center of trajectory segment

List of Acronyms

ACS	Attitude Control System
AGL	Above Ground Level
DCNLP	Direct Collocation with Nonlinear Programming
EI	Entry Interphase
IVS	Integrated Vehicle Simulation
LAP	Launch Assist Platform
LEO	Low Earth Orbit
NLP	Non-Linear Programming
OV	Orbital Vehicle

RV	Re-entry Vehicle
US76	United States Standard Atmosphere, 1976

List of Abbreviations

b	Body reference frame
c.g.	Re-entry vehicle center of gravity
ECEF	Earth-Centered Earth-Fixed reference frame
ECI	Earth-Centered Inertial reference frame
lv	Level Velocity-Fixed reference frame
tb	Trim body reference frame
v	Velocity-fixed reference frame
vv	Vertical Velocity-Fixed reference frame

List of Notation

$\vec{a}$ Vector "a"

a Scalar "a"

$\overline{a}$ Matrix "a"

$\vec{F}_{a_b} = \begin{Bmatrix} F_{a_{b_x}} \\ F_{a_{b_y}} \\ F_{a_{b_z}} \end{Bmatrix}$ Vector of force "a" expressed in the "b" reference frame

$\vec{r}_{a_c} = \begin{Bmatrix} x_{a_c} \\ y_{a_c} \\ z_{a_c} \end{Bmatrix}$ Position vector of "a" expressed in the "c" reference frame

${}_b\overline{T}_a$ Transformation matrix from the "a" reference frame to the "b" reference frame

$\vec{V}^b_{a_c} = \begin{Bmatrix} u^b_{a_c} \\ v^b_{a_c} \\ w^b_{a_c} \end{Bmatrix}$ Velocity of "a" with respect to "b" expressed in the "c" reference frame

List of Constants

$\mu = 1.407646853 \times 10^{16} \frac{ft^3}{s^2}$ Gravitational constant of the Earth

$\omega_e = 7.2921151467 \times 10^{-5} rad/s$ Magnitude of the angular velocity of the Earth

$J_2 = 1.08263 \times 10^{-3}$ Coefficient of Earth's gravitational potential

$J_3 = -2.532 \times 10^{-6}$ Coefficient of Earth's gravitational potential

$J_4 = -1.611 \times 10^{-6}$ Coefficient of Earth's gravitational potential

$R = 1716.567 \frac{ft \cdot lb}{(slug)(^\circ R)}$ Specific gas constant for air

$r_e = 2.09256463 \times 10^7 \, ft$ Equatorial radius of the Earth

Chapter 1: Introduction

The purpose of this thesis is to develop a method to calculate optimal trajectories for fixed-trim atmospheric re-entry vehicles. This method is then tested using vehicle data for the Kistler K-1 launch system. This chapter provides background and motivation for the problem followed by a brief description of the Kistler project. The motivation for considering a collocation approach is also discussed. The chapter concludes with an overview of the thesis.

1.1 Problem Background and Motivation

During a controlled re-entry into Earth's atmosphere, a vehicle must transition from a high altitude, high speed state to a specified low altitude, lower speed state from which it can be recovered. Depending on the chosen vehicle trajectory, this transition can involve potentially catastrophic thermal and pressure loads on the vehicle.[25] It is also frequently necessary to constrain the final position to within tolerances which are small when compared with the trajectory length and average vehicle velocity. For these reasons, many re-entry vehicles have the ability to alter their trajectory in flight using some form of control system. Maneuvering atmospheric re-entry vehicles are used in numerous applications and include strategic weapons and recoverable manned and unmanned spacecraft. This thesis is concerned with the maneuvering of an atmospheric re-entry vehicle in an attempt to achieve a predetermined final position while minimizing the energy required to do so.

The three vehicle configurations most commonly considered for maneuvering re-entry vehicles are the cruciform, variable-trim bank-to-turn, and fixed-trim configurations.[6] The cruciform is the most maneuverable and utilizes aerodynamic control surfaces to produce lift in both the pitch and yaw planes. This configuration would be useful for a ballistic missile attempting to evade defenses. The variable-trim bank-to-turn configuration uses a control surface to vary the amount of lift produced in the pitch plane, and turns by reorienting the lift vector. The orientation of the lift vector is typically described in terms of the bank angle, which is defined as the angle about the

vehicle air-relative velocity vector between the lift vector and the local vertical and is shown in figure 1.1-1. One example of a bank-to-turn vehicle is the space shuttle. A fixed-trim re-entry vehicle has negligible control over its angle of attack or side-slip angle and can only alter its flight path by changing the vehicle bank angle. Examples of fixed-trim re-entry vehicles include the Apollo Command Module and the Soyuz spacecraft. While the fixed-trim configuration is by far the simplest configuration, the maneuverability of the vehicle is severely restricted since the magnitude of the lift vector cannot be controlled during flight.

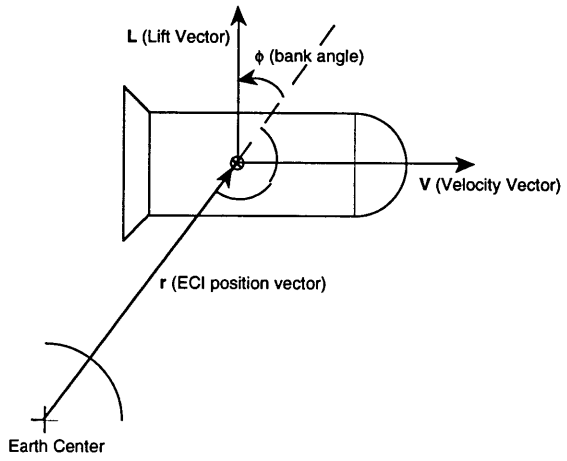


Figure 1.1-1: The vehicle bank angle

This thesis considers trajectories for fixed-trim maneuvering re-entry vehicles. The control history of such a vehicle can be described completely by its bank angle profile. The goal of the optimal control problem for a fixed-trim re-entry vehicle is to determine the bank profile which minimizes some scalar function given a specified set of initial, final, and environmental conditions. In this case, the objective to be minimized is a function of vehicle fuel consumption and final position miss distance.

1.2 The Kistler Program

The trajectories developed in this study were tested using realistic vehicle data. This was accomplished using data for the Kistler K-1 launch system. The Kistler K-1 is a reusable launch system designed to place one or more payloads in low Earth orbit (LEO)

and return to a designated circular landing area 6000 feet in diameter.[26] The vehicle system consists of two stages. The first stage is known as the Launch Assist Platform (LAP) and propels the load-bearing second stage to an altitude from which it can achieve orbit. After separation, the LAP returns to a landing site in the vicinity of the launch site. The second stage is known as the Orbital Vehicle (OV). The OV delivers the payload to LEO and then returns to another landing site near the launch area. Sketches of the system are shown in figures 1.2-1 and 1.2-2. The trajectory generation techniques discussed in this thesis may be applied to the return of either vehicle stage.

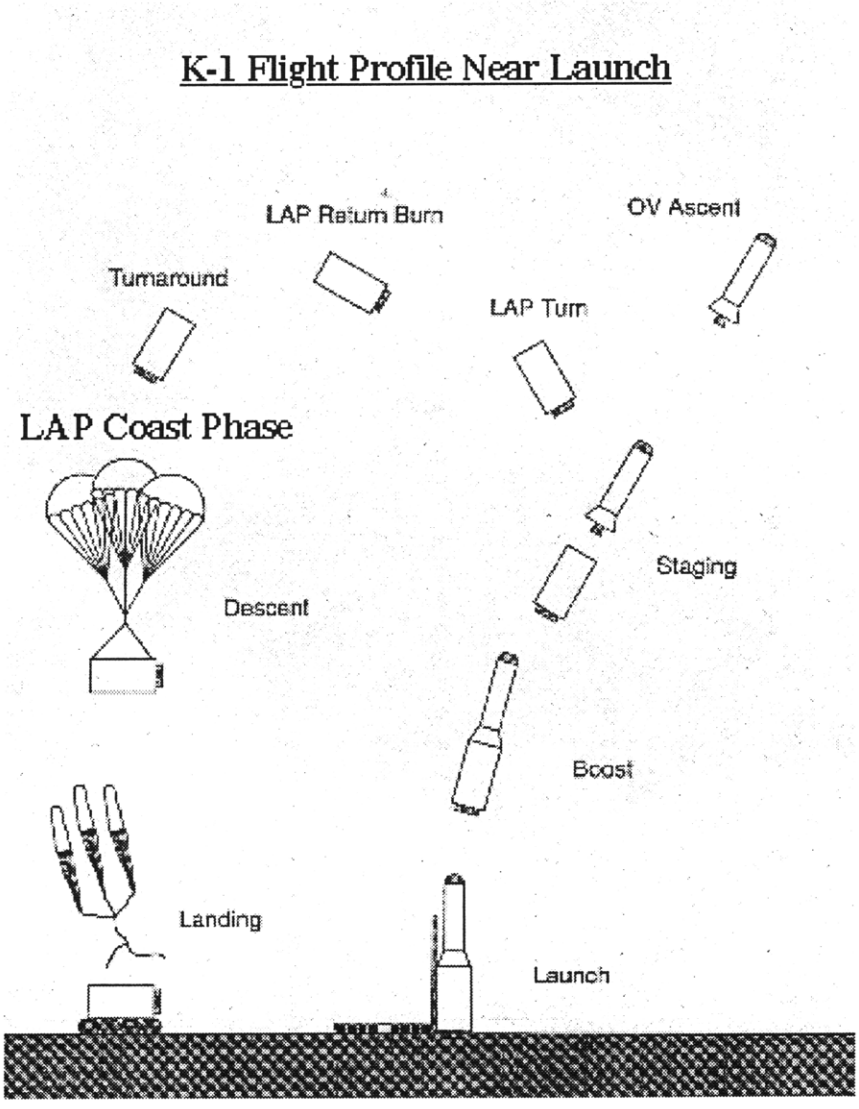


Figure 1.2-1: K-1 Flight profile near launch

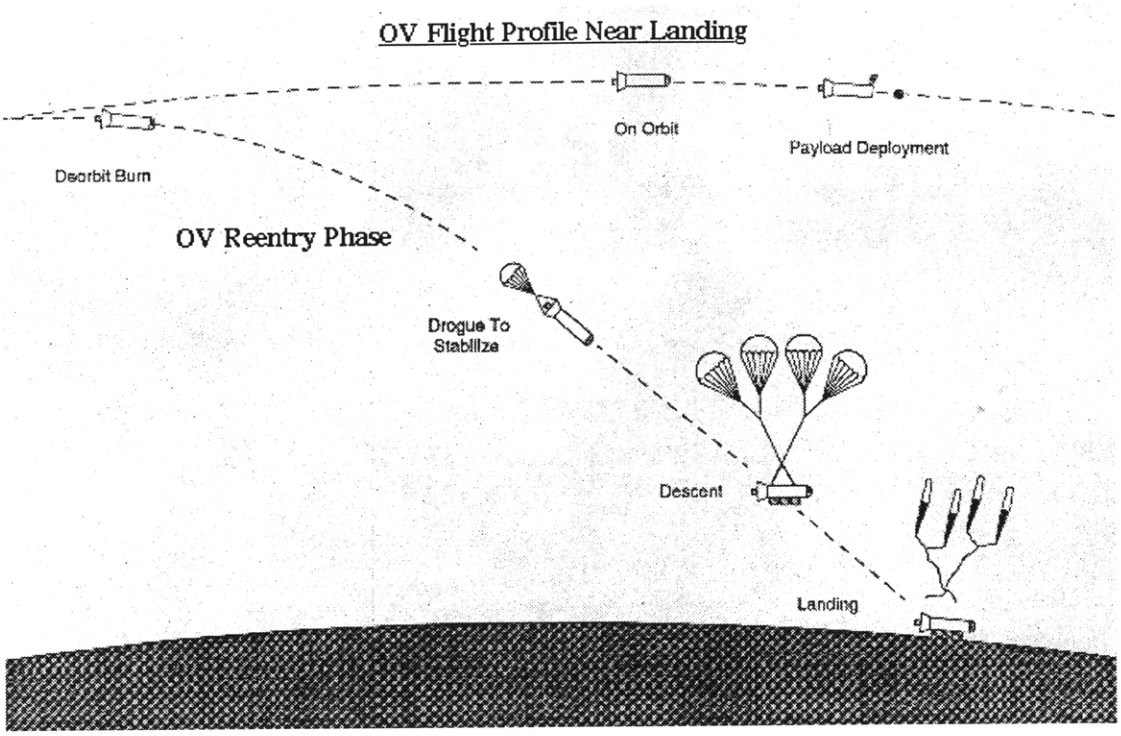


Figure 1.2-2: OV Flight profile near landing

NOTE- THESE DIAGRAMS ARE SCHEMATIC ONLY AND ARE NOT MEANT TO DEPICT THE ACTUAL APPEARANCE OF THE KISTLER LAUNCH SYSTEM

The OV experiences a re-entry phase beginning at entry interface (EI) at an altitude of 400,000 feet above ground level (AGL). The OV is initially at near-orbital speed ($O(26{,}000\,^{ft}/_{s})$). Throughout the re-entry phase the bank angle of the OV may be controlled using the vehicle attitude control system (ACS). The vehicle lands with the aid of several parachutes and an airbag system. For the purpose of this thesis, active control of the vehicle trajectory is assumed to cease after deployment of the first parachute.

Although the LAP does not leave the atmosphere, it does experience a phase of flight similar to the OV re-entry phase. After separation, the LAP executes a return burn and a turn-around maneuver. The LAP then enters a "coast" phase and glides back to the landing site. The coast phase of LAP flight is analogous to the OV re-entry phase. The LAP has a similar ACS and uses the same control strategy as the OV. As with the OV, active control of the LAP trajectory ends with the deployment of the first parachute.

The OV currently uses a predictor-corrector guidance strategy. The software assumes a certain bank angle profile, integrates the equations of motion to predict the resulting

miss distance, and applies an appropriate correction. The nominal bank profile consists of an initial bank maneuver to achieve and maintain a predetermined non-zero bank angle. At a later time, the vehicle undergoes a bank reversal to achieve a second calculated bank angle. The execution time and magnitude of the bank reversal maneuver are determined by the guidance software in real-time such that the miss distance is minimized. The vehicle control system implements the bank profile commanded by the vehicle guidance. The current guidance design assumes that the vehicle loading and heating constraints are controlled implicitly through the selection of an appropriate vehicle state at EI, and has no objective other than to minimize the final position miss distance. The design also assumes that the vehicle is constantly at trim. The vehicle control system attempts to enforce this condition by using the ACS to minimize any deviation from the trim angle of attack or side-slip angle.

The LAP currently has no active guidance strategy, and attempts to hold a constant zero degree bank angle. The LAP coast segment begins much lower in the atmosphere and at a much slower velocity ($O(1000\,^{ft}/_{s})$) than the OV. Due to the nature of its nominal trajectory, the LAP has much less control over its final position than the OV and has little difficulty achieving the specified final position accuracy. The LAP final position is controlled largely by achieving the appropriate vehicle state at the initialization of the coast phase. For these reasons, most of the software testing described in this thesis was done using the OV.

For the purpose of this study, the nominal Kistler guidance software was replaced with the trajectory optimization software discussed in the following chapters. The control software was modified as necessary to implement the resulting vehicle trajectories. All other vehicle characteristics were assumed to be nominal.

1.3 Introduction to Collocation Methods

An alternate guidance strategy is to use a collocation technique to solve an approximation of the optimal control problem. In general, the goal of an optimal control problem is to find the control history $u(t)$ that minimizes a scalar objective function subject to the problem constraints. This objective can be a function of any combination

of the problem states or controls at any time during the trajectory. In this case, the specific optimal control problem is to find the bank angle profile $\phi(t)$ that minimizes some function of the ACS fuel usage and the final position miss distance. The vehicle equations of motion and initial conditions impose constraints on the problem. Additional constraints, such as those related to structural and thermal loading, may also be applied.

The first step in applying a collocation approach is to discretize the problem by considering the vehicle state only at specific points along the trajectory. These points are called nodes. The nodes are connected by trajectory segments or "subintervals"[14] The vehicle equations of motion may be enforced at other points along the trajectory termed collocation points.[29] After the collocation is performed, the optimal control problem is effectively approximated by a parameter optimization problem. This parameter optimization problem may then be solved using existing nonlinear programming techniques.

Collocation methods have been widely used for trajectory planning. This thesis examines the feasibility of applying a collocation technique to a fixed-trim re-entry vehicle. Potential advantages of using a collocation technique include the ability to choose a trajectory which minimizes fuel usage or any other function of the vehicle states or controls. The nature of the parameter optimization problem also allows the addition of constraints that would greatly complicate other optimization techniques, and are often not considered in other guidance strategies. Finally, the collocation technique simultaneously generates an estimate of the vehicle trajectory associated with the recommended control history. This trajectory estimate could potentially be used in some form of feedback system to reduce the effects of system noise or modeling error, or to determine if it is necessary to update the trajectory.

1.4 Thesis Overview

This thesis begins with a discussion of the general background, theory and applications associated with the study. The atmospheric re-entry problem is defined and the collocation procedure is introduced. The nonlinear programming methods and

software used to solve the parameter optimization problem are also discussed. Next, the necessary vehicle and environmental models are described. The vehicle state is chosen and the equations of motion are derived. The design and validation of the collocation software is described in detail. After the software development is described, its performance is analyzed in order to determine its feasibility for use as a guidance strategy. The collocation software is run under various conditions. The node density is varied, and various strategies for updating the control trajectory in flight are considered. Mass, atmospheric density, wind, and entry angle dispersions are also considered. The thesis ends with conclusions and recommendations for future work. A list of the works used in preparing this thesis follows the text.

1.5 Chapter summary

This chapter provided a brief introduction to the re-entry vehicle problem. It described the motivation for attempting to use a collocation procedure to optimize a re-entry vehicle trajectory. The following chapter presents the theoretical background required for this study. Atmospheric re-entry, collocation methods, and nonlinear programming methods are discussed in depth.

Chapter 2: Background and Previous Work

This chapter provides a brief overview of the theory and applications associated with this thesis. Specifically, the atmospheric re-entry problem, collocation methods, and nonlinear programming problem are described.

2.1 The Atmospheric Re-entry Problem

During atmospheric flight at hypersonic speeds, a fixed-trim re-entry vehicle tends to approach a trim condition at which the moments of the vehicle about its center of mass approach zero. The only means of actively controlling the vehicle trajectory is to bank the vehicle about the air-relative velocity vector. The atmospheric re-entry problem for a fixed-trim re-entry vehicle is to calculate and implement a bank angle profile that results in the vehicle hitting the target while somehow managing the structural and heating loads on the vehicle and the control power used.

Prior to re-entry, the initial and final vehicle conditions are selected using trajectory planning techniques. A nominal trajectory and the corresponding control history are designed using assumed vehicle and environmental conditions. During flight, systems on board the re-entry vehicle must ensure that the actual re-entry trajectory meets the system design parameters, accounting for any unplanned disturbances or deviations from the assumed re-entry conditions. To accomplish this goal, the vehicle is provided various navigation, guidance, and control capabilities.[17] Navigation systems provide the vehicle position, velocity, and attitude with respect to a chosen reference frame. Using this information, the guidance system determines the desired vehicle trajectory and issues the necessary bank angle commands to the flight control system. The vehicle flight control system implements the guidance commands in order to follow the desired trajectory. This thesis is primarily concerned with the guidance portion of the re-entry vehicle problem. The guidance software developed in this thesis assumes the navigation and control capabilities of the re-entry vehicle to be ideal. That is, it is assumed that the vehicle position and velocity are available to the guidance software with perfect accuracy, and that the vehicle control system can maintain the precise bank angle

commanded by the guidance software. For testing purposes, the navigation and control software designed for the K-1 launch system are used with minor modifications to allow them to interface with the new guidance strategy. The actual K-1 navigation and control capabilities are included in the simulation used to test the guidance software.

Various guidance strategies have been considered for use with fixed-trim re-entry vehicles. For example, the Apollo Command Module guidance strategy involved calculating the component of the vehicle lift in the plane of the vehicle inertial position and air-relative velocity vectors. The magnitude of "in-plane" lift necessary to reach the target was continuously calculated. The vehicle bank angle was chosen such that the in-plane component of lift was held to this desired value, and any unwanted out-of-plane lift component was nulled by periodically rolling the vehicle to reverse the sign of the lateral lift. This periodic bank reversal was called "lateral switching."[15] Interestingly, a similar behavior was observed in certain cases when an in-flight trajectory optimization was performed for the Kistler OV, and will be described in Chapter 5.

Another traditional guidance strategy is "diveline guidance."[10],[25] Prior to flight, one or more "divelines" are established. These divelines are straight lines through the Earth's atmosphere, with the final diveline ending at the target. During flight, the vehicle simply attempts to keep its lift vector oriented toward the diveline. This strategy tends to keep the vehicle moving toward the desired diveline, such that the vehicle is in the vicinity of the diveline when it intersects the target.

A more sophisticated potential guidance strategy is to formulate the re-entry vehicle guidance problem as an optimal control problem. This strategy has previously been considered as an off-line trajectory planning technique[11], but increasing computer speed creates the potential for the trajectory optimization to be performed multiple times in flight. This is the approach considered in this thesis.

2.2 The Optimal Control Problem

The general optimal control problem is to find the design parameters $\vec{b}$, the control history $\vec{u}(t)$, and the state trajectory $\vec{x}(t)$that minimize the scalar performance index:

$$J = \int_{t_0}^{t_f} f_0(\vec{x},\vec{u},\vec{b})dt \tag{2.1}$$

subject to the differential constraint:

$$\dot{\vec{x}} = f(\vec{x},\vec{u},\vec{b}) \tag{2.2}$$

The problem is also subject to the specified initial or final conditions, boundary constraints on the states or controls, or any other general constraint affecting $\vec{x}$ or $\vec{u}$. For the problem considered in this thesis the vehicle design parameters are set and are not considered to be independent variables in the optimal control problem. As previously described, the fixed trim re-entry vehicle considered in this thesis has only one control variable, the bank rate $\dot{\phi}$. The initial problem time is also defined to be:

$$t_o = 0 \tag{2.3}$$

The optimal control problem for the fixed-trim re-entry vehicle becomes:

Minimize (over the control variable $\dot{\phi}$):

$$J = \int_0^{t_f} f_0(\vec{x},\dot{\phi})dt \tag{2.4}$$

Subject to:

$$\dot{\vec{x}} = f(\vec{x},\phi)$$

$$\vec{x}(t=0) = constant$$

additional general constraints

Research into the solution of optimal spacecraft trajectories has historically fallen into one of two categories.[9] Indirect methods use calculus of variations and Pontryagin's maximum principle to express the re-entry problem as a multi-point boundary value problem (BVP). The resulting BVPs are typically solved numerically.[28] Direct methods

involve the discretization of the optimal control problem into a parameter optimization problem. The resulting parameter optimization problem is then solved using nonlinear programming techniques. In general, indirect methods attempt to find an approximate solution to the exact problem, while direct methods seek a solution to an approximation of the re-entry problem.

Boundary value problems resulting from realistic re-entry vehicle problems can rarely be solved analytically. Problems solved using Pontryagin's Maximum Principle usually require numerical methods and significant computation.[13] These problems can be very difficult to solve because they require a very accurate initial guess for the solution.[9] Therefore, for non-trivial control problems it is usually necessary to discretize the problem and apply direct solution methods. This approach was chosen for this thesis.

2.3 Direct Solution Methods for the Optimal Control Problem

The goal of any direct method for solving the optimal control problem is to approximate the continuous problem with a parameter optimization problem. Since the problem state is only considered at discrete nodes, the state history must be integrated between these nodes using some form of quadrature rule. Direct solution methods may be categorized according to the type of integration scheme they use. An explicit integration scheme requires the value of the current problem state to be known in order to perform the next integration step. An implicit integration scheme has no such limitation and must use an iterative predictor-corrector approach.[14]

Hull[14] provides a recent and thorough survey of direct methods for solving optimal control problems. He describes a way of classifying direct solution methods by the unknown problem parameters. Hull considers four general methods for converting the optimal control problem into a parameter optimization problem. For a given problem the independent variables may be: (a) the controls, (b) the controls and some of the states, (c) the controls and all of the states, or (d) all of the states.[14] When estimates are calculated

for all of the states (i.e. all of the states are treated as independent variables) an implicit scheme may be used. Otherwise, an explicit integration scheme is required.

Explicit integration methods are simpler than implicit methods because they allow the entire trajectory to be integrated in one pass. The control variables at each node are treated as independent variables, and the state variables at each node may be calculated from the state and control variables at the previous node. These kinds of explicit methods are also known as "direct shooting methods".[14] Disadvantages of these methods are high sensitivity to the initial guess for the unknown parameters and reasonably high integration errors. A slight modification of direct shooting techniques involves providing guesses for the problem state at some nodes. The state variables at these nodes are treated as independent variables, and the equations of motion would therefore only be explicitly integrated over a portion of the full trajectory. These methods seek to reduce the integration error and are called "direct multiple shooting methods."[14]

Implicit integration methods require at a minimum an initial guess for the problem state at every node. The initial state guess most likely will not satisfy the problem differential constraint, and the degree of constraint violation between nodes is quantified as a set of residuals or "defects" which are driven to zero as part of the optimization process. Implicit methods may be further distinguished by the manner in which they treat the problem control variables.

The collocation technique used in this thesis was first described by Hargraves and Paris[11], and treats both the control and state variables as independent variables in the problem formulation. The state history between nodes is approximated using a series of cubic polynomials fitted between neighboring nodes (the choice of cubic polynomials is discussed by Hargraves and Paris[11]). The control history is assumed to be linear between nodes. This direct, implicit method for converting an optimal control problem into a parameter optimization problem is commonly called Direct Collocation with NonLinear Programming (DCNLP).

Hargreaves, Paris, and Vlases described how their technique was developed into a software package called OTIS (Optimal Trajectories by Implicit Simulation) for the U.S. Air Force.[12] The OTIS software is a FORTRAN program that is used to optimize point mass (3 degree-of freedom) and rigid body (6 degree-of-freedom) aerospace vehicle

trajectories. The OTIS software can handle multi-stage and multi-vehicle problems, and optimize both continuous control histories and vehicle design parameters. This software package is typically used for trajectory planning. In addition to atmospheric trajectory problems, DCNLP methods have been applied to a variety of other control problems including robotics problems[8], spacecraft attitude control[27], and orbital mechanics problems[9].

The final direct method that uses implicit integration is called differential inclusion. This is the fourth class of solutions considered by Hull, and treats only the state variables as independent variables. In this method, the control variables are completely removed from the parameter optimization problem. Instead, the controls are implemented as bounds on the allowable state variable time derivatives. Post-processing is required to determine the control history necessary to implement the desired state trajectory. The goal of this technique is to reduce the size of the resulting non-linear programming problem.[16] However, Conway and Larson have recently showed that differential inclusion in fact results in a larger problem than other direct methods because it requires the states to be interpolated linearly between nodes.[7] The use of a lower order quadrature rule requires the use of more nodes to achieve the same degree of accuracy, and actually increases the total size of the required NLP problem. A summary of solution methods for optimal control problems is shown in figure 2.2-1.

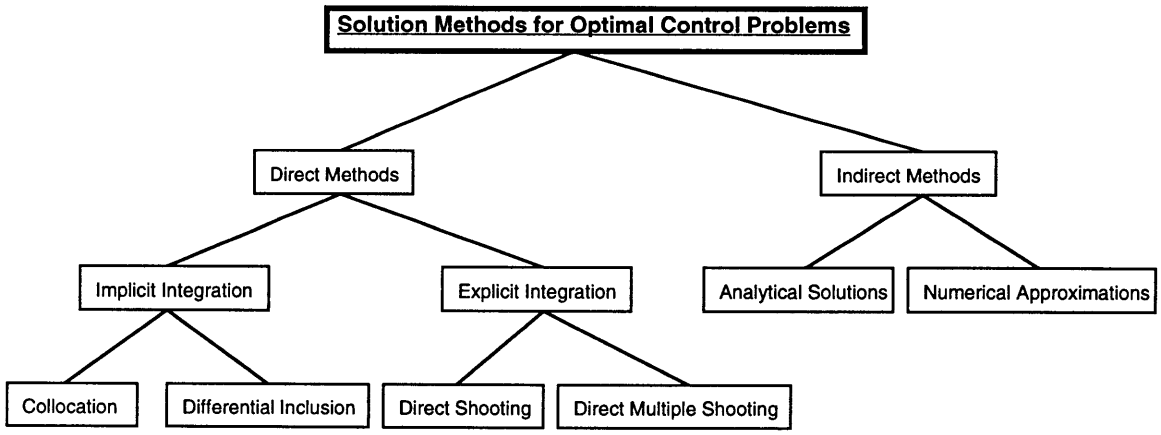


Figure 2.2-1: Solution methods for optimal control problems

2.4 Mathematical Programming Methods

Once the optimal control problem is approximated by a parameter optimization problem, the discrete problem must be solved. Parameter optimization problems are typically solved using mathematical programming. Numerous texts are available on the subject of mathematical programming[18],[31], and an extensive knowledge of the subject is not necessary to use the current generation of mathematical programming computer software. Therefore the subject is only briefly discussed here.

We begin by considering the linearly constrained programming problem, because it provides insight into the concepts of mathematical programming and will eventually be used as an intermediate step in solving the nonlinear problem. The general linear programming problem is to optimize a linear function of variables called the "objective function" subject to a set of linear equalities and/or inequalities called "constraints"[31]:

$$\text{Minimize } \vec{c}^T\vec{x} \tag{2.1a}$$

$$\text{Subject to } \overline{A}\vec{x} = \vec{b} \tag{2.b}$$

$$\text{And } \vec{x} \geq \vec{0} \tag{2.1c}$$

where $\vec{x}$ is the problem state n-vector, $\overline{A}$ is a $n \times m$ matrix, $\vec{b}$ is a m-vector, and $n \geq m$. Constraint inequalities are handled by introducing bounded "slack" variables. For example, the constraint:

$$x_1 \geq c \tag{2.2}$$

becomes:

$$x_1 + s_1 = b \tag{2.3a}$$

where:

$$s_1 \geq 0 \tag{2.3b}$$

A linear program with inequality constraints therefore becomes:

$$\text{Minimize: } \vec{c}^{\,T}\vec{x} \tag{2.4a}$$

$$\text{Subject to } \bar{A}\vec{x} + \bar{I}\vec{s} = \vec{b} \tag{2.4b}$$

$$\text{And } \vec{x} \geq \vec{0} \tag{2.4c}$$

where $\vec{s}$ is the set of slack variables. These slack variables may now be treated in the same manner as other problem state variables.

Any value of $\vec{x}$ that satisfies the constraint equations is called a "feasible" solution, and the feasible solution with the lowest possible objective value is called the "optimal" solution. A "basic" solution to the linear programming problem is created when any $n-m$ elements of the state vector $\vec{x}$ are set to zero, leaving a non-singular $m \times m$ system of linear equations. The elements of the state vector that are set to zero are called "non-basic" variables, while the remaining elements are the "basic" variables.

It can be shown that the optimal solution to any linear programming problem is a basic feasible solution.[18] This result is called the "Corner Point Theorem" or the "Fundamental Theorem of Linear Programming." The Fundamental Theorem of Linear Programming can be easily explained graphically for the two-dimensional case. Consider the two-dimensional x-y plane shown in figure 2.4-1. A finite number of inequality constraints define a subset of the plane in which a solution is feasible. Since the constraints are linear, this region is a polygon. For each point on the x-y plane there is an associated objective value, which may be sketched as a series of contour lines across the plane. Each contour line represents a constant objective value. Note that since the objective is linear, all of these contour lines must be straight. From inspection, it is clear that the minimum objective value in the feasible region will occur at a corner of the polygon. At this corner at least two of the inequality constraints are at their bounds, and the corresponding slack variables are set to zero. In this case these two slack variables are the non-basic variables. The remaining slack variables and the problem states (in this case x and y) are between their bounds and are determined by the problem constraints, since there is only one point in the plane where the two chosen constraints are exactly at their bounds.

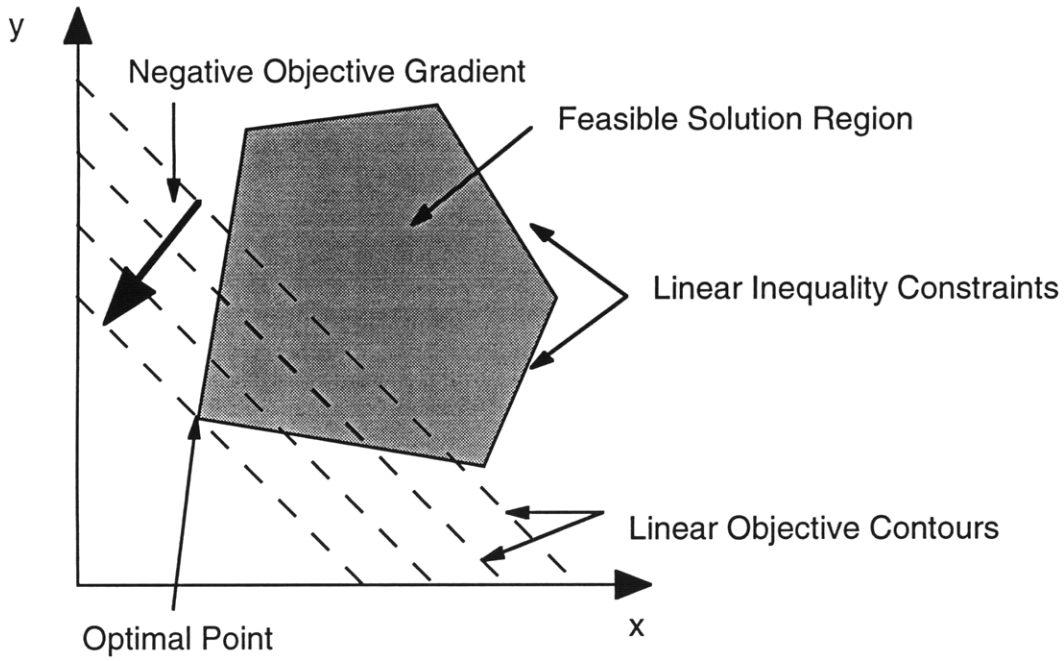


Figure 2.4-1: Graphical Interpretation of linearly constrained problem with linear objective

Since there are a finite number of basic feasible solutions, the optimal solution can be found by examining all of the basic feasible solutions and choosing the one with the lowest objective value. The commonly used simplex method is one way of moving from one basic feasible solution to another so as to continuously decrease the value of the objective function until a minimum is reached.[18]

A linearly constrained program with a non-linear objective function can be understood in a similar manner, as shown in figure 2.4-2. In this example there are still a certain number of basic and non-basic variables. In addition, there is a set of "super-basic" variables that are free independent variables. In the geometric example shown, the optimal point is located on a constraint line. The slack variable associated with this constraint is at its bound and is a non-basic variable. There is a super-basic (or independent) variable which represents the freedom of the optimal point to be located anywhere along the constraint line. The remainder of the variables are basic (or dependent) because they are determined by the general constraints. In a multi-dimensional problem, the number of super-basic variables is a measure of the "non-linearity" of the problem.

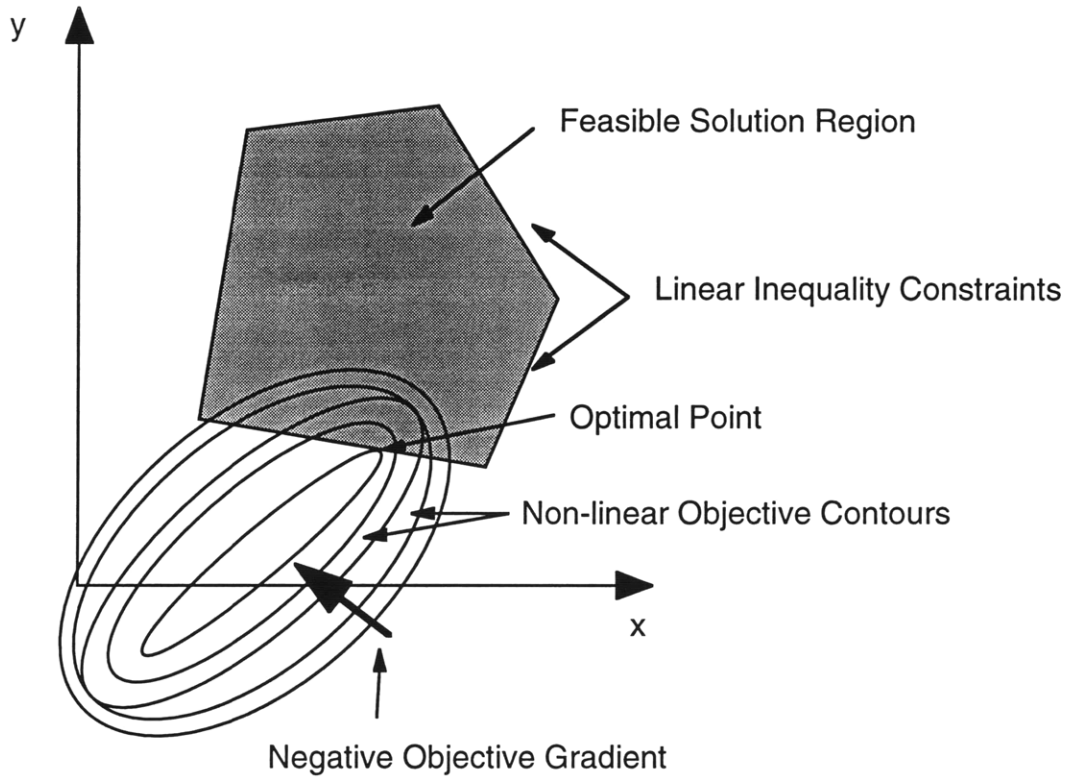


Figure 2.4-2: Graphical Interpretation of linearly constrained problem with nonlinear objective

One method for solving linearly constrained problems with nonlinear objective is the reduced gradient algorithm. In general, the algorithm chooses sets of basic, non-basic, and super-basic variables. The super-basic variables are varied in the direction of the negative objective gradient, and the basic variables are adjusted as necessary to satisfy the constraints. When no further progress can be made, some of the non-basic variables are added to the set of super-basic (independent) variables, allowing the algorithm to "search" for the solution in a different direction.

It is also important to note that the non-linearity of the problem introduces the possibility of locally optimal solutions. Currently the only realistic procedure available for determining if the solution is a global optimum is to vary the initial guess for the state trajectory and check to see that the programming software converges on the same solution.[12] Many software packages are available that attempt to solve nonlinear programming problems. The MINOS software package chosen for use in this study has

the ability to solve nonlinearly constrained problems with nonlinear objectives, and is discussed in the following section.

2.4.1 The MINOS Software

MINOS (Modular In-core Nonlinear Optimization System) is a FORTRAN-based software package produced by the Systems Optimization Laboratory at Stanford University. MINOS is designed to solve large optimization problems expressed in the following form:

$$\text{Minimize: } F(\vec{x}) + c^T\vec{x} + d^T\vec{y} \tag{2.5a}$$

subject to:

$$\vec{f}(\vec{x}) + A_1\vec{y} = \vec{b}_1 \tag{2.5b}$$

$$A_2\vec{x} + A_3\vec{y} = \vec{b}_2 \tag{2.5c}$$

$$\vec{l} \le \begin{pmatrix} \vec{x} \\ \vec{y} \end{pmatrix} \le \vec{u} \tag{2.5d}$$

where:

c, d, b_1, b_2, l, u, A_1, A_2, A_3 are constant

F(x) and f(x) are smooth

The n_1 components of x are the nonlinear variables, and the n_2 components of y are the linear variables. Similarly, the constraints may be categorized as linear or nonlinear.

The re-entry vehicle problem involves minimizing a nonlinear objective subject to nonlinear constraints. This problem is even more general than the linearly constrained non-linear objective problem discussed in the previous section. To solve this type of

problem, MINOS uses a method called an "augmented Lagrangian algorithm." The software performs a series of "major iterations," each of which solves a linearly constrained "sub-problem." The nonlinear constraints are linearized about the current estimate for each problem state:

$$\tilde{\vec{f}}(\vec{x},\vec{x}_k) = \vec{f}(\vec{x}_k) + J_k(\vec{x}_k)(\vec{x}-\vec{x}_k) \tag{2.6}$$

where $\vec{x}_k$ is the state estimate at the beginning of the kth major iteration, $\vec{f}(\vec{x}_k)$ is the constraint vector at the start of the major iteration, $\tilde{\vec{f}}(\vec{x},\vec{x}_k)$ is the estimate of the constraint vector linearized about $\vec{x}_k$, and J_k is the associated Jacobian matrix at the same point. The problem becomes:

$$\text{Minimize: } (F(x)+c^T\vec{x}+d^Ty) - \vec{\lambda}_k^T\left(f-\tilde{f}\right)+\frac{1}{2}\rho\left(f-\tilde{f}\right)^T\left(f-\tilde{f}\right) \tag{2.7a}$$

subject to:

$$\tilde{\vec{f}}(\vec{x}) + A_1\vec{y} = \vec{b}_1 \tag{2.7b}$$

$$A_2\vec{x} + A_3\vec{y} = \vec{b}_2 \tag{2.7c}$$

$$\vec{l} \le \begin{pmatrix}\vec{x}\\ \vec{y}\end{pmatrix} \le \vec{u} \tag{2.7d}$$

The new objective function is called the "augmented Lagrangian." The first term in this function is the original objective function. The second term is the Lagrangian term where λ_k is the estimate of the Lagrange multipliers for the current major iteration. The final term is called the "quadratic penalty function" where ρ is the "penalty parameter". This term penalizes the algorithm for using linearized estimates of the constraints that differ greatly from the real nonlinear constraints. The penalty parameter is set by the user, and raising the value of this parameter may aid in the convergence of highly nonlinear problems.[23]

This linearized sub-problem is solved using a reduced-gradient algorithm similar to the algorithm qualitatively discussed in the previous section.[23] The solution to the linearized problem is used as the starting point for the next major iteration. In addition to the penalty parameter, there are several other parameters that may be adjusted by the user to affect the software performance. The "major damping parameter" sets the maximum percent change in the problem state variables that are allowed between major iterations. This adds damping between major iterations and can help encourage convergence. The "minor iteration limit" sets the maximum number of steps of the reduced gradient algorithm that may be performed when solving the linearized sub-problems. In the interest of speed, it is not always desirable to solve the sub-problems exactly, and depending on the specific problem minor inaccuracies in the sub-problem solutions may have little effect. Increasing the minor iteration limit tends to increase the time required to complete each major iteration, but may reduce the number of major iterations required. Conversely, decreasing the limit reduces the time required to complete each major iteration, but if the limit is set too low more major iterations may be required to reach the desired solution. The penalty parameter, major damping parameter, and minor iterations limit are all determined experimentally for a given problem.

2.5 Chapter summary

This chapter provided background information in the theory and applications used in this thesis. The atmospheric re-entry problem and the optimal control problem were discussed. This was followed descriptions of direct solution methods and nonlinear programming methods. The MINOS computer software was introduced and described. The following chapter describes the necessary components of the system model.

Chapter 3: Vehicle and Environmental Models

The first step in solving for an atmospheric re-entry trajectory is to choose a system model. The re-entry problem requires a complex model to describe the vehicle behavior to the required degree of accuracy. Each component of the system must be modeled including the vehicle dynamics, atmospheric properties, motion of the atmosphere in the inertial reference frame, gravitational field of the Earth, and vehicle aerodynamics. This chapter describes each of these individual models.

3.1 Fixed Trim Re-entry Vehicle Model

We shall begin by formulating the mathematical model for an atmospheric re-entry vehicle. The motion of the vehicle center of mass is determined by the sum of the forces acting upon the vehicle. For this analysis, it is assumed that only aerodynamic and gravity forces are acting on the vehicle. The forces associated with the ACS jet firings, or any other forces that may exist, are assumed to be negligible. The procedure for writing the vehicle equations of motion is to write all of the vehicle forces in the same reference frame and apply Newton's second law:

$$\vec{F} = m\vec{a} \tag{3.1}$$

The full vehicle equations of motion have three translational and three rotational degrees of freedom. All translational degrees of freedom must be modeled to produce a meaningful control history. However, it is not necessary to model all of the rotational degrees of freedom. The three rotational degrees of freedom may be expressed as bank angle, angle of attack, and side-slip angle. The bank angle is previously defined in section 1.1. The angle of attack is defined as the angle between the longitudinal axis of the vehicle and the projection of the air-relative velocity vector in the vehicle roll-yaw plane. Similarly, the side-slip angle is defined as the angle between the longitudinal axis of the vehicle and the projection of the air-relative velocity vector in the vehicle roll-pitch plane. The angle of attack and side-slip angles are shown in figures 3.1-1 and 3.1-2.

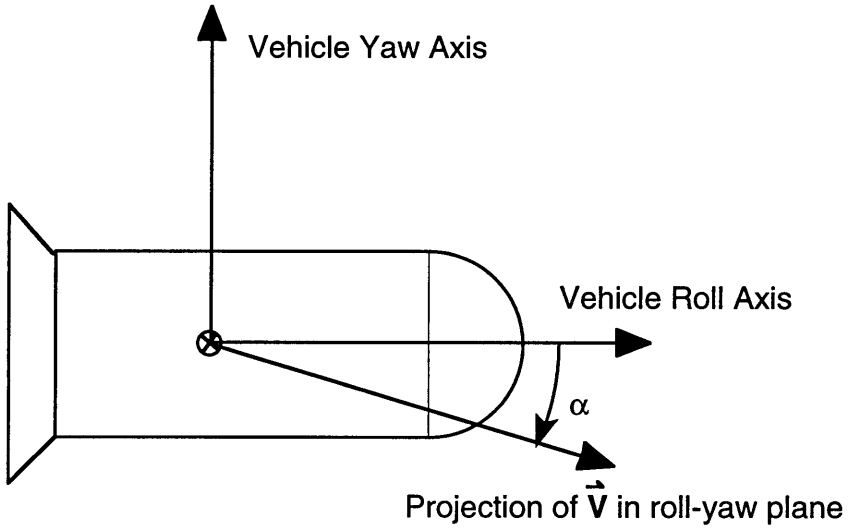


Figure 3.1-1: The re-entry vehicle angle of attack

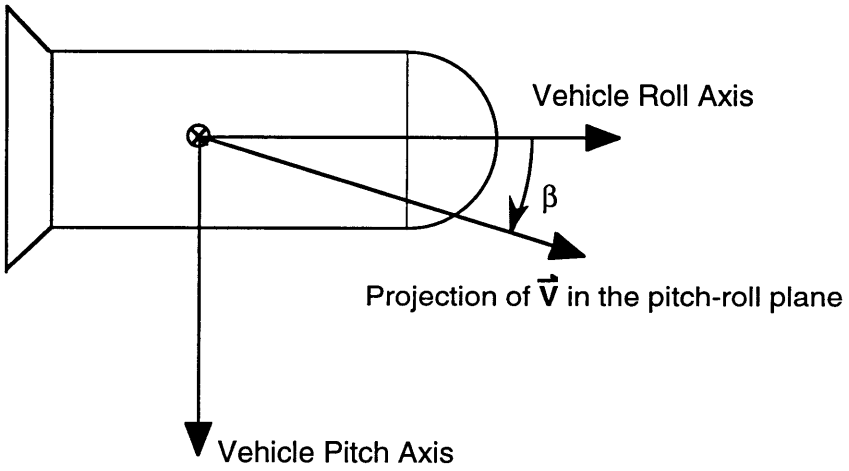


Figure 3.1-2: The re-entry vehicle side-slip angle

It is known from previous analysis of this vehicle that the angle of attack and side-slip behavior are both high frequency oscillations about trim values. Since by its nature the collocation solution only evaluates state values at discrete nodes, the resulting solution would not capture behavior occurring above a certain frequency. Analysis shows that the frequencies of the aerodynamic trim oscillations are several orders of magnitude above the maximum frequency that could be captured using a reasonable

number of nodes. Output from the IVS simulation indicates that the lower altitude aerodynamic frequencies are on the order of 1-2 seconds, while the collocation software is limited by memory constraints to approximately 30 nodes. For an average 700 second trajectory length, the smallest nodal spacing that could reasonably be achieved is on the order of 20 seconds. Since the true angle of attack and side-slip behavior of the re-entry vehicle can not be accurately captured using the collocation scheme, the vehicle is assumed to remain at a trim condition. The validity of this assumption is considered again in section 4.5.1. The corresponding lift and drag coefficients are also assumed to remain at their trim values. The assumptions made are :

$$\alpha = \alpha_{trim} \tag{3.2a}$$

$$\beta = \beta_{trim} \tag{3.2b}$$

Therefore:

$$C_l = C_{l_{trim}} \tag{3.2c}$$

$$C_d = C_{d_{trim}} \tag{3.2d}$$

Once the four degree of freedom vehicle model is chosen, the next step is to choose the vehicle states. The collocation procedure requires that the system be written as a series of coupled first order equations. Since the forces on the vehicle are proportional to the various accelerations, it is necessary to choose states to represent each degree of freedom and the associated rate. The bank rate is chosen as a control, and is treated separately. The vehicle states are chosen to be the bank angle (as defined in section 1.1) and the vehicle position and velocity expressed in the Earth Centered Inertial (ECI) reference frame:

$$\vec{x} = \begin{Bmatrix} x_{RV_{ECI}} \\ y_{RV_{ECI}} \\ z_{RV_{ECI}} \\ u^{ECI}_{RV_{ECI}} \\ v^{ECI}_{RV_{ECI}} \\ w^{ECI}_{RV_{ECI}} \\ \phi \end{Bmatrix} \tag{3.3}$$

The equations of motion will therefore be of the form:

$$\dot{\vec{x}} = \frac{d}{dt}\begin{Bmatrix} x_{RV_{ECI}} \\ y_{RV_{ECI}} \\ z_{RV_{ECI}} \\ u^{ECI}_{RV_{ECI}} \\ v^{ECI}_{RV_{ECI}} \\ w^{ECI}_{RV_{ECI}} \\ \phi \end{Bmatrix} = \vec{f}(\vec{x}) \tag{3.4}$$

The ECI reference frame is fixed with respect to distant stars and has its origin at the center of the Earth. The ECI x_{ECI}-axis is in the equatorial plane and aligned with the direction of the mean vernal equinox of date at the initial simulation time. The ECI z_{ECI}-axis passes through the north rotational pole of the Earth. The ECI y_{ECI}-axis is orthogonal to the x_{ECI} and z_{ECI}-axes.

In order to evaluate the equations of motion, the air-relative velocity vector must also be calculated. The air-relative velocity vector is the velocity vector of the air expressed in the inertial frame subtracted from the vehicle inertial velocity vector:

$$\vec{V}^{air}_{RV_{ECI}} = \vec{V}^{ECI}_{RV_{ECI}} - \vec{V}^{ECI}_{air_{ECI}} \tag{3.5}$$

The next step in deriving the vehicle equations of motion is to express the aerodynamic forces in the inertial reference frame. This requires introducing several additional reference frames and transformations. The procedure is similar to that outlined in Bladt.[5]

3.1.1 Necessary Reference Frames and Transformations

The first reference frame is chosen with its x-axis aligned with the air-relative velocity vector $\vec{V}_{RV_{ECI}}^{air}$ and the z-axis aligned with the lift vector $\vec{L}$. This is defined to be the "velocity-fixed" reference frame, and is shown in figure 3.1.1-1. Note that by definition the lift vector is always perpendicular to the air-relative velocity vector and the drag vector is always in the direction opposite the air-relative velocity vector. Also note that when the angle of attack and side-slip angle remain constant, the velocity-fixed reference frame remains fixed with respect to the vehicle.

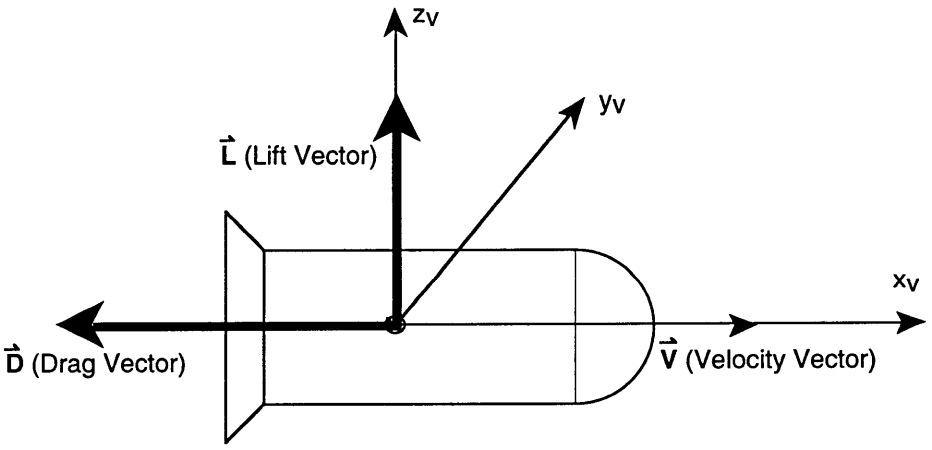


Figure 3.1.1-1: The velocity-fixed reference frame

The aerodynamic force vector acting on the vehicle is easily expressed in this reference frame:

$$\vec{F}_{aero_V} = \begin{Bmatrix} -D \\ 0 \\ L \end{Bmatrix} \tag{3.6}$$

The magnitudes of the aerodynamic forces are functions of the dynamic pressure Q and the vehicle lift and drag coefficients:

$$D = QSC_d \tag{3.7}$$

$$L = QSC_l \tag{3.8}$$

The dynamic pressure and aerodynamic coefficients are determined from the vehicle inertial position and velocity through the use of atmospheric and aerodynamic models, which are discussed in the following sections.

The goal is to express the aerodynamic forces in the inertial reference frame, and then add terms to account for the gravity force. This is accomplished by a series of rotations using direction cosine matrices (DCM). The first step is to consider a rotation about the air-relative velocity vector such that the resulting z-axis is in the plane of the velocity vector and the ECI z_{ECI}-axis. The necessary angle of rotation is called the "inertial bank angle," ϕ_i. The resulting reference frame is called the "vertical velocity-fixed" reference frame, and is shown in figure 3.1.1-2.

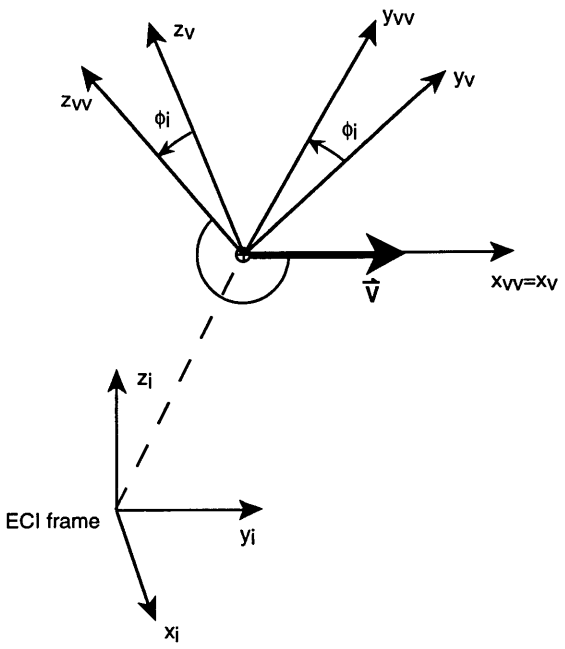


Figure 3.1.1-2: The vertical velocity-fixed reference frame

The direction cosine matrix for this transformation is:

$$\begin{Bmatrix} x_{RV_{vv}} \\ y_{RV_{vv}} \\ z_{RV_{vv}} \end{Bmatrix} = {}_{vv}T_{v} \begin{Bmatrix} x_{RV_{v}} \\ y_{RV_{v}} \\ z_{RV_{v}} \end{Bmatrix} \tag{3.9}$$

Where:

$${}_{vv}T_{v} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi_i & -\sin\phi_i \\ 0 & \sin\phi_i & \cos\phi_i \end{bmatrix} \tag{3.10}$$

The next transformation is to rotate about the y_{vv}-axis such that the resulting z-axis is aligned with the ECI z-axis. The necessary angle is called the "inertial flight path angle" γ_i and is shown in figure 3.1.1-3. The resulting reference frame is called the "level

velocity-fixed" reference frame since it moves with the velocity vector and its x-y plane remains parallel to the ECI x-y plane.

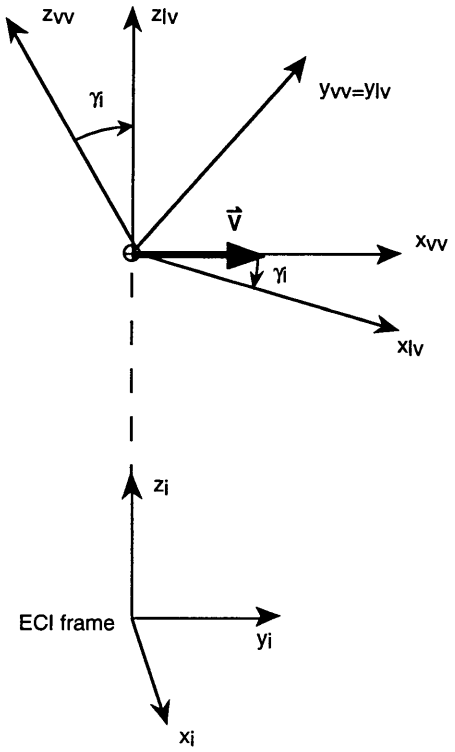


Figure 3.1.1-3: The level velocity-fixed reference frame

The corresponding DCM is:

$$\begin{Bmatrix} x_{RV_{lv}} \\ y_{RV_{lv}} \\ z_{RV_{lv}} \end{Bmatrix} = {}_{lv}T_{vv} \begin{Bmatrix} x_{RV_{vv}} \\ y_{RV_{vv}} \\ z_{RV_{vv}} \end{Bmatrix} \tag{3.11}$$

Where:

$${}_{lv}T_{vv} = \begin{bmatrix} \cos\gamma_i & 0 & -\sin\gamma_i \\ 0 & 1 & 0 \\ \sin\gamma_i & 0 & \cos\gamma_i \end{bmatrix} \tag{3.12}$$

The final rotation is about the z_{lv}-axis such that the resulting reference frame is completely aligned with the ECI frame. The necessary angle is called the "inertial azimuth angle" ψ_i: and is shown in figure 3.1.1-4.

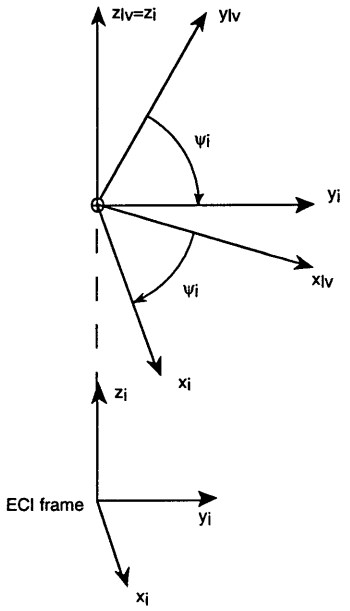


Figure 3.1.1-4: The Earth Centered Inertial (ECI) reference frame

The DCM for this transformation is:

$$\begin{Bmatrix} x_{RV_{ECI}} \\ y_{RV_{ECI}} \\ z_{RV_{ECI}} \end{Bmatrix} = {}_{ECI}T_{lv} \begin{Bmatrix} x_{RV_{lv}} \\ y_{RV_{lv}} \\ z_{RV_{lv}} \end{Bmatrix} \tag{3.13}$$

Where:

$${}_{ECI}T_{lv} = \begin{bmatrix} \cos\psi_i & -\sin\psi_i & 0 \\ \sin\psi_i & \cos\psi_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.14}$$

The aerodynamic forces acting on the vehicle may now be expressed in the ECI frame by combining these transformations:

$$\vec{F}_{aero_{ECI}} = {}_{ECI}T_v \begin{Bmatrix} -D \\ 0 \\ L \end{Bmatrix} \tag{3.15}$$

Where:

$${}_{ECI}T_v = {}_{ECI}T_{lv\,lv}T_{vv\,vv}T_v \tag{3.16}$$

This expression may be further simplified by writing the rotation angles in terms of the air-relative velocity. The inertial flight path is the angle between the air relative velocity vector and the ECI x-y plane. From inspection this is:

$$\sin\gamma_i = \frac{w^{air}_{RV_{ECI}}}{\left\|\vec{V}^{air}_{RV_{ECI}}\right\|} \Rightarrow \gamma_i = \sin^{-1}\left(\frac{w^{air}_{RV_{ECI}}}{\sqrt{(u^{air}_{RV_{ECI}})^2 + (v^{air}_{RV_{ECI}})^2 + (w^{air}_{RV_{ECI}})^2}}\right) \tag{3.17}$$

Similarly, the inertial azimuth angle is the angle between the ECI x-axis and the projection of the air relative velocity vector in the ECI x-z plane. This angle is given by:

$$\tan\psi_i = \frac{v^{air}_{RV_{ECI}}}{u^{air}_{RV_{ECI}}} \Rightarrow \psi_i = \tan^{-1}\left(\frac{v^{air}_{RV_{ECI}}}{u^{air}_{RV_{ECI}}}\right) \tag{3.18}$$

The inertial bank angle is the sum of the bank angle ϕ (as defined in section 1.1) and the angle ϕ_r between the plane of the velocity vector and the position vector and a plane parallel to the velocity vector and the z_{ECI}-axis :

$$\phi_i = \phi + \phi_r \tag{3.19}$$

This can be visualized more easily in the following special case. Consider the case when the velocity vector is aligned with the inertial y_{ECI}-axis. Figure 3.1.1-5 shows the inertial bank angle in a cross-section view of the vehicle (i.e. with the velocity vector "coming out of the paper").

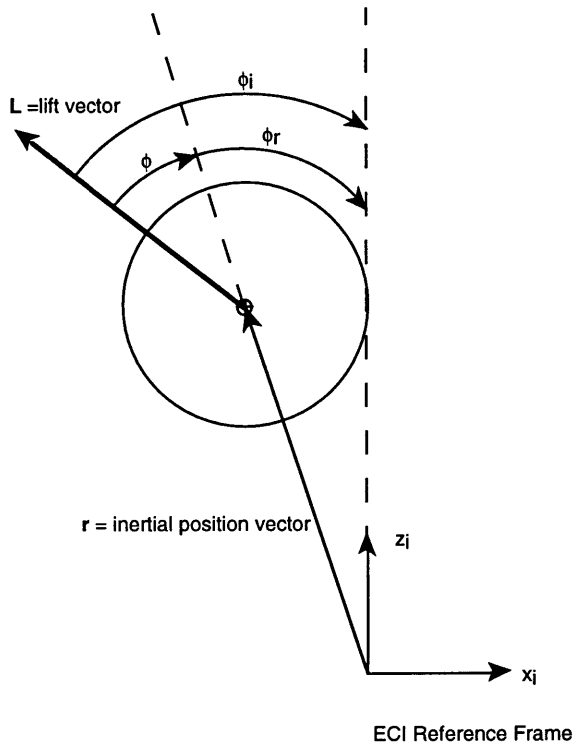


Figure 3.1.1-5: The inertial bank angle

The angle ϕ_r represents the component of ϕ_i due to the vehicle inertial position. ϕ_r may be determined by finding the angle between any vector normal to the first plane and any other vector normal to the second. A vector normal to the plane of the air-relative velocity vector and the position vector is:

$$\begin{Bmatrix} x_{RV_{ECI}} \\ y_{RV_{ECI}} \\ z_{RV_{ECI}} \end{Bmatrix} \times \begin{Bmatrix} u^{air}_{RV_{ECI}} \\ v^{air}_{RV_{ECI}} \\ w^{air}_{RV_{ECI}} \end{Bmatrix} = \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ x_{RV_{ECI}} & y_{RV_{ECI}} & z_{RV_{ECI}} \\ u^{air}_{RV_{ECI}} & v^{air}_{RV_{ECI}} & w^{air}_{RV_{ECI}} \end{vmatrix} = \begin{Bmatrix} y_{RV_{ECI}} w^{air}_{RV_{ECI}} - v^{air}_{RV_{ECI}} z_{RV_{ECI}} \\ z_{RV_{ECI}} v^{air}_{RV_{ECI}} - x_{RV_{ECI}} w^{air}_{RV_{ECI}} \\ x_{RV_{ECI}} v^{air}_{RV_{ECI}} - y_{RV_{ECI}} u^{air}_{RV_{ECI}} \end{Bmatrix} \quad (3.20)$$

A vector normal to the plane of the z_{ECI}-axis and the air-relative velocity vector is:

$$\begin{Bmatrix}0\\0\\1\end{Bmatrix}\times\begin{Bmatrix}u^{air}_{RV_{ECI}}\\v^{air}_{RV_{ECI}}\\w^{air}_{RV_{ECI}}\end{Bmatrix}=\begin{vmatrix}\hat{i}&\hat{j}&\hat{k}\\0&0&1\\u^{air}_{RV_{ECI}}&v^{air}_{RV_{ECI}}&w^{air}_{RV_{ECI}}\end{vmatrix}=\begin{Bmatrix}-v^{air}_{RV_{ECI}}\\u^{air}_{RV_{ECI}}\\0\end{Bmatrix} \tag{3.21}$$

The angle between these two vectors is:

$$\cos\phi_r=\frac{\left(\begin{Bmatrix}0\\0\\1\end{Bmatrix}\times\vec{V}^{air}_{RV_{ECI}}\right)\cdot\left(\vec{r}_{RV_{ECI}}\times\vec{V}^{air}_{RV_{ECI}}\right)}{\left\|\begin{Bmatrix}0\\0\\1\end{Bmatrix}\times\vec{V}^{air}_{RV_{ECI}}\right\|\left\|\vec{r}_{RV_{ECI}}\times\vec{V}^{air}_{RV_{ECI}}\right\|} \tag{3.22}$$

The gravitational acceleration experienced by the vehicle is calculated by the gravitational model and is returned already decomposed into its ECI components. Therefore the total force acting on the vehicle is:

$$\vec{F}_{total_{ECI}}=\vec{F}_{aero_{ECI}}+m\vec{g}_{ECI} \tag{3.23}$$

Substituting and applying Newton's second law yields:

$$\dot{\bar{x}}=\frac{d}{dt}\begin{Bmatrix}x_{RV_{ECI}}\\y_{RV_{ECI}}\\z_{RV_{ECI}}\\u^{ECI}_{RV_{ECI}}\\v^{ECI}_{RV_{ECI}}\\w^{ECI}_{RV_{ECI}}\\\dot{\phi}\end{Bmatrix}=\begin{Bmatrix}u^{ECI}_{RV_{ECI}}\\v^{ECI}_{RV_{ECI}}\\w^{ECI}_{RV_{ECI}}\\F_{Total_{ECI_x}}/m\\F_{Total_{ECI_y}}/m\\F_{Total_{ECI_z}}/m\\\dot{\phi}\end{Bmatrix} \tag{3.24}$$

$$= \begin{Bmatrix} u_i \\ v_i \\ w_i \\ -D/m \cos\gamma_i \cos\psi_i + L/m \cos(\phi + \phi_r) \sin\gamma_i \cos\psi_i + L/m \sin(\phi + \phi_r) \sin\psi_i + g_{ECI_x} \\ -D/m \cos\gamma_i \sin\psi_i + L/m \cos(\phi + \phi_r) \sin\gamma_i \sin\psi_i - L/m \sin(\phi + \phi_r) \cos\psi_i + g_{ECI_y} \\ D/m \sin\psi_i + L/m \cos(\phi + \phi_r) \cos\gamma_i + g_{ECI_z} \\ \dot{\phi} \end{Bmatrix} \quad (3.25)$$

(Where the necessary angles are defined above)

This is a system of first-order, coupled, nonlinear equations which describes the state derivative in terms of the current state and the control variable. The next step is to determine the magnitude of the aerodynamic forces using environmental models.

3.2 Atmospheric Model

An atmospheric model is needed to estimate the local air density and speed of sound as a function of inertial position. The local air density is directly proportional to the dynamic pressure experienced by the re-entry vehicle, and therefore to the magnitude of the resulting aerodynamic forces:

$$Q = \frac{1}{2} \rho \left\| \vec{V}_{RV_{ECI}}^{air} \right\|^2 S_{ref} \quad (3.26)$$

The local speed of sound a is necessary to determine the Mach number of the vehicle:

$$M = \frac{\left\| \vec{V}_{RV_{ECI}}^{air} \right\|}{a} \quad (3.27)$$

The Mach number of the vehicle must be calculated to estimate the vehicle aerodynamic parameters.

The United States Standard Atmosphere of 1976 (US76) was chosen as the atmospheric model. The code structure used to perform the necessary calculations was adapted directly from the Kistler IVS simulation. The code first uses the vehicle ECI position to calculate the altitude about an ellipsoidal approximation of the Earth. Note that since the ellipsoid is symmetric about the rotational axis of the Earth, the model does not account for any longitudinal variations, and the rotation of the Earth about its pole may be neglected when using the US76 model. The resulting density and speed of sound profiles are shown in figures 3.2-1 and 3.2-2.

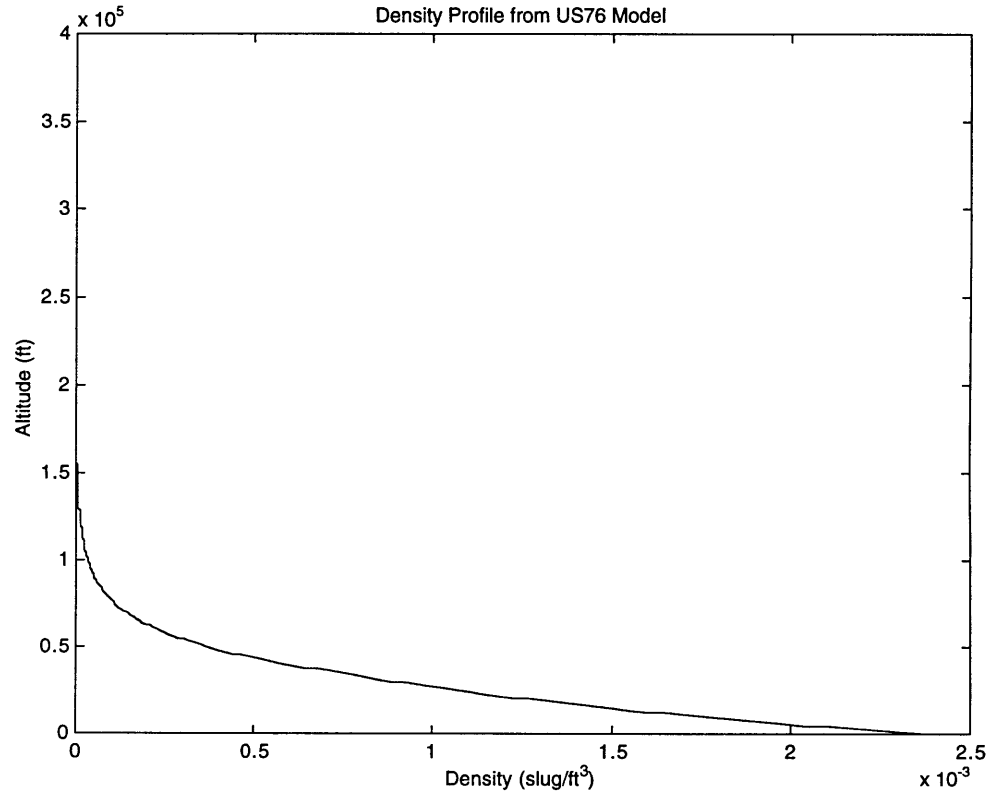


Figure3.2-1: US76 Atmospheric density profile

Other atmospheric models could easily be substituted into the collocation software system model. The only requirement is that the atmospheric model return the current atmospheric density and speed of sound given the current ECI position. Another such model, used in the development of the Kistler launch system, is the GRAM95 model. This model accounts for latitudinal, longitudinal, and seasonal variations in the

atmosphere. The K-1 vehicle system also has the ability to accept updated atmospheric data in flight. Any of these methods of generating density and speed of sound profiles would be appropriate for use with the collocation software developed in this thesis. The US76 model was chosen for this study because of its relative simplicity.

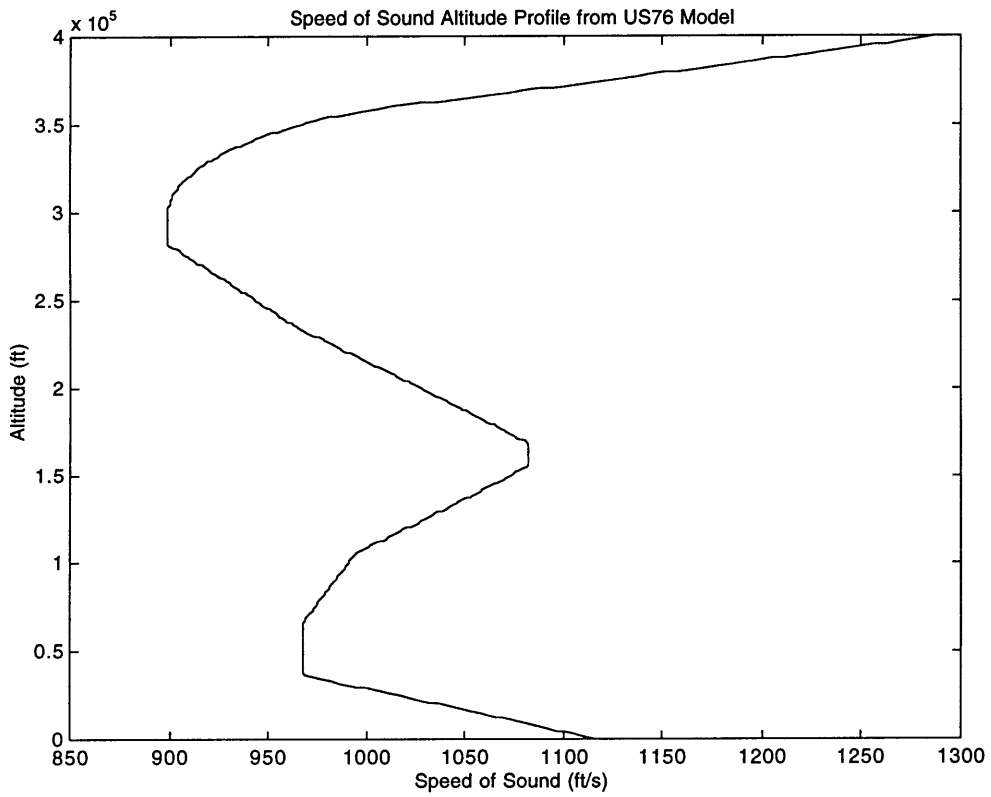


Figure3.2-1: US76 Atmospheric speed of sound profile

3.3 Motion of the Atmosphere in the ECI Frame

In addition to the static properties of the Earth's atmosphere, the motion of the air has a significant impact on the motion of a re-entry vehicle. The magnitude of the aerodynamic forces acting on a flight vehicle are directly proportional to the dynamic pressure experience by the vehicle, which is in turn proportional to the square of the air-

relative velocity of the vehicle. It is therefore necessary to model the motion of the Earth's atmosphere in the inertial reference frame.

Motion of the Earth's atmosphere in the inertial reference frame is caused by two distinct sources. First, the atmosphere is moving with the surface of the Earth in the inertial frame due to the Earth's rotation. This motion is equivalent to a rotation about the polar axis of the Earth at the angular frequency of the Earth's rotation. This rotational motion is much more significant to the re-entry problem than wind and is considered first. The air is also moving with respect to the local surface of the Earth due to wind caused by local variations in air pressure. The collocation software does not contain a wind prediction model, but the software does provide the opportunity to add the wind velocity to the air relative velocity vector if the actual wind profile is known.

Neglecting wind, the atmosphere is assumed to rotate about the rotational pole of the Earth at the same frequency as the Earth itself, as shown in figure 3.3-1. The angular velocity of the Earth may be found from its period:

$$\omega_e = \frac{2\pi}{86164.10s} = 7.292115 \cdot 10^{-5} \, rad/s \tag{3.28}$$

The velocity of the atmosphere with respect to the ECI frame at the re-entry vehicle location is:

$$\vec{V}_{air_{ECI}}^{ECI} = \begin{Bmatrix} 0 \\ 0 \\ \omega_e \end{Bmatrix} \times \vec{r}_{RV_{ECI}} \tag{3.29}$$

The velocity of the vehicle with respect to the air may now be calculated as described in section 3.1, and the magnitude of this velocity may be used to calculate the correct dynamic pressure and Mach number of the vehicle. The magnitude of the motion of the atmosphere in the inertial reference frame is approximately $O(1000 \, ft/s)$.

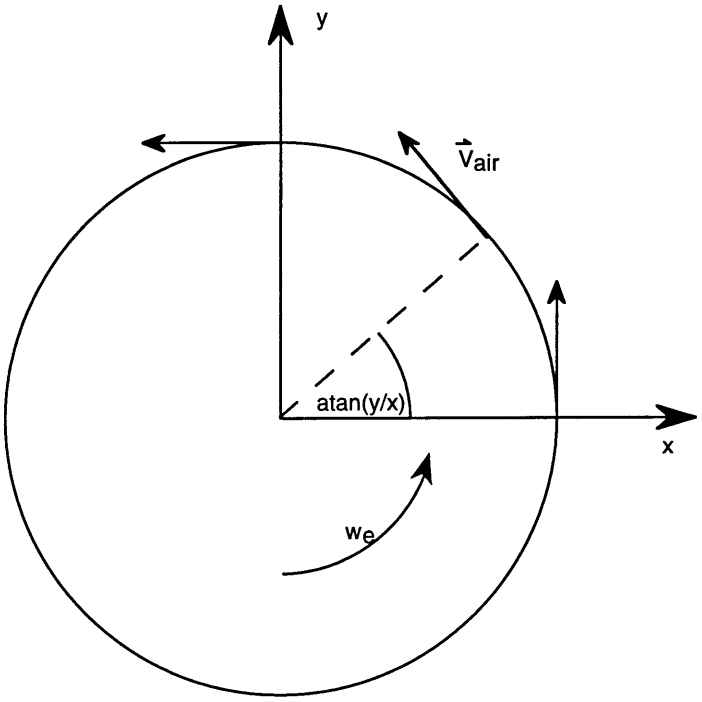


Figure 3.3-1: Velocity of the air due to Earth's rotation

3.4 Gravitational Model

The effects of the Earth's gravity on the re-entry vehicle are significant. It is therefore necessary to accurately model the Earth's gravitational field. The same model is used for the collocation software that was used in the IVS simulation. One approximation of the gravitational potential of the Earth at the re-entry vehicle position is[4]:

$$\Phi = \frac{\mu}{\left\|\vec{r}_{RV_{ECI}}\right\|}\left(\begin{array}{l} 1-\frac{J_2}{2}\left(\frac{r_e}{\left\|\vec{r}_{RV_{ECI}}\right\|}\right)^2(3\sin^2 L-1)-\frac{J_3}{2}\left(\frac{r_e}{\left\|\vec{r}_{RV_{ECI}}\right\|}\right)^3(5\sin^3 L-3\sin L) \\ -\frac{J_4}{8}\left(\frac{r_e}{\left\|\vec{r}_{RV_{ECI}}\right\|}\right)^4(35\sin^4 L-30\sin^2 L+3)+H.O.T \end{array}\right) \tag{3.30}$$

where L is the geocentric latitude:

$$L = \sin^{-1}\left(\frac{z_{RV_{ECI}}}{\left\|\vec{r}_{RV_{ECI}}\right\|}\right) \tag{3.31}$$

and J_n are coefficients determined by experimental observation. Following the derivation described in Bate, the gravitational acceleration is the gradient of the potential function:

$$\vec{g}_{ECI} = \nabla\Phi = \begin{Bmatrix} \frac{\delta\phi}{\delta x_{RV_{ECI}}} \\ \frac{\delta\phi}{\delta y_{RV_{ECI}}} \\ \frac{\delta\phi}{\delta z_{RV_{ECI}}} \end{Bmatrix} \tag{3.32}$$

Differentiating:

$$g_{ECI_x} = \frac{\delta\Phi}{\delta x} = -\frac{\mu x_{RV_{ECI}}}{\left\|\vec{r}_{RV_{ECI}}\right\|^3}\left(\begin{array}{l} 1 - J_2\frac{3}{2}\left(\frac{r_e}{\left\|\vec{r}_{RV_{ECI}}\right\|}\right)^2\left(5\frac{z_{RV_{ECI}}^{\ 2}}{\left\|\vec{r}_{RV_{ECI}}\right\|^2} - 1\right) + J_3\frac{5}{2}\left(\frac{r_e}{\left\|\vec{r}_{RV_{ECI}}\right\|}\right)^3\left(3\frac{z_{RV_{ECI}}}{\left\|\vec{r}_{RV_{ECI}}\right\|} - 7\frac{z_{RV_{ECI}}^{\ 3}}{\left\|\vec{r}_{RV_{ECI}}\right\|^3}\right) \\ -J_4\frac{5}{8}\left(\frac{r_e}{\left\|\vec{r}_{RV_{ECI}}\right\|}\right)^4\left(3 - 42\frac{z_{RV_{ECI}}^{\ 2}}{\left\|\vec{r}_{RV_{ECI}}\right\|^2} + 63\frac{z_{RV_{ECI}}^{\ 4}}{\left\|\vec{r}_{RV_{ECI}}\right\|^4}\right) + H.O.T. \end{array}\right) \tag{3.33}$$

$$g_{ECI_y} = \frac{\delta\Phi}{\delta y} = \frac{y_{RV_{ECI}}}{x_{RV_{ECI}}} g_{ECI_x} \tag{3.34}$$

$$g_{ECI_z} = \frac{\delta\Phi}{\phi z} = -\frac{\mu z_{RV_{ECI}}}{\left\|\vec{r}_{RV_{ECI}}\right\|^3}\left(\begin{array}{l} 1 + J_2\frac{3}{2}\left(\frac{r_e}{\left\|\vec{r}_{RV_{ECI}}\right\|}\right)^2\left(3 - 5\frac{z_{RV_{ECI}}^{\ 2}}{\left\|\vec{r}_{RV_{ECI}}\right\|^2}\right) + J_3\frac{3}{2}\left(\frac{r_e}{\left\|\vec{r}_{RV_{ECI}}\right\|}\right)^3\left(10\frac{z_{RV_{ECI}}}{\left\|\vec{r}_{RV_{ECI}}\right\|}\right. \\ \left. -\frac{35}{3}\frac{z_{RV_{ECI}}^{\ 3}}{\left\|\vec{r}_{RV_{ECI}}\right\|^3} - \frac{r}{z}\right) - J_4\frac{5}{8}\left(\frac{r_e}{\left\|\vec{r}_{RV_{ECI}}\right\|}\right)^4\left(15 - 70\frac{z_{RV_{ECI}}^{\ 2}}{\left\|\vec{r}_{RV_{ECI}}\right\|^2} + 63\frac{z_{RV_{ECI}}^{\ 4}}{\left\|\vec{r}_{RV_{ECI}}\right\|^4}\right) + \ldots \end{array}\right) \tag{3.35}$$

Note that these equations only include the zonal harmonics. Therefore the magnitude of gravitational acceleration is independent of longitude. This means that the rotation of the Earth about its pole may be neglected when gravitational acceleration is calculated.

3.5 Aerodynamic Model

The aerodynamic model is unique to each specific re-entry vehicle. Given the current vehicle flight conditions, it is necessary to compute the magnitude of the aerodynamic forces acting on the vehicle. As previously described, the aerodynamic forces are expressed as lift and drag in the vehicle velocity-fixed reference frame. Aerodynamic tables are provided for the K-1 launch system which list the lift, drag, and moment coefficients as functions of Mach number and total angle-of-attack α^*. The total angle of attack is defined as the angle between the vehicle longitudinal axis and the air relative velocity vector. Assuming the trim angles are small, the total angle of attack is approximately:

$$\alpha^* \approx \sqrt{\alpha^2 + \beta^2} \tag{3.36}$$

This study assumes the vehicle is constantly at trim, so a smaller aerodynamic table may be constructed of trim aerodynamic coefficients versus Mach number. For each Mach number entry in the full aerodynamic table, linear interpolation is used to estimate the lift and drag coefficient corresponding to a zero moment coefficient about the nominal vehicle center of gravity. The resulting trim aerodynamic values are shown in figures 3.5-1 and 3.5-2. Note that if the vehicle center of gravity location were to deviate from nominal the moment about the c.g. caused by the aerodynamic forces would change. Therefore the trim conditions would be altered, and the full aerodynamic tables would be needed to calculate the new vehicle trim condition.

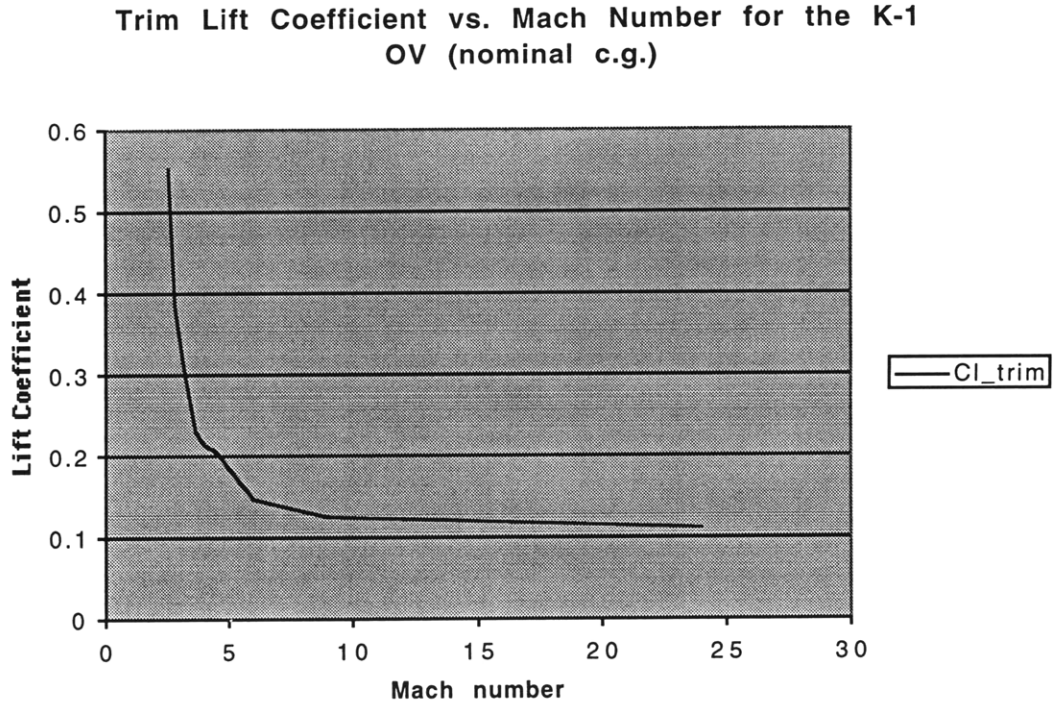


Figure 3.5-1: Trim Cl values for the K-1 OV

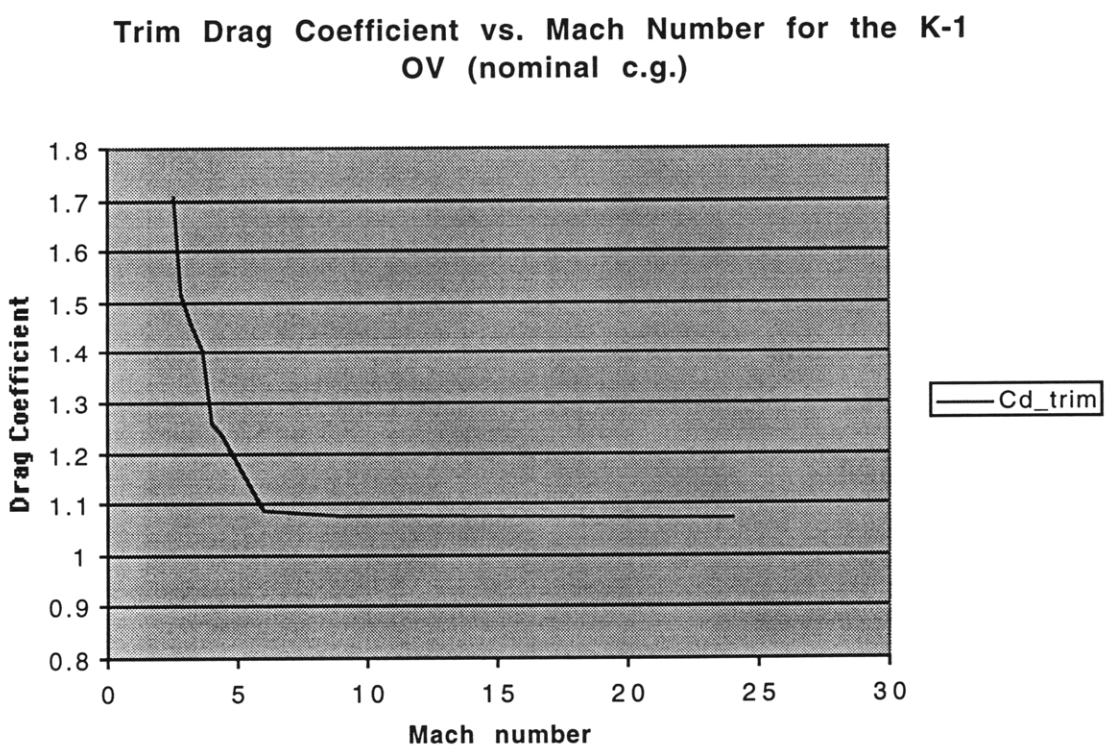


Figure 3.5-2: Trim Cd values for the K-1 OV

3.6 Chapter Summary

This chapter presented the various components of the system model used to describe the atmospheric re-entry problem. The necessary coordinate frames and transformations were introduced and the re-entry vehicle equations of motion were derived. The atmospheric and gravity models used in the collocation software were described. The motion of the air with respect to the inertial reference frame was calculated. The method for estimating the vehicle aerodynamic parameters was also described. The following chapter outlines the design and initial validation of the collocation software.

Chapter 4: Collocation Software Design and Validation

This chapter describes the design and validation of the collocation software. First, the specific collocation scheme used in the software is derived. Then the nonlinear program constraints and objective are defined. The code structure is described, and the software is validated by comparing predicted trajectories for predetermined bank profiles to trajectories generated by the IVS simulation. Finally, the implementation of the collocation software in the IVS simulation is discussed.

4.1 The Collocation Scheme

Direct Collocation with Non-Linear Programming (DCNLP) is a method of converting an optimal control problem into a parameter optimization problem. The parameter optimization problem is expressed as a series of nonlinear objectives and constraints. The resulting parameter optimization problem is then solved with available nonlinear programming software.

As described in chapter 2, the initial discretization is performed by representing the vehicle state and controls at a finite number N nodes along the vehicle trajectory. Each node corresponds to a unique time, t. The nodes do not have to be equally spaced, although for simplicity they will be equally spaced in this implementation. The time interval between the i^{th} pair of nodes is Δt_i. The nodes are connected by a number of trajectory segments, as shown in figure 4.1-1.

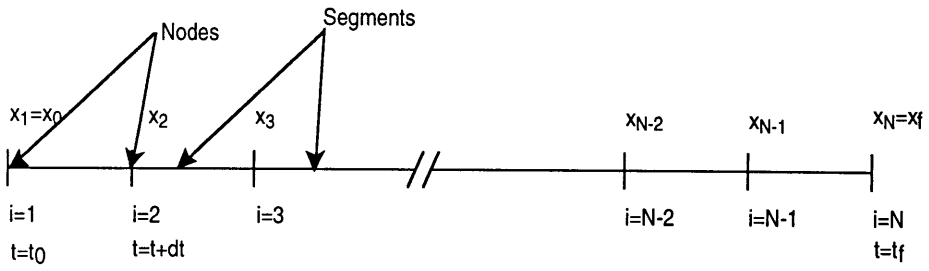


Figure 4.1-1: Nodes and trajectory segments

The vehicle state at the i[th] node is represented as $\vec{x}_i$, and the control is $\vec{u}_i$. Since the current problem formulation only has one control variable (bank rate), the rest of the analysis will treat u_i as a scalar. The state derivative at each node is a function of the current state and control, and is determined by the vehicle equations of motion described in chapter 3:

$$\dot{\vec{x}}_i = \vec{f}(\vec{x}_i, u_i) \tag{4.1}$$

Note that the vehicle equations of motion are expressed as a system of coupled, first order, non-linear equations. The number of equations is equal to the number of vehicle states.

The first step in the DCNLP process is to estimate the values of the states and controls along the trajectory segments. The vehicle states are estimated by a series of cubic polynomials determined by the state values and state derivative values at the neighboring nodes. These are Hermite polynomials because they depend only on the states and their first derivatives at the endpoint of each segment. There will be a separate polynomial for each vehicle state between each pair of consecutive nodes. Fitting a polynomial to any trajectory segment:

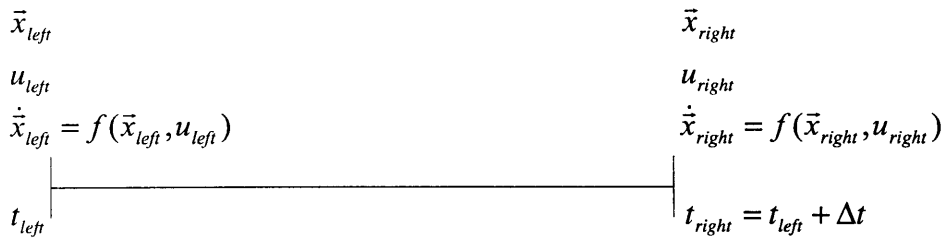


Figure 4.1-2: A trajectory segment

The general form for each polynomial is:

$$\vec{x}_H(t) = \vec{a}(t - t_{left})^3 + \vec{b}(t - t_{left})^2 + \vec{c}(t - t_{left}) + \vec{e} \tag{4.2}$$

where:

$$t_{left} \le t \le t_{right}$$

Differentiating:

$$\dot{\vec{x}}_H = 3\vec{a}(t - t_{left})^2 + 2\vec{b}(t - t_{left}) + \vec{c} \tag{4.3}$$

Matching the boundary conditions:

At the left node:

$$\vec{x}_{left} = \vec{x}_H(t_{left}) = \vec{e} \tag{4.4}$$

$$\vec{e} = \vec{x}_{left} \tag{4.5}$$

$$\dot{\vec{x}}_{left} = \dot{\vec{x}}_H(t_{left}) = \vec{c} \tag{4.6}$$

$$\vec{c} = \dot{\vec{x}}_{left} \tag{4.7}$$

At the right node:

$$\vec{x}_{right} = \vec{a}\Delta t^3 + \vec{b}\Delta t^2 + \dot{\vec{x}}_{left} + \vec{x}_{left} \tag{4.8}$$

$$\dot{\vec{x}}_{right} = 3\vec{a}\Delta t^2 + 2\vec{b}\Delta t + \dot{\vec{x}}_{left} \tag{4.9}$$

Rearranging and solving for $\vec{a}$ and $\vec{b}$:

$$\vec{a} = \frac{2(\vec{x}_{left} - \vec{x}_{right})}{\Delta t^3} + \frac{(\dot{\vec{x}}_{left} + \dot{\vec{x}}_{right})}{\Delta t^2} \tag{4.10}$$

$$\vec{b} = \frac{3(\vec{x}_{right} - \vec{x}_{left})}{\Delta t^2} + \frac{(\dot{\vec{x}}_{right} - \dot{\vec{x}}_{left}) - 3(\dot{\vec{x}}_{left} + \dot{\vec{x}}_{right})}{2\Delta t} \tag{4.11}$$

Therefore the Hermite polynomial fit to the trajectory segment is :

$$\vec{x}_H(t) = [\frac{2(\vec{x}_{left} - \vec{x}_{righ})}{\Delta t^3} + \frac{(\dot{\vec{x}}_{left} + \dot{\vec{x}}_{right})}{\Delta t^2}](t - t_{left})^3 + [\frac{(\dot{\vec{x}}_{right} - \dot{\vec{x}}_{left})}{2\Delta t} + \frac{3(\vec{x}_{right} - \vec{x}_{left})}{\Delta t^2} - \frac{3(\dot{\vec{x}}_{left} + \dot{\vec{x}}_{right})}{2\Delta t}](t - t_{left})^2 + \dot{\vec{x}}_{left}(t - t_{left}) + \vec{x}_{left} \tag{4.12}$$

$$where:$$

$$t_{left} \le t \le t_{left} + \Delta t$$

The first derivative of the Hermite polynomial approximation is:

$$\dot{\vec{x}}_H(t) = 3[\frac{2(\vec{x}_{left} - \vec{x}_{righ})}{\Delta t^3} + \frac{(\dot{\vec{x}}_{left} + \dot{\vec{x}}_{right})}{\Delta t^2}](t - t_{left})^2 + 2[\frac{(\dot{\vec{x}}_{right} - \dot{\vec{x}}_{left})}{2\Delta t} + \frac{3(\vec{x}_{right} - \vec{x}_{left})}{\Delta t^2} - \frac{3(\dot{\vec{x}}_{left} + \dot{\vec{x}}_{right})}{2\Delta t}](t - t_{left}) + \dot{\vec{x}}_{left} \tag{4.13}$$

The control variable is assumed to vary linearly between consecutive nodes. The value of u along each segment is estimated using linear interpolation:

$$u(t) = u_{left} + (u_{right} - u_{left})\frac{(t - t_{left})}{\Delta t} \tag{4.14}$$

where:

$$t_{left} \le t \le t_{left} + \Delta t$$

The vehicle equations of motion are enforced at collocation points set at the center of each trajectory segment. Therefore it is necessary to estimate the values of the vehicle state and control at the center of each segment. Evaluating the Hermite cubic and its derivative at the center of a segment yields:

$$\vec{x}_{H_C} = \frac{(\vec{x}_{left} + \vec{x}_{right})}{2} + \frac{\Delta t(\dot{\vec{x}}_{left} - \dot{\vec{x}}_{right})}{8} \tag{4.15}$$

$$\dot{\vec{x}}_{H_c} = \frac{3(\vec{x}_{right} - \vec{x}_{left})}{2\Delta t} - \frac{(\dot{\vec{x}}_{left} + \dot{\vec{x}}_{right})}{4} \tag{4.16}$$

The estimated control value at the center of each segment is simply:

$$u_c = \frac{u_{left} + u_{right}}{2} \tag{4.17}$$

The vehicle equations of motion give the state derivative in terms of the current vehicle state. If the Hermite polynomials are good approximation to the states, then the derivative of the polynomial should closely match the state derivative predicted by the equations of motion at the same point. For each element of the vehicle state, the value of the derivative of the Hermite cubic is compared to the derivative value obtained from evaluating the equations of motion. The result is called the defect. The defects associated with each vehicle state are assembled into the defect vector:

$$\vec{d} = \dot{\vec{x}}_{EOM} - \dot{\vec{x}}_{H_c} \tag{4.18}$$

Where:

$$\dot{\vec{x}}_{EOM} = f(\vec{x}_{H_c}, u_C) \tag{4.19}$$

Substituting from above:

$$\vec{d} = f(\frac{(\vec{x}_{left} + \vec{x}_{right})}{2} + \frac{\Delta t(\dot{\vec{x}}_{left} - \dot{\vec{x}}_{right})}{8}, \frac{u_{left} + u_{right}}{2}) + \frac{3(\vec{x}_{left} - \vec{x}_{right})}{2\Delta t} + \frac{(\dot{\vec{x}}_{left} + \dot{\vec{x}}_{right})}{4} \tag{4.20}$$

Each collocation point has an associated defect vector, and each defect vector has an element for each vehicle state. When all of the elements of all of the defect vectors are zero, the trajectory estimate satisfies the vehicle equations of motion at all of the collocation points. It is assumed that if the equations of motion are satisfied at the collocation points, they are approximately satisfied elsewhere. Once the discretized equations of motion are satisfied, additional constraints may be applied to the re-entry

vehicle problem as necessary. All of these constraints and the objective are then input into the MINOS software and an optimal control history is calculated.

4.2 The Non-Linear Program Constraints

As described in section 2.3, a nonlinear program is specified as a finite number of constraints and a scalar objective to be minimized. The nonlinear program variables consist of every vehicle state and control at each trajectory node and the final trajectory time. This vector is sometimes referred to as the NLP state:

$$\vec{X} = \begin{Bmatrix} \vec{x}_1 \\ \vec{x}_2 \\ \vdots \\ \vec{x}_N \\ \phi_1 \\ \phi_2 \\ \vdots \\ \phi_N \\ t_f \end{Bmatrix} \tag{4.21}$$

The objective and the constraints currently applied to this problem are all nonlinear in the most general case. The nonlinear constraints are input into the MINOS software as one vector containing all of the compiled constraints. The software also requires the Jacobian matrix, which is the matrix containing the partial derivatives of all of the constraint vector elements with respect to all of the NLP variables. The specific nonlinear constraints associated with each aspect of the re-entry vehicle problem are described in the following sections. One advantage to the collocation approach is that additional problem constraints may be easily added.

4.2.1 The Defect Vector Constraint

The largest portion of the constraint matrix is the defect vector constraint. For the equations of motions to be correctly enforced at each collocation point, every element of each defect vector constructed in section 4.1 must be successfully constrained to zero:

$$\vec{d}_i = \vec{0} \tag{4.22}$$

The number of nonlinear constraints imposed by the defect vector constraint is:

$$m_{defect} = (N-1) \cdot length(\vec{x}) \tag{4.23}$$

In order to enforce these constraints, the optimization software must be provided with a subroutine to calculate every defect vector as a function of the current variable values. It is also necessary to calculate the gradient of each component of every defect vector with respect to each NLP variable. The compilation of all of these gradient vectors yields a matrix with rows corresponding to each element in each defect vector and columns corresponding to each problem state. Note from section 4.1 that the value of each defect vector depends only on the vehicle state and control values at the neighboring nodes and the time duration between nodes. Most of the non-zero entries in the constraint Jacobian are associated with the defect vector constraint. The portion of the total constraint Jacobian resulting from the defect vector is a banded block matrix:

$$\bar{J}_{defect} = \begin{bmatrix} \bar{F}(\vec{d}_1,\vec{x}_1) & \bar{F}(\vec{d}_1,\vec{x}_2) & \bar{0} & \cdots & \cdots & \cdots & \bar{0} \\ \bar{0} & \bar{F}(\vec{d}_2,\vec{x}_2) & \bar{F}(\vec{d}_2,\vec{x}_3) & \bar{0} & \cdots & \cdots & \vdots \\ \bar{0} & \bar{0} & \bar{F}(\vec{d}_3,\vec{x}_3) & \bar{F}(\vec{d}_3,\vec{x}_4) & \cdots & \cdots & \vdots \\ \bar{0} & \bar{0} & \bar{0} & \ddots & \ddots & \bar{0} & \vdots \\ \vdots & \vdots & \vdots & \vdots & \bar{F}(\vec{d}_{N-2},\vec{x}_{N-2}) & \bar{F}(\vec{d}_{N-2},\vec{x}_{N-1}) & \bar{0} \\ \bar{0} & \bar{0} & \cdots & \bar{0} & \bar{0} & \bar{F}(\vec{d}_{N-1},\vec{x}_{N-1}) & \bar{F}(\vec{d}_{N-1},\vec{x}_N) \end{bmatrix} \tag{4.24}$$

where:

$$\overline{F}(\vec{d}_i,\vec{x}_j)=\begin{bmatrix} \frac{\partial d_{i_1}}{\partial x_{j_1}} & \frac{\partial d_{i_1}}{\partial x_{j_2}} & \cdots & \frac{\partial d_{i_1}}{\partial x_{j_6}} & \frac{\partial d_{i_1}}{\partial x_{j_7}} \\ \frac{\partial d_{i_2}}{\partial x_{j_1}} & \frac{\partial d_{i_2}}{\partial x_{j_2}} & \cdots & \frac{\partial d_{i_2}}{\partial x_{j_6}} & \frac{\partial d_{i_2}}{\partial x_{j_7}} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \frac{\partial d_{i_6}}{\partial x_{j_1}} & \frac{\partial d_{i_6}}{\partial x_{j_2}} & \cdots & \frac{\partial d_{i_6}}{\partial x_{j_6}} & \frac{\partial d_{i_6}}{\partial x_{j_7}} \\ \frac{\partial d_{i_7}}{\partial x_{j_1}} & \frac{\partial d_{i_7}}{\partial x_{j_2}} & \cdots & \frac{\partial d_{i_{71}}}{\partial x_{j_6}} & \frac{\partial d_{i_7}}{\partial x_{j_7}} \end{bmatrix} \tag{4.25}$$

assuming:

$$\vec{d}_i=\begin{Bmatrix} d_{i_1} \\ d_{i_2} \\ d_{i_3} \\ d_{i_4} \\ d_{i_5} \\ d_{i_6} \\ d_{i_7} \end{Bmatrix} \quad and \quad \vec{x}_j=\begin{Bmatrix} x_{i_1} \\ x_{i_2} \\ x_{i_3} \\ x_{i_4} \\ x_{i_5} \\ x_{i_6} \\ x_{i_7} \end{Bmatrix} \tag{4.26}$$

Since the defect vectors also depend on the time step between nodes, there will also be non-zero entries in the constraint Jacobian corresponding to the partial derivative of each defect vector with respect to the final trajectory time:

$$\frac{\partial d_{i_j}}{\partial t_f}=\frac{\partial d_{i_j}}{\partial \Delta t}\frac{\partial \Delta t}{\partial t_f}=\frac{1}{N-1}\frac{\partial d_{i_j}}{\partial \Delta t} \tag{4.27}$$

The total number of non-zero elements p in the constraint Jacobian associated with the defect constraint is:

$$p_{defects}=2\cdot length(\vec{x})\cdot length(\vec{d})\cdot(N-1)+length(\vec{d})(N-1) \tag{4.28}$$

For the re-entry vehicle problem, as currently defined, there will therefore be $7(N-1)$ elements of the total constraint vector and $105(N-1)$ non-zero elements of the constraint Jacobian matrix introduced by the defect constraint. The complexity of the system model makes it all but impossible to analytically calculate these partial derivatives. Therefore the necessary elements of the Jacobian matrix are estimated numerically using a first-order difference approximation:

$$\frac{\partial d_{i_j}}{\partial x_{i_n}} = \frac{d_{i_j}(\vec{x}_i + \vec{\varepsilon}^T \vec{x}_i, \vec{x}_{i+1}, u_i, u_{i+1}) - d_{i_j}(\vec{x}_i, \vec{x}_{i+1}, u_i, u_{i+1})}{\varepsilon} \tag{4.29}$$

where $\vec{\varepsilon}$ is defined as :

$$\vec{\varepsilon} = \begin{Bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \varepsilon_4 \\ \varepsilon_5 \\ \varepsilon_6 \\ \varepsilon_7 \end{Bmatrix}, \quad \varepsilon_i = \begin{cases} \varepsilon \; if \; i = n \\ 0 \; otherwise \end{cases} \tag{4.30}$$

and ε is an arbitrary small number. The necessary formulas to implement the defect constraint in the nonlinear programming software have now been derived.

4.2.2 The Initial Condition Constraint

The initial condition constraint is perhaps the simplest component of the constraint matrix. This constraint enforces the initial vehicle state in the nonlinear programming software. The initial state is specified for each particular re-entry guidance problem, and consists of the vehicle inertial position, inertial velocity, bank angle, and bank rate. Note that the bank rate at the first node is not a free control variable.

For the nominal re-entry guidance problem as stated in section 2.1, it is not strictly necessary to consider the vehicle initial condition as a constraint. These values could simply be treated as constants in the nonlinear programming software. However, in addition to its role as guidance software, the collocation software could potentially be used as part of a trajectory planning algorithm. With minor modifications the software could be set up to determine the optimal vehicle entry interface state given a nominal bank profile and target. This possibility is discussed further in Chapter 6. For the purpose of completeness the vehicle initial state is implemented here as a constraint. The constraint is simply:

$$\vec{x}_1 = \begin{Bmatrix} x_{RV_{ECI}}(t=t_0) \\ y_{RV_{ECI}}(t=t_0) \\ z_{RV_{ECI}}(t=t_0) \\ u^{ECI}_{RV_{ECI}}(t=t_0) \\ v^{ECI}_{RV_{ECI}}(t=t_0) \\ w^{ECI}_{RV_{ECI}}(t=t_0) \\ \phi(t=t_0) \end{Bmatrix} \tag{4.31}$$

$$u_1 = \dot{\phi}(t=t_0) \tag{4.32}$$

and the corresponding Jacobian is:

$$J_{IC} = \begin{bmatrix} \frac{\partial x_{1_1}}{\partial x_{1_1}} & \frac{\partial x_{1_1}}{\partial x_{1_2}} & \cdots & \frac{\partial x_{1_1}}{\partial x_{1_7}} \\ \frac{\partial x_{1_2}}{\partial x_{1_1}} & \ddots & & \\ \vdots & & \ddots & \\ \frac{\partial x_{1_7}}{\partial x_{1_1}} & & & \frac{\partial x_{7_7}}{\partial x_{7_7}} \end{bmatrix} = \bar{I} \tag{4.33}$$

The initial condition constraint therefore adds 7 constraints to the nonlinear programming problem and 7 non-zero elements to the constraint Jacobian. Note that the initial bank rate is still assumed to be zero and is not explicitly implemented as a constraint, since the K-1 software currently nulls the vehicle body rates before handing

over control to the re-entry software. If some non-zero rate were to be allowed, the initial bank rate could be implemented as a constraint using the same procedure shown above. In this case the inclusion of a non-zero initial bank rate constraint would add one more element to the total constraint vector and one more non-zero element to the total constraint Jacobian.

4.2.3 The Final Condition Constraint

There are several possible ways to deal with the desired final state of the vehicle. The vehicle final state may be treated as either a constraint or as part of the objective function. Early experimental results indicated that enforcing the vehicle final position as a constraint greatly reduced the ability of the nonlinear programming software to converge on a solution. Therefore, for the remainder of this study the desired final position was included as part of the objective function, with the objective being to minimize the distance to the desired landing site at the end of the vehicle trajectory. However, the final position constraint could be implemented in the same manner as the initial position constraint. In this case the final position constraint would add 3 elements to the constraint vector, and 3 non-zero elements to the constraint Jacobian.

4.2.4 The Loading Constraint

One of the advantages to using the collocation procedure is that additional constraints may be easily added to the problem. One such constraint is a limit on the longitudinal acceleration experienced by the vehicle. The design of the re-entry vehicle imposes limits on the structural load which may be applied to the vehicle. The current design specification for the Kistler K-1 requires that the vehicle be subjected to no more than 8 g's of longitudinal acceleration.

This design constraint creates one more general constraint per node in the nonlinear programming problem. It is assumed for simplicity that it is sufficient to

constrain the vehicle loading at each node. In reality the point of maximum structural loading could occur between nodes.

In order to propagate the vehicle state, the inertial acceleration of the vehicle is calculated at each node. The longitudinal acceleration is found by expressing the inertial acceleration of the vehicle in the vehicle body frame:

$$\vec{a}_{RV_b}^{ECI} = \begin{Bmatrix} \dot{u}_{RV_b}^{ECI} \\ \dot{v}_{RV_b}^{ECI} \\ \dot{w}_{RV_b}^{ECI} \end{Bmatrix} = {}_bT_{ECI}\,\vec{a}_{RV_{ECI}}^{ECI} \tag{4.34}$$

The transformation from the ECI reference frame to the body reference frame is found by:

$${}_bT_{ECI} = {}_bT_{vv}\,T_{ECI} = {}_bT_v({}_{ECI}T_v)^T \tag{4.35}$$

The transformation from the velocity-fixed to the ECI frame is derived in section 3.1.1. The transformation from the vehicle body frame to the velocity-fixed reference frame involves a rotation about the vehicle pitch axis by the angle of attack magnitude, followed by a rotation about the resulting z-axis by the side-slip angle magnitude (refer to figures 3.1-1 and 3.1-2 for the definition of angle of attack and side-slip angles). Finally, a rotation of 180 degrees about the vehicle longitudinal axis must be performed to account for the fact that the body z-axis is opposite the lift vector for a zero degree bank angle. The trim side-slip angle for the nominal K-1 launch system is approximately zero. Therefore:

$${}_bT_{vv} \approx \begin{bmatrix} \cos(\alpha) & 0 & -\sin(\alpha) \\ 0 & 1 & 0 \\ \sin(\alpha) & 0 & \cos(\alpha) \end{bmatrix} \tag{4.36}$$

The longitudinal acceleration is then simply applied as a constraint in the nonlinear programming problem. The MINOS software also requires the partial derivative of the

longitudinal acceleration with respect to all of the parameter optimization problem states. Since the vehicle acceleration is only a function of the current vehicle state, the only non-zero gradient terms are those with respect to the vehicle state at the current node. These derivatives are estimated numerically as described in section 4.2.1. The g-loading constraint introduces another N constraints into the total constraint vector and $7N$ non-zero elements in the constraint Jacobian.

Summing from the above sections, the total constraint vector has $7(N-1)+7+N=8N$ elements, and the constraint Jacobian is a $8N\times(7N+1)$ matrix with $105(N-1)+7+N=106N-98$ non-zero elements.

4.3 The Objective Function

Once the necessary constraints have been defined, an objective function may be added to the optimization software. In this case an "optimal" fixed-trim re-entry trajectory has two goals: to minimize the amount of ACS fuel used and to minimize the deviation from the desired final position. The corresponding objective function therefore has two distinct parts. An arbitrary weighting factor must be applied to determine the relative importance of these two goals.

The first part of the objective is to minimize the fuel usage. Since, in this model, there is no mechanism to damp the vehicle bank rate, it is reasonable to assume that fuel usage is directly proportional to the acceleration applied to the bank angle. From Newton's second law:

$$n_{control} = I_{bank}\ddot{\phi} \tag{4.37}$$

where $n_{control}$ is the control moment applied about the bank axis (which is aligned with the air relative velocity vector), and I_{bank} is the vehicle inertia about the bank axis. It can be assumed that the rate of fuel consumption is directly proportional to the absolute value of the control moment. Since the vehicle inertia is assumed to be constant, the rate of fuel consumption is therefore directly proportional to the absolute value of the bank

acceleration. The total fuel usage f_{total} is therefore proportional to the integral the absolute value of the bank acceleration over the vehicle trajectory:

$$f_{total} \propto \int_{trajectory} |\ddot{\phi}| dt \tag{4.38}$$

The average bank acceleration between any two nodes may be estimated using a finite difference approximation:

$$\ddot{\phi}_k \approx \frac{\dot{\phi}_{k+1} - \dot{\phi}_k}{\Delta t_k} \tag{4.39}$$

Substituting:

$$f_{total} \propto \sum_{k=1}^{N-1} \left\| \frac{\dot{\phi}_{k+1} - \dot{\phi}_k}{\Delta t_k} \right\| \tag{4.40}$$

The objective is to minimize the sum on the right hand side of the equation above. In order to accomplish this, the optimization software must be provided the gradient of the fuel objective with respect to each vehicle state and control. Since in this case the nodes are evenly spaced, Δt_k is constant and may be removed from the quantity to be minimized. The gradient is:

$$\frac{\partial f_{total}}{\partial \dot{\phi}_k} = \begin{cases} 1 \; if \; \dot{\phi}_k > \dot{\phi}_{k+1} \\ 0 \; if \; \dot{\phi}_k = \dot{\phi}_{k+1} \\ -1 \; if \; \dot{\phi}_k < \dot{\phi}_{k+1} \end{cases} \tag{4.41}$$

The partial derivative of total fuel usage with respect to any other vehicle state is zero. Therefore the fuel objective introduces N non-zero elements to the objective gradient vector.

The second part of the objective is to minimize the position error between the predicted and desired final positions. The predicted final position error e_p is:

$$e_p = \sqrt{\left(x_{final} - x_{final_{desired}}\right)^2 + \left(y_{final} - y_{final_{desired}}\right)^2 + \left(z_{final} - z_{final_{desired}}\right)^2} \tag{4.42}$$

The objective to be minimized is the square of the miss distance. In this case minimizing the square is appropriate, since it is fair to put a quadratic weight penalty on larger miss distances. The square of the final position error is:

$$e_p{}^2 = (x_{final} - x_{final_{desired}})^2 + (y_{final} - y_{final_{desired}})^2 + (z_{final} - z_{final_{desired}})^2 \tag{4.43}$$

Again, the optimization software must be provided the partial derivatives of the objective function with respect to any vehicle states. Differentiating:

$$\frac{\partial e_p}{\partial x_{final}} = 2(x_{final} - x_{final_{desired}}) \tag{4.44}$$

$$\frac{\partial e_p}{\partial y_{final}} = 2(y_{final} - y_{final_{desired}}) \tag{4.45}$$

$$\frac{\partial e_p}{\partial z_{final}} = 2(z_{final} - z_{final_{desired}}) \tag{4.46}$$

The partial derivative of the position error with respect to any other NLP variable is zero.

In order to combine the two components of the objective function, a weighting factor W is introduced . A linear combination of the two objective functions is taken such that a value of W=1 only takes into account the position error, while a value of W=0 only considers the fuel usage. The total objective function O is:

$$O = (1 - W)\sum_{k=1}^{N-1}\left|\dot{\phi}_{k+1} - \dot{\phi}_k\right| + We_p{}^2 \tag{4.47}$$

The objective gradient also has the weighting factors applied:

$$\frac{\partial O}{\partial \dot{\phi}_k} = (1 - W)\frac{\partial O}{\partial \dot{\phi}_k} \tag{4.48}$$

$$\frac{\partial O}{\partial x_{final}} = 2W(x_{final} - x_{final_{desired}}) \tag{4.49}$$

$$\frac{\partial O}{\partial y_{final}} = 2W(y_{final} - y_{final_{desired}}) \tag{4.50}$$

$$\frac{\partial O}{\partial z_{final}} = 2W(z_{final} - z_{final_{desired}}) \tag{4.51}$$

The partial derivative of the objective function with respect to any other NLP variable is zero. Note that there are a total of $N+3$ non-zero elements in the objective gradient vector. Now that the objective function and its gradient have been derived, they may be inserted into the optimization software and the optimization performed.

4.4 The Code Structure

Computer code was developed to implement the collocation scheme and system model described in the previous sections. The structure of the collocation software is shown in figure 4.4-1. The MINOS software is available as a FORTRAN subroutine. The program "OV" was written to set up the problem, set the necessary nonlinear programming software parameters, read the vehicle initial conditions from the IVS simulation, and call the MINOS subroutine. This program also reads any existing control history stored in the K-1 control software and uses this profile to construct an initial guess for the vehicle trajectory.

The "OV" program solves three nonlinear programming problems every time it is run. First, the program solves for the trajectory using a predetermined trajectory length and control history. If no previous control history is available as an initial guess the software assumes a constant zero bank rate. The vehicle initial state is provided as the first guess for the vehicle state at every node. The solution to the first nonlinear programming problem provides a refined guess of the vehicle trajectory. The program then attempts to solve the same problem with the same bank profile but with free final

time. The trajectory resulting from this problem is then used as the initial guess for the full trajectory optimization problem, with free final time and bank profile. Experience indicates that the collocation software is much more robust when it attempts to solve the re-entry vehicle problem in these three stages.

Experience with the software also suggests that the g-loading constraint should only be applied when a good initial guess is supplied for the vehicle trajectory. The g-loading is very dependent upon the vehicle trajectory, and if a poor initial guess is supplied the loading constraint may prevent the software from converging on the desired solution. Therefore the g-loading constraint is only applied in the third stage of the trajectory optimization.

The "OV" program calls the "minoss" subroutine once for every nonlinear program solved. This subroutine is included as part of the MINOS software package and attempts to solve a generic nonlinear programming problem. The "minoss" subroutine must call additional subroutines which calculate the problem constraint vector, objective value, and the necessary derivatives. To calculate the current objective value, the subroutine calls the "funobj" subroutine, which is also included as part of the MINOS software package. This subroutine was modified to call a user-defined subroutine, in this case named "ovobj." The "ovobj" subroutine is given the current NLP program variables and returns the current objective value and gradient.

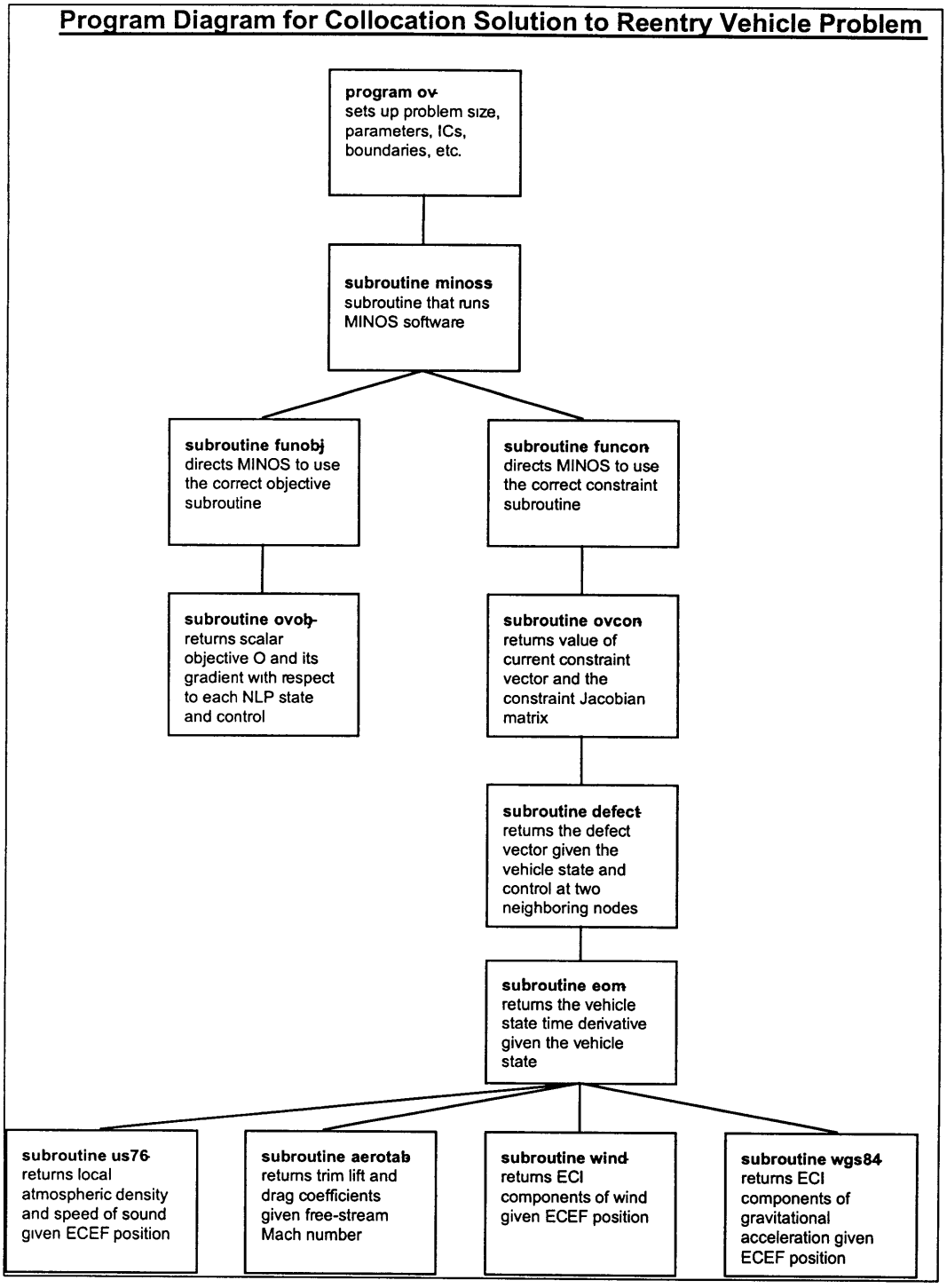


Figure 4.4-1 : Functional flow diagram for the collocation software

The "minoss" subroutine also calls the "funcon" subroutine. The "funcon" subroutine returns the current constraint vector and Jacobian, which it retrieves from the user-defined subroutine "ovcon.". In order to make these calculations, the "ovcon"

subroutine repeatedly calls the "defect" subroutine, which calculates the defect vector at any collocation point given the vehicle state and control at the neighboring two nodes and the segment length. The "defect" subroutine calls the "eom" subroutine as necessary,

Program/Subroutine Name	Inputs	Outputs
OV	-Vehicle Initial Condition -Problem Size -NLP software parameters -Past Control Histories (if available)	-Optimal control history -State trajectory -Final trajectory time
minoss	-Problem size and setup -Boundaries on NLP variables	-NLP variables associated with optimal control history and trajectory -Final values of all constraints
funobj	-problem number and values for NLP variables	-Current objective and gradient
ovobj	-Current values for NLP variables	-Current objective and gradient
funcon	-problem number and values for NLP variables	-Current constraint vector and Jacobian
ovcon	-Current values for NLP variables	-Current constraint vector and Jacobian
defect	-Vehicle states and controls at two neighboring nodes -Length of trajectory segment	-Defect vector at collocation point between consecutive nodes
eom	-Current vehicle state and control	-Current vehicle state derivative
us76	-Current ECI position	-Local atmospheric density and speed of sound
aerotab	-Current free-stream Mach number	-Vehicle trim aerodynamic parameters for nominal c.g. location
wind	-Current true altitude	-Current wind (if known, otherwise assumed to be zero)
wgs84	-Current ECI position	-Current gravitational acceleration vector

Table 4.4-1 : Subroutines used in the collocation software

which uses the vehicle equations of motion to compute the state derivative given any vehicle state. The "eom" subroutine in turn calls the "us76","aerotab","wind",and "wgs84" subroutines. These routines calculate the local atmospheric properties, aerodynamic parameters, current known wind value (if available), and current gravitational acceleration respectively. The necessary inputs and outputs to the various subroutines are summarized in table 4.4-1.

4.5 Testing the Accuracy of the Collocation Solution

Once the collocation software is complete, testing should be performed to ensure that the software correctly predicts the re-entry vehicle motion. First, the assumption (made in section 3.1) that the high frequency aerodynamic oscillations may be neglected is tested. Also, the vehicle trajectories predicted by the collocation software for a predetermined control history may be compared with similar trajectories generated by the IVS simulation.

4.5.1 Validating Initial Assumptions

When the re-entry vehicle equations of motion were derived, the angle of attack and sideslip oscillations were neglected. This assumption was justified experimentally using the Draper IVS simulation. The IVS was modified so that the angle of attack and sideslip angle are set to their estimated trim values after each control cycle.

The Draper IVS simulation uses a 6-degree of freedom model of the vehicle, corresponding to a full 12-state model. The translational states were chosen to be the inertial (ECI) position and the inertial velocity expressed in the vehicle (or body) reference frame. The rotational states are represented by a quaternion describing the transformation from the inertial reference frame to the vehicle (or body) reference frame

and the attitude rates expressed in the body frame. All of the information about the attitude of the vehicle is contained in the inertial to body quaternion. This quaternion may be modified at any step in the numerical integration to instantaneously "place" the vehicle at any desired attitude.

The Kistler re-entry control software calculates the current and trim angle of attack and side-slip angles during each control cycle. From these values, it is possible to calculate the deviations from trim:

$$\Delta\alpha = \alpha - \alpha_{trim} \tag{4.52}$$

$$\Delta\beta = \beta - \beta_{trim} \tag{4.53}$$

A "trim body" reference frame is introduced corresponding to the orientation of the body frame if the vehicle were at trim. The transformation from the actual body frame to the trim body frame involves a rotation about the body z_b-axis by the sideslip angle deviation and a rotation about the resulting y-axis by the angle of attack deviation. The transformation is:

$$ {}_{tb}T_b = \begin{bmatrix} \cos(\bar{\alpha}) & 0 & -\sin(\bar{\alpha}) \\ 0 & 1 & 0 \\ \sin(\bar{\alpha}) & 0 & \cos(\bar{\alpha}) \end{bmatrix} \begin{bmatrix} \cos(\bar{\beta}) & \sin(\bar{\beta}) & 0 \\ -\sin(\bar{\beta}) & \cos(\bar{\beta}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.54}$$

$$ {}_{tb}T_b = \begin{bmatrix} \cos(\bar{\alpha})\cos(\bar{\beta}) & \cos(\bar{\alpha})\sin(\bar{\beta}) & -\sin(\bar{\alpha}) \\ -\sin(\bar{\alpha}) & \cos(\bar{\beta}) & 0 \\ \sin(\bar{\alpha})\cos(\bar{\beta}) & \sin(\bar{\alpha})\sin(\bar{\beta}) & \cos(\bar{\alpha}) \end{bmatrix} \tag{4.55}$$

The corresponding (scalar-first) quaternion transformation is:

$$ {}_{tb}q_b = \begin{Bmatrix} \tfrac{1}{2}\sqrt{1+\cos(\bar{\alpha})\cos(\bar{\beta})+\cos(\bar{\beta})+\cos(\bar{\alpha})} \\ \tfrac{1}{2}\sqrt{1+\cos(\bar{\alpha})\cos(\bar{\beta})-\cos(\bar{\beta})-\cos(\bar{\alpha})}\,\mathrm{sgn}(-\sin(\bar{\alpha})\sin(\bar{\beta})) \\ \tfrac{1}{2}\sqrt{1-\cos(\bar{\alpha})\cos(\bar{\beta})+\cos(\bar{\beta})-\cos(\bar{\alpha})}\,\mathrm{sgn}(\sin(\bar{\alpha})\cos(\bar{\beta})+\sin(\bar{\alpha})) \\ \tfrac{1}{2}\sqrt{1-\cos(\bar{\alpha})\cos(\bar{\beta})-\cos(\bar{\beta})+\cos(\bar{\alpha})}\,\mathrm{sgn}(\cos(\bar{\alpha})\sin(\bar{\beta})+\sin(\bar{\alpha})) \end{Bmatrix}^T \tag{4.56}$$

Where "sgn" is the signum function. This quaternion may be multiplied with the inertial to body quaternion in the vehicle state calculations to place the vehicle at trim:

$$ {}_{tb}q_{ECI} = {}_{tb}q_{b\,b}\,q_{ECI} \tag{4.57} $$

The appropriate quaternion multiplication techniques are found in Ostrander.[24] The inertial to "trim body" quaternion is then substituted back into the IVS simulation, and the vehicle is placed at trim. Several simulation runs were made with the angle of attack and/or side-slip oscillations eliminated. A 700 second run with a constant zero degree bank angle was used for testing purposes. Runs were made with only the side-slip oscillation removed, only the angle of attack oscillation removed, and both oscillations removed simultaneously.

The change in the final vehicle position was used as an indicator of the relative importance of the aerodynamic oscillations. When only the angle of attack oscillation was removed, the final vehicle position differed from the nominal final position by 720 feet. When only the side-slip oscillation was removed the final position changed by 270 feet. When both oscillations were removed simultaneously the final position was 700 feet away from the nominal final position. The effects of the aerodynamic oscillations are reasonably small when compared to the required one nautical mile accuracy in landing position. The Kistler flight control software also attempts to damp the aerodynamic oscillations, rendering the effects on the trajectory even less significant. Therefore it is reasonable to neglect the oscillations (and the associated two degrees of freedom) when optimizing the vehicle trajectory.

4.5.2 Orbital vehicle (OV) Model Validation

The accuracy of the collocation method and the system models may be tested by estimating the trajectory that results from a predetermined bank history. In this case, the collocation software is used to calculate the trajectory that results when the OV is held at

a constant zero degree bank angle. The results are compared to the same trajectory calculated by the IVS simulation.

Plots of the OV zero bank trajectory are included in figure 4.5.2-1 and figure 4.5.2-2. In this case only ten nodes were used to discretize the vehicle trajectory. Note that the vehicle trajectory predicted by the collocation software closely agrees with the trajectory predicted by the IVS simulation. The predicted final vehicle positions disagree by approximately 1.6 miles. Here, the collocation software uses a time step of approximately 80 seconds between nodes while the fourth order Runge-Kutta integration scheme used by the IVS simulation has a time step of 0.02 seconds. Possible sources of error include modeling errors and numerical error associated with both the discretization and the computer implementation. In addition to the aerodynamic oscillations discussed above the IVS simulation also models vehicle forces and mass changes caused by the ACS jets.

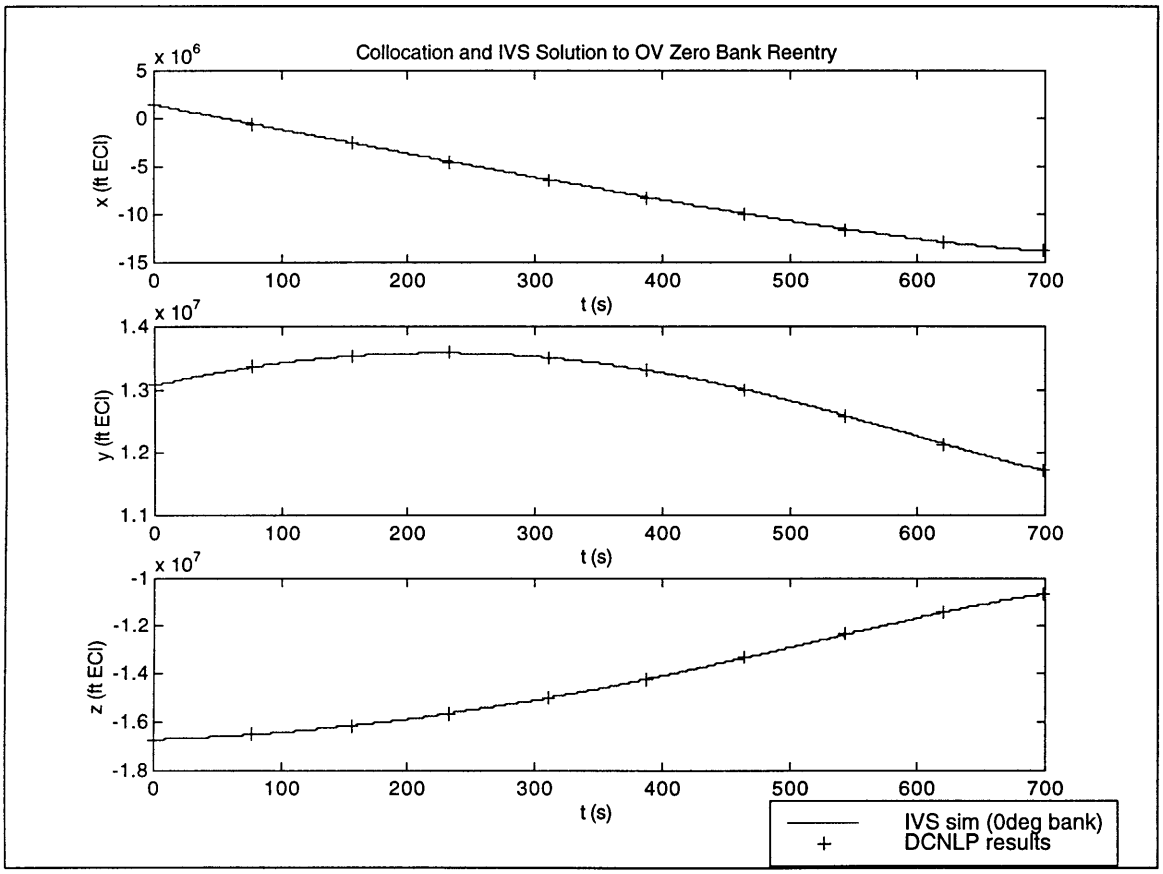


Figure 4.5.2-1 OV Inertial position profile (10 Nodes, Constant 0 deg bank angle)

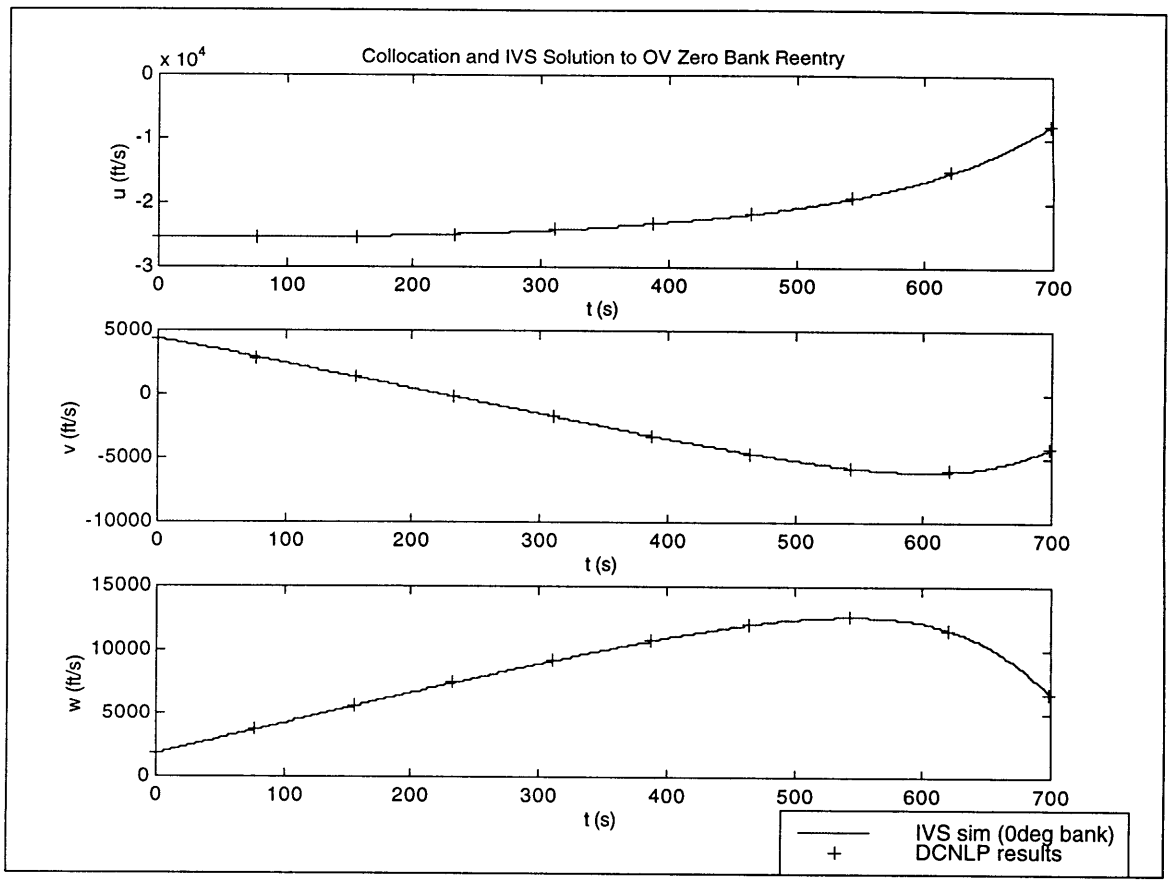


Figure 4.5.2-2 OV Inertial velocity profile (Constant 0 deg bank angle)

A key parameter affecting the accuracy of the collocation solution is the number of nodes chosen to discretize the problem. Figure 4.5.2-3 shows the relationship between the number of nodes chosen and the final position error of the vehicle. Note that the error starts to level off when more than approximately 7 nodes are used. The error approaches 0.8 nautical miles when more than 15 nodes are used.

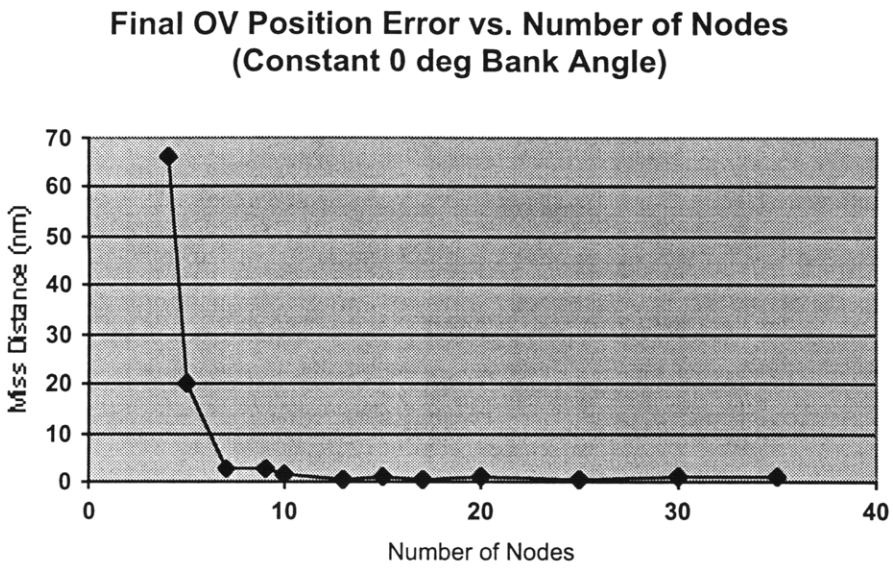


Figure 4.5.2-3 Final OV position error vs. number of nodes (Constant 0 deg bank angle)

4.5.3 Launch Assist Platform (LAP) Model Validation

A similar analysis may be used to consider the accuracy of the LAP solution. Plots of the LAP trajectory are shown in figure 4.5.3-1 and figure 4.5.3-2. For a zero bank re-entry the collocation software agrees well with the IVS simulation and the final position error is less than 0.5 nautical miles.

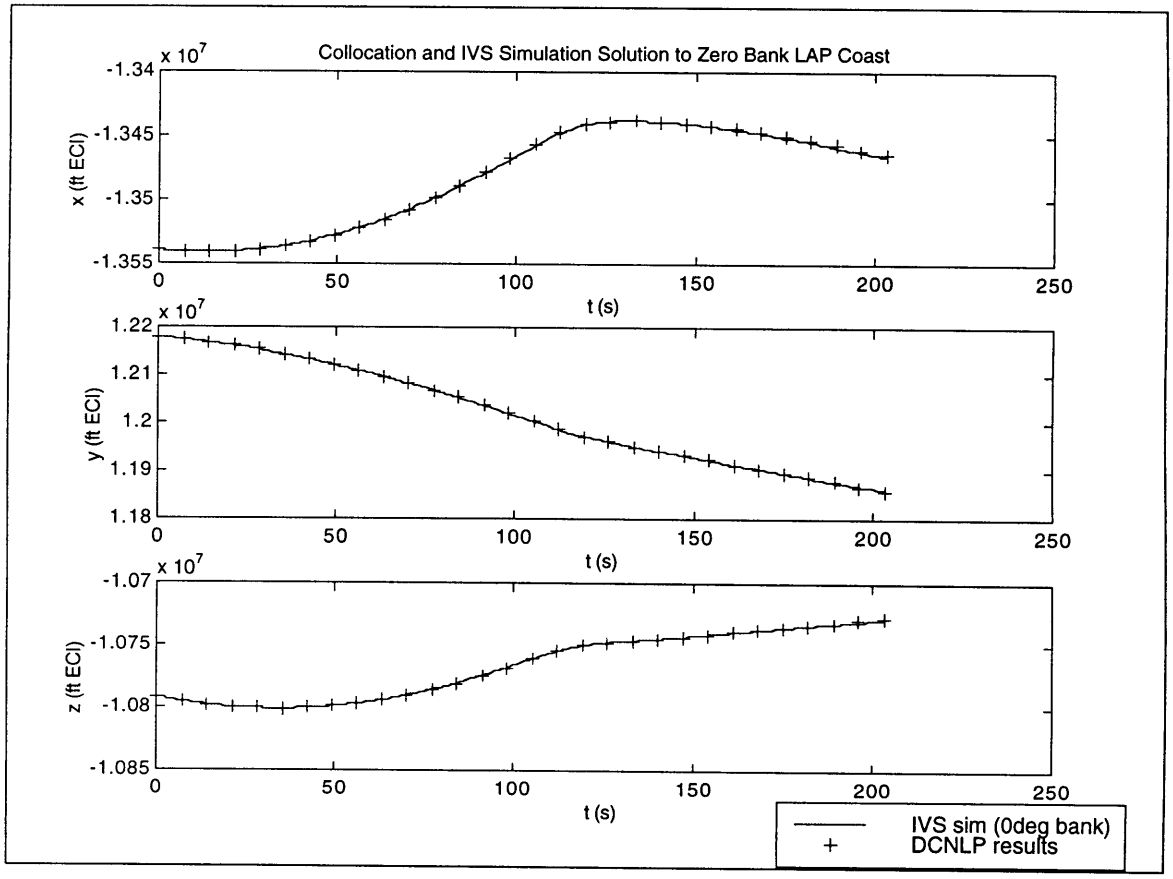


Figure 4.5.3-1 LAP Inertial position profile (Constant 0deg bank angle)

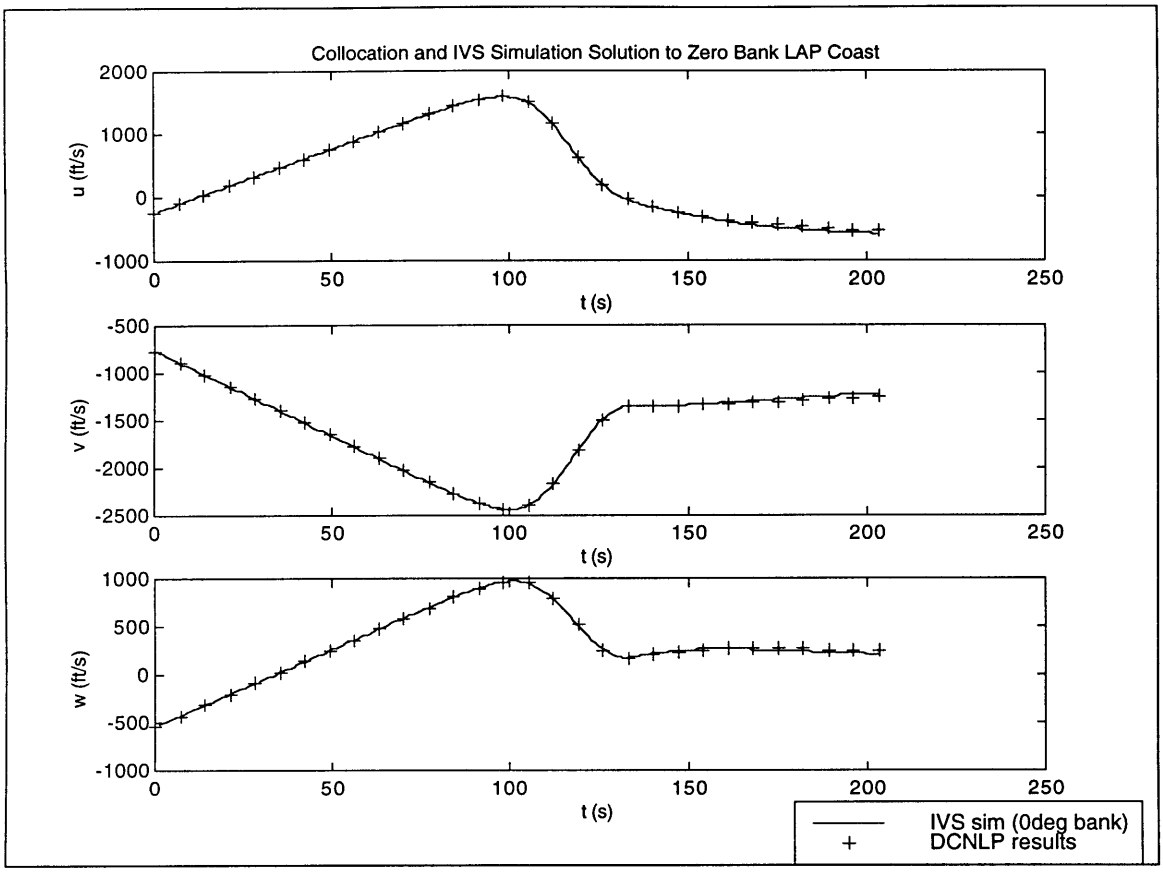


Figure 4.5.3-2 LAP Inertial velocity profile (Constant 0deg bank angle)

As with the OV, the final position error of the LAP appears to approach a constant as more than 7 nodes are used. Figure 4.5.3-3 shows the relationship between the final position error of the LAP and the number of nodes. Note that the final position accuracies achieved with the LAP and the OV models can not be directly compared because the vehicle trajectories are substantially different. The OV has a much longer and faster re-entry trajectory, and therefore the error in predicted final position is expected to be greater than that for the LAP.

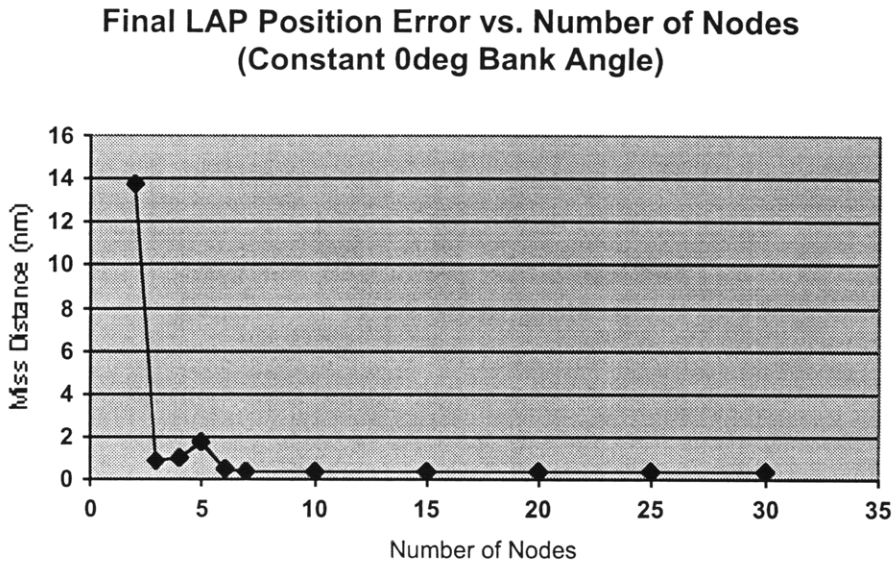


Figure 4.5.2-3 Final LAPposition error vs. number of nodes (Constant 0deg bank angle)

4.6 Implementing the Optimal Trajectories in the IVS Simulation

The trajectories developed by the collocation software must now be implemented by the re-entry vehicle control software. The current control software for the Kistler K-1 vehicle was modified and used for this purpose.

Although the collocation procedure uses the vehicle bank rate as the control variable, it is more appropriate to use bank angle as an input to the controller in the IVS simulation. The "optimal" bank rates are extremely small, and are much smaller than the width of the rate deadbands used by the controller. If the bank rate were to be used as the only input to the control software, it would be overwhelmed by noise. Note that it is still advantageous to use bank rate as the control variable in the collocation because the resulting bank angle history is continuously differentiable with respect to time. Using bank rate as the control variable in the collocation also makes it more straightforward to calculate the objective function.

The optimal bank histories are implemented using the existing vehicle controller. The re-entry control code for the Kistler vehicle receives a desired bank angle command from the re-entry guidance software. The control code estimates the current bank angle from the current vehicle position, attitude, and velocity. The software then issues rotation commands in the vehicle roll and yaw axes which result in the ACS jets firing to maintain the desired bank angle. If the difference between the desired and actual bank angle is small, the controller commands the jets to hold the bank angle inside a deadband of predetermined width. If the desired bank angle input is discontinuous with respect to time, the vehicle can not closely track the desired bank angle. Therefore, if the difference between the desired and actual bank angles is above some threshold, the controller plans a three-part bank maneuver to reach the desired bank angle. The bank maneuver consists of an acceleration phase, a coasting phase, and a decceleration phase.

The collocation procedure assumes that the control variable is linear between the control values at neighboring nodes. Therefore the bank rate profiles produced by the optimization software are piecewise linear. Since the bank rate profiles are continuous, the bank angle profile is both continuous and smooth. The control software is therefore able to track the desired bank angle profile without using the three-part bank maneuvers described above. If the control software did attempt to design a three part bank maneuver, it would most likely pass through the desired bank angle during the acceleration phase of the maneuver, and require a large additional correction to return to the correct bank angle. In order to prevent this from happening, the bank angle error threshold was set to an arbitrary high value.

The optimization software returns the desired bank angle and bank angle rate at each node. The desired bank angle at any given time is then the value of the Hermite polynomial fit between the two neighboring nodes. Node that this polynomial depends on both the bank angle and rate at the neighboring nodes. Using the general calculations shown in section 4.1, the desired bank angle at any time is approximated by a cubic polynomial :

$$\phi(t) = [\frac{2(\phi_{left} - \phi_{right})}{\Delta t^3} + \frac{(\dot{\phi}_{left} + \dot{\phi}_{right})}{\Delta t^2}](t - t_{left})^3 + [\frac{(\dot{\phi}_{right} - \dot{\phi}_{left})}{2\Delta t} + \frac{3(\phi_{right} - \phi_{left})}{\Delta t^2} - \frac{3(\dot{\phi}_{left} + \dot{\phi}_{right})}{2\Delta t}](t - t_{left})^2 + \dot{\phi}_{left}(t - t_{left}) + \phi_{left} \quad (4.58)$$

Since the bank rate varies linearly between nodes:

$$\phi_{right} = \phi_{left} + \frac{\dot{\phi}_{right} + \dot{\phi}_{left}}{2}\Delta t \quad (4.59)$$

Substituting into 4.58:

$$\phi(t) = [\frac{(\dot{\phi}_{right} - \dot{\phi}_{left})}{2\Delta t} + \frac{3\phi_{left}}{\Delta t^2}](t - t_{left})^2 + \dot{\phi}_{left}(t - t_{left}) + \phi_{left} \quad (4.60)$$

where the "left" and "right" subscripts denote the values at the left and right neighboring nodes. As expected for a linear bank rate profile, the bank angle varies quadratically between nodes. This bank angle is used as a continuous input to the control software.

4.7 Chapter Summary

This chapter discussed the design and validation of the collocation software. The construction of the nonlinear programming problem from the trajectory optimization problem was outlined in detail, and the size of the resulting NLP problem was quantified. The code structure was described. The accuracy of the collocation software was demonstrated for both the OV and the LAP by integrating the vehicle equations of motion for a given bank profile. The implementation of the collocation solution in the IVS simulation was also discussed. The next chapter discusses testing of the collocation software. The collocation software is used to solve for an optimal bank profile given the vehicle initial state and desired final position.

Chapter 5: Results

This chapter discusses the testing and evaluation of the trajectory optimization procedure described in the previous chapters. The feasibility of the collocation procedure is analyzed, and various factors which may affect performance are examined. The chapter begins with a description of the test targets and cases chosen. Various test runs dealing with the collocation strategy are then described. The chapter concludes by examining the effects of various unknown system dispersions.

5.1 Choosing Target Positions

The first step in testing the performance of the collocation strategy is to choose appropriate target positions. The Kistler vehicle stages both have pre-determined landing sites that are fixed in ECEF coordinates. Under the currently used guidance strategy, the vehicle state at entry interface (EI) is chosen based on the estimated environmental conditions to allow the vehicle to reach the target using the nominal control history. For the purpose of testing the collocation software, it was necessary to examine a variety of different feasible combinations of initial and final conditions. These conditions were determined by first choosing the vehicle state at EI . The set of nominal EI conditions used for developing the Kistler control and guidance software were chosen. Various target positions were then found by integrating the vehicle equations of motion using a predetermined trajectory length and control history. In all cases the trajectory length was chosen to be 700 seconds, which is the approximate length of the nominal vehicle re-entry phase. The control histories were chosen to be constant bank angle profiles of various magnitudes. The resulting target positions should represent a reasonable distribution of the set of all possible targets that the vehicle could reach given its initial condition.

Ten target positions were chosen for use in testing the collocation software. The first target corresponds to the position of the OV at 700 seconds into the nominal re-entry phase (using the standard Kistler guidance strategy). The next nine positions correspond to the vehicle positions at 700 seconds into the re-entry phase resulting from constant

bank angles of 0, +/-15, +/-30, +/-45, and +/-60 degrees. A summary of the chosen target positions is included in table 5.1-1.

Note that the initial vehicle altitude is set by the definition of EI to 400,000 ft above ground level (AGL). Varying the longitude of the initial vehicle position has no effect on the resulting control history, since the chosen atmospheric and gravity models are independent of longitude. The only effect would be to shift the vehicle target position by the same amount of longitude. Varying the latitude of the vehicle initial position by a small amount would have minimal effects on the corresponding control history, but these effects are considered negligible for the purpose of this study.

OV Target #	**X (ECEF) (ft x10^7)**	**Y (ECEF) (ft x10^7)**	**Z (ECEF) (ft x10^7)**	**Distance from Target #1 (nm)**	**Target Found by:**
1	-1.29981	1.23951	-1.08914	0	Nominal OV position @t=700s
2	-1.3191	1.24094	-1.06833	46.8	0 degree bank held for 700s
3	-1.31757	1.23928	-1.07176	41.0	+15 degree bank held for 700s
4	-1.31592	1.24383	-1.06848	36.2	-15 degree bank held for 700s
5	-1.31073	1.23891	-1.07907	24.5	+30 degree bank held for 700s
6	-1.30740	1.24808	-1.07247	35.8	-30 degree bank held for 700s
7	-1.29791	1.24006	-1.09071	4.2	+45 degree bank held for 700s
8	-1.29286	1.25378	-1.08090	29.5	-45 degree bank held for 700 s
9	-1.27821	1.24322	-1.10705	46.7	+60 degree bank held for 700s
10	-1.27174	1.26080	-1.09447	58.7	-60 degree bank held for 700s

Table 5.1-1: OV target positions

The collocation software was tested using all of these targets. In all cases the software converged on a solution. Typical trajectories and control histories output from the collocation software are included in figures 5.1-1 through 5.1-3.

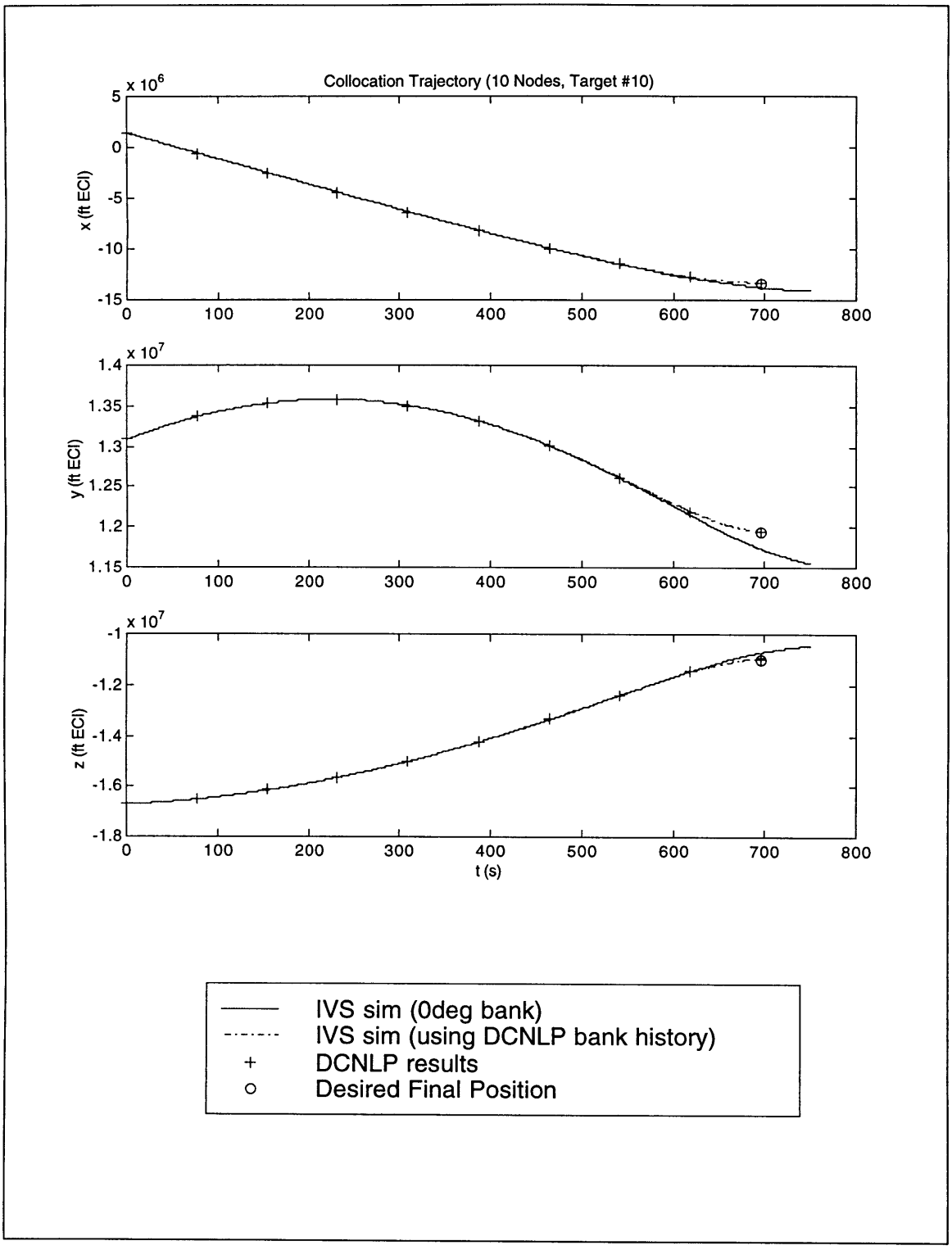


Figure 5.1-1: OV position trajectory (10 Nodes, Target #10)

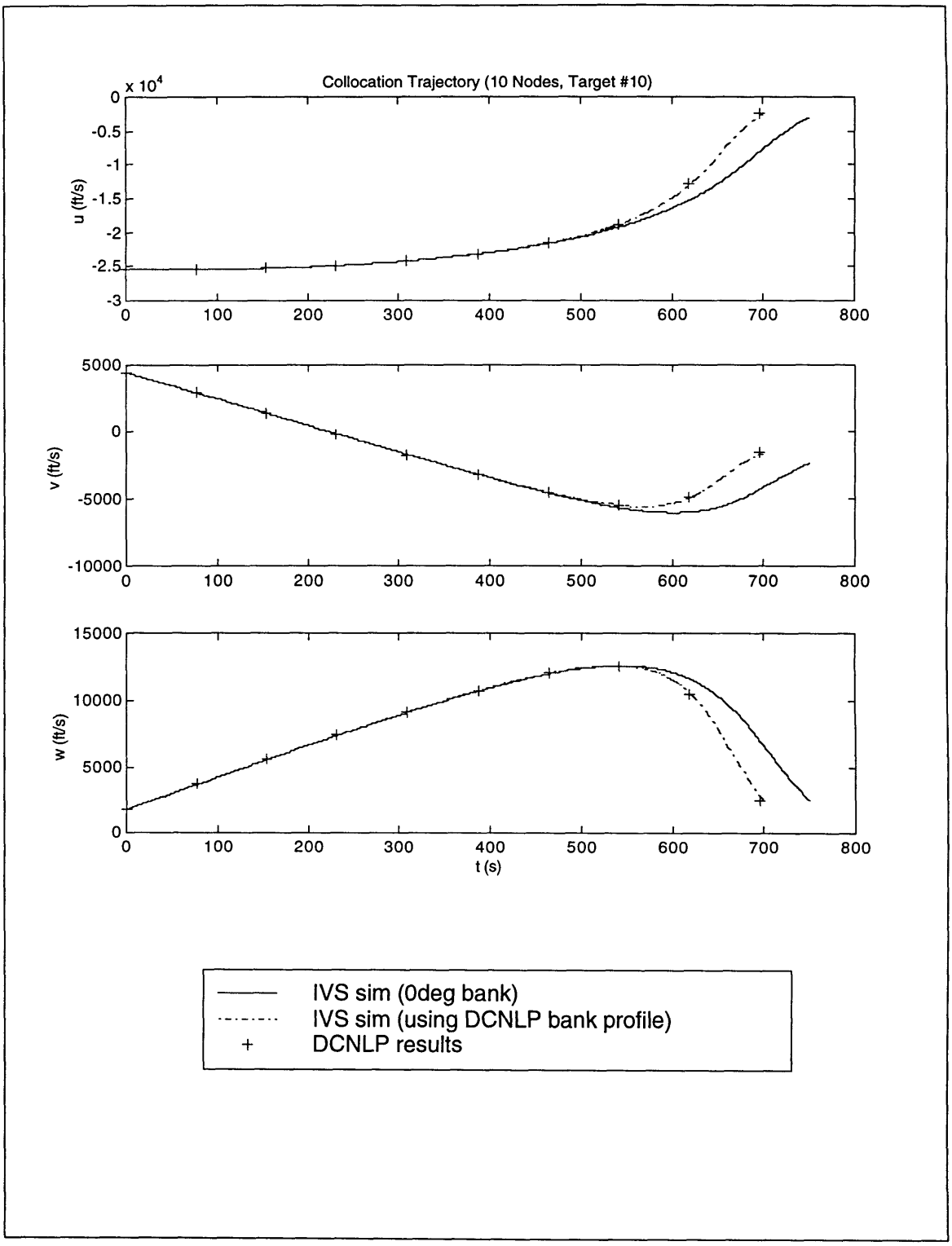


Figure 5.1-2: OV velocity trajectory (10 Nodes, Target #10)

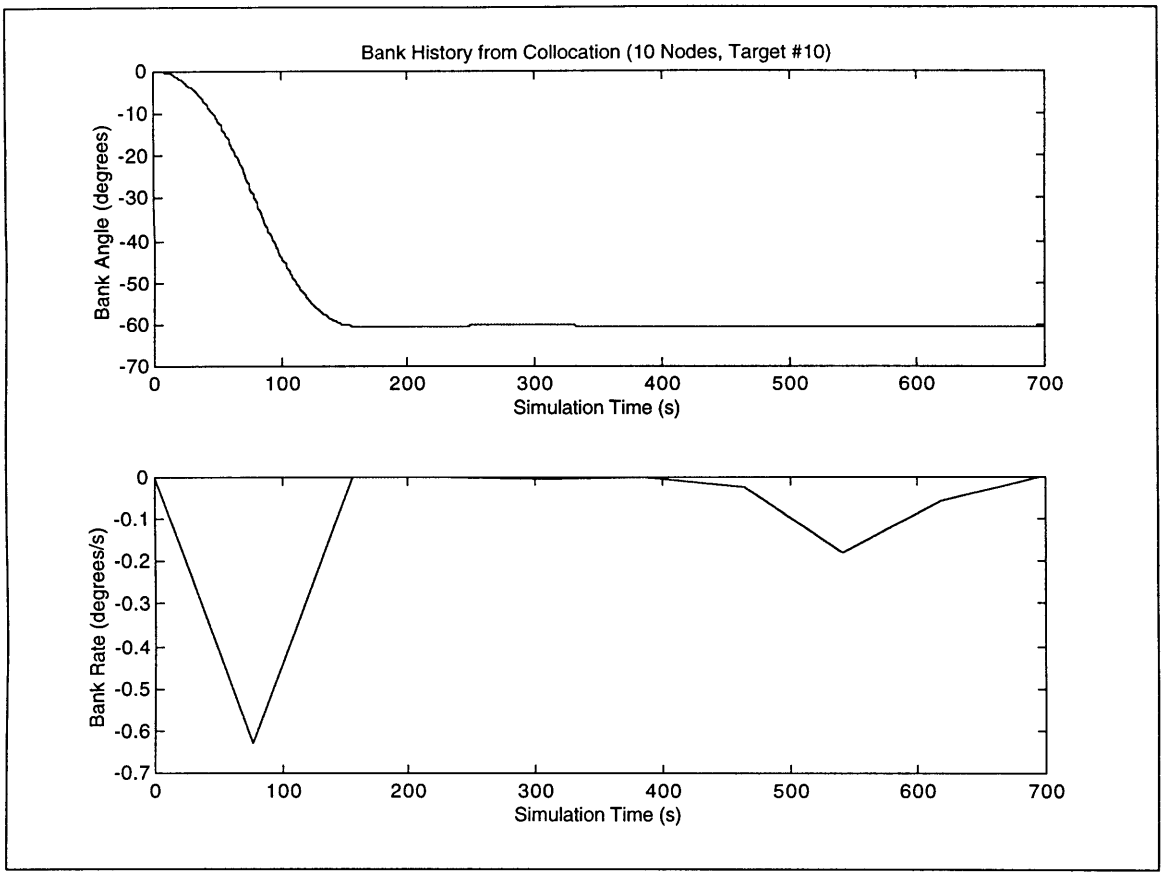


Figure 5.1-3: OV commanded bank history (10 Nodes, Target #10)

Target #	Predicted Miss Distance (nm)	Actual Miss Distance (nm)
1	0.061	2.33
2	0.0024	1.7
3	0.159	2.14
4	0.083	2.63
5	0.033	5.8
6	0.043	2.95
7	0.052	3.99
8	0.344	4.98
9	0.030	6.97
10	0.68	5.34

Table 5.1-2: OV miss distances for various targets

The vehicle zero bank trajectory is also shown in the plots for comparison. When the resulting trajectories were implemented, the associated miss distances were small, but still greater than the 1 nautical mile accuracy design specification. A list of the resulting miss distances is included in table 5.1-2. The "predicted miss distance" is the difference between the desired target and the final vehicle position predicted by the collocation software. The "actual miss distance" is the distance between the target and the final vehicle position predicted by the IVS simulation.

Target position #1 is a particularly interesting case. Target #1 is the vehicle position reached by the nominal Kistler guidance strategy at 700 seconds into re-entry. Since in this case the collocation guidance software is attempting to reach the same final position as the nominal guidance, the performance of the two strategies may be directly compared. The nominal Kistler guidance strategy used 66.2 lbm of ACS fuel and achieved a final position miss distance of approximately 1,000 feet. The collocation guidance strategy developed in this thesis used 30.6 lbm of fuel and achieved an open loop miss distance of 2.3 nautical miles. If the miss distance associated with the collocation technique can somehow be reduced, these results suggest that the collocation approach offers a substantial fuel savings. More detailed tests of the collocation guidance strategy are discussed in the following sections.

5.2 Choosing the Test Cases

In order to further evaluate the collocation software, it was necessary to choose a number of test cases. The purpose of these test cases is to demonstrate that the DCNLP method is a feasible approach to solving the re-entry vehicle problem. (Additional testing would be required if the collocation guidance strategy were to be implemented in actual flight code.) The test cases are summarized here, and the results are discussed in detail in later sections. Two distinct categories of test cases were chosen. The first group of test runs involved varying the collocation scheme parameters. The number of nodes and strategy for updating the control histories were considered. The second category of test runs involved analyzing the sensitivity of the collocation scheme to unknown system

dispersions, including vehicle mass, atmospheric density, wind, and entry angle dispersions.

The first several test cases were discussed in the previous section, and were chosen to show that the collocation software worked for various target positions. One run was made for each of the 10 chosen target positions chosen in the previous section, and the resulting control histories were implemented in the IVS. These runs are "open-loop" runs because the desired control histories were not updated in flight to account for any deviation from the desired trajectory. All of the open-loop runs were made using ten nodes. These runs served as a comparison for all of the additional runs and demonstrated that the collocation software was functioning as expected.

The next set of test runs was chosen to analyze the effects of varying the number of nodes. The number of nodes is an important parameter in any collocation scheme, and effects both the speed and accuracy of the solution. Three of the desired final positions were chosen and 10 runs were made with each desired position varying the number of nodes. Target #1 was chosen because the resulting trajectories can easily be compared to the nominal Kistler guidance scheme. As discussed in the previous section, target #6 was found by integrating the vehicle trajectory given a constant +30 degree bank angle for 700 seconds. Target #7 was found by integrating the vehicle trajectory given a constant –45 degree bank angle for 700 seconds. Since the bank angle held was of opposite sign and different magnitude, targets #6 and #7 should be fairly representative of all other possible target positions. The number of nodes was varied from 3 to 30 nodes. Note that a similar set of runs for the constant zero bank angle case was previously discussed in section 4.5.2. These runs resulted in plots of final position error versus the number of nodes used for each of the desired final positions tested. The goal of this set of test runs was to estimate the smallest number of nodes that may reasonably be used. As with the previous set of runs, this set was "open-loop" because the control history was not updated in flight.

If the collocation techniques described in this thesis are to be used for vehicle guidance, there must be some mechanism for updating the control histories in flight. This is necessary to account for uncertainty in the vehicle and environmental parameters, as well as any unplanned disturbances that may occur during flight. The next set of test runs

involved updating the optimal control history once during the re-entry phase. As before, three of the test target positions were chosen for these tests. In all cases, the optimization software was run at the beginning of the vehicle re-entry phase. The resulting control histories were loaded into the vehicle control software, and the IVS simulation was run. At some point during the trajectory, the IVS simulation was paused and the collocation software was re-run using the current vehicle state as the new initial condition. The updated control history was then loaded into the IVS simulation and the simulation was continued. Note that if the optimization software were actually to be run in flight, there would be some time lag while the software computes the new trajectory. That is, by the time the collocation software returns a control history the vehicle would no longer be at the specified initial condition. The effects of this time lag are assumed to be small and are not considered in this thesis. The goal of this set of test runs was to determine if there appeared to be an optimal time during the trajectory to update the control history.

The next logical step was to update the collocation solution at multiple points during the trajectory. Again, three of the test target positions were chosen for analysis. For each target position, runs were made in which the optimization software was re-run multiple times. In all cases, the optimization software was re-run after the vehicle had completed a predetermined fraction of the estimated remaining trajectory time. This set of test runs yielded plots of miss distance versus number of optimization updates per run. The goal of this set of test runs was to confirm that the final miss distance is reduced as the optimization software is updated more often in flight.

The next category of test runs examined the sensitivity of the collocation method to unknown system dispersions. Any effective guidance strategy must be able to cope with unpredictable disturbances which lead the re-entry vehicle away from the desired trajectory. Each set of dispersion runs began with an "open loop" run to show that the particular dispersion being considered significantly influenced the final position accuracy. Multiple trajectory updates were then performed in flight to demonstrate that the final vehicle position converged on the target. Each set of dispersion runs was used to create plots of vehicle miss distance versus the number of optimization updates performed during re-entry. The first set of dispersion runs involved the re-entry vehicle mass. Several of the desired final positions were chosen, and gains were applied to the

value for vehicle mass in the collocation software vehicle model. The net result was that the value of vehicle mass used in the IVS simulation differed from the value in the collocation software vehicle model. The vehicle trajectory calculated by the IVS simulation therefore tended to diverge from the trajectory predicted by the collocation software. The next set of test runs involved a similar analysis for unknown density dispersions. In this case, the vehicle mass was set to the correct value and gains were applied to the collocation software density model. In addition to these constant gains, several runs were made using more realistic density dispersions obtained from a more sophisticated atmospheric model. A slightly different approach was taken with wind dispersions. Experience indicates that the vehicle trajectory during the re-entry phase is not largely affected by wind. Therefore the approach taken was to show that a large "worst-case" wind had little effect on the vehicle trajectory. Cross-winds and head-winds were considered separately. Finally, the effects of varying the vehicle entry angle were considered. The vehicle entry angle is the angle between the vehicle air-relative velocity vector and the local horizontal at entry interface. The vehicle trajectory is extremely sensitive to this angle.

All of the test runs described in this chapter were done using data for the Kistler K-1 OV. At the time these runs were performed, the LAP was not capable of holding non-zero bank angles to the degree of accuracy required for this study. Although its control software and ACS are similar to those found on the OV, the LAP was designed only to hold a constant zero bank angle. The shorter, slower trajectory of the LAP also creates less opportunity to vary the final vehicle position. In fact, it is difficult to control the LAP to land outside of the specified landing tolerance. Therefore, it was decided that the current LAP configuration was not appropriate to use in this study.

5.3 The Collocation Setup Test Cases

A series of cases were considered in order to determine the effects of varying the manner in which the collocation is performed. The collocation may be performed using different node structures, and the software may be re-run at various points in the vehicle trajectory. All of the test runs in this section used data for the Kistler K-1 OV. Three

different target positions (positions #1, #6, and #7 described above) were used for these tests.

5.3.1 Varying the Node Density

The effects of varying the node density were examined. In all cases the nodes were evenly spaced along the vehicle trajectory. The number of nodes was varied from 3 to 30. Plots of vehicle miss distance for various targets are included in figures 5.3.1-1 through 5.3.1-3. Table 5.3.1-1 lists the data in tabular form for the target #1 case. The results for the other targets were similar.

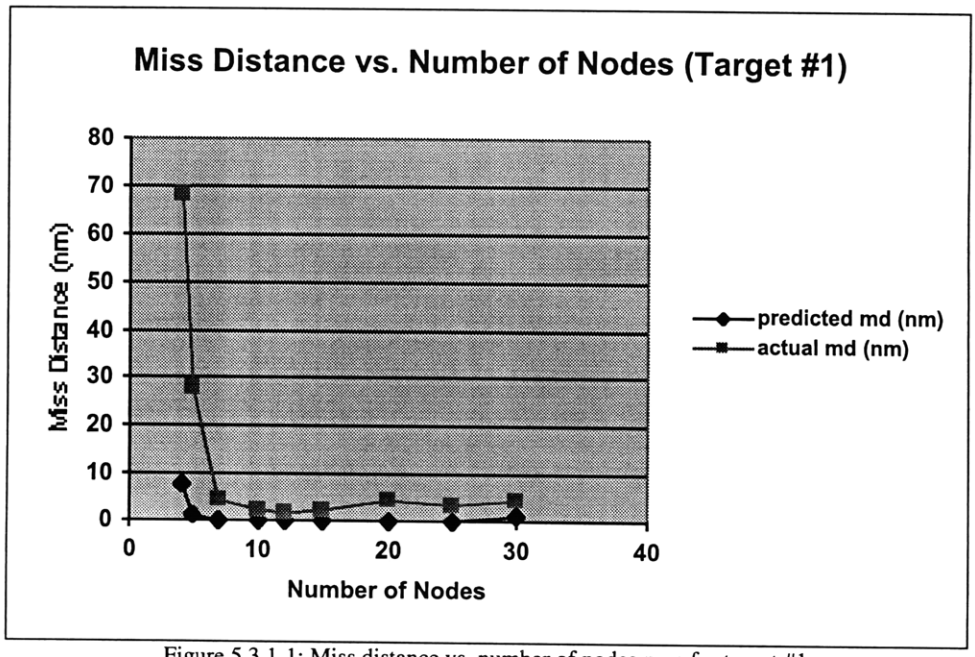


Figure 5.3.1-1: Miss distance vs. number of nodes runs for target #1

# of Nodes	Predicted Miss Distance (nm)	Actual Miss Distance (nm)
4	8.0	68.4
5	0.935	28.0
7	0.0595	4.34
10	0.061	2.33
12	0.0956	1.47
15	0.0319	2.20
20	0.0826	4.40
25	0.039	3.59
30	0.0123	4.47

Table 5.3.1-1: Miss distance vs. number of nodes runs for target #1

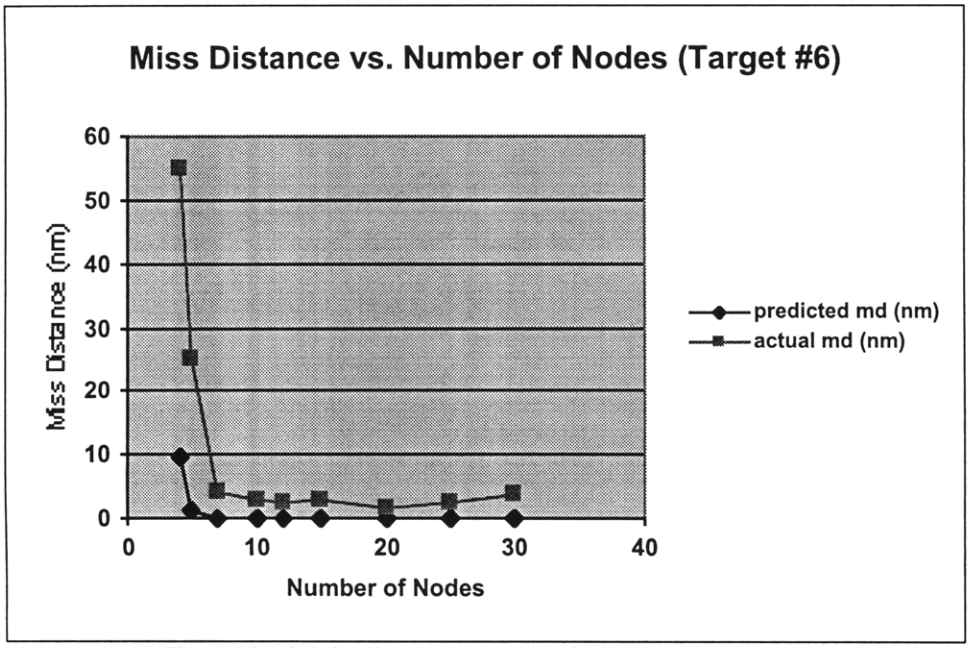


Figure 5.3.1-2: Miss distance vs. number of nodes runs for target #6

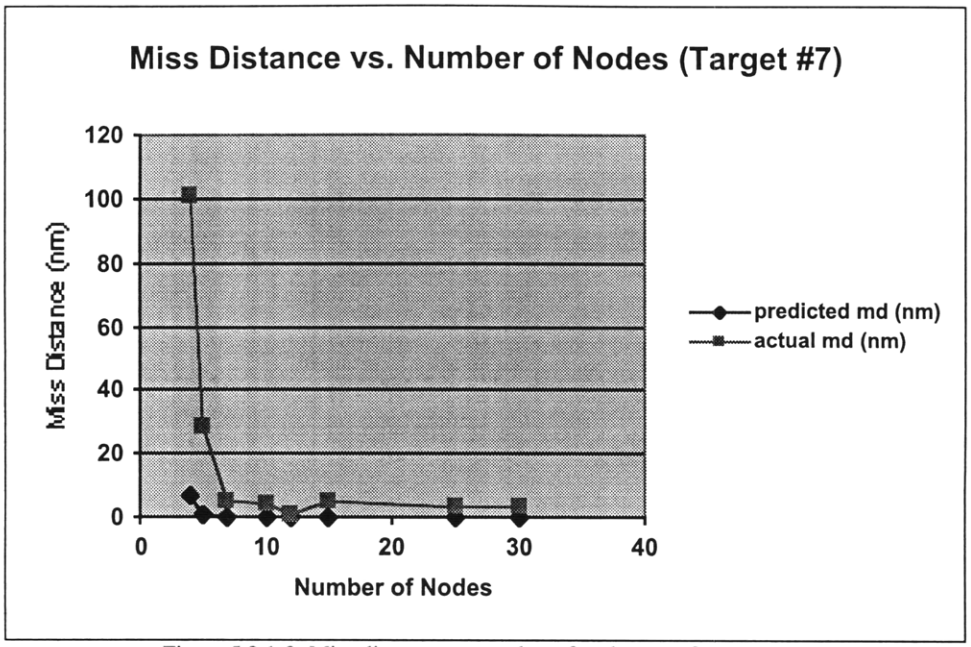


Figure 5.3.1-3: Miss distance vs. number of nodes runs for target #7

Note that the predicted miss distance was small (under the one nautical mile limit) with as few as 5 nodes. Since the collocation software only outputs the bank rate at the discrete nodes, increasing the number of nodes allows the software to design more complicated bank profiles. The fact that the predicted miss distance was small indicated that the collocation software had satisfactory bank profile resolution to design trajectories which terminated at the desired target. As with the constant zero degree bank angle case, the actual miss distance appeared to approach a constant value when more than 7 nodes are used. This suggests that, while the necessary control resolution is present at approximately 5 nodes, more nodes are required to accurately integrate the vehicle equations of motion. Therefore, the number of nodes that must be used in the collocation scheme is limited by the accuracy required to integrate the equation of motion. It is desirable to use as few nodes as possible in order to limit the problem size and required solution time. Considering the above plots, the choice of 10 nodes appeared to offer a reasonable compromise between accuracy and speed. Therefore 10 nodes were used for the remainder of this study. Also note from the above data that the actual miss distances are reasonably small when more than 7 nodes are used. The vehicle has the ability to change its final position on the order of one hundred miles. Actual miss distances on the

order of several miles indicate that the collocation software is working as planned. In order to meet the 1 nautical mile accuracy design specification, some form of feedback scheme will be necessary.

5.3.2 Updating the Trajectory Once During Re-entry

Various disturbances, discretization errors and modeling inaccuracies make it impossible for the vehicle to experience the exact trajectory predicted by the collocation software. One way to improve the final position accuracy of the vehicle is to re-run the optimization software during re-entry. Figures 5.3.2-1 through 5.3.2-6 show the effects of updating the trajectory at various times for different target positions. In all cases the trajectory was updated after a certain fraction of the trajectory length had passed.

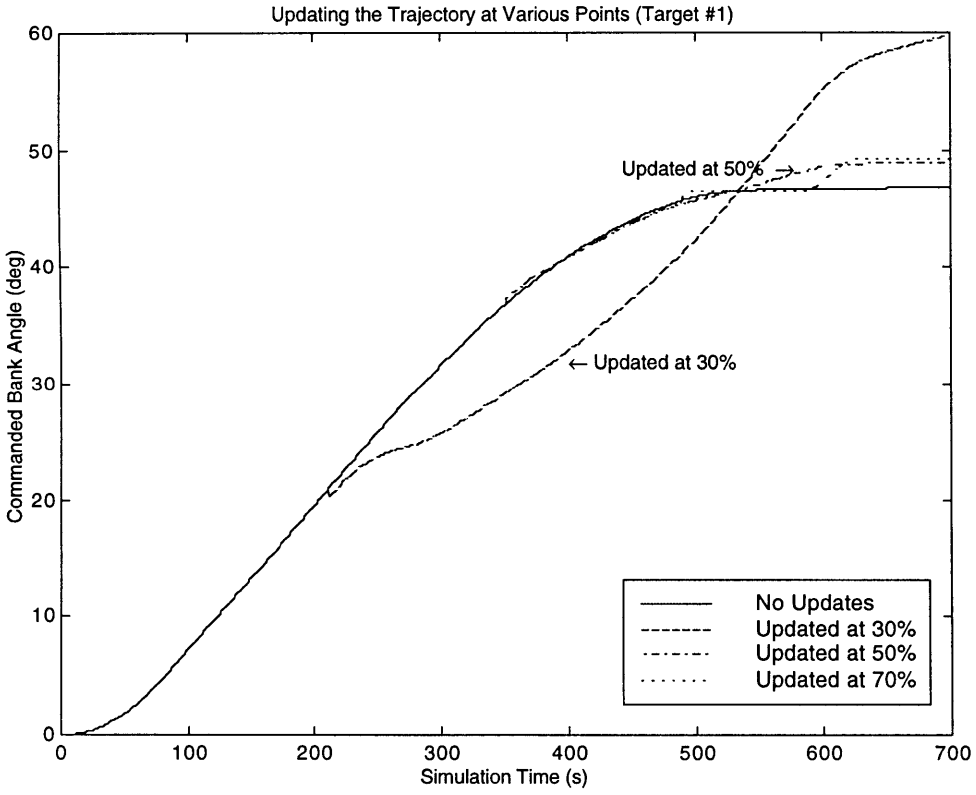


Figure 5.3.2-1: Updating the vehicle trajectory at various times for target #1

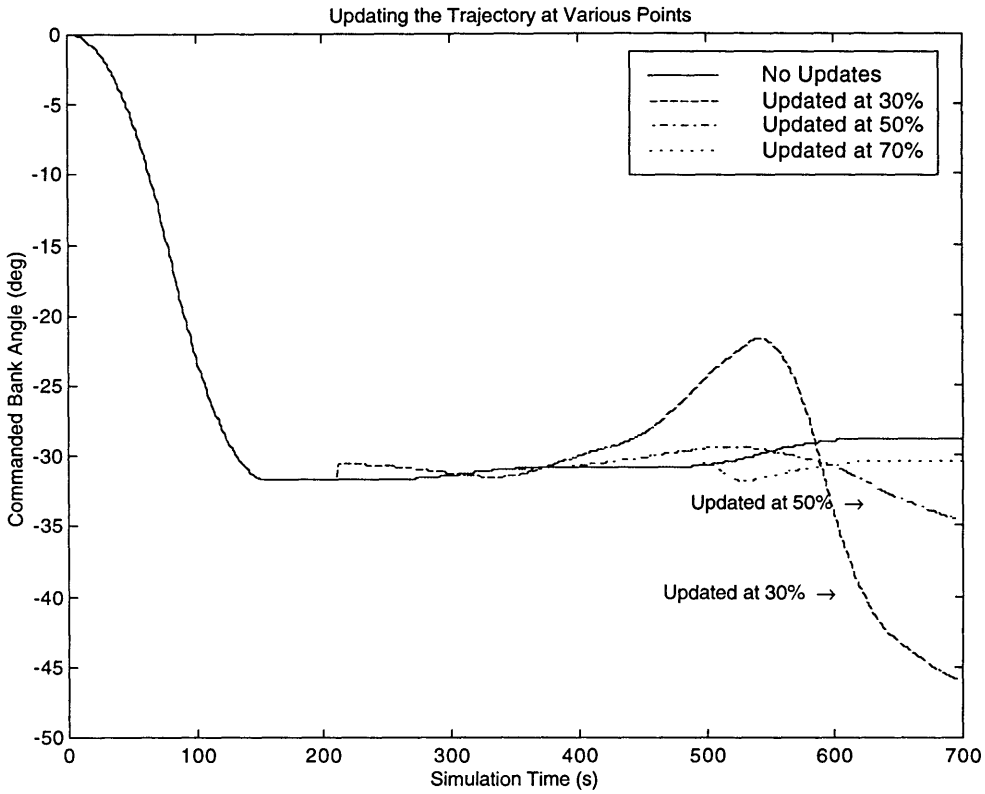


Figure 5.3.2-2: Updating the vehicle trajectory at various times for target #6

Note that in the first two cases (fig. 5.3.2-1 and 5.3.2-2) the trajectory update at 30% of the initial trajectory length appears to differ substantially from the trajectories updated later in flight. When the update was performed later in the flight the resulting trajectories more closely resembled the original trajectory. This is likely a case of the software finding a different locally optimal (or "suboptimal") trajectory. In both cases the update at 30% uses almost the same amount of fuel as the update at 50% (29.86 lbm vs. 31.70 lbm for target #1 and 60.77 vs. 58.58 lbm for target #2) and achieves a similar miss distance (0.75 nm vs. 0.6 nm for target #1 and 0.30 nm vs. 0.62 nm for target #2). Since the nonlinear programming software seeks only to minimize the objective, it is reasonable that it might "jump" between two different trajectories with similar objective

values (since one trajectory has no advantage over the other). Since the trajectories do have similar objective values and all of the constraints are met, this phenomena appears to have no effect on the vehicle performance.

Also note that there sometimes appears to be a slight discontinuity in bank angle when an update is performed (specifically the update at 30% in fig. 5.3.2-2). When the trajectory update is performed, the initial vehicle bank angle in the collocation software is constrained to the current actual vehicle bank angle, not the current commanded bank angle. If the actual and commanded bank angle do not agree exactly the result is a small discontinuity in the commanded bank angle profile when the updated trajectory is loaded into the control software.

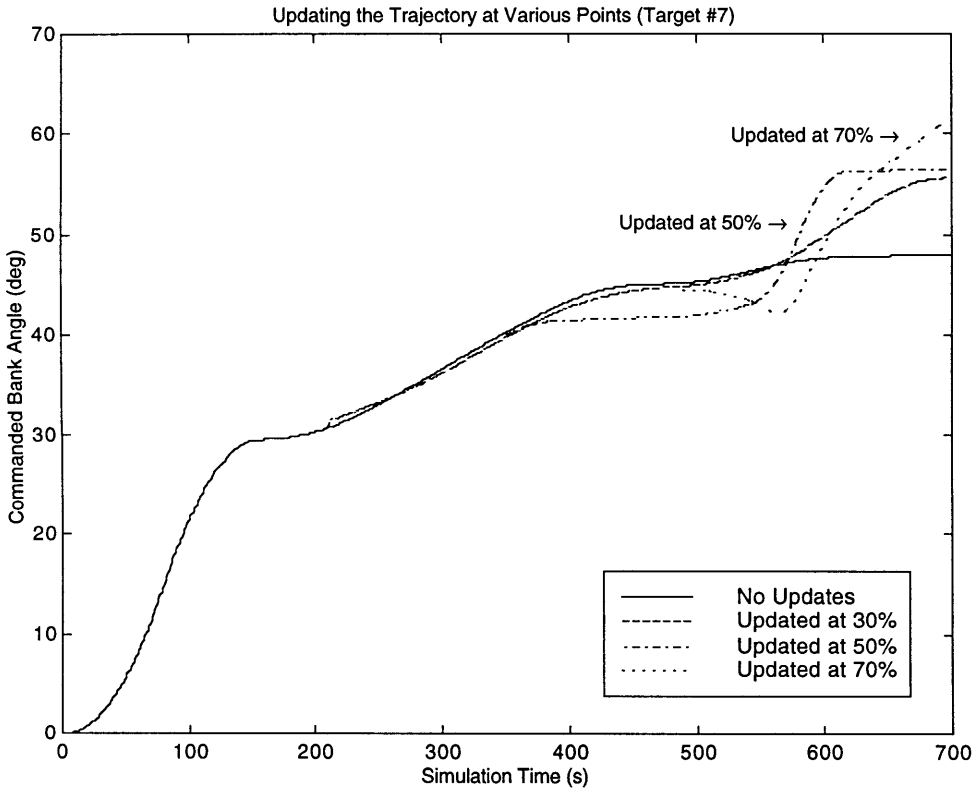


Figure 5.3.2-3: Updating the vehicle trajectory at various times for target #7

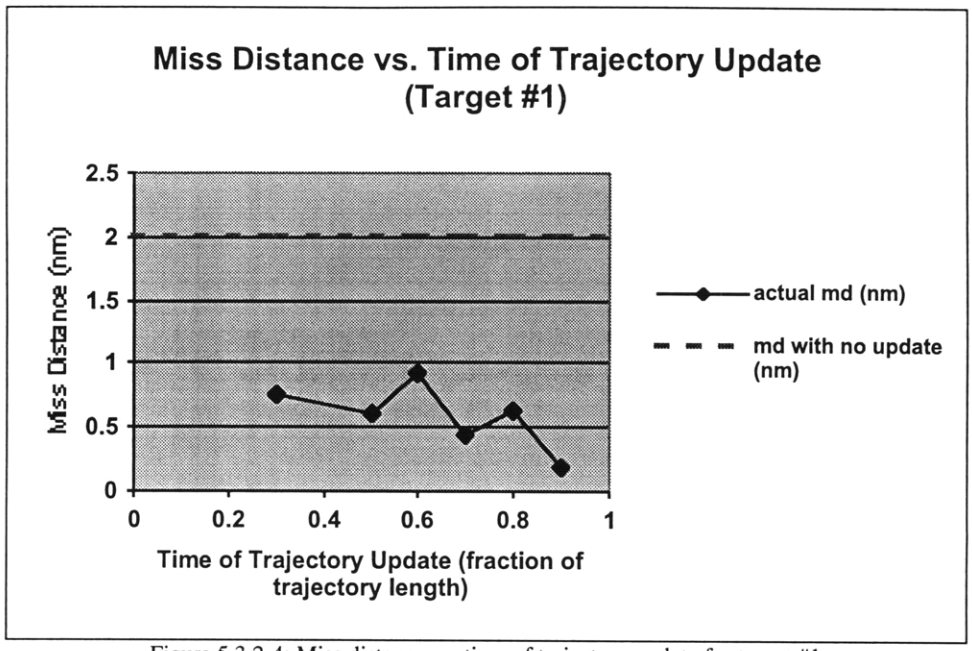


Figure 5.3.2-4: Miss distance vs. time of trajectory update for target #1

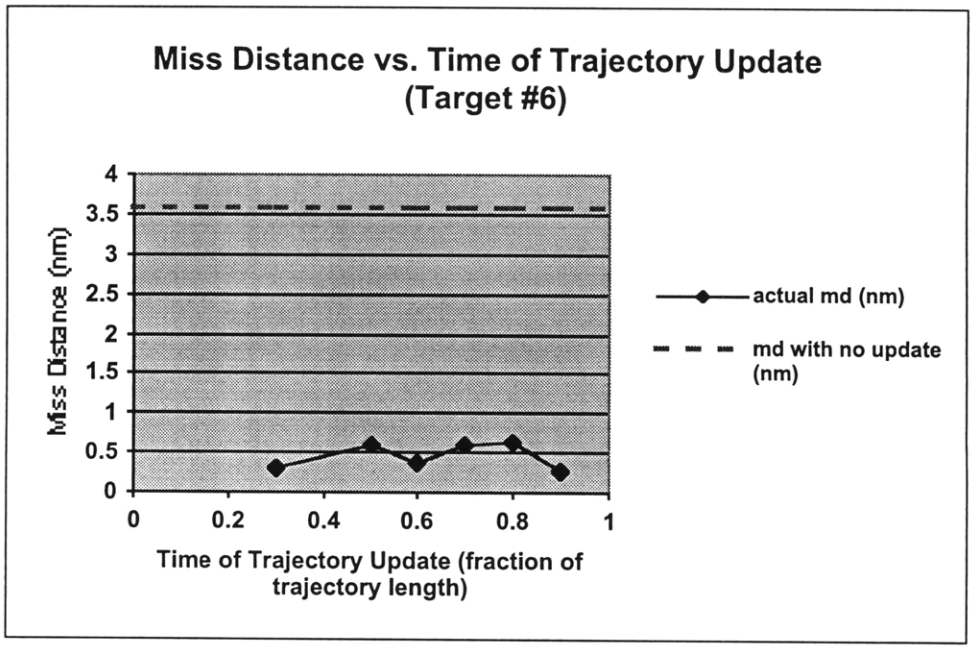


Figure 5.3.2-5: Miss distance vs. time of trajectory update for target #6

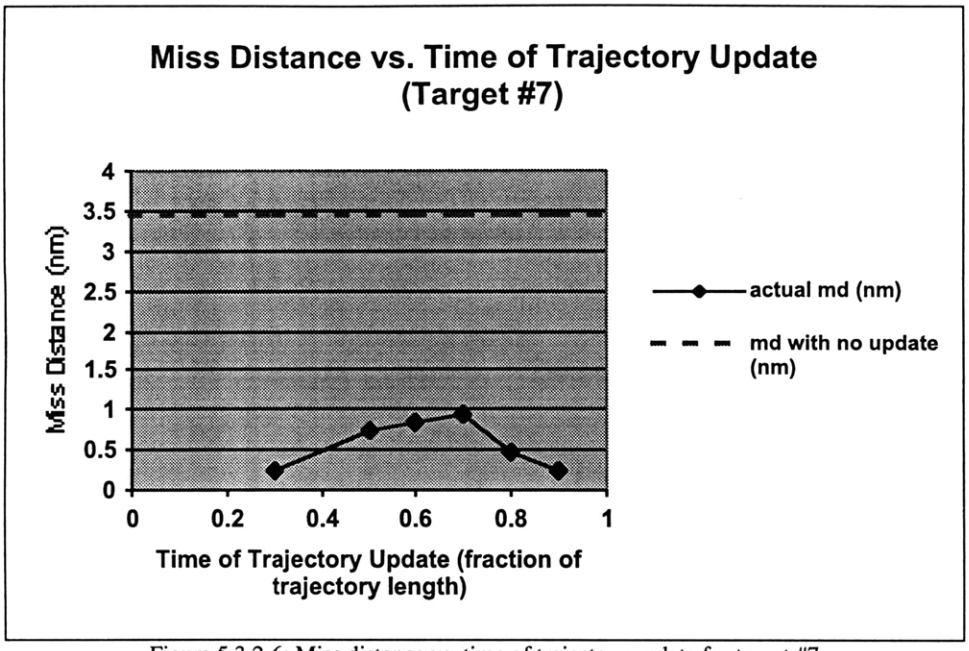


Figure 5.3.2-6: Miss distance vs. time of trajectory update for target #7

Note that in all cases updating the trajectory once in flight reduced the final position error to within 1 nautical mile. There appears to be little correlation between miss distance and exactly when the collocation software is re-run. When the update is performed early in the trajectory, there is a large amount of time left for disturbances to be introduced to the trajectory and propagate. However, when the update is performed later in the trajectory the vehicle has less ability to influence its final position (since it is lower in the atmosphere) and must make more drastic maneuvers to do so (since there is less time remaining to make a correction). These tests suggest that these two effects tend to cancel each other out. If the final position accuracy is to be improved, the vehicle control history must be updated multiple times in flight. For simplicity, all future trajectory updates will be performed when the vehicle is 50% through the re-entry trajectory. In the case of multiple trajectory updates, updates will be performed midway between the previous trajectory update and the predicted trajectory termination time.

5.3.3 *Updating the Trajectory Multiple Times During Reentry*

During an actual flight, the optimization software could be run continuously if necessary. For the next series of tests, the optimization software was re-run multiple times during reentry. In all cases when the optimization software was re-run, it was done when 50% of the previously calculated trajectory had passed. Figures 5.3.3-1 through 5.3.3-6 show the effects of updating the control history multiple times in flight.

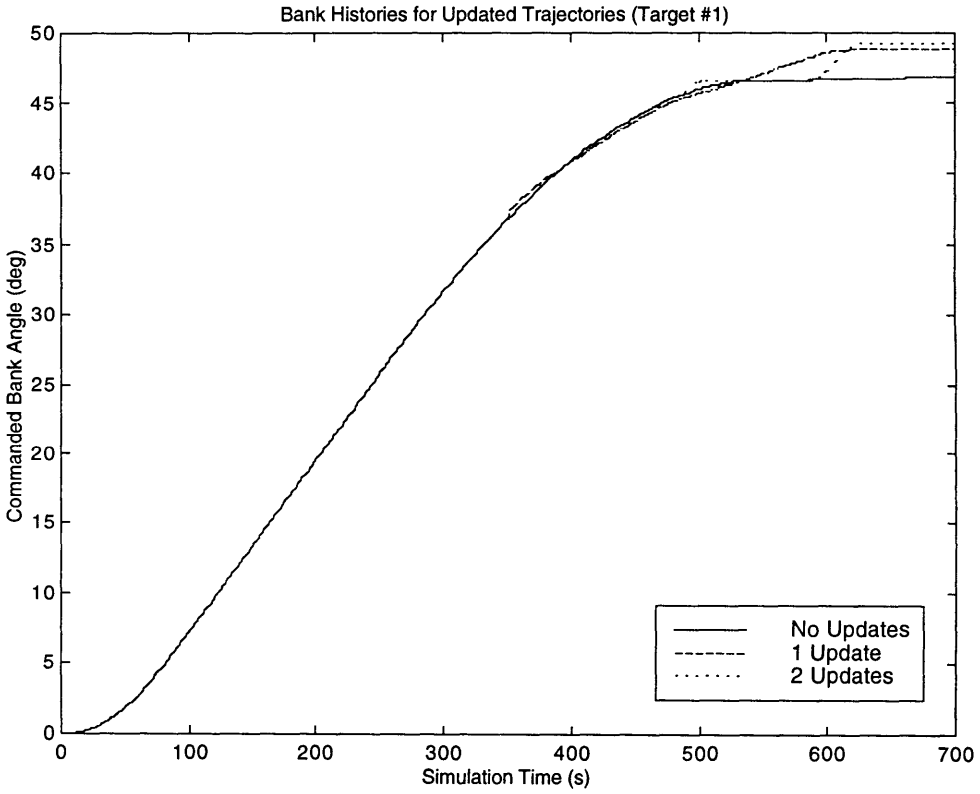


Figure 5.3.3-1: Bank angle profile with multiple trajectory updates for target #1

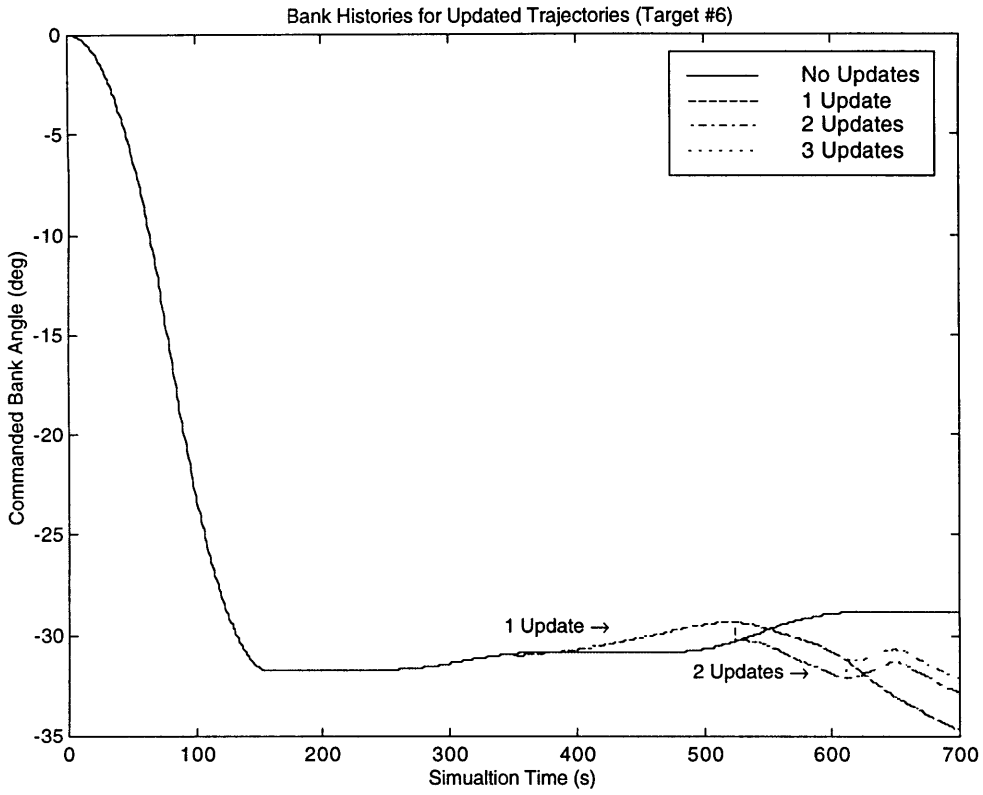


Figure 5.3.3-2: Bank angle profile with multiple trajectory updates for target #6

As before, the results for Target #1 may be directly compared with the nominal guidance strategy results. When the trajectory optimization software was run six times during reentry, the final position miss distance was 838 feet (of the same order as the nominal strategy) and the vehicle used 31.6 lbm of ACS fuel (versus 66.2 for the nominal strategy). Therefore, at this stage in the analysis, the collocation guidance strategy appears to be a promising alternative to the nominal guidance strategy.

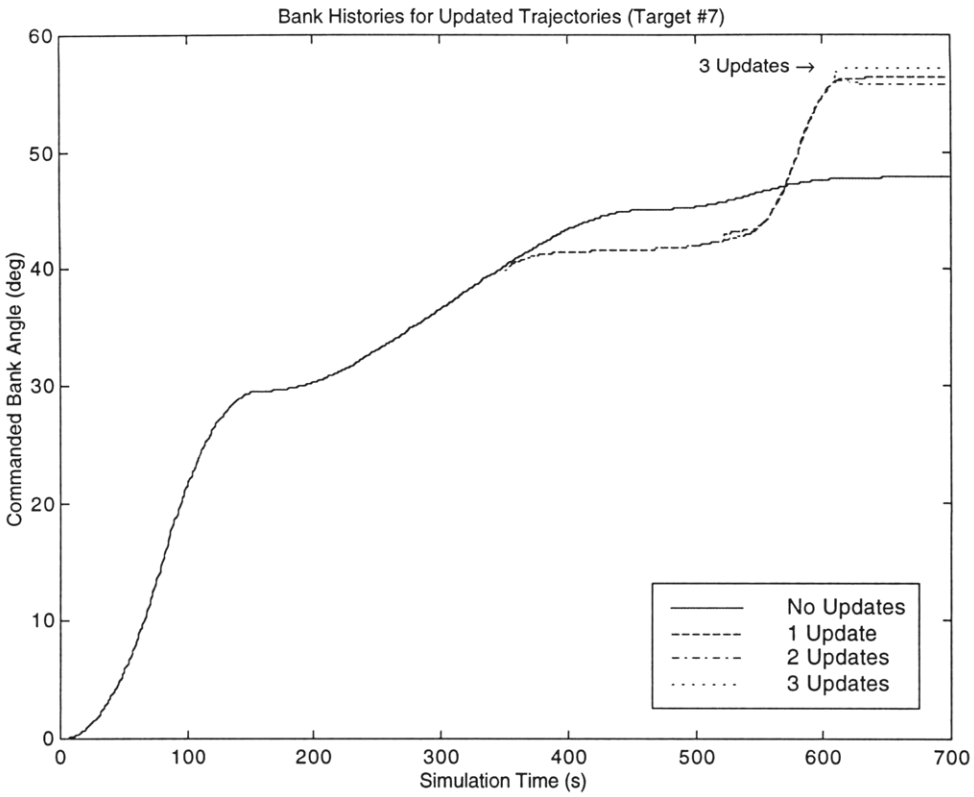


Figure 5.3.3-3: Bank angle profile with multiple trajectory updates for target #7

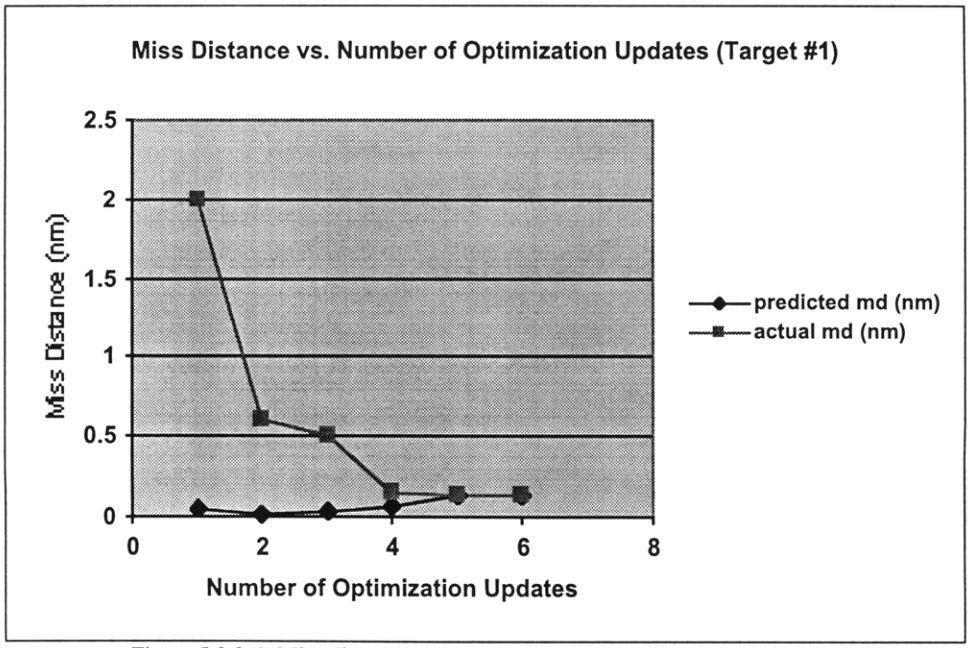


Figure 5.3.3-4: Miss distance vs. number of optimization runs for target #1

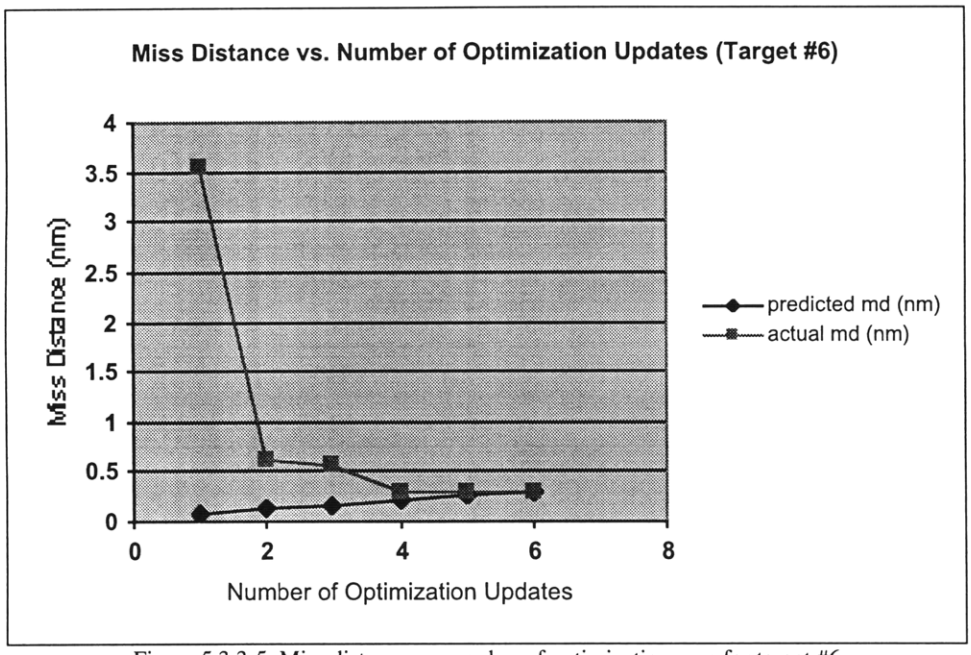


Figure 5.3.3-5: Miss distance vs. number of optimization runs for target #6

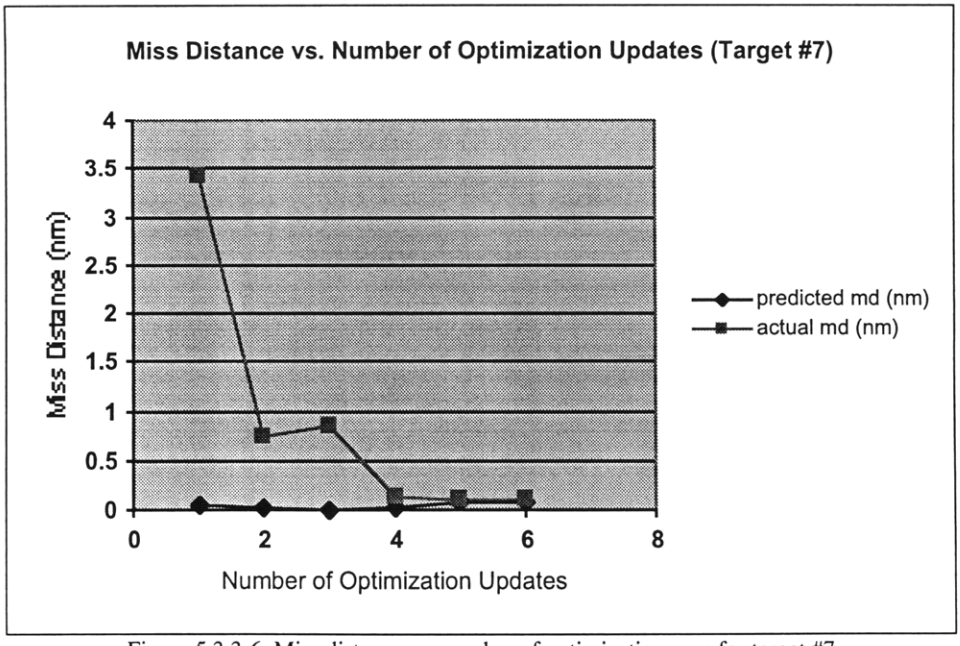


Figure 5.3.3-6: Miss distance vs. number of optimization runs for target #7

In all cases the miss distance decreased below 1 nautical mile after one mid-flight optimization update. As more updates were made the actual miss distance converged with the predicted miss distance. In all cases the actual miss distance stopped improving significantly when the optimization software was run more than 4 times. Under the current scheme for updating the trajectory, the 5^{th} and 6^{th} updates are performed very late in the trajectory, and the vehicle has limited ability to control its final position. If this technique were to be implemented in flight code, other strategies for deciding when to update the trajectory should be examined. Note that the optimization software was able to more accurately predict the miss distance as more updates were performed and the remaining trajectory length became shorter.

For a reentry trajectory approximately 700 seconds long with 4 trajectory updates using the current update strategy, the shortest time interval between collocation software runs would be approximately 90 seconds between the third and fourth run. The collocation software averaged approximately 16 seconds of CPU time on the computer used for these tests. The actual time required to run the software will vary depending on the speed of the particular flight computer used, but the Kistler flight computer performance is expected to be similar to the computer used for these tests. It therefore appears feasible to run the software in flight. This simple scheme to re-run the optimization software results in satisfactory vehicle performance with only four runs of the optimization software. If the optimization software were to run continuously on-board the vehicle, many more than four runs would be made during flight, and the software performance could potentially be improved.

5.4 The Dispersion Test Cases

In spite of attempts to model the system as accurately as possible, there will always be some modeling error. Various vehicle and environmental parameters will not be known exactly. If the collocation technique is to be used as part of a guidance algorithm, it must be robust with respect to these various sources of modeling error. The collocation approach was tested by deliberately including some error in the models for various system parameters. The effects of unknown perturbations in the vehicle mass, winds, and atmospheric density were all examined. Finally, the effects of varying the initial entry angle of the vehicle were examined.

5.4.1 Mass Dispersions

There is always some estimation error associated with the mass of the re-entry vehicle. Also, even though the current model assumes constant mass, the total vehicle mass decreases during flight due to ACS firings. To test the sensitivity of the collocation approach to unknown mass perturbations, a multiplier was applied to the vehicle mass value in the collocation software model. (e.g. a multiplier of 1.05 means that the collocation software assumes the vehicle is 5% heavier than it actually is). Figures 5.4.1-1 through 5.4.1-4 show the effects of unknown mass perturbations on the vehicle trajectory.

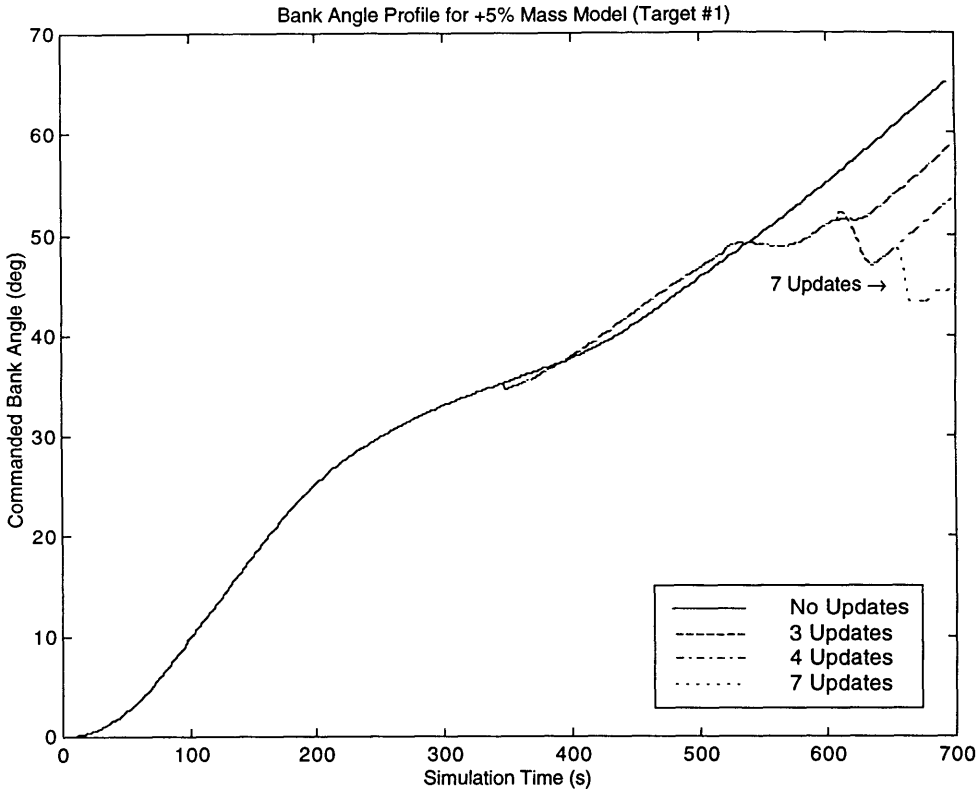


Figure 5.4.1-1: Bank angle profile for 1.05 mass multiplier (Target #1)

In the first case the actual mass of the vehicle was less than the mass assumed by the optimization software. The aerodynamic forces acting on the vehicle were therefore more significant compared to the momentum of the vehicle. Since the drag force acting on the vehicle had a more pronounced effect, the predicted vehicle final position tended to be short of the target. When the optimization software was re-run, the algorithm attempted to correct for this error by decreasing the bank angle, and increasing the component of vehicle lift in the downrange direction. Since in this case the model was not updated to account for this error, a similar but smaller correction was needed each time the vehicle control history was updated. It is possible that software could be used to compare the actual vehicle trajectory to the trajectory predicted by the collocation software, and estimate any necessary corrections to the vehicle model. This case used more ACS fuel than the non-dispersion case, because multiple corrections were needed to ensure that the vehicle hit the target. The vehicle used 46.2 lbm of ACS fuel when the

optimization software was run 7 times in flight, compared with 30.6 lbm when the collocation software used the correct vehicle mass estimate.

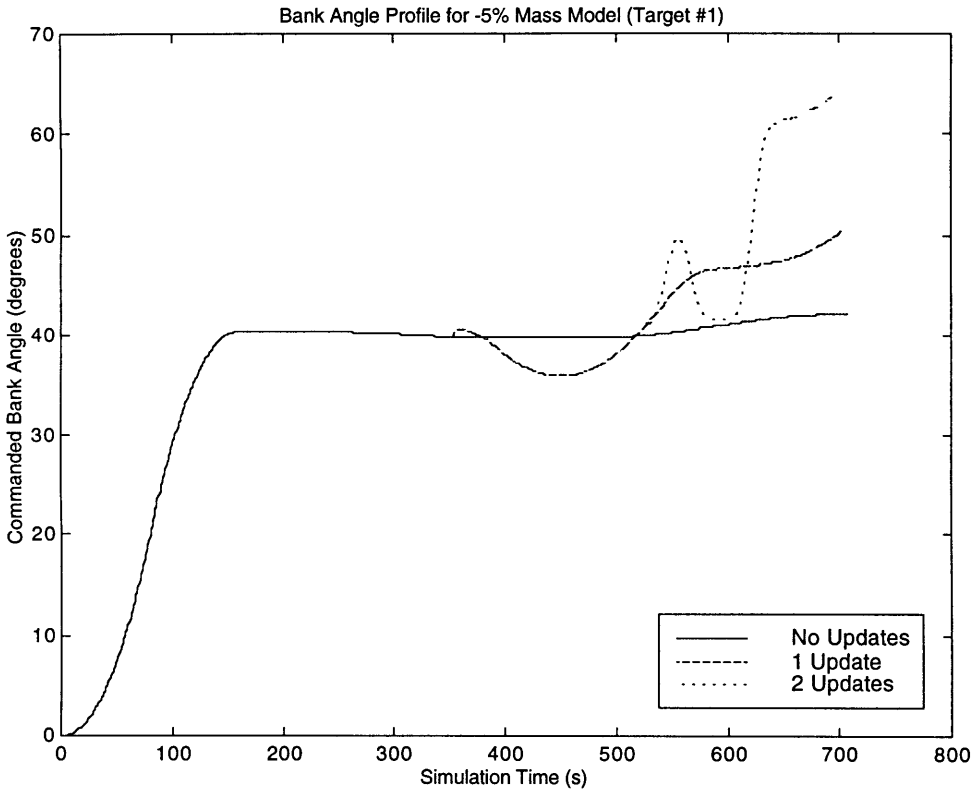


Figure 5.4.1-2: Bank angle profile for 0.95 mass multiplier (Target #1)

The next case considered the possibility that the vehicle mass was greater than the value assumed by the collocation software. Therefore the aerodynamic forces on the vehicle were less significant compared to the vehicle inertia. Since the aerodynamic drag was less effective at slowing the vehicle, a downrange final position error was introduced. The software corrects for this by decreasing the amount of lift in the downrange direction. In order to do this without introducing cross-range error, the vehicle must first bank in one direction, and then reverse. This is similar to the "lateral switching" technique used for the Apollo Command Module (described in section 2.1). Note that each additional trajectory update introduced another "switching" maneuver superimposed on the previously calculated control history. This case used substantially more fuel than the 1.05 mass multiplier case (111.2 lbm for 7 in-flight updates), because the sign of the

bank rate was frequently reversed to build the switching maneuvers. It clearly takes less fuel to correct for an upwind final position error than for a downwind error.

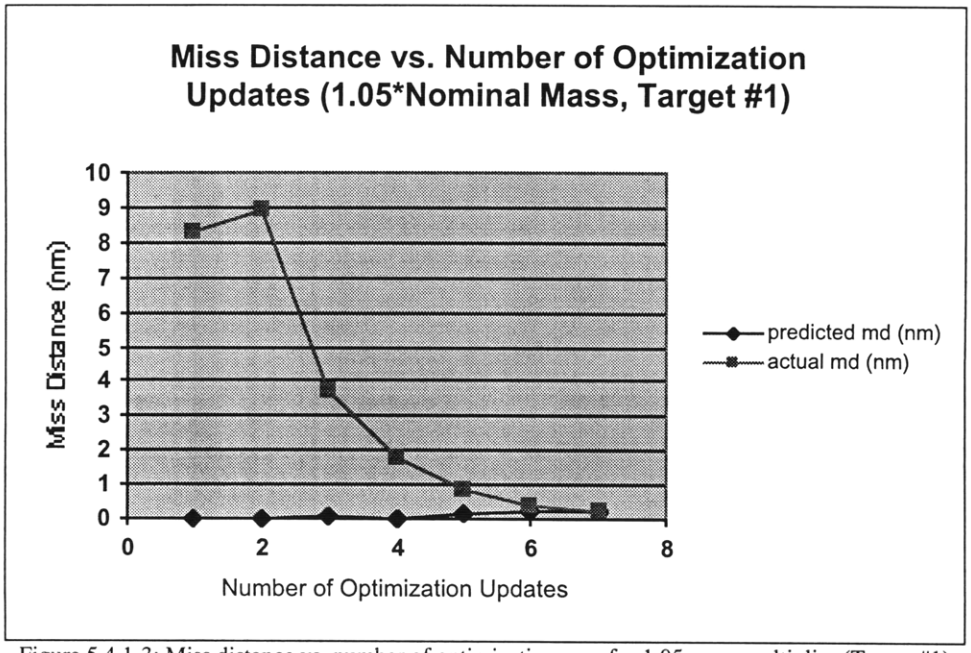


Figure 5.4.1-3: Miss distance vs. number of optimization runs for 1.05 mass multiplier (Target #1)

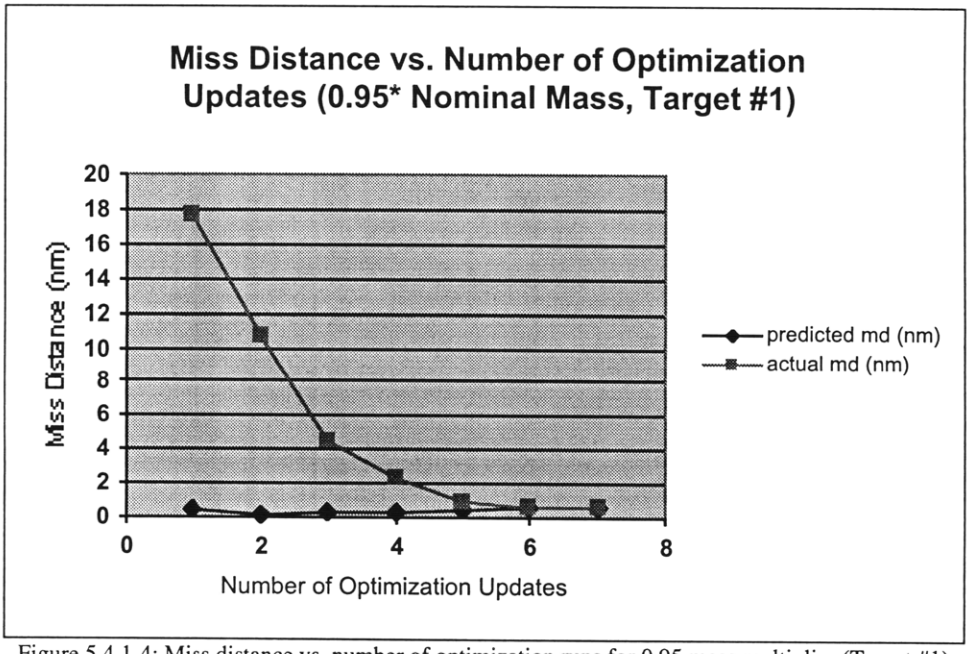


Figure 5.4.1-4: Miss distance vs. number of optimization runs for 0.95 mass multiplier (Target #1)

Note that in both cases the vehicle mass perturbation caused a significant miss distance when no in flight updates were performed. However, when the trajectory was

repeatedly updated in flight the actual and predicted miss distances converged well inside the 1 nautical mile specification. This suggests that continuously optimizing the vehicle trajectory in flight effectively deals with unknown mass dispersions.

5.4.2 Wind Dispersions

Unknown wind dispersions are not expected to be a major factor for this guidance algorithm, since any reasonable wind dispersion would be small compared with the magnitude of the vehicle air relative velocity. Experience with the K-1 vehicle has shown that unknown wind dispersions generally only have significant effects on the vehicle trajectory after the parachutes have deployed. This thesis only deals with the trajectory of the vehicle before parachute deployment. It is assumed that some form of guidance software performs the necessary calculations relating to the parachute descent and determines the desired final position of the vehicle at the drogue parachute deployment. Therefore it is expected that unknown wind dispersions will not have a large effect on the trajectory optimization procedures described in this study. In order to validate this assumption, the performance of the software was analyzed when large magnitude unknown headwinds and crosswinds are introduced into the system model. For a chosen wind magnitude, the direction of the headwind or crosswind was calculated and the resulting perturbation was added to the vehicle air-relative velocity vector:

$$\vec{V}^{air}_{RV_{ECI_{perturbed}}} = \vec{V}^{air}_{RV_{ECI_{actual}}} + \vec{V}^{air}_{wind\ perturbation_{ECI}} \tag{5.1}$$

A headwind (or tailwind) would be expected to introduce downrange error and increase the vehicle miss distance. For the purpose of this thesis the direction of the headwind was defined to be directly opposite the direction of the vehicle air-relative velocity vector. Note that as defined here the "headwind" has both vertical and horizontal components in the local vertical reference frame:

$$\vec{V}^{air}_{headwind\ perturbation_{ECI}} = V_{headwind} \frac{-\vec{V}^{air}_{RV_{ECI}}}{\left|\vec{V}^{air}_{RV_{ECI}}\right|} \tag{5.2}$$

The crosswind perturbation acts in the direction normal to the plane of the vehicle inertial position vector and the air-relative velocity vector:

$$\vec{V}^{calm\ air}_{crosswind\ perturbation_{ECI}} = V_{crosswind} \frac{\vec{r}_{RV_{ECI}} \times \vec{V}^{air}_{RV_{ECI}}}{\left|\vec{r}_{RV_{ECI}} \times \vec{V}^{air}_{RV_{ECI}}\right|} \tag{5.3}$$

First, a large constant headwind magnitude of 100 ft/s was assumed. The wind perturbation was introduced to the collocation software system model, and the resulting trajectories were implemented in the IVS simulation. A plot of actual and predicted miss distance versus number of optimization updates is shown in figure 5.4.2.1.

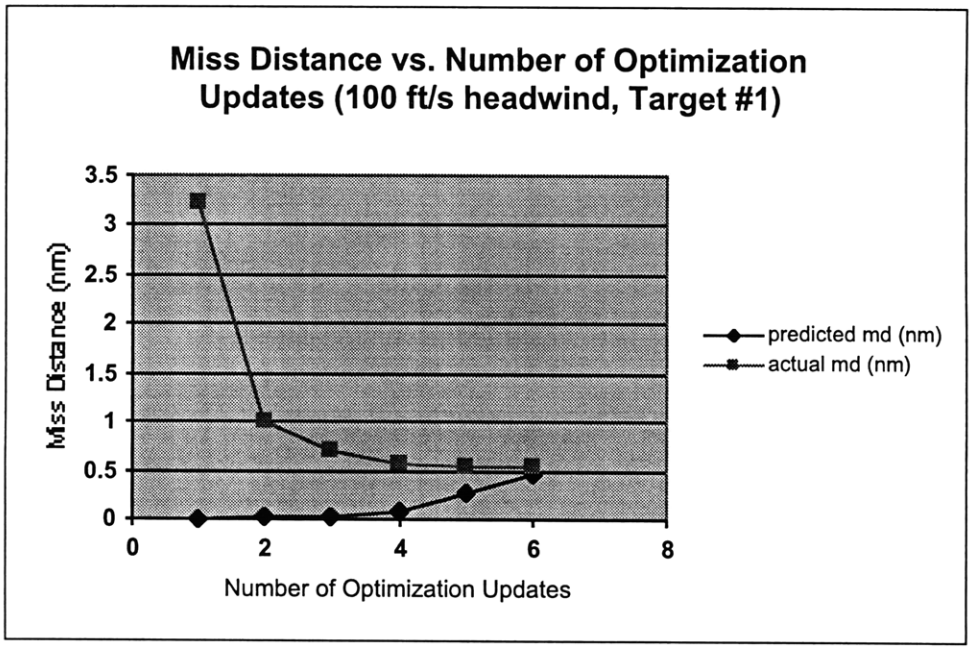


Figure 5.4.2-1: Miss distance vs. number of optimization runs for 100 ft/s headwind (Target #1)

Note that when no optimization updates are performed the final position miss distance is approximately 3 nautical miles, compared with a 1.6 nautical mile miss distance when no wind perturbation is applied. Clearly, an unknown headwind

component of even a reasonably large magnitude has little effect on the vehicle trajectory. As before, the actual miss distance quickly converged below the required 1 nautical mile accuracy when the control history was updated multiple times in flight. The same analysis was performed for a constant 100 ft/s right crosswind. The results are shown in figure 5.4.2-2.

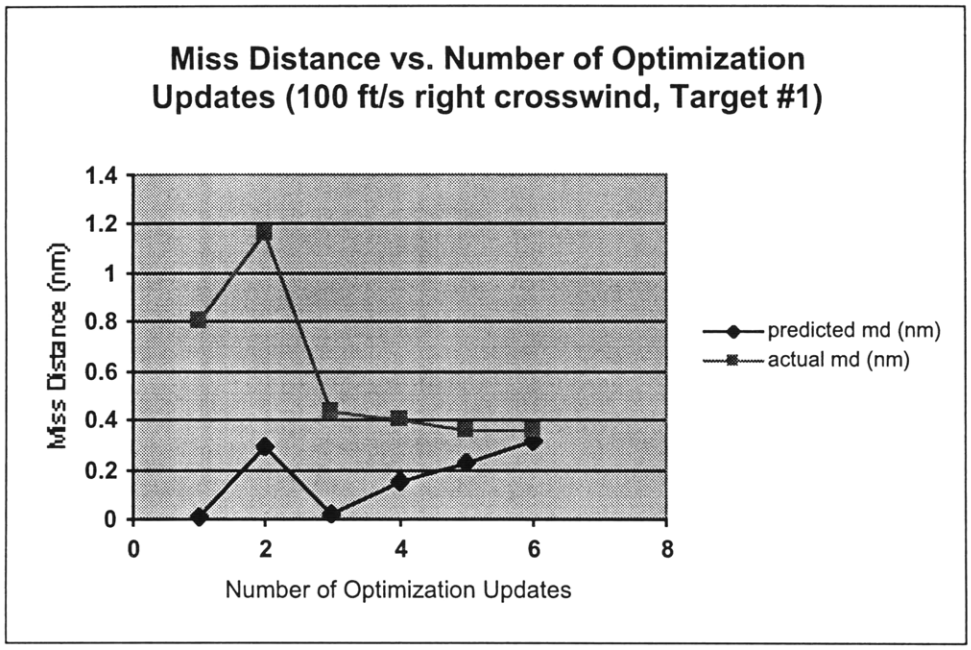


Figure 5.4.2-2: Miss distance vs. number of optimization runs for 100 ft/s crosswind (Target #1)

Again, the final position miss distance is small even when no optimization updates are performed (in this case even smaller than the no-wind miss distance). Therefore, unknown wind dispersions appear to have little effect on the vehicle trajectory. Figure 5.4.2-3 shows the results when a 100 ft/s right crosswind and 100 ft/s headwind are applied simultaneously. Since the effects of the wind dispersions are so small, no noticeable changes are created in the control history, and the bank angle plots are not included here. The total fuel usage is relatively unchanged as a result of the wind dispersions. The headwind case uses 34.0 lbm of fuel compared with 31.6 lbm for the nominal run, while the crosswind case uses 42.3 lbm and the combined case uses 41.2 lbm.

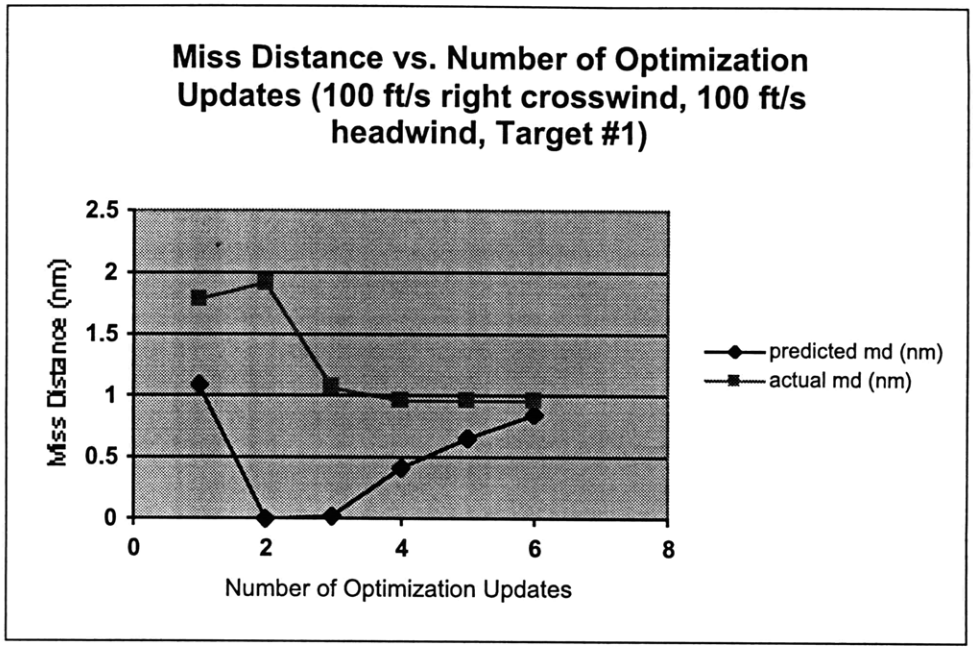


Figure 5.4.2-3: Miss distance vs. number of optimization runs for 100 ft/s crosswind and 10 ft/s headwind (Target #1)

Notice that the miss distance still converges below the required value. This series of tests indicated that unknown wind dispersions would not have a significant effect on the collocation guidance software. Therefore the effects of unknown wind dispersions were not considered further in this study.

5.4.3 Density Dispersions

Unknown density dispersions can have a significant effect on the vehicle trajectory. Several different approaches were used to demonstrate that the collocation guidance technique could successfully deal with errors in the density model. First, a constant multiplier was applied to density value obtained from the collocation software model. As with the mass dispersions, a multiplier greater than one implies that the collocation software assumes a density value higher than the real value. Density multipliers of 1.05 and 0.95 were applied to the US76 standard atmospheric model used in the collocation software. The results are shown in figures 5.4.3-1 through 5.4.3-4.

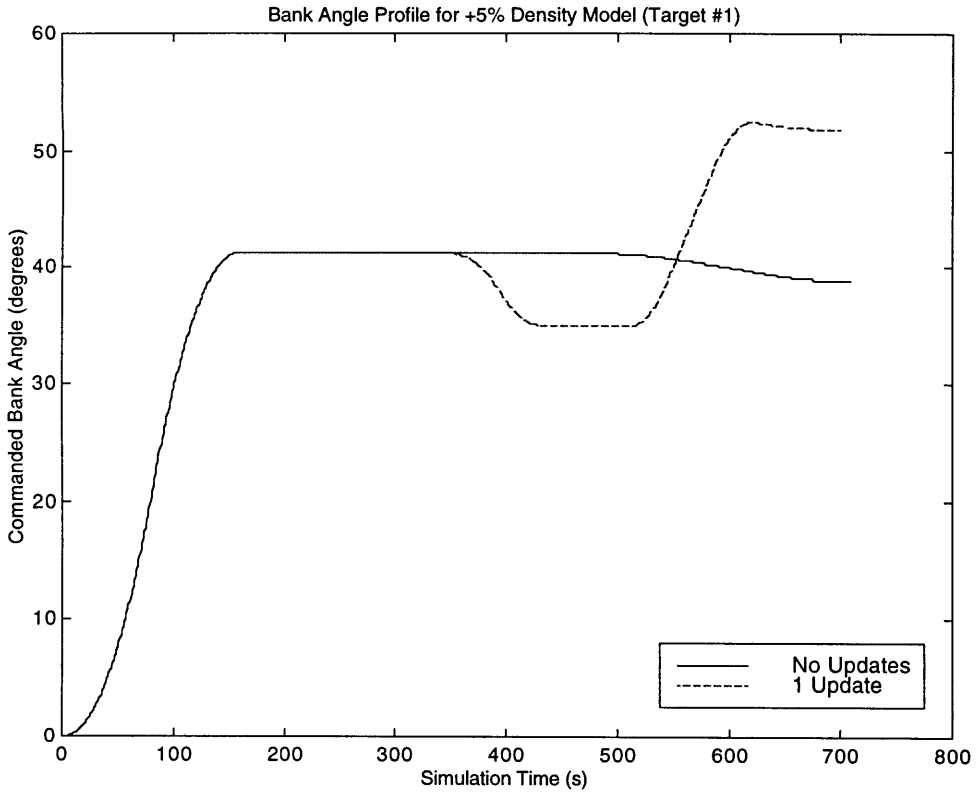


Figure 5.4.3-1: Bank angle profile for 1.05 density multiplier (Target #1)

In this case the atmospheric density in the IVS simulation was less than the value assumed by the collocation software. Therefore the aerodynamic forces on the vehicle were smaller than predicted. The lower drag created a significant downrange error, similar to the error produced when a mass multiplier smaller than one is used. When the trajectory was updated at mid-flight, the software attempted to correct for this downrange error by performing a switching maneuver similar to that described in section 5.4.1. Note that the magnitude of the switching error is approximately the same as that for the 0.95 mass multiplier case (Figure 5.4.1-2). This is as expected, since (neglecting the gravitational force) the vehicle accelerations are directly proportional to the dynamic pressure, and inversely proportional to the vehicle mass.

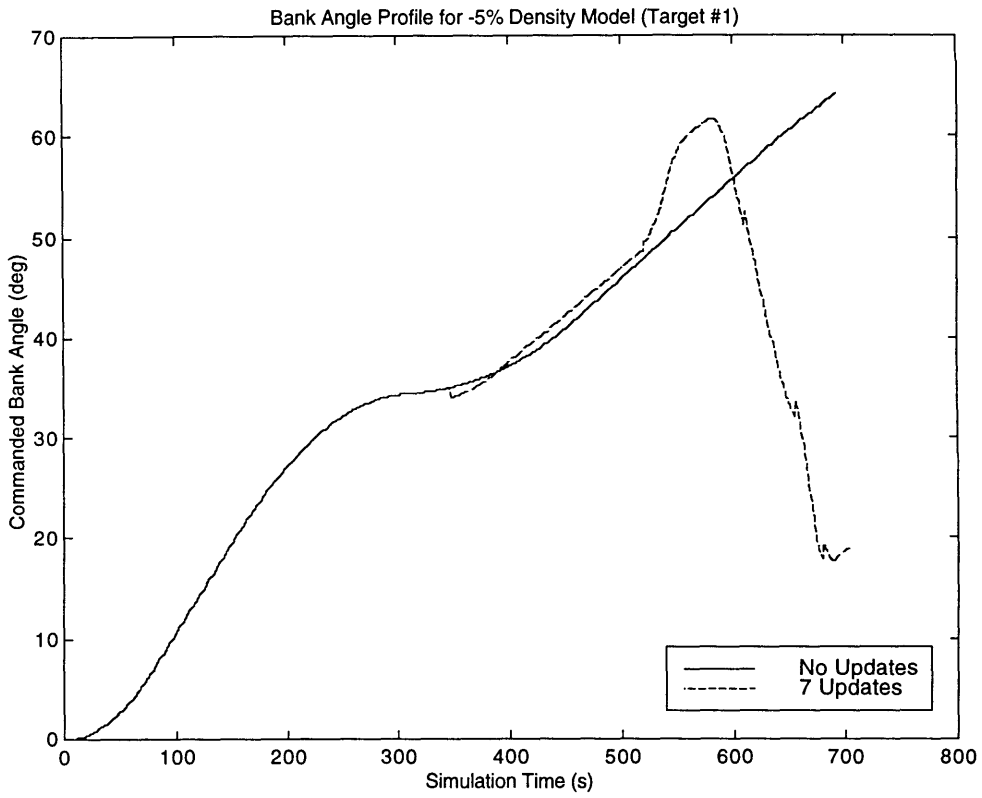


Figure 5.4.3-2: Bank angle profile for 0.95 density multiplier (Target #1)

In this case the actual density was greater than that assumed by the collocation software. The aerodynamic forces were therefore also greater and the vehicle tended to be short of the target. The software corrected for this error by lowering the bank angle, which increased the component of vehicle lift in the downrange direction and reduced the final position error. Th effect on the control history was similar to the case when a mass multiplier greater than one was applied.

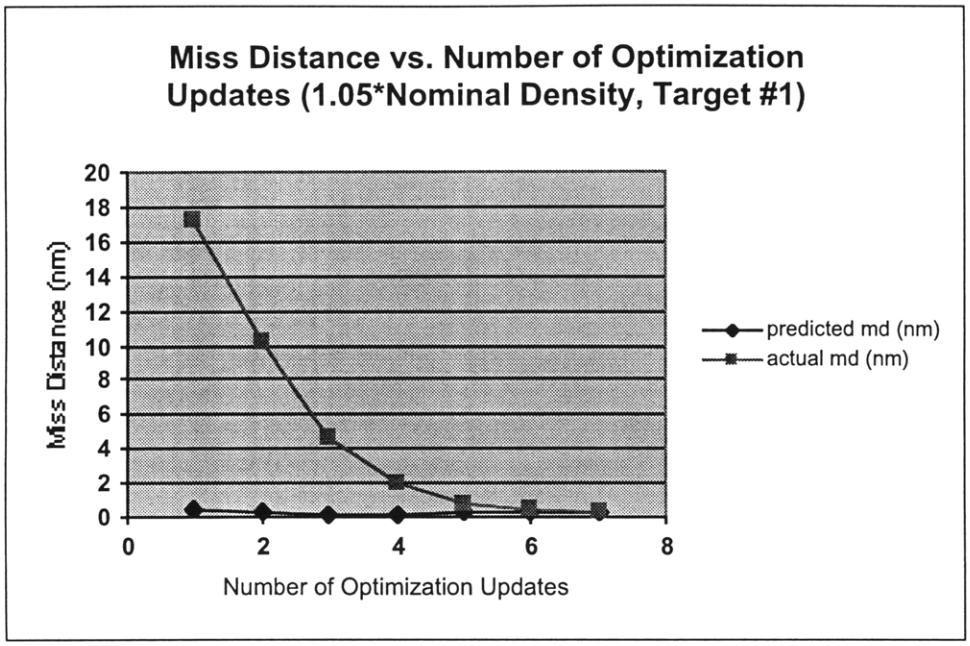


Figure 5.4.3-3: Miss distance vs. number of optimization runs for 1.05 density multiplier

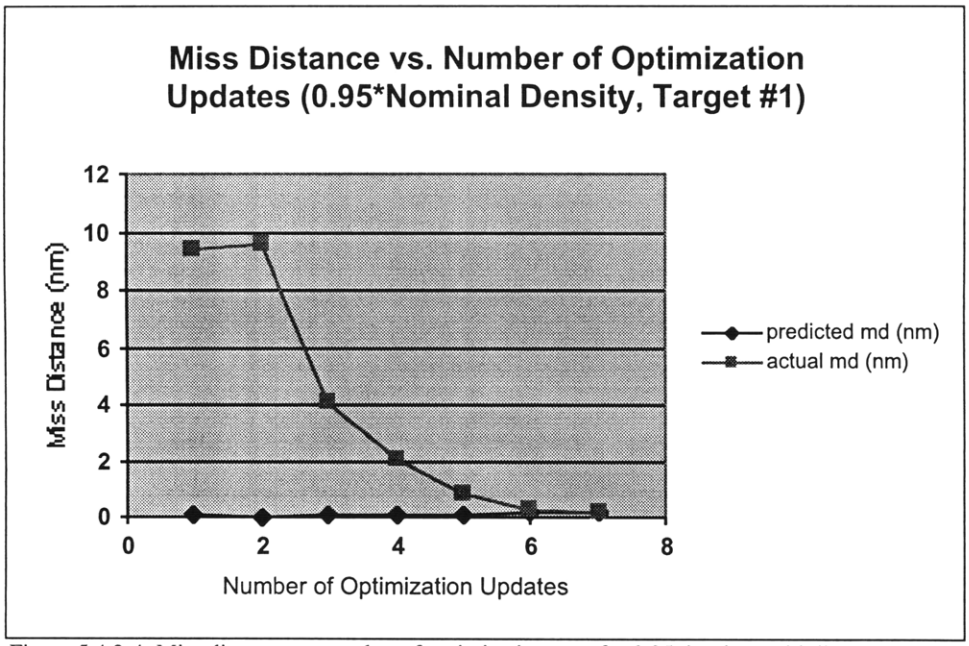


Figure 5.4.3-4: Miss distance vs. number of optimization runs for 0.95 density multiplier (Target #1)

The plots show that a substantial "open loop" final position error was introduced by the density multipliers. In both cases the vehicle successfully achieved the specified

final position accuracy when the trajectory was repeatedly optimized in flight. The 1.05 density multiplier run used 93.2 lbm of ACS fuel when 7 optimization updates were performed in flight. The 0.95 density multiplier run only required 6 optimization updates to converge, and used 75.6 lbm of ACS fuel. In both cases the amount of fuel required increased substantially over the 31.6 lbm required for the nominal case. As was discovered with the mass dispersion cases, it seems to required significantly less fuel to correct for upwind position error.

In the cases described above the density model was perturbed by a constant ratio. This represents a "worst case" scenario. In reality, the actual atmospheric density deviation from normal would vary as a function of position. This was modeled using the GRAM95 atmospheric model. The GRAM95 model uses the Global Upper Air Climatic Atlas (GUACA) compiled by the National Oceanic and Atmospheric Administration and the United States Naval Oceanography Command. The model is capable of producing realistic density dispersions based on more than a decade of real atmospheric data. The nominal vehicle trajectory and a random number seed were input into the GRAM95 model. Using the trajectory position coordinates, the model returned the ratio of dispersed versus standard density at a number of altitudes. In reality the dispersed density profile would modify the vehicle trajectory, and the density altitude profile along the modified trajectory would differ very slightly from the profile along the nominal trajectory. Assuming that the atmospheric density does not vary significantly with small changes in latitude or longitude, this effect may be ignored. A typical random density dispersion is shown in figure 5.4.3-5.

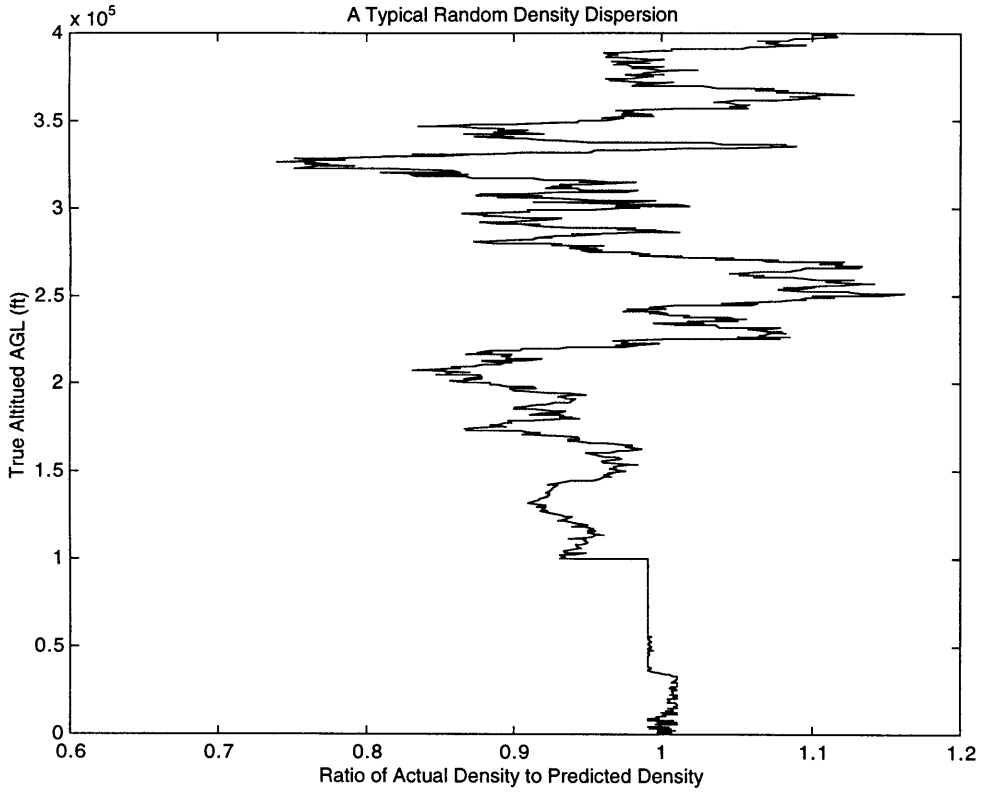


Figure 5.4.3-5: A typical random density dispersion from the GRAM95 model

As before, the density perturbations were applied to the collocation model. A subroutine was introduced to calculate and apply the density perturbations. The resulting control histories and miss distances are shown in figure 5.4.3-6 and figure 5.4.3-7. Note that the collocation software was forced to make large corrections each time it is run. The final position miss distance barely converged inside the one nautical mile limit, and 125.6 lbm of ACS fuel was required to achieve this degree of accuracy. These figures should improve if the optimization software is run continuously in flight. Also, the collocation software does not have the ability to compare the predicted aerodynamic forces to the actual forces. The nominal Kistler guidance scheme uses an estimator to calculate the predicted aerodynamic forces on the vehicle, and compares these predictions to the actual forces measured using accelerometers. Scale factors are then applied to the predicted aerodynamic forces so that they more closely track the measured forces. A similar approach, using similar or even identical estimation algorithms, could be applied

to the collocation guidance software. While the collocation guidance strategy just barely meets the required accuracy in this case, there are several potential improvements that could be made if flight code were to be developed. These potential improvements will be briefly summarized in section 6.2. This series of test runs has showed that the collocation guidance strategy can compensate for trajectory deviations caused by unknown errors in the atmospheric model.

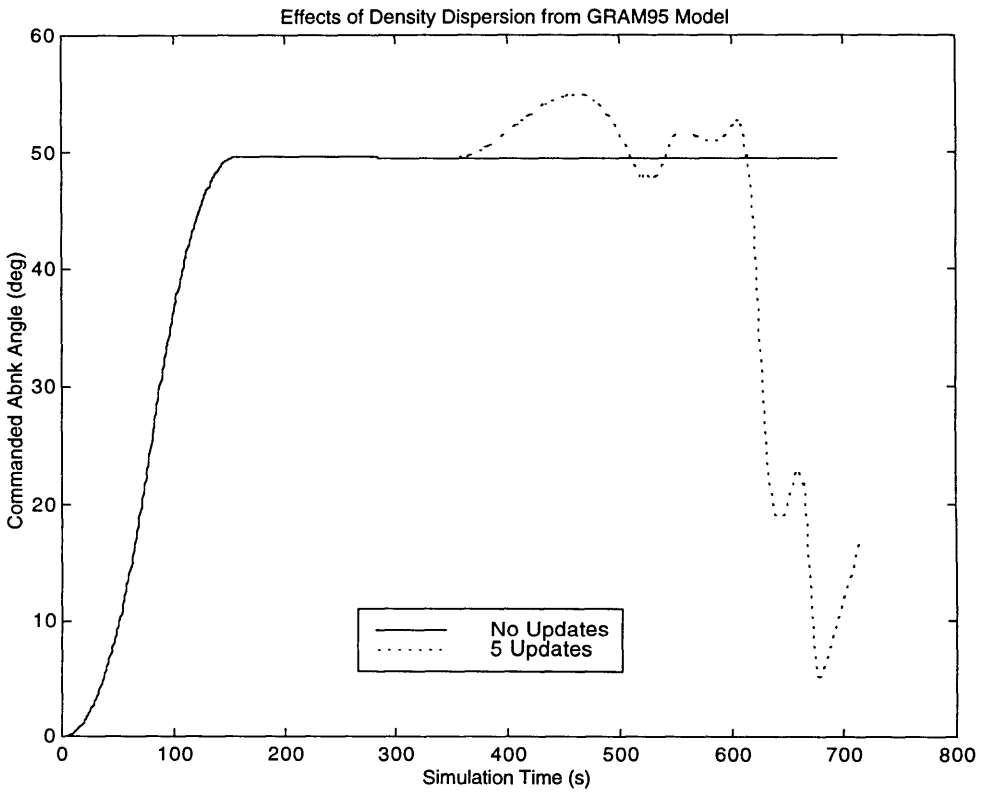


Figure 5.4.3-6: Control histories for GRAM95 density dispersion model case (Target #1)

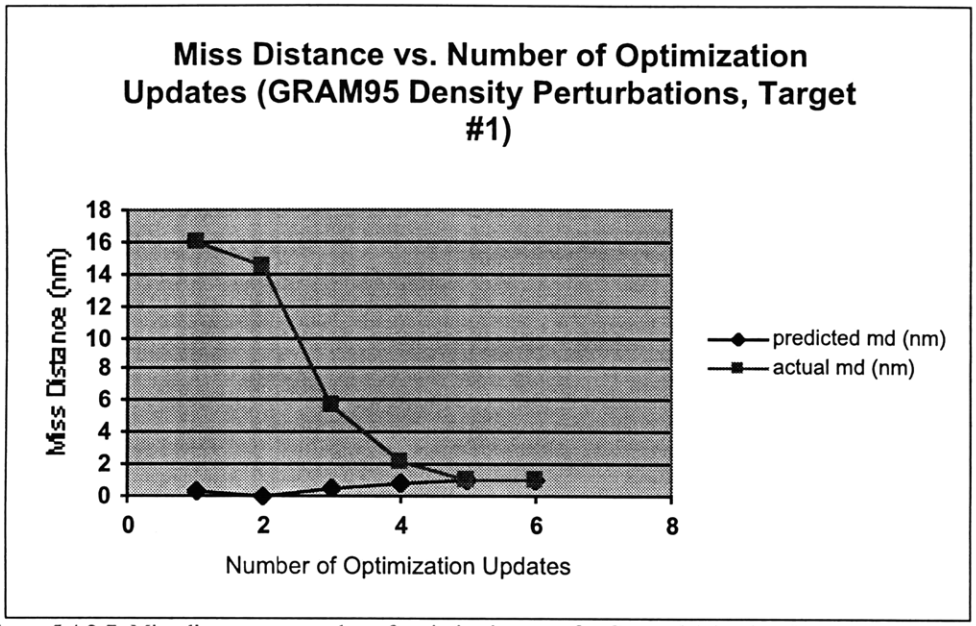


Figure 5.4.3-7: Miss distance vs. number of optimization runs for GRAM95 density dispersion (Target #1)

5.4.4 Entry Angle Dispersions

The re-entry vehicle entry angle is defined as the angle between the inertial velocity vector and the local horizontal plane at EI:

$$\xi = \cos^{-1}\left(\frac{\vec{V}^{ECI}_{RV_{ECI}} \cdot \vec{r}_{RV_{ECI}}}{\left\|\vec{V}^{ECI}_{RV_{ECI}}\right\|\left\|\vec{r}_{RV_{ECI}}\right\|}\right) - 90^{\circ} \tag{5.4}$$

The re-entry vehicle trajectory is very sensitive to the entry angle. Therefore it is important to consider the effects of varying the vehicle entry angle. Assuming the vehicle inertial position and the magnitude of the inertial velocity remain constant, the entry angle may be changed by performing a simple rotation of the vehicle. In order to perturb the entry angle, a simple rotation was performed about the vehicle y-axis. Several preliminary runs were made to determine the sensitivity of the vehicle trajectory to entry angle. It was determined that entry angle variations on the order of hundredths of a degree were significant and could result in the vehicle's inability to reach the desired target. Therefore the vehicle entry angle errors on the order of +/- 0.01 degree were

considered. First, a positive entry angle error was considered. The initial inertial velocity vector in the collocation software was altered to raise the entry angle by 0.01 degree. The resulting control history and miss distance values are shown in figures 5.4.4-1 and 5.4.4-2.

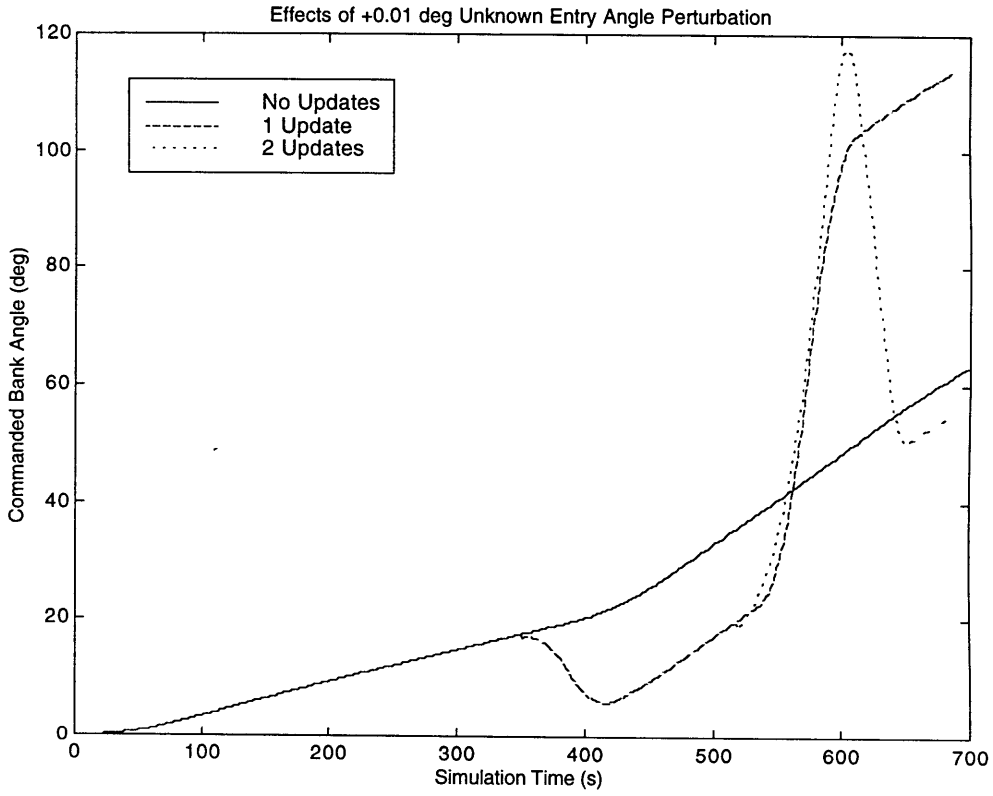


Figure 5.4.4-1: Control Histories for +0.01 degree entry angle dispersion (Target #1)

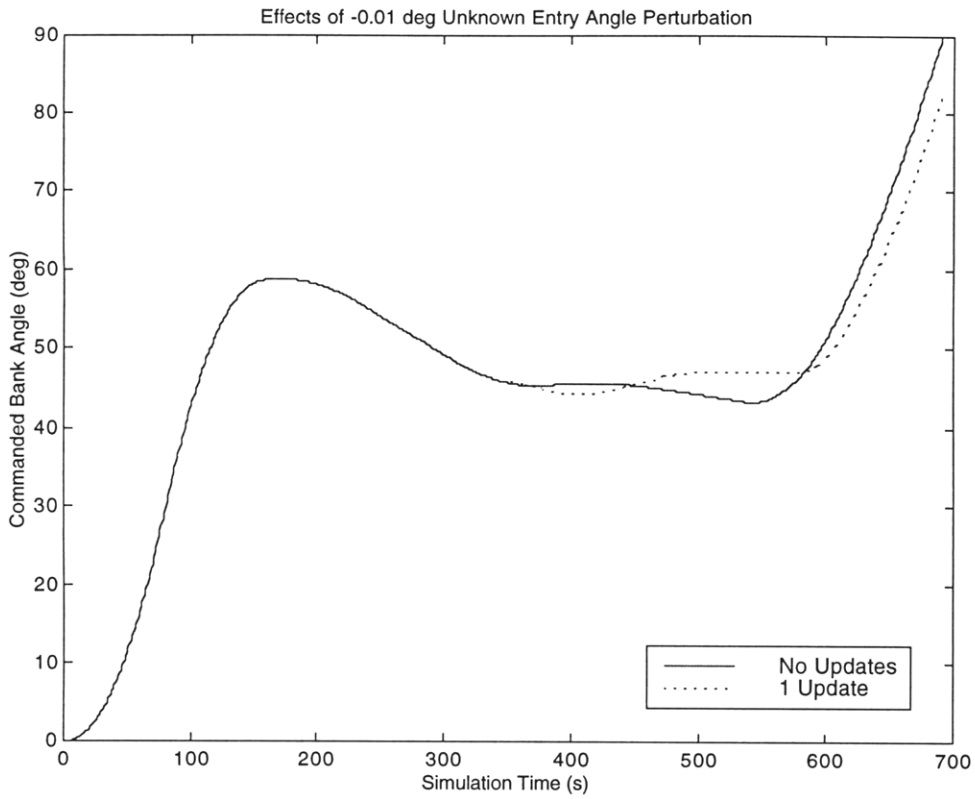


Figure 5.4.4-2: Control Histories for -0.01 degree entry angle dispersion (Target #1)

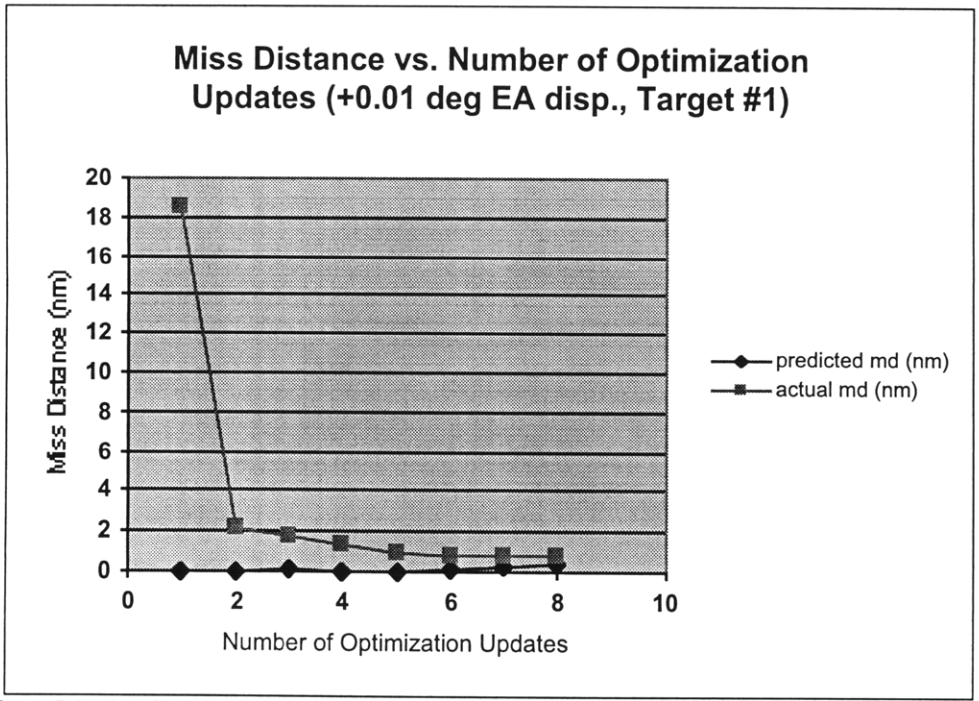


Figure 5.4.4-3: Miss distance vs. number of optimization runs for +0.01 entry angle dispersion (Target #1)

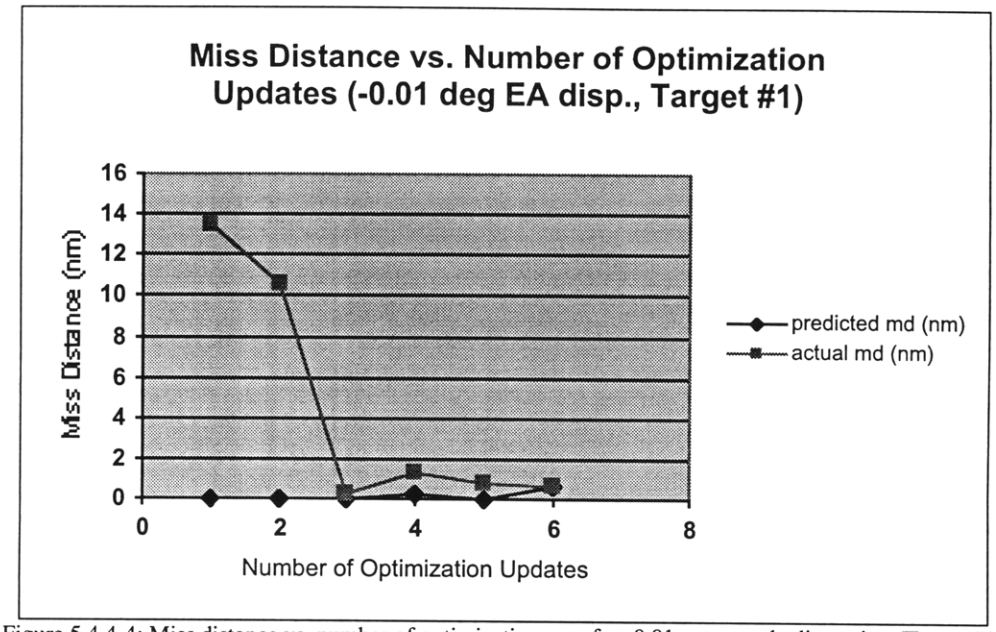


Figure 5.4.4-4: Miss distance vs. number of optimization runs for -0.01 entry angle dispersion (Target #1)

Note that in both cases repeatedly running the optimization software reduces the final position miss distance to acceptable values. In both cases, the amount of ACS fuel necessary to reach the target increased substantially (84.3 lbm for the +0.01 entry angle dispersion, and 125.4 lbm for the –0.01 degree entry angle dispersion). These results indicate that the collocation guidance software can effectively correct for trajectory deviations cause by entry angle errors on the order of +/- 0.01 degrees.

5.5 Chapter Summary

This chapter discussed the testing and evaluation of the trajectory optimization software. The software was evaluated using the Kistler K-1 IVS simulation. A variety of test cases were considered which analyzed various aspects of the collocation scheme setup and the effects of unknown system dispersions. It was shown that a collocation scheme using as few as ten nodes and updated multiple times in flight could meet the guidance requirements for the Kistler OV. The g-loading constraint was successfully enforced for all of the chosen test cases. Some selected results from the various test runs

are summarized in table 5.5-1. All results are for target #1, and all dispersions are applied to the collocation software model (e.g. "Mass *1.05" implies that a gain of 1.05 was applied to the value used for the vehicle in the collocation software system model.) For the nominal case, the collocation guidance strategy appeared to offer a significant fuel savings. Other fuel consumption results were also quite reasonable. In all cases the final position error converged below 1 nautical mile. Further improvements to the collocation guidance technique that may further reduce miss distances are briefly discussed in section 6.2. In general, the collocation guidance strategy appears to be the most sensitive to the entry angle, moderately sensitive to the density profile and vehicle mass estimate, and less sensitive to unknown wind dispersions. The test cases demonstrated that the collocation method is a feasible approach to solving the re-entry vehicle problem.

Dispersion Quantity	# of Optimization Runs	Predicted Miss Distance (nm)	Actual Miss Distance (nm)	ACS Fuel Usage (lbm)
No Dispersion	6	0.133	0.138	31.6
Mass * 1.05	7	0.217	0.248	46.2
Mass * 0.95	7	0.585	0.601	111.2
100 ft/s Crosswind	6	0.467	0.554	42.3
100 ft/s Headwind	6	0.312	0.361	34.0
100 ft/s Crosswind and 100 ft/s Headwind	6	0.841	0.960	41.2
Density * 1.05	8	0.343	0.353	93.2
Density * 0.95	7	0.181	0.201	75.6
GRAM Density Dispersion	6	0.998	0.989	125.6
+0.01 degree Entry Angle	8	0.387	0.797	84.3
-0.01 degree Entry Angle	6	0.620	0.681	125.4

Table 5.5-1: Summary of results for selected dispersion cases (Target #1)

NOTE- THIS TABLE CONTAINS VALUES RELATING TO THE TECHNIQUES DESCRIBED IN THIS THESIS ONLY, AND IS NOT REPRESENTATIVE OF ACTUAL KISTLER FLIGHT CODE PERFORMANCE.

Chapter 6: Conclusions and Recommendations

The goal of this thesis was to study the feasibility of using a collocation technique for the trajectory planning and guidance of a re-entry vehicle. The conclusions of this study and recommendations for future work are presented in this chapter.

6.1 Conclusions

The main objective of this thesis was to demonstrate the feasibility of applying the direct collocation with non-linear programming (DCNLP) technique to the fixed-trim re-entry vehicle guidance problem. The design requirements were that the vehicle return to within one nautical mile of a designated landing position starting from an altitude of 400,000 feet and orbital velocity. The problem was introduced and the necessary background information was described in the first two chapters of this thesis. The third and fourth chapters described the necessary system model and the development of the collocation software. The fifth chapter described the methods used to evaluate the collocation guidance software. The results show that such a method is a reasonable approach to re-entry vehicle guidance. Several different targets and various unknown dispersion cases were considered. Neglecting any unplanned disturbances that may occur after parachute deployment, the final position accuracy specification was met for all of the cases considered in this study. Based upon this work, it appears feasible to use the DCNLP technique as part of an on-line guidance strategy.

6.2 Recommendations for Future Work

There are numerous opportunities for future research pertaining to this topic. Several steps are necessary to develop the techniques presented in this thesis into flight guidance software. First, the collocation software would have to be fully integrated into the re-entry vehicle control software and run on the vehicle flight computer. For the purposes

of this thesis, the vehicle simulation was paused and the actual collocation software was run off-line. Also, data from the vehicle sensors could be used to correct for deviations in the environmental data. The current Kistler guidance software uses accelerometers to compare the actual current aerodynamic forces to the estimated forces. Corrections are then applied to the future estimated aerodynamic forces. These gains were not used in this thesis, and could potentially enhance the performance of the collocation software. Another potential addition to the software is the thermal loading constraint. Using the atmospheric and vehicle models, the heat load applied to the vehicle could be calculated as a function of the current vehicle state, and introduced as an inequality constraint in the non-linear programming problem. One benefit of the collocation approach is that such additional constraints may be added with minimal difficulty.

Another area for future work is to develop a more efficient scheme for running the optimization software in flight. The technique used in this thesis was to re-run the software when a predetermined portion of the trajectory had passed, regardless of the deviation from the desired trajectory. A possible alternate technique would be to measure the vehicle deviation from the previously predicted trajectory (which is conveniently returned by the collocation software), and re-run the software when the error reaches a certain threshold. This would eliminate the need to re-run the software if no trajectory correction were necessary.

An additional likely use of the collocation software is for trajectory planning. Collocation techniques are traditionally used for trajectory planning rather than guidance, and the software developed for this thesis could be modified to determine the optimal entry interface conditions for a given target position. Instead of applying the entire EI state as a constraint, the EI state could be constrained to the appropriate altitude (400,000 ft) and an obtainable inertial velocity. The software could then calculate the optimal EI position and re-entry trajectory. The additional degrees of freedom would likely increase the time required to run the software, but such trajectory planning could be done off-line prior to re-entry.

6.3 Chapter Summary

This final chapter summarized the conclusions of this study. The thesis demonstrated that the DCNLP technique could feasibly be applied to the fixed-trim re-entry vehicle guidance problem. The thesis concluded with recommendations for future work.

Works Cited

[1] Adby, P.R., and Dempster, M.A.H., Introduction to Optimization Methods, Chapman and Hall (London: 1974).

[2] Anderson, John D., Introduction to Flight, McGraw-Hill, Inc. (New York: 1989).

[3] Aoki, Masanao, Introduction to Optimization Techniques, The Macmillan Co. (New York: 1971).

[4] Bate, Roger R., et al, Fundamentals of Astrodynamics, Dover Publications, Inc. (New York: 1971).

[5] Bladt, J., Trajectory Planning for a Fixed Trim Re-entry Vehicle, M.S. Thesis, Massachusetts Institute of Technology, Dec 1986.

[6] Cliff, E.M., et al, "Fixed-Trim Re-Entry Guidance Analysis," *Journal of Guidance, Control, and Dynamics*, Vol. 5, No. 8, 1981, pp. 558-563

[7] Conway, B.A., and Larson, K.M., "Collocation versus Differential Inclusion in Direct Optimization," *Journal of Guidance, Control, and Dynamics*, Vol. 21, No. 5, 1998, pp.

[8] Coverstone-Carroll, V. L., and Wilkey, N. M., "Optimal Control of a Satellite-Robot System Using Direct Collocation with Non-Linear Programming", *Acta Astronautica*, Vol. 36, No. 3, 1995, pp. 149-162.

[9] Enright, Paul J., and Conway, Bruce A., "Optimal Spacecraft Trajectories Using Collocation and Nonlinear Programming," *Journal of Guidance, Control, and Dynamics*, Vol. 14, No. 5, 1991, pp.981-985.

[10] Gracey, C., et al, "Fixed-Trim Re-entry Guidance Analysis," Journal of Guidance, Navigation, and Control, Vol. 5, No. ?, ?, pp.558-563.

[11] Hargraves, C.R., and Paris, S. W., "Direct Trajectory Optimization Using Nonlinear Programming and Collocation," *Journal of Guidance, Control, and Dynamics*, Vol. 10, No. 4, 1987, pp.338-342.

[12] Hargraves, C.R., et al, "OTIS Past, Present, and Future," AIAA Paper 92-4530, AIAA Guidance, Navigation, and Control Conference, Hilton Head Island, SC, August 10-12, 1992.

[13] Hocking, Leslie M., Optimal Control: An Introduction to the Theory with Applications, Clarendon Press (Oxford: 1991).

[14] Hull, David G., "Conversion of Optimal Control Problems into Parameter Optimization Problems," *Journal of Guidance, Control, and Dynamics*, Vol. 20, No. 1, 1997 pp.57-60.

[15] Johnson, Madeline S., and Giller, Donald R., Apollo Guidance, Navigation, and Control, Volume V: The Software Effort, Charles Stark Draper Laboratory (Cambridge: 1971).

[16] Kumar, R.R., and Seywald, H., "Should Controls Be Eliminated While Solving Optimal Control Problems via Direct Methods?," *Journal of Guidance, Control, and Dynamics*, Vol. 19, No. 2, 1996, pp.59-68.

[17] Lin, Ching-Fang, Modern Navigation, Guidance, and Control Processing, Prentice Hall (New Jersey: 1991).

[18] Luenberger, David G., Introduction to Linear and Nonlinear Programming, Addison-Wesley Publishing Company (Reading, MA: 1965).

[19] Mangasarian, Olvi L., Nonlinear Programming, McGraw Hill (New York: 1969).

[20] Masanao, Aoki, Introduction to Optimization Techniques, Macmillan Co. (New York: 1972).

[21] McCormick, Garth P., Nonlinear Programming, John Wiley & Sons (New York: 1983).

[22] McLean, Donald, Automatic Flight Control Systems, Prentice Hall (New York:1990).

[23] Murtagh, Bruce, and Saunders, Michael, MINOS 5.4 User's Guide, Stanford University (Stanford, CA: 1995).

[24] Ostrander, Damon D., An Engineer's Approach to Quaternions.

[25] Regan, Frank, and Anandakrishnan, Satya, Dynamics of Atmospheric Reentry, AIAA inc. (Washington, DC: 1993).

[26] Rubenstein, David, and Carter, David, "Attitude Control System Design for Return of the Kistler K1 Orbital Vehicle," AIAA Paper 98-4420. Guidance, Navigation, and Control Conference and Exhibit, Boston, MA, August 10-12, 1998.

[27] Rubenstein, David, and Melton, Robert, "Multiple Rigid Body Reorientation Using Relative Motion with Constrained Final System Configuration," AIAA Paper 95-415. AAS/AIAA Astrodynamics Specialist Conference, Halifax, Nova Scotia, Canada, August 14-17, 1995.

[28] Seywald, Hans, "Trajectory Optimization Based on Differential Inclusion," *Journal of Guidance, Control, and Dynamics*, Vol. 17, No. 3, 1994, pp. 480-487.

[29] Strang, Gilbert, Introduction to Applied Mathematics, Wellesley-Cambridge Press (Wellesley, MA: 1986).

[30] Sultan, Alan, Linear Programming, Academic Press Inc. (Boston: 1993).

[31] Taha, Hamdy A., Operations Research: An Introduction, MacMillan Publishing Co., Inc. (New York: 1971).

[32] Turner, Peter R., Guide to Numerical Analysis, Macmillan (London: 1989).

[33] Vandergraft, James S., Introduction to Numerical Computations, Academic Press (New York: 1983).

[34] Vihn, Nguyen X., et al, Hypersonic and Planetary Entry Mechanics, The University of Michigan Press (Ann Arbor: 1980).

[35] Vihn, Nguyen X., Optimal Trajectories in Atmospheric Flight, Elsevier Scientific Publishing Co. (New York: 1981).

[36] Zangwill, Willard I., Nonlinear Programming: A Unified Approach, Prentice-Hell Inc. (Englewood Cliffs, N.J.: 1969).