The Pennsylvania State University

The Graduate School

College of Engineering

VERTICAL TAKEOFF VERTICAL LANDING SPACECRAFT TRAJECTORY OPTIMIZATION VIA DIRECT COLLOCATION AND NONLINEAR PROGRAMMING

A Thesis in

Aerospace Engineering

by

Michael Joseph Policelli

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

August 2014

The thesis of Michael Joseph Policelli was reviewed and approved* by the following:

David B. Spencer
Professor of Aerospace Engineering
Thesis Advisor

Joseph F. Horn
Professor of Aerospace Engineering

George A. Lesieutre
Professor of Aerospace Engineering
Head of the Department of Aerospace Engineering

*Signatures are on file in the Graduate School

ABSTRACT

Rocket-powered translational Vertical Takeoff Vertical Landing (VTVL) maneuvers are a promising lander spacecraft mobility method as compared with or in addition to rovers for certain mission profiles. Such a VTVL vehicle would take off vertically under rocket propulsion, translate a specified horizontal distance, and vertically return softly to the surface.

Previous literature suggested that the propellant required to perform such a maneuver could be estimated via an impulsive-ballistic trajectory using the "ideal" rocket equation. This analysis was found to be inadequate. Any feasible trajectories will always require additional propellant to compensate for gravity losses while lifting off and landing. Additionally, there is an asymmetry between the takeoff and landing phases of the maneuver due to the propellant mass used over the course of the flight. Lastly, various potential spacecraft propulsion system architectures impose a number of possible constraints on the allowable path and boundary conditions.

An adaptable Optimal Control Problem (OCP) was developed instead to model the basic dynamics and required propellant consumption of various VTVL spacecraft trajectory profiles for a range of constraints, spacecraft parameters, and translation distances. The model was discretized into a Nonlinear Programming (NLP) problem, and a Direct Collocation (DC) method utilizing implicit Simpson-Hermite integration was used to ensure the feasibility of solutions with sufficient accuracy.

MATLAB's Nonlinear Programming *fmincon* routine with the sequential quadratic programming solver was able to converge on the optimal VTVL trajectory in terms of minimizing the required propellant use within the spacecraft and mission constraints. Trades were performed to determine the impact of various parameters on the required propellant including thrust to initial weight

ratios, propellant specific impulse, the allowable range and angular rate of change of the spacecraft thrust vector, translation distances, maximum altitude, flight times, and boundary conditions.

The VTVL trajectory optimization model developed was found to be robust and able to handle a wide range of various spacecraft and mission parameters. Results were compared against the required propellant use and nominal time of flight determined via the ballistic-impulse burn-coast-burn analysis. For the finite model developed herein, the required propellant use and optimal flight times exceeded the ideal impulsive case by 5-30% depending on the specific spacecraft and mission parameters and constraints implemented. These results can help guide future mission planners in deciding whether to utilize VTVL as a mobility method.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I owe a debt of gratitude to Dr. David Spencer for his guidance and close assistance over my entire time as a graduate student. I am especially grateful for his above and beyond assistance in helping to refine this thesis from concept to a finished product while working through the gaps in my prior knowledge. I would like to sincerely thank Michael Paul for his mentoring and dedication over the past few years in creating the amazing opportunities for myself and many others to gain invaluable experience, find my passion, and meet many good friends along the way. My education would not have been possible without the support and funding of The Applied Research Laboratory and the Penn State University as a whole in backing the Penn State Lunar Lion XPRIZE team.

Dr. Patrick Reed's class on Evolutionary Algorithms inspired my interest in optimization techniques that eventually led to the work herein. Dr. Jacob Langelaan repeatedly provided critically important advice in developing my understanding of direct trajectory optimization and helped me get past several roadblocks along the way. Dr. Joseph Horn's instruction on dynamics and controls were essential in developing my understanding and knowledge needed for this work.

I am eternally beholden to my parents, family, and friends for their love and encouragement that has given me the confidence to reach for the stars, and the audacity to try. In particular I would like to thank Mallori Hamilton for her tireless support and assistance over the past several months of writing, coding, and revising.

Nomenclature

Acronyms and Abbreviations

CGL	Chebyshev-Gauss-Lobatto
DC	Direct Collocation
DCNLP	Direct Collocation and Nonlinear Programming
MOI	Moment of Inertia
NEO	Near-Earth Object
NLP	Nonlinear Programming
NLP_CONTROL	Nonlinear Programing Control Variable
NLP_STATE	Nonlinear Programing State Variable
NLPV	Nonlinear Programing Variable
OCP	Optimal Control Problem
SSTO	Single Stage to Orbit
TOF	Time of Flight

Symbols

α	Flight path angle
$g_{\oplus}$	Nominal Earth surface gravity $9.806\ m/s\text{^}2$
g_{local}	Local surface gravity
I_{sp}	Effective Specific Impulse
J	Scalar Objective Function
k	Node index
θ_{min}	Minimum Thrust Angle
θ_{max}	Maximum Thrust Angle
$\dot{\theta}_{max}$	Maximum allowable rate of change of the thrust angle
n	Number of discritization nodes
t	Time
t_{burn}	Rocket burn duration
t_k	Time at nod k
Δt_k	Node width
τ_k	Trajectory segment k
$Thrust_{min}$	Minimum Thrust Magnitude
$Thrust_{max}$	Maximum Thrust Magnitude
$U_{exhaust}$	Exhaust velocity
V	Scalar velocity
ΔV	Impulsive velocity change

$\boldsymbol{C_{equality}}$	Equality constraint
$\boldsymbol{C_{inequality}}$	Inequality constraint
$\boldsymbol{\psi}_{initial}$	Initial boundary conditions
$\boldsymbol{\psi}_{final}$	Final boundary conditions
$\mathbf{k}$	Vector of node indices
$\mathbf{p}$	Time independent problem parameters
$\mathbf{s}(t)$	Vector of state variable functions
$\dot{\mathbf{s}}(t)$	Vector of state equations
$\mathbf{s}_{lower}$	Lower bounds of state variables
$\mathbf{s}_{upper}$	Upper bounds of state variables
$\mathbf{s}[:,k]$	Matrix of state variable function values at nodes
$\mathbf{t}$	Time vector
$\mathbf{u}(t)$	Vector of control variable functions
$\mathbf{u}_{lower}$	Lower bounds of control variable
$\mathbf{u}_{upper}$	Upper bounds of control variables
$\mathbf{u}[:,k]$	Matrix of control variable function values at nodes
$\mathbf{z}$	Dynamic variable equations
$\mathbf{Z}_k$	Individual node of the solution matrix
$\mathbf{Z}$	Solution matrix

State Variable Abbreviations

Abbreviation	Variable	Units
$xPos$	X Position	$meters$
$xVel$	X Velocity	$meters/second$
$zPos$	Z Position	$meters$
$zVel$	Z Velocity	$meters/second$
$Mass$	Mass	$kilogram$

Control Variable Abbreviations

$Thrust$	Thrust Magnitude	$Newtons$
θ	Thrust Angle	$degrees$

Mathematical Notation

Bold font is used to identify the state and control function vectors and all matrices. Subscripts refer to a particular index of a vector or matrix, or to differentiate between variables at specific times or conditions. Colons are used with matrices or vectors to indicate multiple members along a dimension are being referred to simultaneously.

Chapter 1

Introduction to Spacecraft Mobility

1.1 Problem Definition and Objective

This thesis is focused on trajectory optimization for space-based vehicles performing Vertical Takeoff Vertical Landing (VTVL) translational "hopping" maneuvers over the surface of a body with significant gravity and minimal air resistance, such as the Moon or Mars. The goal is to determine the minimal propellant required to accomplish a desired VTVL translation maneuver for a given distance with an idealized spacecraft propulsion system using direct optimization methods. The results are intended to guide future mission planning in developing a higher fidelity yet adaptable model to predict the required propellant mass for a desired mobility capability, e.g. number and magnitude of desired translation maneuvers (hops). Starting from a simplified parameterized model, appropriate constraints can be added to capture the limitations of a range of possible spacecraft propulsion systems and/or mission architectures to increase the solution fidelity.

1.2 Introduction to Spacecraft Lander Mobility Methods

There has been a rich and interesting variety of spacecraft lander designs and architectures considered and built since the dawn of the space age as unique and complex as the mission requirements which spawned them. Despite their differences, almost all of them have employed vertical high thrust chemical rocket powered maneuvers for their terminal landing, stretching from the first lunar missions to Curiosity's "Sky Crane."[1] Recently, there has been renewed interest from the public and private sector in visiting or revisiting the surface of the Moon, Mars,

and Near Earth Objects (NEOs) for science, tourism, resource exploitation, exploration, and human settlement. Development of the next generation of spacecraft landers is ongoing.

Many of the first landers in the lunar and Martian programs had no secondary mobility capabilities. Accomplishing the soft landing was arguably the most challenging and important part of these early missions. These spacecraft were limited in their capabilities and could only study the area immediately surrounding their landing sites. However, once mission planners were confident in their ability to accomplish soft landings, they quickly sought to explore a greater area than possible from a fixed location.

Later lunar missions included rovers and astronauts to explore the surrounding terrain. Staged spacecraft landers were designed to return human or geological payloads back to Earth. Given that crewed missions typically cost orders of magnitude more than robotic missions, wheeled rovers have been the mobility method of choice for recent missions.

1.3 Mobility through Vertical Takeoff Vertical Landing

It is possible to execute "hopping" maneuver(s) using a spacecraft propulsion system to Vertically Takeoff, translate a desired distance, and Vertically Land (VTVL). Several reasons may favor using this mobility method in lieu of or in addition to a wheeled rover such as:

- The ability to cross terrain which would be impassable to most rovers such as craters and/or boulder fields.
- Decreased mission cost, complexity, and risk by eliminating the development of a separate roving vehicle with independent subsystems and its own unique failure modes.
- The potential to visit multiple sites of interest (geological or otherwise) which are significant distances apart within a compressed time span.

- Executing a VTVL hop prior to a rover deployment could decrease the accuracy needed for the initial entry, descent, and landing.

Short duration hops could be made into or over permanently shadowed craters on the Moon or large valleys on Mars to quickly cover significant ground. Some Near Earth Objects (NEOs) such as asteroids could be sampled at multiple locations similar to *Hayabusa's* "kiss" method where a wheeled rover would be impractical due to the low gravity[2,a]. For lunar missions, this could eliminate the need for the significant power sources required to survive the lunar night if surface operations could be completed within one lunar day.

If the same guidance and propulsion system that was used to originally land on the surface was used for the hop, and sufficient margin existed within the propellant tank(s) volume, the mass and cost could theoretically be as low as the additional propellant required to perform the maneuver. The required propellant mass, if it were known, could be compared to the mass and complexity of a separate rover when deciding which mobility method, if any, to use for a particular mission. Key to deciding whether to further pursue the development of VTVL as a spacecraft mobility method depends on the creation of sufficiently accurate models to determine how much propellant is required in order to perform hopping maneuvers.

1.4 Previous VTVL Vehicles

Surveyor 3 (1967) accidentally hopped due to a fault in the radar interpreting algorithm. The engines did not cut off when the spacecraft touched the surface – twice. The craft "bounced" until ground control sent a cut-off signal[3]. Luckily, the craft survived.

---

[a] A wheeled rover would have to move prohibitively slowly in microgravity to prevent slipping since the available friction for traction is proportional to the weight of the craft. Obstacles could be impassible.

Surveyor 6 (1967) intentionally performed a lateral 2.4 meter hop in order to study the lunar regolith's surface mechanical properties. After the hop, the craft used its cameras to inspect the indentations left from the primary landing and to study the effects of plume impingement with the surface[4]. This was done to help determine if the surface was suitable for manned missions.

In the early to mid-1990's, the McDonnell Douglas DC-X program attempted to develop the technologies needed for a Single Stage to Orbit (SSTO) VTVL reusable launch vehicle. While the program never progressed to orbit, it did complete several successful suborbital test flights. Funding was ultimately canceled after a crash and subsequent fire that destroyed the second vehicle[5].

While not a rocket powered VTVL craft, the *Hayabusa* Mission also included a hopping robot *MINERVA* that was intended to "bounce" along the surface of the asteroid *Itokawa*. The internal reaction wheels would have been spun up and down abruptly to create torques so that the entire rover would tumble along the surface. This would have been the first deployment of such a technique in space. Unfortunately, in 2005 upon arrival at the asteroid, *MINERVA* was deployed faster than *Itokawa*'s escape velocity by mistake. *MINERVA* survived for several hours but never reached the surface[2].

1.5 Current VTVL vehicle development

There are currently several ongoing efforts by NASA, private companies and individuals, and Penn State University to develop VTVL vehicles as software and hardware testbeds, as spacecraft for use on the Moon, and reusable VTVL launch vehicles for suborbital and orbital altitudes. A sample of current vehicles under development is shown in Table 1.1.

For any of these current or future vehicles, (or similar vehicles not listed), a degree of translational capability is needed for their respective mission profiles including precision landings, planetary "hopping" maneuvers, or "return-to-pad" launch vehicle stage fly-backs. Naturally, it is desirable to determine the most propellant efficient trajectories as possible. This thesis addresses this capability through the use of Direct Collocation and Nonlinear Programming for trajectory optimization.

Table 1.1 Selected VTVL Vehicles Under Development

Organization	Craft name	Type	Propellant(s)
Penn State University	*Puma*[6]	Lunar Lander Software Demonstrator	Monopropellant hydrogen peroxide
NASA Marshall Spaceflight Center	*Mighty Eagle*[7]	Lunar Lander Technology Demonstrator	Monopropellant hydrogen peroxide
NASA Johnson Space Center	*Morpheus*[8]	Lunar Lander and Green Propellant Technology Demonstrator	Liquid oxygen and Methane
Masten Aerospace	*Xaero B*[9] (among others)	Reusable suborbital payload delivery	Liquid oxygen and isopropyl alcohol
Blue Origin	*New Shepard*[10]	Technology demonstrator	Hydrogen peroxide and kerosene
Space Exploration Technologies	*Grasshopper*[11]*/Falcon 9 Reusable First Stage*[12]	Reusable orbital launch vehicle first stage	Liquid oxygen and kerosene

1.6 Thesis Outline and Scope

In evaluating the use of Direct Collocation and Nonlinear Programming as applied to VTVL spacecraft, this thesis addresses this topic in several chapters. Chapter 2 details the process of deciding which control variables and assumptions were appropriate to model a generic VTVL spacecraft. Chapter 3 develops a state variable dynamic system model to represent the spacecraft and VTVL translational maneuver as an Optimal Control Problem (OCP). Chapter 4 discusses the background and implementation of Direct Collocation with Nonlinear Programming (DCNLP), a direct trajectory optimization method. Chapter 5 presents the results across a variety of various spacecraft parameters and mission profiles. A summary of the work completed and suggestions for future work to increase the model fidelity and usefulness are covered in Chapter 6.

Several appendices have been added to assist readers who are new to the concepts of trajectory optimization. These include:

Appendix A: Notes on Direct Collocation

Appendix B: Time as a Fixed or Free Variable, and Node Distribution

Chapter 2

Dynamic Variables used for Modeling VTVL Spacecraft Propulsion Systems

2.1 Introduction to State Variable Dynamic Modeling

The goal of this chapter is to determine which state and control variables are required to create an adaptable model of a VTVL trajectory and spacecraft, and what simplifying constraints and assumptions can be made. As discussed herein, the goal is to produce the most agnostic propulsion system model as possible while preserving the ability to introduce constraints which can capture the specific limitations of a particular spacecraft propulsion system design.

2.2 Vertical Takeoff versus Vertical Landing

Compared to vertical landing, vertical takeoff is significantly easier. Constant thrust engines will quickly accelerate a spacecraft away from the ground, and slight variations of thrust magnitude or vector usually do not result in a complete loss of vehicle. Vertical landing is intrinsically harder as determination of the state variables rely on imperfect sensor data and dynamic system models. Small errors can quickly lead to failures and/or large errors in landing accuracy/location.

If it were possible to perfectly determine the system state variables and model the propulsion system, as well as eliminate all errors and noise, the most propellant efficient method to vertically land a spacecraft would be to fire the engines at maximum thrust for the exact duration required to bring the spacecraft to a halt just at the surface with zero residual velocity. The control law would be reduced to a switching function of when to start the engines[13]. Unfortunately, reality is more complicated.

2.3 Necessity of Throttling, Thrust Vectoring, and Closed Loop Control

To actually accomplish a soft landing requires complex guidance, navigation, control, and propulsion subsystems in a feedback loop. While it may be possible to use non-throttleable high thrust (e.g. typical solid) engines to zero out most of the incoming velocity of a lander spacecraft during its initial approach to the central body, the terminal descent propulsion system needs to be capable of thrust vectoring and throttling to account for performance variations, sensor inaccuracies, and system noise such as propellant sloshing or sensor drift. The degree of throttling required depends on the specific spacecraft architecture, mass, and engine thrust levels.

Depending on the central body in question, it may be possible and highly preferable to use the same engines and control systems for the initial terminal descent and landing as for the hopping maneuver(s). If this is the case, and sufficient margin existed within the propulsion systems propellant/pressurant tank volumes, the mass penalty to perform the hopping maneuver could be as low as the extra propellant required.

Figure 2.1 details a theoretical VTVL spacecraft control architecture. The Flight Computer outputs commands to the Propulsion Subsystem to achieve a desired trajectory, attitude or spin rate change, etc. The Propulsion Subsystem would consist of all the valves, plumbing, tanks, propellant, engines, gimbals, etc., that generate the required force vectors to bring the spacecraft from an initial system state to a desired final system state. Additional hardware could be utilized as needed.

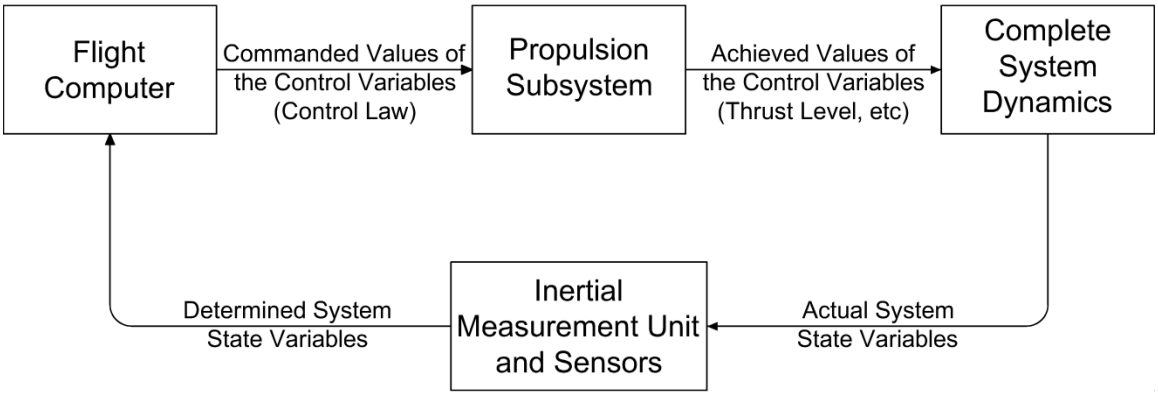

Figure 2.1 Possible VTVL Spacecraft Control Architecture

The Complete System Dynamics would be the actual laws of physics that the spacecraft encounters. The actual dynamics slightly differ from the models used to approximate them due to unmodeled (or unknown) forces and behaviors, e.g. local variances in gravity due to central body shape and density irregularities, and/or differences in expected versus achieved propulsion system performance. Meanwhile the Inertial Measurement Unit (IMU) and sensors would be attempting to keep track of the spacecraft's state variables such as position, velocity, acceleration, rotation, etc., but this data is noisy and only as accurate as the sensors can provide. Some state variables cannot be directly measured and must rely on imperfect software models, which adds additional system noise.

The flight software needs to operate in a closed loop to be able to make (hopefully only slight) adjustments on the fly to compensate for noise and constantly ensure that the spacecraft is following the desired trajectory. The Flight Computer may need to generate a new control law in real time to correct for accumulated errors or perhaps change the landing site if an unexpected obstacle (e.g. boulder) is encountered. The work presented herein focuses on designing the optimal trajectory before flight, but this effort can guide the development of robust flight software in the future.

2.4 Potential Propulsion Architectures

While many possible propulsion system architectures are possible, all of them must be able to effectively vector their net thrust in order to successfully land because of the noise concerns discussed earlier. This same thrust vectoring is used to translate horizontally, though specific propulsion system designs will have unique limitations such as a maximum thrust angle and a maximum thrust angle angular rate of change.

Figure 2.2 shows several examples with the black lines representing the individual engine forces and the red line representing the resultant net thrusts and/or torques generated.

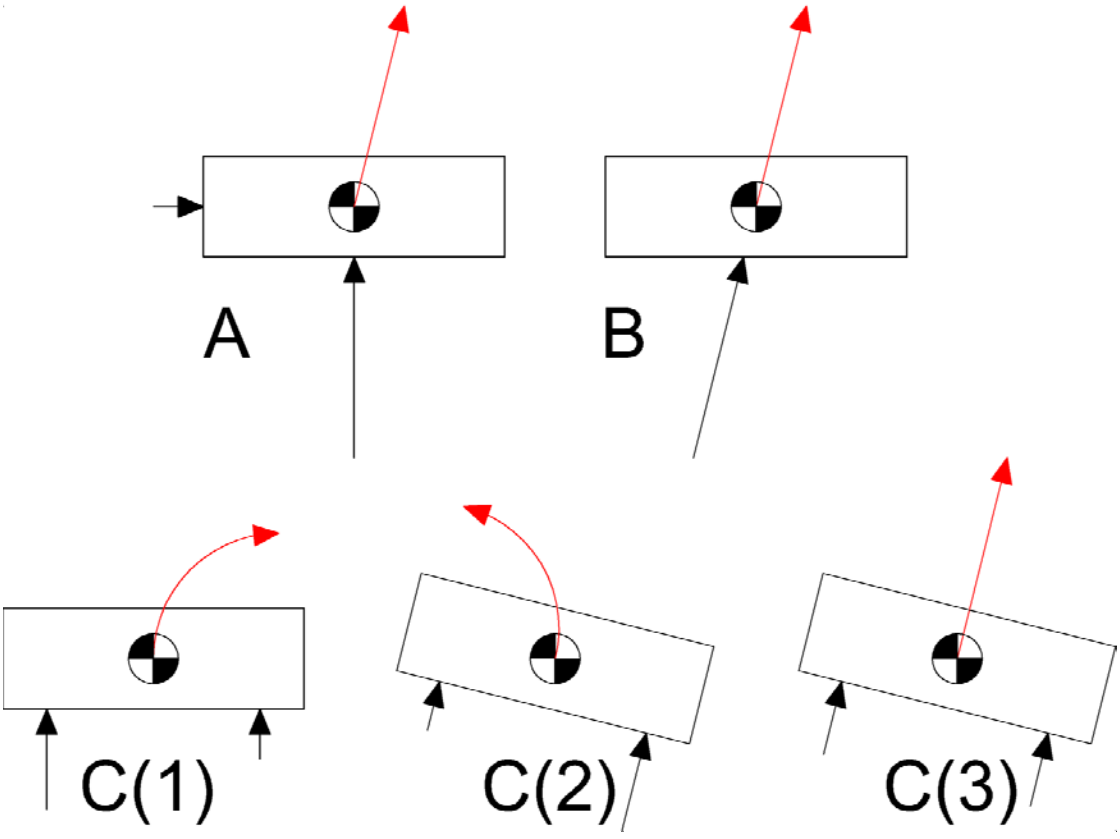

Figure 2.2: Various Possible Propulsion Systems Thrust Vectoring Methods

Craft A utilizes a main lifting engine and vernier engines for translation. Craft B employs a gimbaled main engine. Craft C has fixed engines which engage a positive[b] pitching maneuver, C(1), a negative pitching maneuver, C(2), and then translates holding a pitch angle, C(3).

Similarly, there are many possible methods to accomplish throttling. Direct proportional control of the propellant mass flow rate is possible in some engine designs, though there may be some losses in efficiency and certain throttle ranges that are out of bounds. For example, the Descent Propulsion System (DPS) for the Apollo missions was only capable of either throttling between 10-60%, or running at full 100% thrust[14]. For engines capable of pulsing[c], the duty cycle can be adjusted to effectively lower the time-averaged thrust and achieve a range of effective throttling. Combinations of proportional, pulsing, and constant thrust control engines are all possible as well.

2.5 Central Body Approximations of Uniform Gravity and Lack of Air Resistance

In this study, the hopping distances and maximum heights reached during any translation maneuvers are far less than 1% of the central body radius. Modeling gravity as a uniform constant equal to the nominal surface gravity is considered sufficiently accurate. Similarly, air resistance was neglected as the key central bodies of interest have little to virtually no atmosphere.

2.6 Reduction to Two Dimensions

The initial landing trajectory to reach the central body is not considered; therefore neither is any propulsion system specific orientation bias that would favor traveling in a specific direction. Thus, there exist an infinite number of equipotential possible secondary landing sites that lie along a circle with radius equal to the translation distance surrounding the primary landing site. The coordinate system origin is thus chosen such that the origin is centered at the primary spacecraft

---

[b] Positive and negative pitching angles are defined as per the convention shown in Figure 2.4
[c] Pulsing is operating an engine at a high frequency instead of continuously.

landing site, (prior to any hopping maneuver), and the positive X axis extends from the primary landing site through the secondary landing site as shown in Figure 2.3.

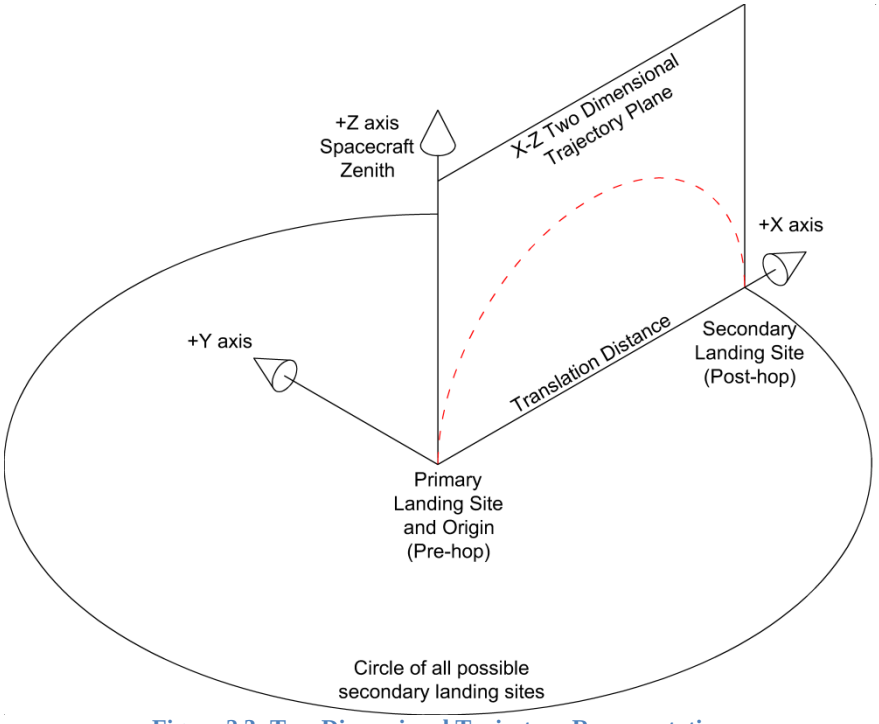

Figure 2.3: Two Dimensional Trajectory Representation

The positive Z axis extends from the ground through the zenith of the spacecraft. The Y-axis is perpendicular to the X and Z axis as according to the right-hand rule. Anywhere the spacecraft could land, a straight line could be traced from its takeoff to landing site; and the coordination frame could be rotated to align. Any trajectories that laterally deviated out-of-plane in the +/- Y axis midflight would require a disturbing and restoring force to return within plane. Generating these forces would require additional propellant with no benefit and would result in a suboptimal trajectory. It is thus reasoned that optimal trajectories will lie entirely with the X-Z plane and therefore state variables representing two spatial dimensions are sufficient to model this VTVL maneuver.

2.7 Spacecraft Point-Mass Approximation

Since offsetting the coordination system to compensate for the height of the vehicle would not change the underlying physics, the initial height is considered to be zero to simplify the results. To keep the model as propulsion system agnostic as possible, the only spacecraft state variable tracked is the overall mass – no moment of inertia (MOI) matrix is calculated, and roll, pitch, or yaw maneuvers are not modeled. The thrust was modeled to act directly on the center-of-mass of the vehicle. Therefore the spacecraft is essentially modeled as a point mass.

While a point mass does not have a meaningful attitude, the spacecraft's coordinate system is assumed to coincide with the central body coordinate system origin. The standard conventions of spacecraft roll, pitch, and yaw being rotations around the X, Y, and Z axis are made only to discuss their disuse.

2.8 State Variable Selection

Since the problem is restricted to two spatial dimensions, the only forces acting on the craft are thrust and gravity, and the thrust is assumed to act on the center-of-mass; only five *time dependent* state variables are required to fully represent the spacecraft VTVL trajectory problem; the velocity and position along the X and Z axes, and the spacecraft mass. The simulation time duration, (the Time of Flight for the VTVL maneuver), can also be fixed, or free to float between an upper and lower bound.

Note that the spacecraft mass determines the magnitude of acceleration that the spacecraft experiences for a given thrust level, and the acceleration will grow as propellant mass is depleted. The lower the exhaust velocity/specific impulse of the rocket engines used, the greater the mass flow rate will be for a given thrust.

2.9 Control Variable Selection

With the previous simplifications, the control variables can be reduced to a net thrust vector with a specific magnitude, $Thrust$, and thrust angle, θ, as shown in Figure 2.4. The thrust angle is defined so that zero corresponds with the nadir. In this thesis $|\theta_{max}|$ always equals $|\theta_{min}|$, though this is not strictly required in general. Note the spacecraft is shown with a pitch angle equal to the thrust angle purely for visualization purposes.

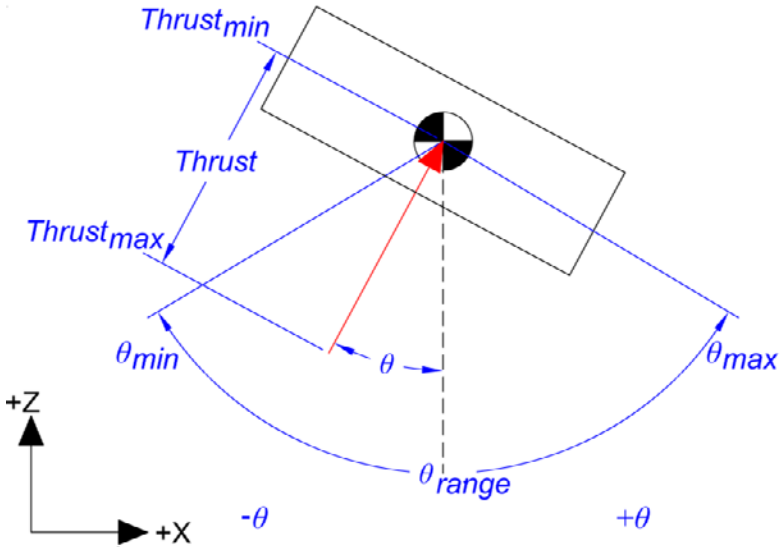

Figure 2.4: Spacecraft Thrust Control Modeling

2.10 Basic Control Law Constraints for Increasing Model Fidelity

The most basic constraints made on the control variables are their maximum and minimum range of values. Every spacecraft propulsion system has an upper limit to the net thrust it can produce[d], and as previously mentioned, some engines/systems have lower limits and/or restricted throttling ranges. As discussed herein, introducing additional constraints can reflect a range of spacecraft propulsion architecture-specific limitations and increase the fidelity of results.

---

[d] A VTVL spacecraft should reserve some of its throttling capability in order to account for system noise, e.g. create a control law with a planned max throttle limit of 80% of peak thrust for maneuvers, reserving the remaining 20% in case the spacecraft was accelerating or decelerating slower than anticipated and was at risk of impacting. This is referred to as *control authority* and varies by spacecraft design. For this study, the thrust values given are assumed to be after any control authority margin is reserved.

For a spacecraft with a gimbaled main engine, (Craft B in Figure 2.2), the gimbal likely has a restricted range of motion. For a fixed engine spacecraft, (Craft C in Figure 2.2), there may be a maximum pitch angle allowed due to sensor limitations or to lower risk, etc. These specific limitations can be captured by setting the appropriate values for the minimum and maximum allowable thrust angle.

Similarly, constraints on the thrust angle angular rate of change can be implemented in order to account for the maximum allowable or possible pitch rate of change of a spacecraft or speed at which the spacecraft can gimbal its main engine. By constraining the allowed thrust angle angular rate of change to the achievable pitch rate of a spacecraft, higher fidelity results can be realized without using a spacecraft's full momentum of inertia in the state equations.

There can also be constraints on the allowable liftoff and landing values of the thrust angle. While a spacecraft with a gimbaled main engine may be able to take off and land at a slight angle, a spacecraft with fixed engines would likely need to take off almost vertically, i.e. the thrust angle must equal zero at the beginning and the end of the flight. Some residual final speed could be tolerated upon landing, but too much in either the X or Z direction could damage the craft and/or cause it to dig into the surface and/or flip over.

2.11 Development of State Equations

Once the control variables are chosen, sufficiently accurate equations that model the problem dynamics are developed. These state equations include functions that predict the motion of the spacecraft in response to the forces of gravity and thrust, and the change in spacecraft mass as propellant is expelled to create thrust. Once the trajectory problem is represented mathematically,

optimization techniques are employed to refine and improve possible flight profiles and control laws.

2.12 Burn-Coast-Burn Impulsive-Ballistic Minimum Energy Derivation

The lower bound of the minimum propellant use is determined via a "burn-coast-burn" ballistic-impulsive analysis[4,e] similar to modelling the path of a cannonball as shown in Figure 2.5. Such a spacecraft would experience an impulsive velocity change, $\Delta V_{initial}$, (1st burn), that would bring the spacecraft from a rest state to an initial velocity, $V_{initial}$, at a flight path/launch angle, $\alpha_{initial}$, as measured from the horizon. The spacecraft would "coast" in a parabolic arc according to the relevant equations of motion, eventually reaching the ground with a final velocity equal to the initial velocity, $V_{final} = V_{initial}$, and a negative flight path angle, $\alpha_{final} = -\alpha_{initial}$. An equivalent impulsive velocity change would be required at the end of the flight, ΔV_{final}, (2nd burn), to zero out the final velocity, V_{final}, and bring the spacecraft to a rest[f].

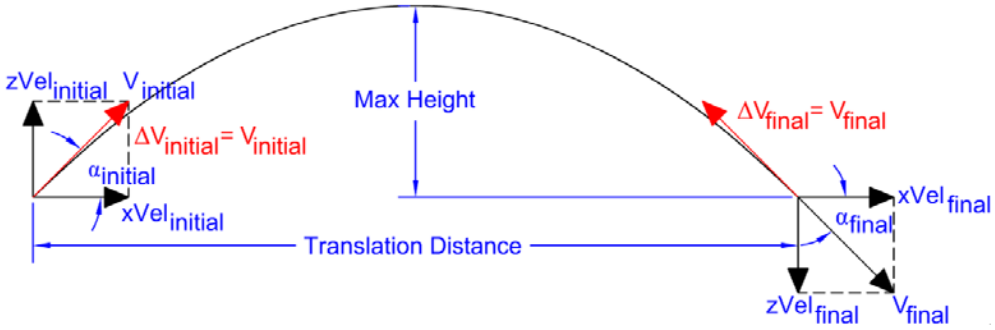

Figure 2.5: "Burn-Coast-Burn" Impulsive-Ballistic Trajectory

While this method has limitations described herein, several useful formulas were derived to establish estimates and values for comparison.

---

[e] There are errata in the formulas listed in this source; the calculations were re-derived.
[f] If the central body in question had exceedingly low gravity, e.g. an asteroid, the spacecraft may be robust enough to survive the impact and save propellant. This could also prevent contamination of the second site by the spacecraft's propellant.

For the most efficient initial flight path angle, $\alpha_{initial}$ = 45 degrees, the ballistic-impulsive TOF and required initial velocity can be solved for in terms of the desired horizontal displacement and the local surface gravity

$$V_{initial,ballistic} = \sqrt{xPos_{final} * g_{local}} \tag{2.1}$$

$$TOF_{ballistic} = \sqrt{\frac{2 * xPos_{final}}{g_{local}}} \tag{2.2}$$

The maximum height reached is simply one quarter of the targeted distance.

$$zPos_{max,ballistic} = \frac{xPos_{final}}{4} \tag{2.3}$$

2.13 Burn-Coast-Burn Impulsive-Ballistic Propellant Mass Derivation

To calculate how much propellant would be required to perform these impulsive ΔV maneuvers, the "ideal" rocket equation is used,

$$\Delta V \ = \ U_{exhaust} * \ln\left(\frac{Mass_{Before\ Burn}}{Mass_{After\ Burn}}\right) \tag{2.4}$$

where the exhaust velocity, $U_{exhaust} = I_{sp} * g_{\oplus}$. Using the ideal rocket equation iteratively and using the relation $|V_{intial}| = \Delta V_{initial} = \Delta V_{final} = |V_{final}|$, the required propellant to perform both burns is found to be

$$Propellant\ Mass_{\ impulsive} = \ Mass_{initial} * \left(1 - \exp\left(-\frac{2 * V_{initial}}{U_{exhaust}}\right)\right) \tag{2.5}$$

By substituting the relation in Eq. (2.1) for finding the required initial velocity to travel a desired horizontal displacement into Eq. (2.5), the impulsive-ballistic propellant mass can be directly solved for in terms of the original spacecraft mass, exhaust velocity, and local gravity.

$$Propellant\ Mass_{impulsive} = Mass_{initial} * \left[1 - \exp(-\frac{2 * \sqrt{xPos_{final} * g_{local}}}{U_{exhaust}})\right] \quad (2.6)$$

Thus Eq. (2.6) places a lower bound on the minimum propellant required in order to perform the hopping maneuver for a given translation distance. This ideal burn-coast-burn trajectory model can be used as an efficiency measure for finite thrust maneuvers. The closer the actual required propellant is to this ideal value, the more efficient the trajectory, but it cannot use less propellant.

2.14 Gravity Losses and Limitations of Impulsive-Ballistic Model

Where the burn-coast-burn model fails is that infinite thrust would be required to accomplish the desired velocity changes instantaneously[15]. Any finite-thrust system needs to fire its engines for a non-zero amount of time in order to perform the initial take-off and landing ΔV's. Because the spacecraft will be experiencing gravity during this time, the actual propellant use required to perform a desired ΔV requires including a *gravity losses* term to the rocket equation as

$$\Delta V = U_{exhaust} * \ln\left(\frac{Mass_{Before\ Burn}}{Mass_{After\ Burn}}\right) - \int_{t=0}^{t_{burn}} g_{local} * dt \quad (2.7)$$

where t_{burn} is the length of time the engines are firing. The consequence of gravity losses is that more propellant is required to accomplish a desired ΔV than the impulsive rocket equation would suggest, e.g. the spacecraft $Mass_{After\ Burn}$ in Eq. (2.7) would be lower than in Eq. (2.4) for the same ΔV. However, the spacecraft is also translating while the thrust accelerates and decelerates the spacecraft at the beginning and end of the flight, so the required coasting distance and initial velocity before coasting would be less than the initial velocity given by Eq. (2.1). Lastly, the spacecraft's propulsion system must compensate in order to achieve a desired flight path angle.

2.15 Free-body diagram of a VTVL hopping spacecraft

While Eq. (2.7) is useful for highlighting the limitations of the impulsive rocket equation, it is ultimately insufficient for calculating the propellant required for a VTVL translation maneuver.

Gravity losses not only increase the amount of propellant required to perform maneuvers, they also change the resultant magnitude and the angle of acceleration of the spacecraft. A free-body diagram of a hopping spacecraft as shown in Figure 2.6 demonstrates the effect. If a flight path angle, α, of 45 degrees was desired, the nominal thrust angle, $\theta_{nominal}$, would need to be steeper in order to compensate for the force of gravity. Additionally, the magnitude of the resultant thrust is lower due to vector addition.

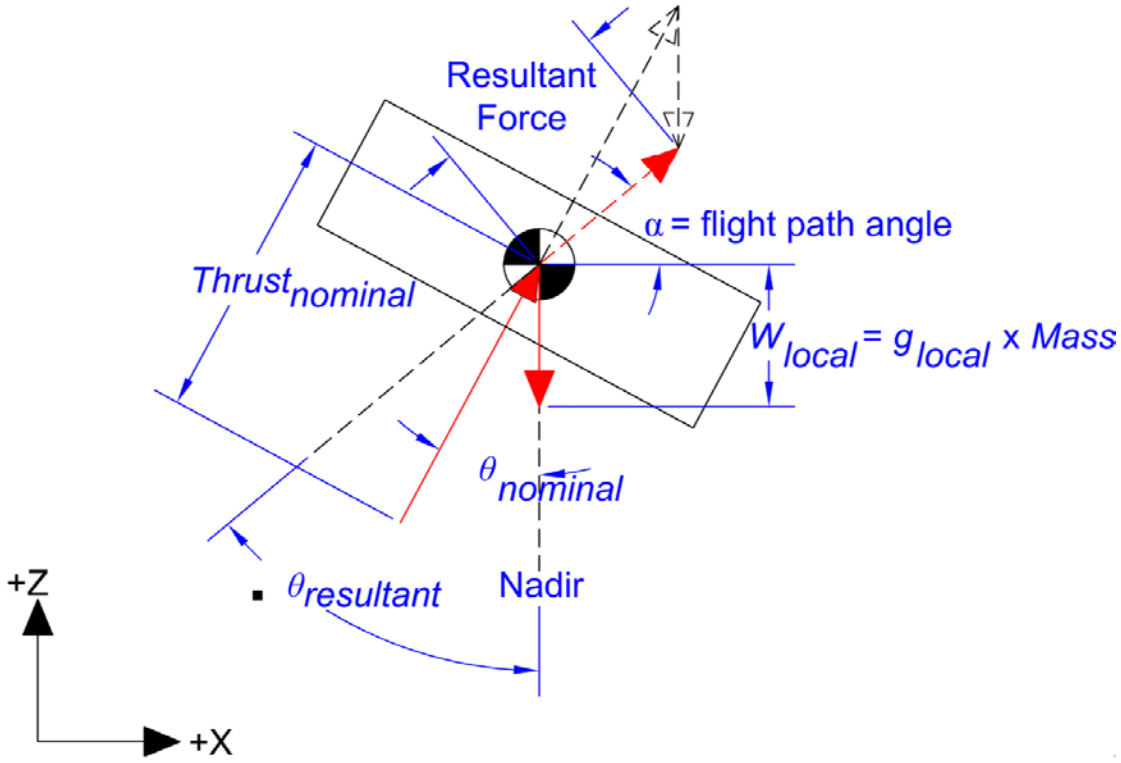

Figure 2.6: Free-body analysis of a VTVL hopping spacecraft

Understanding the problem dynamics can assist mission planners in choosing the required thrust level, throttling capabilities, and thrust vectoring requirement of a VTVL spacecraft propulsion system. Just to get off the ground, the produced thrust must be greater than the spacecraft's local weight. In order to hover and maintain an altitude, the thrust output must continuously decrease to match the spacecraft's current weight, which will exponentially decay as propellant is expelled

from the engines. In order to maintain a specific altitude while translating[g], the magnitude and thrust angle must be balanced such that the vertical component of thrust continuously equals the vehicles' current weight. Lastly, if the vehicle is to keep its engines on during the entire flight[h], throttling below the spacecraft's local weight is required to land.

2.16 Solution Hypothesis

The expected unconstrained solution for a high Thrust to Weight ratio (T/W) spacecraft is to approximate a ballistic trajectory within the limitations of the spacecraft's ability to accelerate as rapidly as possible – essentially a finite-burn, coast, finite-burn, where the second burn is slightly less due to the spacecraft mass decreasing from performing the first burn. From there, various spacecraft parameters or potential solution constraints can be varied to study their impact on the required propellant.

2.17 Comparison to Orbital Trajectory Optimization

Developing a model for VTVL spacecraft trajectory optimization was baselined on a classic optimization problem for a constant thrust[i] spacecraft[16]. For that problem, the objective is finding the thrust-direction history that transfers a spacecraft from an initial circular orbit to the largest possible circular orbit for a given TOF. The initial spacecraft mass, orbital radius, thrust-level, and TOF could all be varied independently.

Modeling a VTVL spacecraft required adding an additional control variable for throttling the thrust, changing the state equations to reflect surface operations, and adding additional spacecraft

---

[g] A flight profile commonly seen of Earth-based VTVL test vehicles is a "top hat" trajectory where the spacecraft vertically ascends, accelerates horizontally, translates, decelerates horizontally, and then vertically lands.

[h] This might be required for non-hypergolic bipropellant propellants where restarts are limited.

[i] These types of trajectories are characteristic of spacecraft utilizing high I_{sp}, low-thrust electric propulsion.

parameters and constraints to increase model fidelity. Orbital trajectory optimization techniques for high-thrust spacecraft that model ΔV maneuvers as impulsive were not helpful[j].

2.18 Developing an Optimal Control Problem

Solving for the optimal VTVL translation trajectory is equivalent to solving for the control law which produces it. Thus a model is developed to represent both the control law and the resulting change in state variables over time. This can be referred to as an Optimal Control Problem. Once a mathematical model is created, optimization techniques can be applied.

[j] The problem dynamics, key assumptions, and solution formats are very different.

Chapter 3

Mathematical Representation of VTVL Translation Maneuver

3.1 Mathematical Representation of VTVL Maneuver as an Optimal Control Problem

The following mathematical representation was adapted in part from a succinct description found in literature describing a typical Optimal Control Problem (OCP)[17,18] and applied to this problem. The VTVL translation maneuver can be readily represented by a system of *dynamic variables* **z,** consisting of a $[5 \times 1]$ column vector of time dependent *state variable functions* $\mathbf{s}(t)$ and a $[2 \times 1]$ column vector of time dependent *control variable functions* $\mathbf{u}(t)$ for any time t within the simulation time range of $[t_{initial}: t_{final}]$. Here $s_j(t)$ for $(j = 1, \cdots, 5)$ refers to any one time dependent state variable function in particular.[k]

$$\mathbf{z} = \begin{bmatrix} \mathbf{s}(t) \\ \mathbf{u}(t) \end{bmatrix} = \begin{bmatrix} s_1(t) \\ s_2(t) \\ \vdots \\ s_5(t) \\ u_1(t) \\ u_2(t) \end{bmatrix} for\ t_{initial} \leq t \leq t_{final} \tag{3.1}$$

The problem dynamics are defined by a $[5 \times 1]$ column vector of parametric differential equations $f_1 \ldots f_5$ called the *state equations* that can be represented as

$$\dot{\mathbf{s}}(t) = \frac{d\mathbf{s}(t)}{dt} = \mathbf{f}[\mathbf{s}(t), \mathbf{u}(t), \mathbf{p}, t] = \begin{bmatrix} \dot{s}_1(t) = f_1(\mathbf{s}(t), \mathbf{u}(t), \mathbf{p}, t) \\ \dot{s}_2(t) = f_2(\mathbf{s}(t), \mathbf{u}(t), \mathbf{p}, t) \\ \vdots \\ \dot{s}_5(t) = f_5(\mathbf{s}(t), \mathbf{u}(t), \mathbf{p}, t) \end{bmatrix} \tag{3.2}$$

where $\mathbf{p}$ is a vector of time independent problem *parameters*. Note that $\mathbf{u}(t)$ is also referred to as the control law, as it gives the values of the control variables over the course of the trajectory.

---

[k] Note that **s** and **u** are later treated as numerical matrices with square bracketsused for indexing.

The state variable functions $\mathbf{s}(t)$ are the integrals of these state equations $\dot{\mathbf{s}}$, and any one state equation in particular is $\dot{s}_j(t)$.

The simulation time start, $t_{initial}$, is defined to be zero, so the simulation duration, (the TOF), is equivalent to, t_{final}. The TOF may be a *free variable* and allowed to float within an upper and lower bound by changing t_{final},

$$t_{final,lower} \leq t_{final} \leq t_{final,upper} \tag{3.3}$$

or a *fixed variable* set by restricting t_{final} to a specific time.

There may be simple global bounds on the state and control variables between lower and upper limits, such as a maximum or minimum allowable altitude, velocity, thrust magnitude, thrust angle, etc.

$$\mathbf{s}_{lower} \leq \mathbf{s}(t) \leq \mathbf{s}_{upper} \tag{3.4}$$

$$\mathbf{u}_{lower} \leq \mathbf{u}(t) \leq \mathbf{u}_{upper} \tag{3.5}$$

The *initial boundary condition* at the simulation start may be defined as a $[8 \times 1]$ column vector

$$\boldsymbol{\psi}_{initial} = \begin{bmatrix} t_{initial} \\ \mathbf{s}(t_{\text{initial}}) \\ \mathbf{u}(t_{\text{initial}}) \end{bmatrix} \tag{3.6}$$

and the desired *final boundary condition* may be defined as a $[8 \times 1]$ column vector

$$\boldsymbol{\psi}_{final} = \begin{bmatrix} t_{final} \\ \mathbf{s}(t_{\text{final}}) \\ \mathbf{u}(t_{\text{final}}) \end{bmatrix} \tag{3.7}$$

The boundary conditions can be described in terms of $\boldsymbol{\psi}_{free}$ and $\boldsymbol{\psi}_{fixed}$ variables subsets. While solving for the optimal trajectory, free variables may float between a range of acceptable limits

given by $\boldsymbol{\psi}_{free,lower}$ and $\boldsymbol{\psi}_{free,upper}$ such as the final mass, or fixed, such as the initial position[l], per VTVL maneuver.

The problem may also be subject to *equality constraints* of the form

$$\boldsymbol{C}_{\boldsymbol{equality}}\text{:}\ \ \mathbf{f}[\mathbf{s}(t), \mathbf{u}(t), \mathbf{p}, t] = 0 \tag{3.8}$$

which must be driven to zero to be fully satisfied, and inequality constraints of the form

$$\boldsymbol{C}_{\boldsymbol{inequality}}\text{:}\ \ \mathbf{f}[\mathbf{s}(t), \mathbf{u}(t), \mathbf{p}, t] \leq 0 \tag{3.9}$$

that only need to be less than or equal to zero in order to be satisfied. Either constraint can introduce *path constraints* or additional limitations on allowable solutions.

The basic optimal control problem is to determine the control law $\mathbf{u}(t)$ that optimizes the fitness function[m], J,

$$J = f[\mathbf{s}(t), \mathbf{u}(\mathrm{t}), \mathbf{p}, t] \tag{3.10}$$

while satisfying all the boundary conditions, upper and lower bounds, and all user-defined inequality and equality constraints. This fitness function must be capable of transitive comparison between any possible trajectories that satisfy all conditions.

3.2 State and Control Functions Are Not Closed Form Analytical Expressions

As is generally the case with most nonlinear coupled dynamics problems, the state equations $f_1 \ldots f_5$[n] cannot be analytically integrated and solved as *closed-form analytical expressions*[o].

---

[l] There needs to be sufficient degrees of freedom, i.e. free variables, or else the problem is over constrained.
[m] This is also called the penalty function, scalar performance index, or objective function.
[n] The state equations are given in Eqs. (3.23)-(3.27).
[o] Also note, for the ballistic-impulsive case described in Section 2.12, the state variable functions can be derived analytically since the state equations are decoupled in the X and Z axes.

Additionally, the control variable functions are usually not closed-form analytical expressions either. The state variable *functions* $s_1 \dots s_5$ can still return a value of the state variables for any time t, but doing so usually requires numerically integrating the state equations from an initial condition while following the control functions.

3.3 Direct Method Overview

The direct method implemented attempts to directly solve for the optimal trajectory by manipulating the values of the state and control variable functions $\mathbf{s}(t)$ and $\mathbf{u}(t)$. This method attempts to simultaneously find the optimal trajectory and the control law which is required to produce it. This technique is well suited to handle the problem dynamics of the VTVL maneuver.

Upper and lower bounds for all state and control variables can readily be enforced so that trajectories stay within a "state-space box," and initial and final conditions can be applied. In this way, the direct method is similar to a *boundary value problem*. However, there is no *a priori* guarantee that the trajectories found during each iteration are *feasible,* i.e., does the control law produce the trajectory given the state equations and additional constraints. The direct method must determine the optimal trajectory only within the feasible subset within the entire state-space.

Depending on how the problem is formulated and the optimization method(s) implemented, the solution state space of all possible trajectories that a given implementation can search may be slightly different, and some solver parameters may require significantly more time and computational power to run. It is important to understand how the method's parameters chosen affect the search space, validity, and usefulness of results.

3.4 Global Time of Flight Bounds

As per Eq. (3.3) the optimal TOF was expected to be greater than the ballistic TOF due to finite burns, but lower than twice the ballistic TOF.

$$t_{final,lower} = \ TOF_{ballistic} = \sqrt{\frac{2 * xPos_{final}}{g_{local}}} \tag{3.11}$$

$$t_{final,upper} = 2 * TOF_{ballistic} = 2 * \sqrt{\frac{2 * xPos_{final}}{g_{local}}} \tag{3.12}$$

3.5 Global State Variable Bounds

In order to reduce the search space, parameterized time-invariant global assumptions are made on the maximum and minimum acceptable state variable values that any optimal solution should be within as per Eq. (3.6). The allowable range of position along the X axis is restricted to be between zero and the final position since any over or undershoot would require restoring forces and thus additional propellant.

$$xPos_{Lower} = 0 \tag{3.13}$$

$$xPos_{Upper} = xPos_{Final} \tag{3.14}$$

Similarly, the minimum X velocity is required to be greater than or equal to zero. The maximum X velocity is assumed to be within twice the ballistic X velocity.

$$xVel_{Lower} = 0 \tag{3.15}$$

$$xVel_{Upper} = 2 * \frac{xPos_{Final}}{TOF_{ballistic}} = \sqrt{2 * xPos_{final} * g_{local}} \tag{3.16}$$

The Z Position, or altitude, is restricted to between zero and half the desired translation distance, which is twice the maximum altitude reached during a ballistic trajectory as per Eq. (2.3).

$$zPos_{Lower} = 0 \tag{3.17}$$

$$zPos_{Upper} = \frac{xPos_{Final}}{2} \tag{3.18}$$

The Z velocity was constrained to be within the initial ballistic velocity. Note that this is the vector total initial ballistic velocity, not only the Z component.

$$zVel_{Lower} = -V_{\text{initial,ballistic}} = -\sqrt{xPos_{final} * g_{local}} \tag{3.19}$$

$$zVel_{Upper} = +V_{\text{initial,ballistic}} = +\sqrt{xPos_{final} * g_{local}} \tag{3.20}$$

The spacecraft's mass can obviously not exceed the original mass for the upper bound. The lower bound was set at the original mass less twice the ballistic propellant mass found via Eq. (2.5).

$$Mass_{Lower} = Mass_{Initial} - 2 * Fuel\ Mass_{\ impulsive} \tag{3.21}$$

$$Mass_{Upper} = Mass_{Initial} \tag{3.22}$$

3.6 Global Control Variable Bounds

The variables used to define the acceptable range of control variable values as per Eq. (3.5) are listed in Table 3.1, and correspond to Figure 2.4. These are global limitations of the spacecraft's propulsion system abilities, though additional constraints can be added to model path constraints or boundary conditions. The specific values used are presented in the results.

Table 3.1: Spacecraft Control Variable Bounds

Parameter	Abbreviation	Units	Use
Minimum Thrust Magnitude	$Thrust_{min}$	Newtons	Lower Bound
Maximum Thrust Magnitude	$Thrust_{max}$	Newtons	Upper Bound
Minimum Thrust Angle	θ_{min}	$Degrees$	Lower Bound
Maximum Thrust Angle	θ_{max}	$Degrees$	Upper Bound

3.7 Determining the Validity of Global Bounds

If any of the bounds on the state variables, control variables, and time of flight are less than needed for the actual optimum trajectory, it is expected that the optimization method should produce trajectories that lie along one or more of the bounding limits. In this case, relaxing the bounds should result in an increase in fitness. If the bounds are too restrictive, the simulation may fail to converge.

3.8 Vector and Matrix Indices

The specific indices of the state and control variable vector functions and the numerical matrices are listed in Table 3.2 for clarity. Note the difference between the function and matrix indexing, further explored in Section 4.2.

Table 3.2: State and Control Variable Function Indices

Function Vector Index	Matrix Indices	Variable	Abbreviation	Units
$s_1(t)$	$\mathbf{s}[1,:]$	X Position	$xPos$	$meters$
$s_2(t)$	$\mathbf{s}[2,:]$	X Velocity	$xVel$	$meters/second$
$s_3(t)$	$\mathbf{s}[3,:]$	Z Position	$zPos$	$meters$
$s_4(t)$	$\mathbf{s}[4,:]$	Z Velocity	$zVel$	$meters/second$
$s_5(t)$	$\mathbf{s}[5,:]$	Mass	$Mass$	$kilograms$
$u_1(t)$	$\mathbf{u}[1,:]$	Thrust Magnitude	$Thrust$	$Newtons$
$u_2(t)$	$\mathbf{u}[2,:]$	Thrust Angle	θ	$degrees$

3.9 State Equations

The problem dynamics are idealized as a vector of parametric differential equations as described in Eqs. (3.23) - (3.27) which correspond to the free body diagram shown in Figure 2.6

$$\dot{s}_1(t) = \dot{x}Pos = \frac{dxPos}{dt} = xVel \tag{3.23}$$

$$\dot{s}_2(t) = \dot{x}Vel = \frac{dxVel}{dt} = \frac{-Thrust * \sin\theta}{Mass} \tag{3.24}$$

$$\dot{s}_3(t) = \dot{z}Pos = \frac{dzPos}{dt} = zVel \tag{3.25}$$

$$\dot{s}_4(t) = \dot{z}Vel = \frac{dzVel}{dt} = \frac{-Thrust * \cos\theta}{Mass} - g_{local} \tag{3.26}$$

$$\dot{s}_5(t) = \dot{M}ass = \frac{dMass}{dt} = \frac{-Thrust}{I_{sp} * g_{\oplus}} \tag{3.27}$$

These equations model the response of the spacecraft to the forces of gravity and thrust under the assumptions previously stated in Chapter 2. The effective specific impulse of the spacecraft, I_{sp}, is assumed to be constant across the entire thrust throttle range. If the specific impulse is expected

to vary significantly throughout the duration of the flight, additional state variables that could be used to model the change would have to be included such as the propellant tank pressure, temperature, etc.

3.10 Optional Constraints

While some VTVL spacecraft may be able to take off at a slight angle, fixed engine crafts such as Craft C in Figure 2.2 are constrained to a vertical takeoff, i.e. $\theta = 0 \text{ @ } t_0$. If this is the case, then the initial thrust level should also be at least the craft's initial weight in the local gravity field in order to take off, i.e. $Thrust = Mass_0 * g_{local} @ t_0$. When landing, the terminal thrust angle may need to be vertical as well, $\theta = 0 \text{ @ } t_{final}$, but the thrust may be higher than the spacecraft's local gravity at the time since it may be shedding residual velocity. These can be enforced by setting appropriate upper and lower bounds for the control variables at t_0 and t_{final} as listed in Table 3.3.

Table 3.3 Optional VTVL Constraint Properties (e.g. Craft C - Figure 2.2)

Variable	Lower Bound @ $t_{initial}$	Upper Bound @ $t_{initial}$	Lower Bound @ t_{final}	Upper Bound @ t_{final}
θ	0	0	0	0
$Thrust$	$Mass_{initial} * g_{local}$	$Thrust_{max}$	$Thrust_{min}$	$Thrust_{max}$

Additionally, the allowable rate of change of the thrust angle can be restricted to model spacecraft constraints by enforcing $\left|\frac{d\theta}{dt}\right| \leq \dot{\theta}_{max}$. A real fixed engine spacecraft has a moment of inertia and cannot instantaneously pitch to change the thrust angle. A gimbaled craft cannot instantaneously change the gimbal position. Additionally, there may be restrictions due to controllability/stability issues, risk reduction concerns, sensor limitations, etc. This is an example of an inequality constraint, and is satisfied as long as the condition in Eq. (3.28) holds.

$$\left|\frac{d\theta}{dt}\right| - \dot{\theta}_{max} \leq 0 \tag{3.28}$$

Chapter 4

Direct Collocation and Nonlinear Programming Trajectory Optimization

4.1 Method History and Literature Review

The method of Direct Collocation to solve optimal control problems was developed by Dickmanns and Well[19] in 1974. Hargraves and Paris[20] pioneered combining Direct Collocation with Nonlinear Programming (DCNLP) for trajectory optimization. Enright and Conway modified the equality constraint by a factor of two thirds the time segment so that it used an implicit Hermite-Simpson integration[21]. Further work has resulted in refined discretization methods to improve algorithm performance for specific cases. These DCNLP methods have been used successfully to optimize the trajectory of both impulsive and finite thrust maneuvers for spacecraft as well as many other optimal control problems[22].

4.2 Direct Transcription

Direct Transcription of the OCP into a Nonlinear Programming[p] (NLP) problem begins with discretizing the simulation time into a numerical monotonic row vector $\mathbf{t}$ of length n in the form

$$\begin{gathered} \mathbf{t} = [t_1, t_2, \dots, t_k, t_{k+1}, \dots, t_n] \\ where \\ t_1 < t_2 < \cdots < t_k < t_{k+1} < \cdots < t_n \end{gathered} \tag{4.1}$$

where t_1 and t_n correspond to the previous terms $t_{initial}$ and t_{final}. Next, the state and control variable functions are discretized into numerical matrices consisting of the *values* of the state and control variables, respectively, at the discrete simulation time points. The individual time points and the corresponding values of the state and control variables are referred to as *node* points. The vector index of any node point is equivalent to the vector index of the corresponding time.

---

[p] The term "programming" in NLP refers to *mathematical programming*, a historical term for optimization.

Matrix indexing is used with matrices, and subscripts are used for vectors. For clarity, the indexing method is $\mathbf{s}[state\ variable\ index, node\ index]$ for the *state variable matrix*, and $\mathbf{u}[control\ variable\ index, node\ index]$ for the *control variable matrix*. The state variable matrix $\mathbf{s}[:,:]$ has dimension of $[5 \times n]$, and the control variable matrix $\mathbf{u}[:,:]$ is $[2 \times n]$, where n is the number of time discretization nodes.

$$\mathbf{t} \rightarrow \mathbf{s}(t) = \begin{bmatrix} s_1(\mathbf{t}) \\ s_j(\mathbf{t}) \\ \vdots \\ s_5(\mathbf{t}) \end{bmatrix} \cong \begin{bmatrix} \mathbf{s}[1,:] \\ \mathbf{s}[j,:] \\ \vdots \\ \mathbf{s}[5,:] \end{bmatrix} = \mathbf{s}[:,:] \tag{4.2}$$

$$\mathbf{t} \rightarrow \mathbf{u}(t) = \begin{bmatrix} u_1(\mathbf{t}) \\ u_2(\mathbf{t}) \end{bmatrix} \cong \begin{bmatrix} \mathbf{u}[1,:] \\ \mathbf{u}[2,:] \end{bmatrix} = \mathbf{u}[:,:] \tag{4.3}$$

An entire solution can be reduced to an $[8 \times n]$ *dynamic solution matrix* $\mathbf{Z}$ consisting of the time vector $\mathbf{t}$ and the corresponding state and control variable values. An individual node $\mathbf{Z}_k$ is an $[8 \times 1]$ column vector cross-section of the solution matrix.

$$\mathbf{Z}_k = \begin{bmatrix} t_k \\ \mathbf{s}[:,k] \\ \mathbf{u}[:,k] \end{bmatrix} \ where\ k = [1{:}n] \tag{4.4}$$

Each node is a unique point within the *search space*, the space of all values the node variables can take for a given node. These individual nodes are represented in Figure 4.1 as red dots, and the solution matrix would be all the nodes from $[1{:}n]$. The line connecting the nodes is the trajectory of the spacecraft in hyper-dimensional dynamic state space, as each node includes the current velocity, mass, and control variables values.

The trajectory segment between any node $\mathbf{Z}_k$ and $\mathbf{Z}_{k+1}$ is referred to as *trajectory segment* τ_k for $k = 1{:}(n-1)$. The *segment width* Δt_k for any *trajectory segment* k is

$$\Delta t_k = t_{k+1} - t_k \tag{4.5}$$

and may or may not be a constant, depending on the node temporal distribution.

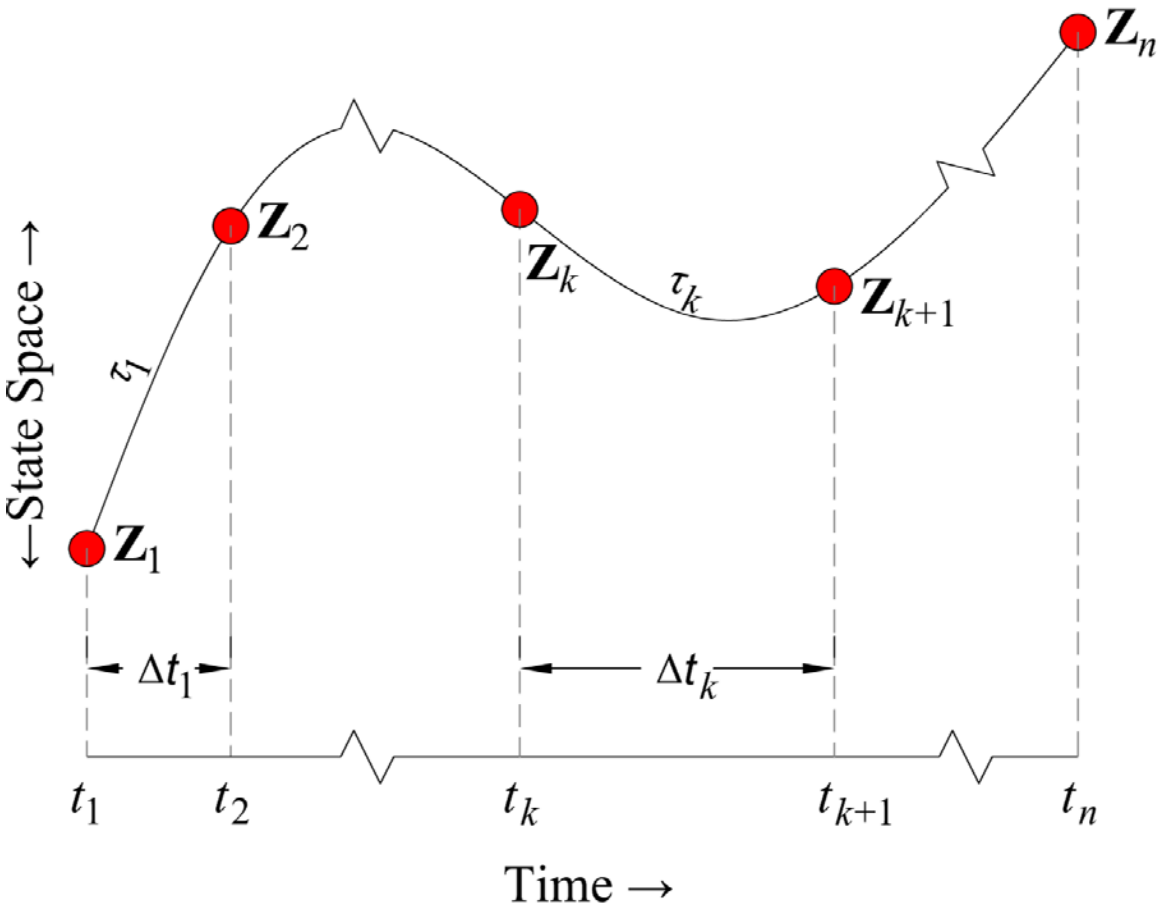

Figure 4.1: Discretization of an Optimal Control Problem into a Nonlinear Programming Problem[q]

4.3 Simple Global Upper and Lower Bounds

Simple global lower and upper bounds on the allowable values of the dynamic variables can be set due to physical, spacecraft, or problem constraints, e.g. minimum and maximum altitudes, thrust levels, etc. These can be imposed on a per node basis to enforce boundary conditions. Setting appropriate constraints can decrease the computational effort required to optimize the

---

[q] Figure 4.1 was initially based off of the work of B. Geiger[23] Note that the vertical axis corresponds to node points within the entire state variable space, not any one state variables in particular.

trajectory by limiting the search space, though care must be taken to ensure that potential optimal trajectories are not excluded.

4.4 Enforcing Feasibility through Equality Constraints

A method must be developed to determine whether a given solution matrix **Z** represents a feasible trajectory, i.e. if the given control law would produce the trajectory and change in spacecraft state variables seen in the solution matrix. Using only the values of the time, state, and control values at the node points bracketing a trajectory segment τ_k, $5 \times (k-1)$ equality constraints in the form of Eq. (3.8) can be developed. If the equality constraint method is valid, when all of the constraints are driven to zero, the solution matrix should represent a feasible trajectory, (though there is no guarantee of optimality).

Essentially, to determine if a trajectory is feasible, the value of the state variables at node $\mathbf{Z}_{k+1}$ are predicted by integrating the state equations $\dot{\mathbf{s}}(t)$ from the values of the state variables at node $\mathbf{Z}_k$ from $t = t_k \rightarrow t_{k+1}$ as per

$$\mathbf{s}[:, k+1] = \mathbf{s}[:, k] + \int_{t_k}^{t_{k+1}} \dot{\mathbf{s}}(t)dt \tag{4.6}$$

where the state equations are evaluated as Eq. (3.2) and therefore at the nodes via

$$\dot{\mathbf{s}}(t_k) = \mathbf{f}(\mathbf{s}[:, k], \mathbf{u}[:, k], \mathbf{p}, t_k) \tag{4.7}$$

Note that the values of the control variables at node $\mathbf{Z}_{k+1}$ and $\mathbf{Z}_k$ are required to calculate the values of the state equations as per Eq. (4.7).

Two inequality constraints are discussed. The first uses simple trapezoidal integration between the nodes. The second utilizes the method of Direct Collocation. As implemented, this results in

an equality constraint that is equivalent to implicit Hermite-Simpson integration, and should be higher fidelity.

4.5 Trapezoidal Feasibility Equality Constraints

Using the notation developed, the trapezoidal approximation of integration is

$$\mathbf{s}[:,k+1] \cong \mathbf{s}[:,k] + \frac{\dot{\mathbf{s}}(t_{k+1}) + \dot{\mathbf{s}}(t_k)}{2} \Delta t_k \tag{4.8}$$

Therefore, if the actual values of the state variables at node $\mathbf{Z}_{k+1}$ in the solution matrix are different than the values predicted by Eq. (4.8), the solution is not feasible. The differences between the predicted and actual state variable values at node $\mathbf{Z}_{k+1}$ are referred to as the *defects*. This process is repeated on a per-node basis and turned into a $[5 \times (n-1)]$ matrix of *trapezoidal feasibility equality constraints,* $\mathbf{C}_{\mathbf{eq}}^{Trap}[:,k]$, as per

$$\mathbf{C}_{\mathbf{eq}}^{Trap}[:,k] = \mathbf{s}[:,k+1] - \mathbf{s}[:,k] + \frac{\dot{\mathbf{s}}(t_{k+1}) + \dot{\mathbf{s}}(t_k)}{2} \Delta t_k \to 0 \tag{4.9}$$

for $k = 1{:}\,(n-1)$, and $\mathbf{C}_{\mathbf{eq}}^{Trap}[:,k]$ is to be driven to zero. Note that this requires that trapezoidal integration is sufficiently accurate to represent the trajectory between nodes.

4.6 Requirements of Simpson-Hermite Integration

Using the notation developed, the Simpson's approximation of integration is expressed as

$$\mathbf{s}[:,k+1] \cong \mathbf{s}[:,k] + \left(\frac{\Delta t_k}{6}\right) * \left(\dot{\mathbf{s}}(t_k) + 4 * \dot{\mathbf{s}}\left(t_k + \frac{\Delta t_k}{2}\right) + \dot{\mathbf{s}}(t_{k+1})\right) \tag{4.10}$$

While the trapezoidal approximation only required the value of the state variables and state equations at the nodes, Simpson's method requires the value of the state equations at the midpoint between the nodes, i.e. $\dot{\mathbf{s}}(t)$ @ $t = t_k + \frac{\Delta t_k}{2}$. However, to calculate the state equations requires the values of the state and control variables at the midpoint as per Eq. (3.2). Thus methods need to be developed first to determine the midpoint values before Simpson's method can be used.

4.7 Control Law Midpoint Linear Interpolation

A simple linear interpolation[r] was used to generate the values of the control variable functions, (the control law), at any simulation time t, using only the values of the control variables at the nodes.

$$\mathbf{u}(t) = \frac{\mathbf{u}[:, k+1] - \mathbf{u}[:, k]}{t_{k+1} - t_k} * (t - t_k) + \mathbf{u}[:, k] \tag{4.11}$$

where k is incremented when $t > t_{k+1}$.

At the midpoints, Eq. (4.11) reduces to

$$\mathbf{u}\left(t_k + \frac{\Delta t_k}{2}\right) = \frac{\mathbf{u}[:, k] + \mathbf{u}[:, k+1]}{2} \tag{4.12}$$

4.8 State Variable Midpoint Interpolation via Direct Collocation

Direct Collocation is a method to estimate the values of the state variables at the midpoint of any trajectory segment τ_k from only the values of the state variables and state equations at the boundary nodes $\mathbf{Z}_k$ and $\mathbf{Z}_{k+1}$, and the segment width Δt_k.

$$\mathbf{s}\left(t_k + \frac{\Delta t_k}{2}\right) = \boldsymbol{f}(\mathbf{s}[:, k], \dot{\mathbf{s}}(t_k), \mathbf{s}[:, k+1], \dot{\mathbf{s}}(t_{k+1}), \Delta t_k) \tag{4.13}$$

The name *Direct Collocation* derives from the central idea of using fictitious piecewise *cubic polynomial functions* juxtaposed or *collocated* alongside the node points of the form

$$\boldsymbol{\sigma}(\tau) = C_0 + C_1 * \tau + C_2 * \tau^2 + C_3 * \tau^3 \tag{4.14}$$

to model the value of the corresponding state variable functions along any trajectory segment τ_k.

$$\boldsymbol{\sigma}(\tau_k) \cong \mathbf{s}(t) \; where \; \tau_k = [0 \to 1] \; for \; t = [t_k \to t_{k+1}] \tag{4.15}$$

For clarity, note that the polynomial functions' independent variable τ is equivalent to the time span $[t_k \to t_{k+1}]$, but mapped to the range $[0 \to 1]$.

---

[r] This assumes that the control variables can change as fast as needed. If limitations exist on the allowable rates of change, these need to be captured through additional constraints, e.g. Eq. (3.28).

The cubic polynomial coefficients can be solved for in terms of their boundary conditions. Substituting $\tau = 0$ and $\tau = 1$ into Eq. (4.14) and its first derivative, the following relations can be developed

$$\begin{bmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\sigma}(0) \\ \dot{\boldsymbol{\sigma}}(0) \\ \boldsymbol{\sigma}(1) \\ \dot{\boldsymbol{\sigma}}(1) \end{bmatrix} \tag{4.16}$$

Matrix inversion allows the cubic coefficients C_0 through C_3 to be defined in terms of the boundary conditions of the polynomial function.

$$\begin{bmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -3 & -2 & 3 & -1 \\ 2 & 1 & -2 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{\sigma}(0) \\ \dot{\boldsymbol{\sigma}}(0) \\ \boldsymbol{\sigma}(1) \\ \dot{\boldsymbol{\sigma}}(1) \end{bmatrix} \tag{4.17}$$

If cubic polynomials can accurately represent the state variable functions along the trajectory segment as per Eq. (4.15), then the boundary conditions of Eq. (4.14) should be equivalent to the values of the state variables and state equations as per

$$\begin{aligned} \boldsymbol{\sigma}(0) &= \mathbf{s}[:, k] \\ \dot{\boldsymbol{\sigma}}(0) &= \dot{\mathbf{s}}(t_k) \\ \boldsymbol{\sigma}(1) &= \mathbf{s}[:, k+1] \\ \dot{\boldsymbol{\sigma}}(1) &= \dot{\mathbf{s}}(t_{k+1}) \end{aligned} \tag{4.18}$$

Therefore, the cubic coefficients can be solved for in terms of the values of the state variables and state equations at nodes $\mathbf{Z}_k$ and $\mathbf{Z}_{k+1}$.

$$\begin{bmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -3 & -2 & 3 & -1 \\ 2 & 1 & -2 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{s}[:, k] \\ \dot{\mathbf{s}}(t_k) \\ \mathbf{s}[:, k+1] \\ \dot{\mathbf{s}}(t_{k+1}) \end{bmatrix} \tag{4.19}$$

And therefore, the values of the state variables at the trajectory segment midpoints, $\left(t_k + \frac{\Delta t_k}{2}\right)$, should be equivalent to Eq. (4.14) evaluated at $\tau = 0.5$.

$$\boldsymbol{\sigma}(0.5) = C_0 + \frac{C_1}{2} + \frac{C_2}{4} + \frac{C_3}{8} \cong \mathbf{s}\left(t_k + \frac{\Delta t_k}{2}\right) \tag{4.20}$$

Using the relations shown in Eq. (4.19) substituted into Eq. (4.20); a relation can be developed to estimate the value of the state variables at the midpoint purely in terms of the boundary conditions.

$$\mathbf{s}\left(t_k + \frac{\Delta t_k}{2}\right) = \frac{\mathbf{s}[:,k] + \mathbf{s}[:,k+1]}{2} + \frac{\Delta t_k * (\dot{\mathbf{s}}(t_k) - \dot{\mathbf{s}}(t_{k+1}))}{8} \tag{4.21}$$

4.9 Simpson-Hermite Feasibility Equality Constraints

Now that the values of the state and control variables at the trajectory segment midpoints can be determined purely from the values of the bounding nodes and state equations as per Eqs. (4.12) and (4.21), the value of the state equations at the midpoint can be derived as per Eq. (3.2).

$$\dot{\mathbf{s}}\left(t_k + \frac{\Delta t_k}{2}\right) = \mathbf{f}\left(\mathbf{s}\left(t_k + \frac{\Delta t_k}{2}\right), \mathbf{u}\left(t_k + \frac{\Delta t_k}{2}\right), \mathbf{p}, t_k + \frac{\Delta t_k}{2}\right) \tag{4.22}$$

Therefore, using the concept of Direct Collocation, the Simpson approximation of integration shown in Eq. (4.10) can be used to construct a $[5 \times (n-1)]$ matrix of *Simpson-Hermite feasibility equality constraints,* $\mathbf{C}_{\mathbf{eq}}^{Simpson}[:,k]$ as per

$$\begin{aligned} \mathbf{C}_{\mathbf{eq}}^{Simpson}[:,k] &= \mathbf{s}[:,k+1] - \mathbf{s}[:,k] \cdots \\ &+ \left(\frac{\Delta t_k}{6}\right) * \left(\dot{\mathbf{s}}(t_k) + 4 * \dot{\mathbf{s}}\left(t_k + \frac{\Delta t_k}{2}\right) + \dot{\mathbf{s}}(t_{k+1})\right) \to 0 \end{aligned} \tag{4.23}$$

where $k = 1{:}\,(n-1)$. As long as the assumption that a cubic polynomial can accurately represent the state variable functions along each trajectory segment is valid, then $\mathbf{C}_{\mathbf{eq}}^{Simpson}[:,k]$ should be able to be driven to zero, and if so, the solution matrix should represent a feasible trajectory. Further notes on Direct Collocation for clarity can be found in Appendix A.

4.10 NLP Variables, Node Distribution, and Time as a Fixed or Free Variable

Each variable that a NLP solver needs to optimize a trajectory with are referred to as Nonlinear Programming Variables, (NLPVs). The distinction of NLP_STATE is made for the subset of NLPVs which are state variables, and NLP_CONTROL for control variables. Since a trajectory solution $\mathbf{Z}$ is represented by an $[8 \times n]$ matrix of the node times, state variables, and control variables, there are $5 \times n$ NLP_STATE variables and $2 \times n$ NLP_CONTROL variables. The NLP solver can freely change the values of any of the NLP_STATE and NLP_CONTROL variables in order to optimize the trajectory within the upper and lower bounds and satisfy the equality and inequality constraints.

The time vector $\mathbf{t}$ was treated differently. Instead of allowing the NLP solver to change the node time value t_k at each node freely, the time vector $\mathbf{t}$ was determined using a linear time distribution from $[t_{initial}: t_{final}]$, where $t_{initial} = 0$ and t_{final} was either fixed, or free to float between an upper and lower bound. If t_{final} was fixed, then the time vector in the solution matrix $\mathbf{Z}$, (the top row), was fixed for the simulation as well.

If t_{final} was a free variable, in order to preserve the same relative node distribution, the time vector $\mathbf{t}$ in the solution matrix was remapped each program iteration to the value of one additional NLPV variable which represented the simulation duration, t_{final}. Therefore the number of NLPVs was reduced to $7 \times n$ if time was fixed, or $(7 \times n) + 1$ if time was free.

4.11 NLPV Upper Bounds, Lower Bounds, and Enforcing Boundary Conditions

Upper and lower time-invariant bounds for each NLP_STATE variable were calculated as per the time-invariant Eqs.(3.13)-(3.22) in the form of $\mathbf{s}[:,:]_{lower}$ and $\mathbf{s}[:,:]_{upper}$ for all nodes. Bounds were set per Table 3.1 for all NLP_CONTROL variables as $\mathbf{u}[:,:]_{lower}$ and $\mathbf{u}[:,:]_{upper}$ as well.

Next, in order to enforce the boundary conditions $\boldsymbol{\psi}_{initial}$ and $\boldsymbol{\psi}_{final}$, the upper and lower bounds were set equal to the boundary conditions for all fixed variables[s] at the column indices corresponding to the first and last nodes.

$$\begin{bmatrix}\mathbf{s}[:,1]_{lower}\\ \mathbf{u}[:,1]_{lower}\end{bmatrix} = \boldsymbol{\psi}_{initial,fixed} = \begin{bmatrix}\mathbf{s}[:,1]_{upper}\\ \mathbf{u}[:,1]_{upper}\end{bmatrix} \tag{4.24}$$

$$\begin{bmatrix}\mathbf{s}[:,n]_{lower}\\ \mathbf{u}[:,n]_{lower}\end{bmatrix} = \boldsymbol{\psi}_{final,fixed} = \begin{bmatrix}\mathbf{s}[:,n]_{upper}\\ \mathbf{u}[:,n]_{upper}\end{bmatrix} \tag{4.25}$$

Free variables at the boundary conditions are still limited to the global limits already applied. If the time duration was a free variable, bounds were set according to Eqs. (3.11) and (3.12).

4.12 Initial Guess Derivation

In order to improve algorithm performance, an initial guess for the simulation duration/time of flight was made according to

$$t_{final,guess} = \ 1.5 * TOF_{ballistic} \tag{4.26}$$

The initial guess time vector, $\mathbf{t}_{guess}$, was then created using a linear distribution between $[0: t_{final,guess}]$. Guess values for the state position and velocity variables were developed from a ballistic trajectory using element-wise multiplication or division along $\mathbf{t}_{guess}$.

---

[s] Note that the colon used in Eq. (4.24) and Eq. (4.25) indicates that multiple row elements are being set, though not necessarily all, since not every state and/or control variable is a fixed boundary condition.

$$xPos_{Guess} = \quad \mathbf{s}[1,:]_{guess} = V_{\text{initial,ballistic}} * \cos(45) * \mathbf{t} \tag{4.27}$$

$$xVel_{Guess} = \quad \mathbf{s}[2,:]_{guess} = V_{\text{initial,ballistic}} * \cos(45) \tag{4.28}$$

$$zPos_{Guess} = \quad \mathbf{s}[3,:]_{guess} = \frac{g_{local}}{2} * \mathbf{t}^2 + V_{\text{initial,ballistic}} * \sin(45) * \mathbf{t} \tag{4.29}$$

$$zVel_{Guess} = \quad \mathbf{s}[4,:]_{guess} = g_{local} * \mathbf{t} + V_{\text{initial,ballistic}} * \sin(45) \tag{4.30}$$

For the spacecraft mass, the guess was a simple linear distribution between the initial mass and the initial mass less the ballistic propellant required.

$$Mass_{Guess} = \quad \mathbf{s}[5,:]_{guess} = Mass_{initial} - \frac{\mathbf{t}}{t_{final,guess}} * Fuel\ Mass_{\ impulsive} \tag{4.31}$$

For the control law, a simple guess was made for the thrust to go from $Thrust_{max}$ to $Thrust_{min}$ at the TOF midpoint and then ramp back to $Thrust_{max}$.

$$Thrust_{Guess} = \mathbf{u}[1,:]_{guess} = \begin{Bmatrix} Thrust_{max} * \left(1 - \frac{2 * \mathbf{t}}{t_{final,guess}}\right) for\ \mathbf{t} \leq \frac{t_{final,guess}}{2} \\ Thrust_{max} * \left(\frac{2 * \mathbf{t}}{t_{final,guess}} - 1\right) for\ \mathbf{t} \geq \frac{t_{final,guess}}{2} \end{Bmatrix} \tag{4.32}$$

For the thrust angle, a guess was made equivalent to the negative flight path angle of a ballistic trajectory, as the control variable θ and flight path angle α shown in Figure 2.6 would be equal in magnitude but opposite in sign in the absence of gravity losses.

$$\theta_{Guess} = \quad \mathbf{u}[2,:]_{guess} = \tan^{-1}\left(\frac{zVel_{Guess}}{xVel_{Guess}}\right) = \tan^{-1}\left(\frac{\mathbf{s}[3,:]_{guess}}{\mathbf{s}[1,:]_{guess}}\right) \tag{4.33}$$

Before being used, any nodes that might have been outside the bounds were set to be within limits on a per element basis according to

$$\mathbf{s}[:,:]_{guess} = \begin{Bmatrix} maximum(\mathbf{s}[:,:]_{guess}, \mathbf{s}[:,:]_{lower}) \\ minimum(\mathbf{s}[:,:]_{guess}, \mathbf{s}[:,:]_{upper}) \end{Bmatrix} \tag{4.34}$$

$$\mathbf{u}[:,:]_{guess} = \begin{Bmatrix} maximum(\mathbf{u}[:,:]_{guess}, \mathbf{u}[:,:]_{lower}) \\ minimum(\mathbf{u}[:,:]_{guess}, \mathbf{u}[:,:]_{upper}) \end{Bmatrix} \tag{4.35}$$

4.13 VTVL and Pitch Rate Constraint Enforcement

If the VTVL constraint is being enforced, then then boundary conditions listed in Table 3.3 are included in the fixed variables, i.e. $\boldsymbol{\psi}_{initial,fixed}$ and $\boldsymbol{\psi}_{final,fixed}$.

For the maximum θ angular rate of change constraint listed in Eq. (3.28), a discrete inequality constraint was used in the form

$$\boldsymbol{C_{inequality}}: \quad \left| \frac{\mathbf{u}[2,k+1] - \mathbf{u}[2,k]}{t_{k+1} - t_k} \right| - \dot{\theta}_{max} \leq 0 \tag{4.36}$$

4.14 Fitness Function

Since the feasibility constraint ensures that the trajectory obeys the problem dynamics, and inequality constraints cover additional spacecraft specific limitations, the fitness function is reduced to maximizing the spacecraft's final mass, i.e. minimizing the propellant use required for a VTVL maneuver. Using the established matrix indexing,

$$J = \ maximize\ \mathbf{s}[5,n] \tag{4.37}$$

4.15 Nonlinear Programming Solver

While any number of nonlinear solvers can be utilized, the *fmincon.m* nonlinear optimization routine was used from MATLAB 2012b with Version 6.2.1 of the Optimization Toolbox. Any changes from the default settings are noted in the results. Note that MATLAB's online documentation and help are excellent references when working with *fmincon* and NLP in general.

4.16 Inherent Error

Note that the Hermite-Simpson feasibility equality constraint is used as it should be more accurate than the trapezoidal. However, note that the equality constraints cannot be driven all the

way to zero, but rather to a small tolerance on the order of $Tol_{con} = 10^{-4}$ to 10^{-8}. There is always some error inherent with discretization, similar to representing a circle as an n-sided polygon. While discretizing the OCP into a greater number of nodes will generally increase the solution fidelity, it also increases the computational time and resources required to produce the result, since the number of NLPVs and equality constraints scales by $5 \times (n-1)$. If a low number of nodes is used and a better solution exists at a higher node resolution than chosen, the solver will not be able to find it. The challenge is to determine when a solution is "good enough" – when the error is sufficiently low – that adding additional nodes is not worth the additional computational time and resources.

Chapter 5

Direct Collocation Results

5.1 Nominal Spacecraft and Solver Parameters

The spacecraft, mission, and solver parameters listed herein were used for all results unless otherwise noted. The time independent parameters, **p**, are shown in Table 3.1 and correspond to Figure 2.3 and Figure 2.4.

Table 5.1: Nominal Spacecraft and Problem Time Independent Parameters

Parameter	Abbreviation	Value	Reasoning
Translation Distance	$xPos_{Final}$	$500\ m$	Significant distance from initial landing
Initial Mass	$Mass_{Initial}$	$100\ kg$	Small scale mission range
Minimum Thrust	θ_{min}	$-60°$	Risk reduction/Spacecraft limit
Maximum Thrust	θ_{max}	$+60°$	Risk reduction/Spacecraft limit
Effective Specific	I_{sp}	$150\ s$	Sufficient to demonstrate asymmetry
Local Gravity	g_{local}	$1.622\ m$	Lunar surface
Max Pitch Rate	$\dot{\theta}_{max}$	∞	Initially unconstrained

Table 5.2 Parameterized Nominal Constraints

Parameter	Abbreviation	Calculation	Value if all nominal values used
Maximum Thrust	$Thrust_{max}$	$3 * Mass_{Initial} * g_{local}$	$486.6\ N$
Minimum Thrust[t]	$Thrust_{min}$	$Thrust_{max}/1000$	$0.4866\ N$

Table 5.3 Nonlinear Programming Parameters

Parameter	Value
Feasibility Equality Constraint	Hermite-Simpson
Number of Discretization Nodes	15

Table 5.4: Parameters used for MATLAB's *fmincon* NLP Solver

Parameter	Value
'Solver'	sequential quadratic programming[u]
'MaxFunEvals'	2e5
'TolCon'	1e-6
'MaxIter'	1e4

[t] See Section 5.3 for why the minimum thrust is slightly above zero.
[u] There is an order of magnitude or more improvement in *fmincon*'s performance in both speed and convergence success when using the 'sqp' solver compared to the default 'interior-point'.

5.2 Unconstrained Solution

Using the nominal spacecraft and problem parameters listed in Section 5.1, the optimal trajectory profile found and the control law which produced it is shown in Figure 5.1. This is essentially a finite equivalent of burn-coast-burn. Accelerate at the maximum thrust available, cutoff thrust and coast, and then perform a similar, though not identical, landing burn.

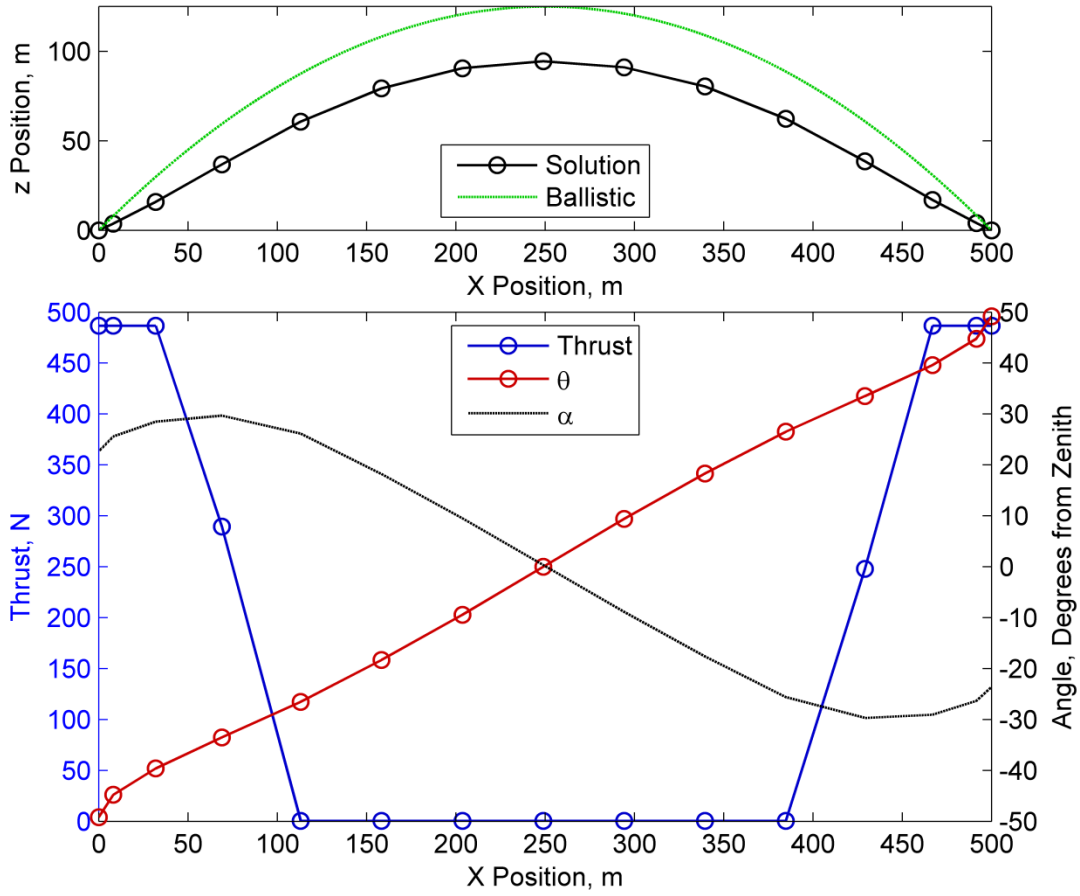

Figure 5.1: Nominal Solution Trajectory Profile and Control Law

The control variables are plotted against the X Position so they align with the XZ trajectory profile. While useful to gain a sense of the flight path of the vehicle, it does not show how the velocity or mass of the vehicle changes over time, and distorts the flight path angle plotline. Note that the flight path angle shown at the first and last node is extrapolated from the interior nodes as $\tan^{-1}\left(\frac{zVel}{xZel}\right)$ is undefined when the velocity is zero for both.

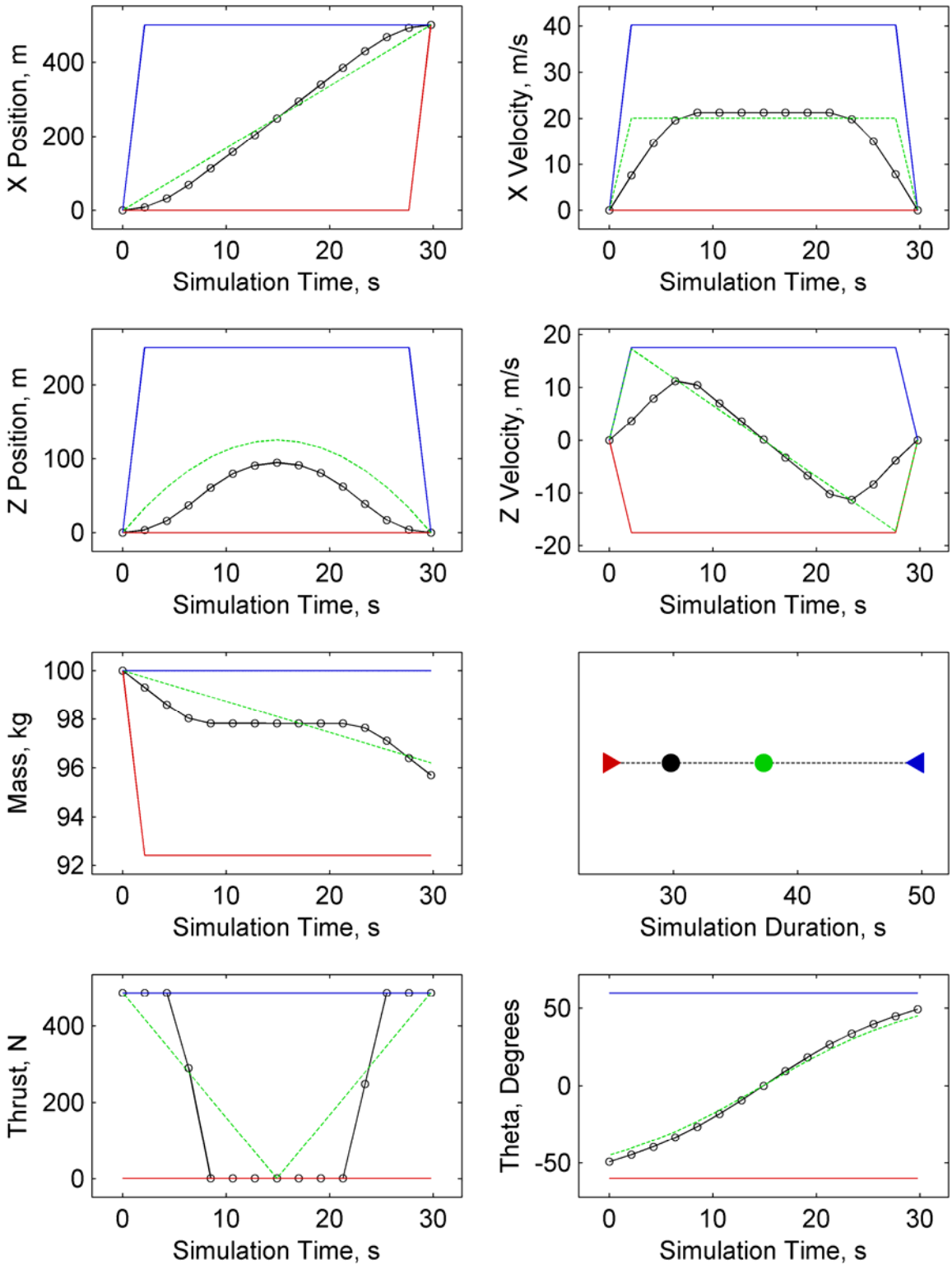

Figure 5.2: Bounded Search Space with Guess and Final Trajectories

Figure 5.2 shows the nominal upper bounds, lower bounds, solution values, and guess values for all NLPVs. For the state and control variables, the actual solution is shown with circular markers, the guess values as a dashed green line, and the lower and upper bounds are shown in red and blue, respectively. The Simulation Duration shows the optimal solution's TOF as a black dot, the guess TOF in green, and the allowable bounds are shown as triangle markers facing inward.

This gives a much more complete picture of the trajectory. The changes in the state and control variables are shown with respect to time instead of position. Each node point at a specific time corresponds to the values of each of the dynamic variables within the solution matrix at the respective node time. Because a linear time distribution was maintained, the temporal spacing is identical between each node.

In the beginning of the flight, the spacecraft gradually accelerates in the X and Z directions. When the thrust cuts off, the spacecraft essentially coasts. The X velocity plateaus and the mass remains constant, though the Z velocity drops due to gravity. The spacecraft then performs a landing maneuver. Because of the time required to accelerate and the corresponding gravity losses, the TOF and propellant use are both greater than the impulsive case. Although difficult to see here since the spacecraft's mass only dropped a few percent, the control law is not symmetric.

The upper and lower bounds are shown mapped to the times corresponding to the final time distribution. It is important to understand that the upper and lower bounds are time-invariant, so while the initial guess for the simulation duration is longer than the final solution's TOF, the bounds are tied to the node points' indices in the solution matrix and are independent of any changes to the time vector while the solver is iterating on a solution. This limits the usefulness of

the simple bounds to enforce additional path constraints as discussed herein. Also note that the initial guesses for the X and Z velocity were modified to be within the boundary conditions as per Eq. (4.34).

These eight graphs collectively illustrate the total variable space that the solver was able to search in as well as the enforcement of the boundary conditions. The solver attempts to optimize the fitness function and satisfy the feasibility equality constraints by adjusting the value of each dynamic variable at each node within the upper and lower limits. Because the upper and lower limits converge for the fixed boundary conditions such as position and velocity, the NLPVs that correspond to those indices in the solution matrix could not be changed. The solution would fail to converge if the solver could not find a feasible set of points connecting from the initial to final conditions.

As previously discussed in Section 4.13, if a solver returns a solution where the NLPVs are only equivalent to the upper or lower bounds at the boundary conditions, this indicates that the optimality of the solution is not constrained by those limits. Since the solution trajectory does not ride along the upper or lower limits for any of the state variables or the thrust angle, the bounds established by Eqs. (3.13)-(3.22) and the θ_{max} and θ_{min} limits should not be artificially distorting the solution under these parameters. However, this does indicate that the optimality of the solution would increase if the maximum allowable thrust was raised.

5.3 Control Asymmetry

To highlight the lack of symmetry between the liftoff and landing burns, the spacecraft effective specific impulse was dropped by an order of magnitude, and the liftoff control law thrust values from the first half of the flight were superimposed over the values of the second half as shown in

Figure 5.3. The area shown in gray is the reduction in total impulse[v] between the "liftoff" and "landing" flight segments. While a propellant with an I_{sp} of 15 seconds is unlikely to ever be considered for use, it is helpful to highlight the lack of symmetry.

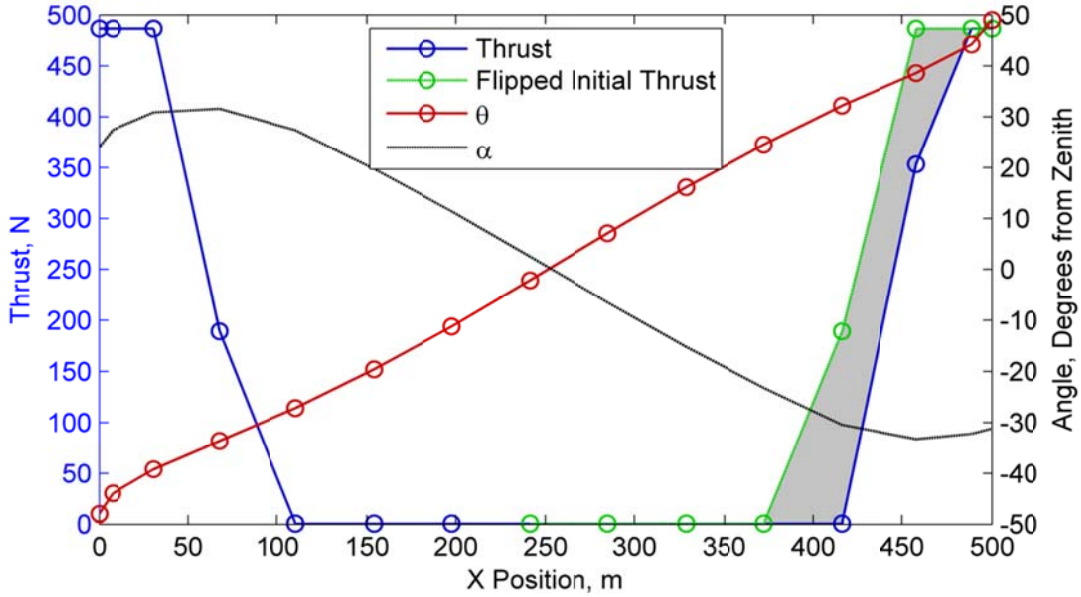

Figure 5.3: Control Law for $I_{sp} = 15$ *seconds* with Liftoff Thrust Mirrored

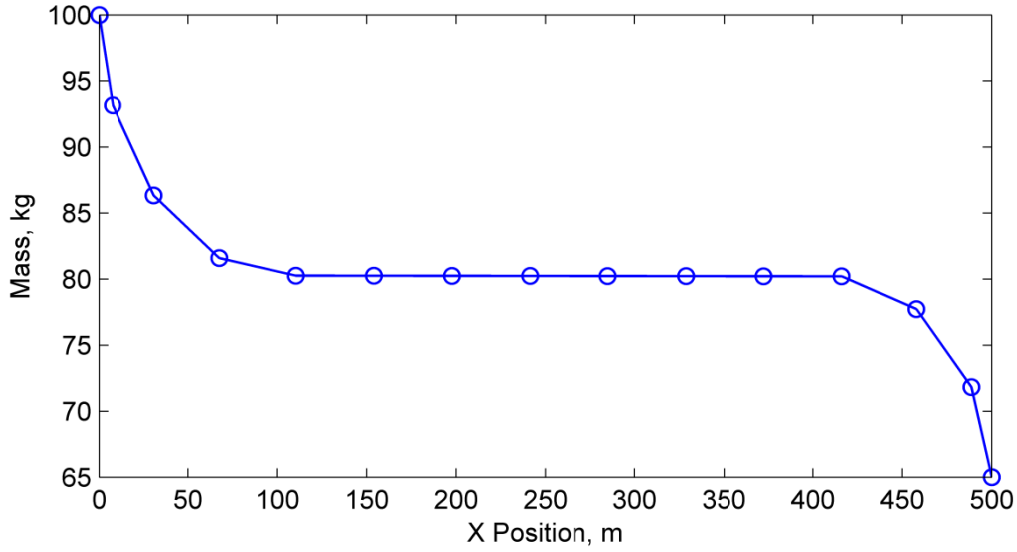

Figure 5.4: Decrease in Spacecraft Mass During Flight with $I_{sp} = 15$ *seconds*

[v] For clarity, impulse here refers to the integral of thrust over time.

With the low I_{sp}, the spacecraft uses a significant amount of propellant performing the liftoff segment, and consequently a smaller impulse is needed for the landing segment. This asymmetry is clearly seen in a plot of the spacecraft's mass over time as in Figure 5.4. Roughly 20 kg of propellant is used for the liftoff burn, while only 15 kg is used for the landing. It is worth noting that the propellant required is an order of magnitude higher than the nominal case where the I_{sp} = 150 seconds.

5.4 Preventing Solver Information Loss

The spacecraft does go up against the minimum thrust bounds in the nominal solution. This is consistent with expectations of the optimal solution attempting to approximate a ballistic burn-coast-burn. While coasting, the optimal thrust should be zero to conserve propellant. While coasting, while the thrust is zero, the thrust angle θ has no effect of the spacecraft's trajectory and therefore fitness. Thus, no information is available to the solver to drive what the optimum θ should be while the spacecraft is coasting. This information loss is reflected by the more or less random values of θ seen in Figure 5.5 between the nodes when the thrust is allowed to go completely to zero, nodes 5-11. While the rapidly changing θ values when the thrust is zero have no effect on the fitness and are not really meaningful in this simulation, it was desired to have the results match as closely as possible to the behavior of a real spacecraft and prevent this behavior.

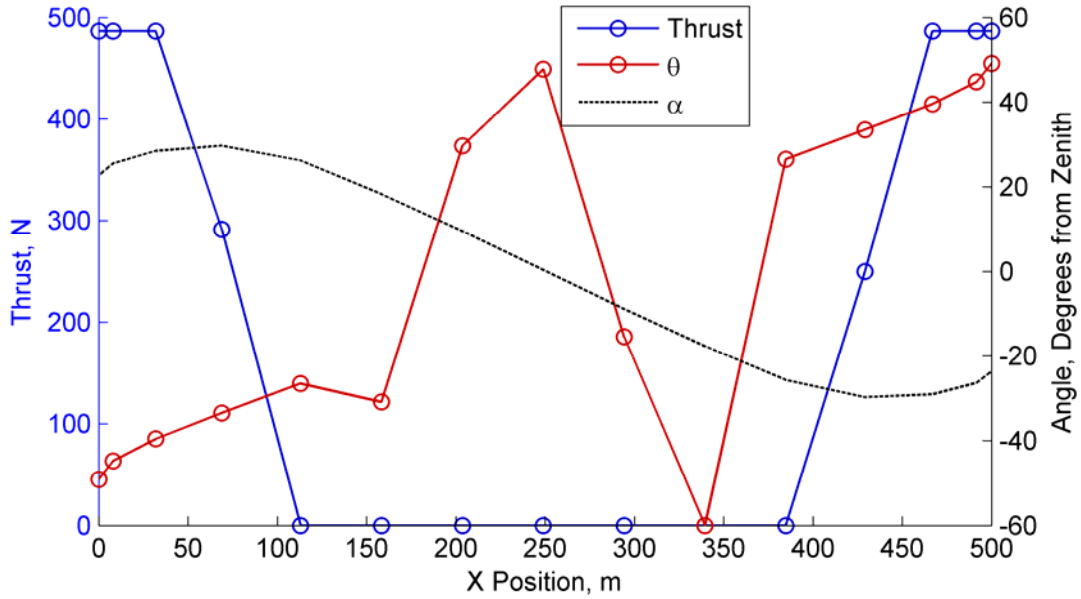

Figure 5.5: Control Law with $Thrust_{min} = 0$

In order to smooth out the observed control law, the thrust angular rate of change, $\frac{d\theta}{dt}$, could be constrained to be below some maximum value, $\dot{\theta}_{max}$, through an inequality constraint as described in Eqs. (3.28) and (4.36). A value of $\dot{\theta}_{max} = 4°/second$ was found to generate very smooth thrust angle curves as seen in Figure 5.6 without increasing the required propellant.

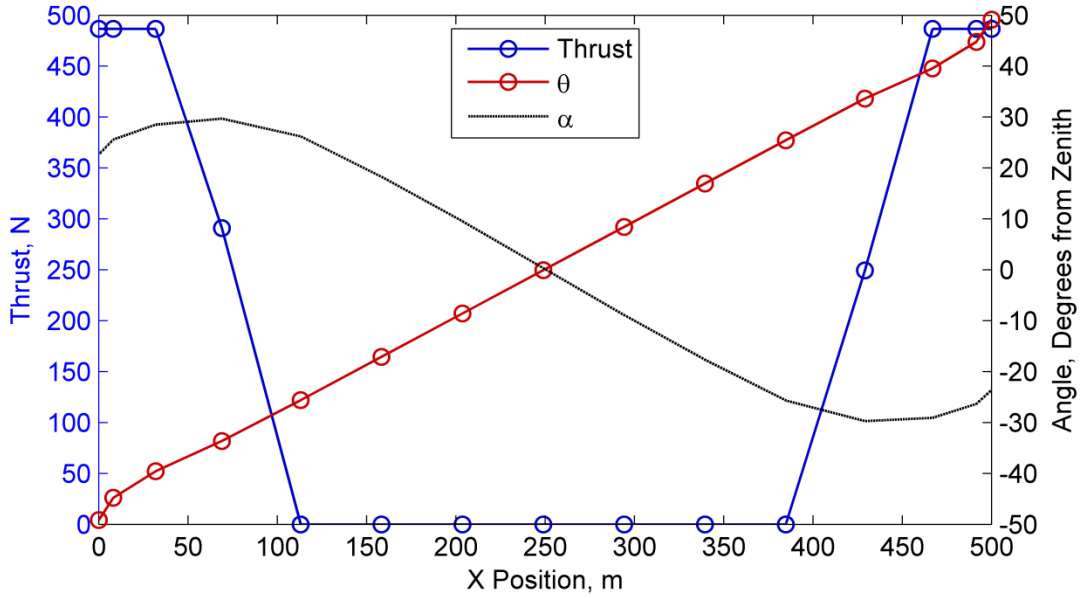

Figure 5.6: Control Law with $Thrust_{min} = 0$ and $\dot{\theta}_{max} = 4°/second$

The easiest method found to smooth the thrust angle while the spacecraft is coasting is to prevent the thrust from actually going to zero which eliminates the need to use the $\dot{\theta}_{max}$ inequality constraint. This is essentially a small numerical fix to give the solver enough information to prevent the random walk[w] seen in in Figure 5.5. Preventing the thrust from going to zero does require additional propellant, on the order of the $Thrust_{min}$:$Thrust_{max}$ ratio, 10^{-3}. This small amount is considered to be within the noise.

5.5 Increasing the Thrust over Weight Ratio

The solver also hit the upper thrust bound in the nominal solution shown in Figure 5.2, indicating that increasing the T/W[x] ratio of the spacecraft should result in an increase in solution fitness. Simulations were run across several T/W ratios to determine the effect of increasing and decreasing the thrust on the required propellant and optimal TOF, as shown in Figure 5.7 and Figure 5.8, respectively.

The results are consistent with expectations. Increasing the thrust reduces gravity losses and reduces the TOF since a smaller amount of time is needed to accelerate and decelerate the spacecraft prior to the coasting period. With increasing thrust, the required propellant and TOF both asymptotically approach the theoretical ballistic-impulsive minimum propellant and ballistic TOF calculated using Eqs. (2.2) and (2.6) for this translation distance. Note that the propellant and TOF ratios over the ballistic quantities are shown on the right axis.

---

[w] The path is not technically "random" as NLP is deterministic for a given set of conditions.
[x] As the spacecraft's mass and weight decrease over the flight; the T/W ratio is set from the initial weight.

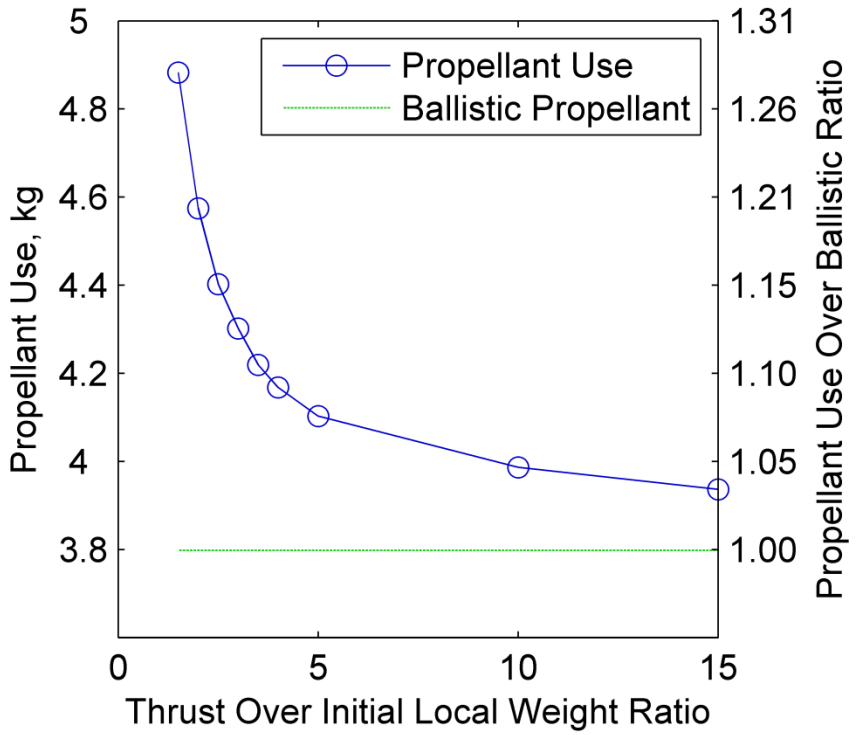

Figure 5.7: Effect of Increasing T/W Ratio on Fitness

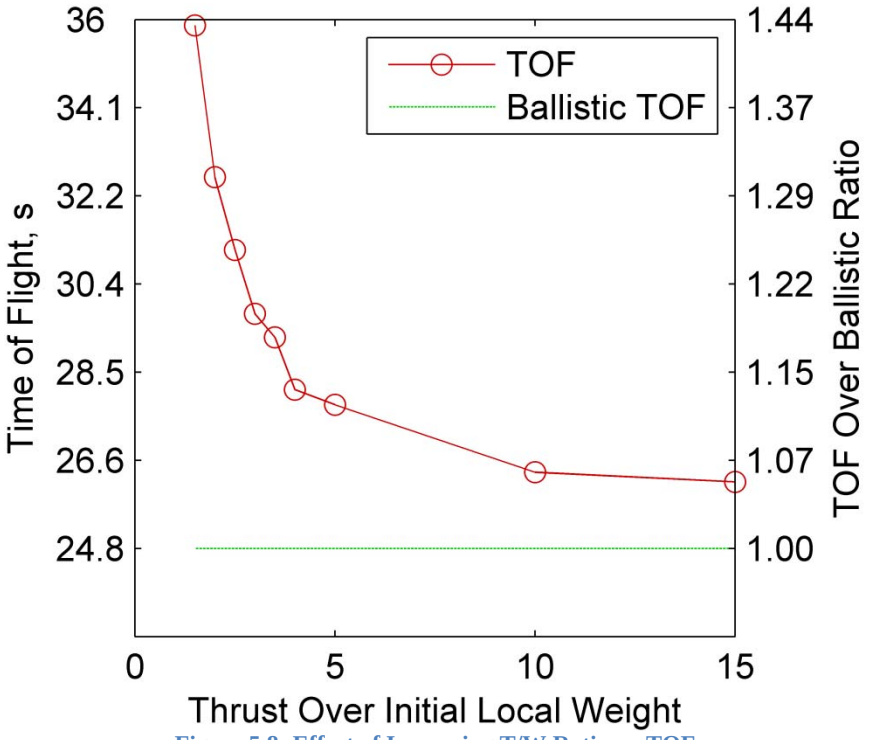

Figure 5.8: Effect of Increasing T/W Ratio on TOF

As discussed in Section 2.3, spacecraft capable of landing typically require a propulsion system able to not only throttle below the local weight, but also operate in a closed loop system capable of small adjustments in real-time to account for noise. Thus, the higher the nominal thrust level of the engine, the higher the required throttle range (minimum thrust level) needs to be in order to land. The relative throttle sensitivity requirements also increase as well. For example, a propulsion system with a T/W of 3 might need to throttle between 15-30% to land, but a spacecraft with a T/W of 9 might need to throttle between 5-10% to land.

Since engine mass and size typically scale with higher peak thrust outputs, eventually the propellant mass savings from increasing the thrust are offset by the increased engine mass and support structures. The additional throttling requirements need to be considered as well. This type of T/W analysis is useful to drive VTVL spacecraft propulsion system design requirements and/or engine selection.

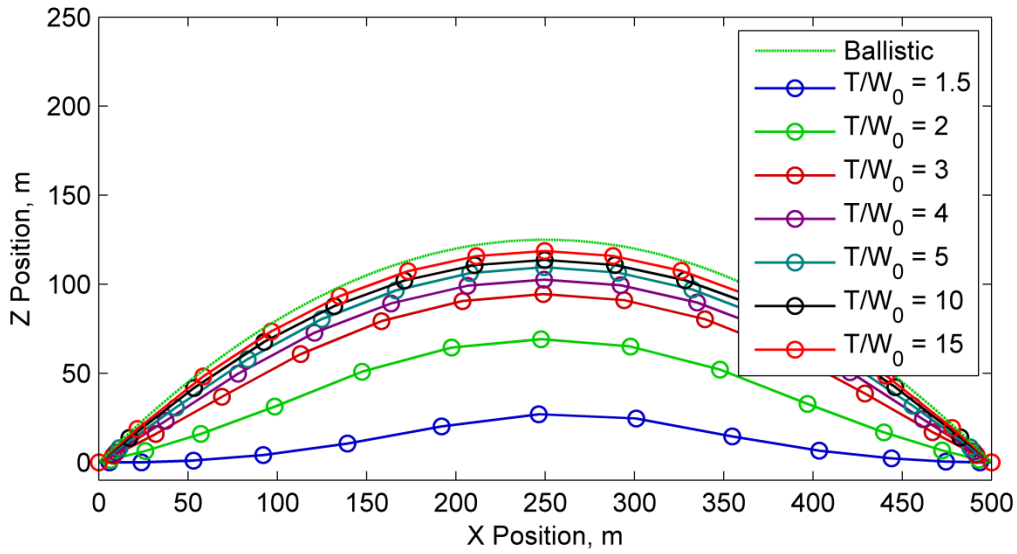

Figure 5.9: Selected Solution Trajectories Resulting from Varying T/W Ratios

Changing the T/W ratios also dramatically affects the XZ trajectory profile. Optimum trajectory profiles generated using selected T/W ratios are shown in Figure 5.9. The profile shape,

maximum height reached, and initial flight path angle also asymptotically approach the ballistic case with a higher T/W ratio. For a lower T/W ratio, the spacecraft barely leaves the ground for the first and last 50 meters.

5.6 Enforcing a Floor Constraint

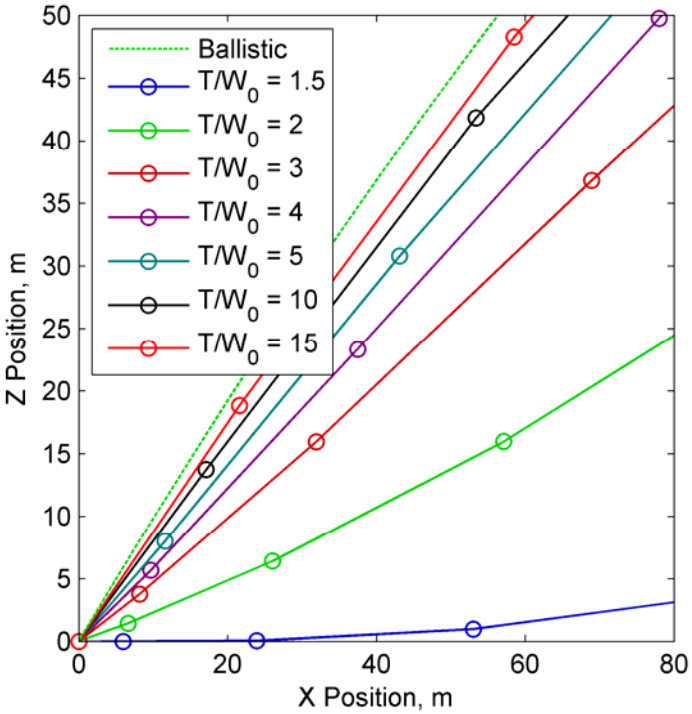

Figure 5.10: Initial Flight Paths Resulting from Varying T/W Ratios

A zoomed in view of Figure 5.9 near the origin is shown in Figure 5.10. Note that for the low T/W profiles, a "floor" constraint may be required to ensure the spacecraft maintains an acceptable clearance from the surface.

A possible way of doing this, though not optimal, is to raise the Z Position lower bound for all interior nodes (all nodes except for the first and last). In Figure 5.11, a T/W ratio of 1.5 was used in all cases, and the Z Position lower bound swept across $[0:1:2]$. Note that the trajectory profiles are raised significantly for only a small increase in the required solution propellant mass.

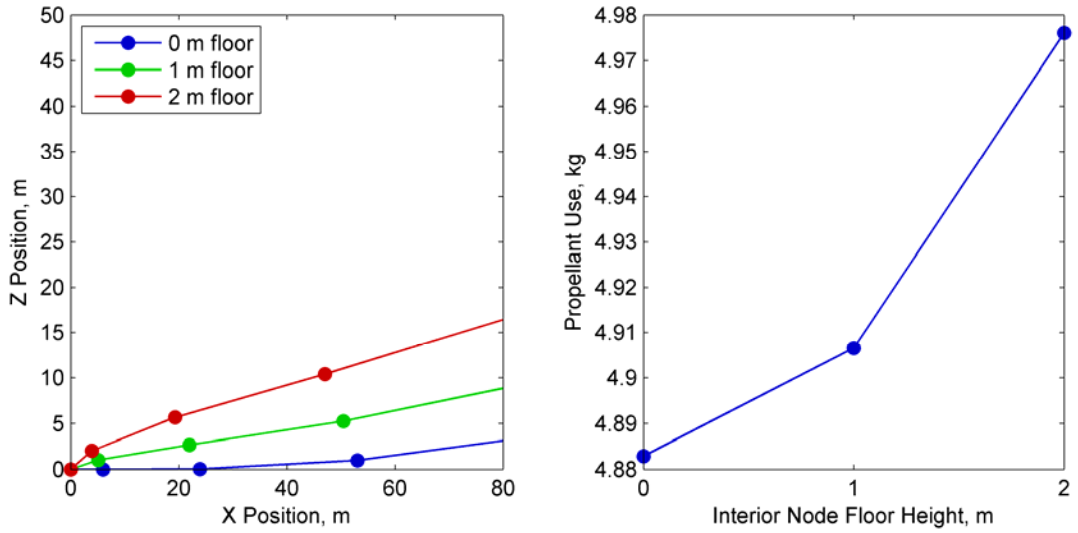

Figure 5.11: Effect of Raising Z Position Lower Bound on Fitness and Initial Flight Path for T/W = 1.5

However, enforcing floor constraints through manipulating the Z Position lower bound can be problematic since it depends on the number of discretization nodes used and the hopping distance. With an increased number of nodes, it may not be possible to reach the Z Position lower bound before the next node point in the time required. With a shorter hopping distance, the optimal height of the next node may be lower than the constraint. This can lead to an artificially constrained solution or a failure to converge.

Although enforcing a floor constraint through the node lower bounds was sufficient for the maneuvers studied herein (in part because of the default number of nodes used), it would be better to create a position-dependent inequality constraint in the form

$$\boldsymbol{C_{inequality}}\colon\ zPos_{floor} - zPos \leq 0, \quad while\ 0 > xPos < xPos_{final} \tag{5.1}$$

to enforce an interior floor constraint.

5.7 Effects of Increasing the Number of Discretization Nodes

A sweep of changing the number of discretization nodes was done for $n = [5{:}\,5{:}\,25]$. Figure 5.12 and Figure 5.13 indicate that both the required propellant mass and TOF slightly decrease as the number of nodes is increased. As contrasted with increasing the T/W ratio, both of these behaviors result from an increase in the fidelity of the results, not an actual increase in the potential solution fitness. This is why both graphs asymptotically approach a value greater than the theoretical ballistic minimum propellant and TOF. Note that the ratios of propellant use and TOF to the ballistic case are shown on the right axes.

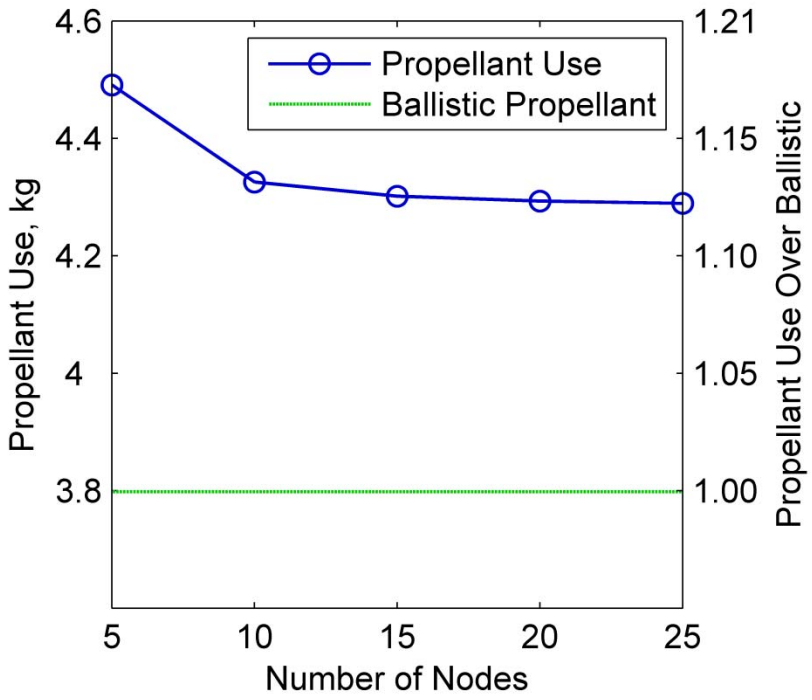

Figure 5.12: Effect of Increasing Number of Discretization Nodes on Propellant Use

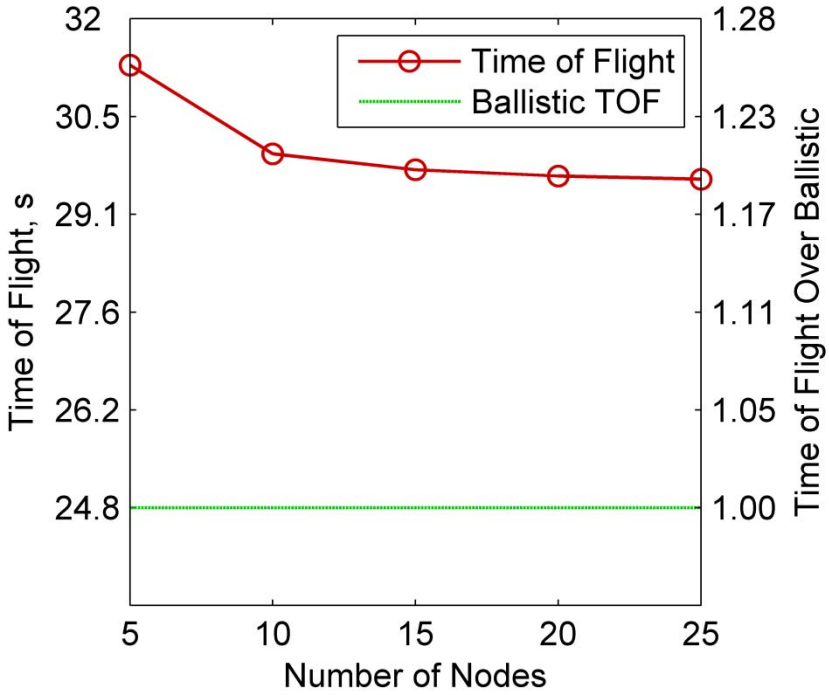

Figure 5.13: Effect of Increasing Number of Nodes on Fitness and the Optimal TOF

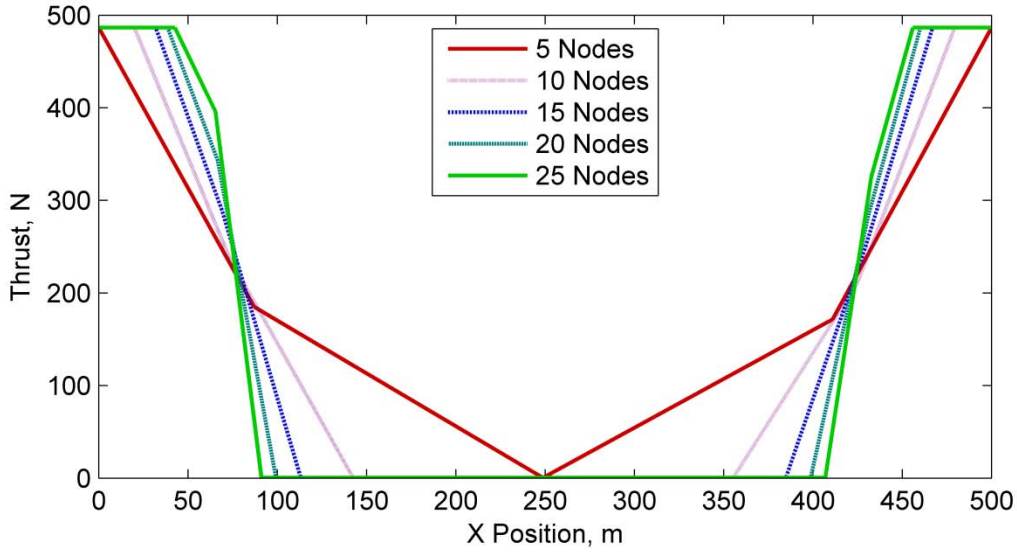

Figure 5.14: Value of the Optimal Thrust Control Law with Increasing Nodes

Figure 5.14 shows how the optimal thrust control law changes with an increase in nodes. As the number of discretization nodes increases, the node width decreases. Instead of the thrust slowly falling off and slowly ramping back up as seen when $n = 5$, the optimal behavior as the number of

nodes increases is for the thrust to rapidly switch from $Thrust_{max}$ to $Thrust_{min}$ as fast as possible, ideally over one node width. This indicates that the minimum energy solution and therefore optimal solution would be for thrust to instantaneously switch off at a specific point in time as $n \to \infty$. This closer approximation to the minimum energy case drives the decrease in propellant use and TOF seen in Figure 5.12 and Figure 5.13.

This increase in fidelity and fitness comes at a cost. Figure 5.15 highlights the power law relationship between the number of function evaluations (NFE) the solver performs and the number of discretization nodes. A default of 15 nodes was chosen to strike a balance between the computation time and accuracy of results.

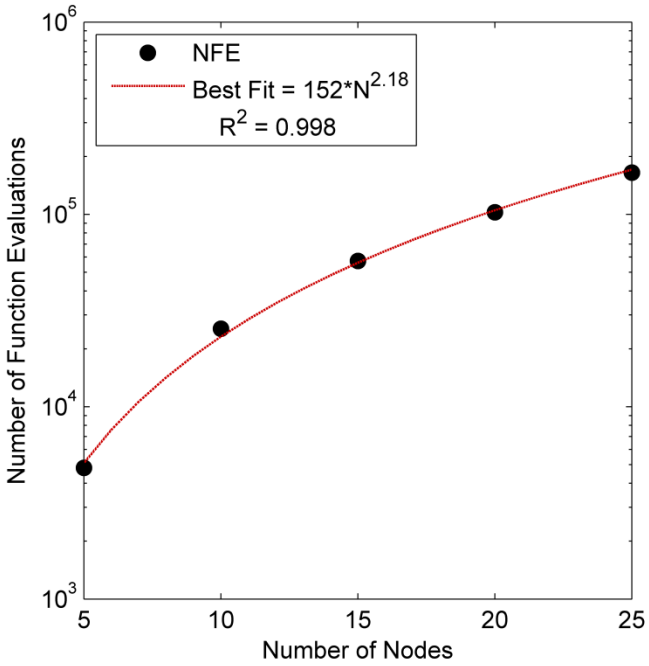

Figure 5.15: Increase in Number of Function Evaluations versus Number of Discretization Nodes

5.8 Demonstration of Optimal Solution Time of Flight

To show that the solver was successfully optimizing the TOF in addition to the propellant mass, a range of fixed TOFs were used centered on the optimal TOF previously found for the nominal solution. The trajectory profiles shown in Figure 5.16 and results shown in Figure 5.17 indicate

that if the fixed TOF is less than the optimal TOF, the spacecraft takes a shallower, faster path, and uses more propellant. If the fixed TOF is greater than the optimal TOF, the spacecraft goes higher to essentially "waste" the extra time, using extra propellant in the process to fight gravity.

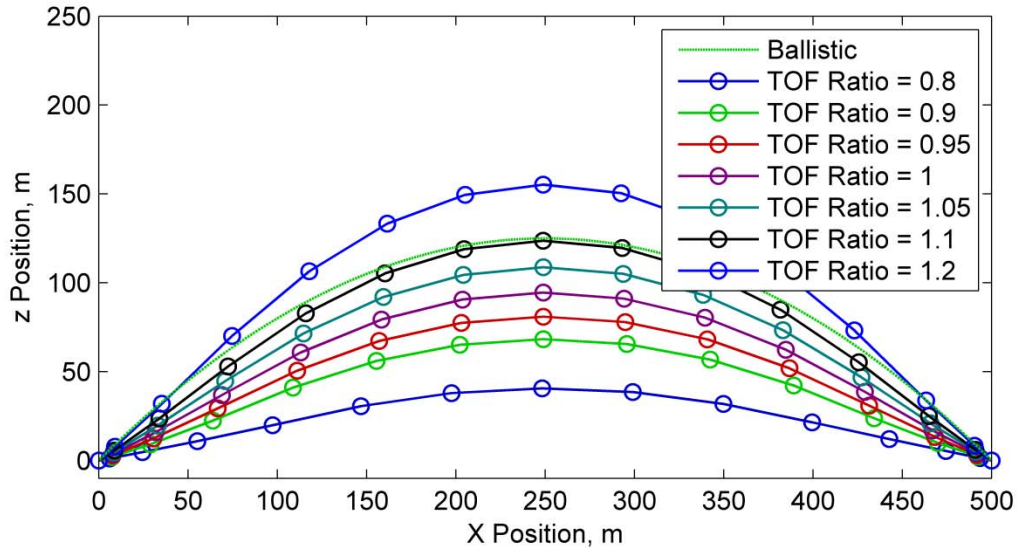

Figure 5.16: Trajectory Profiles with Various Fixed TOFs

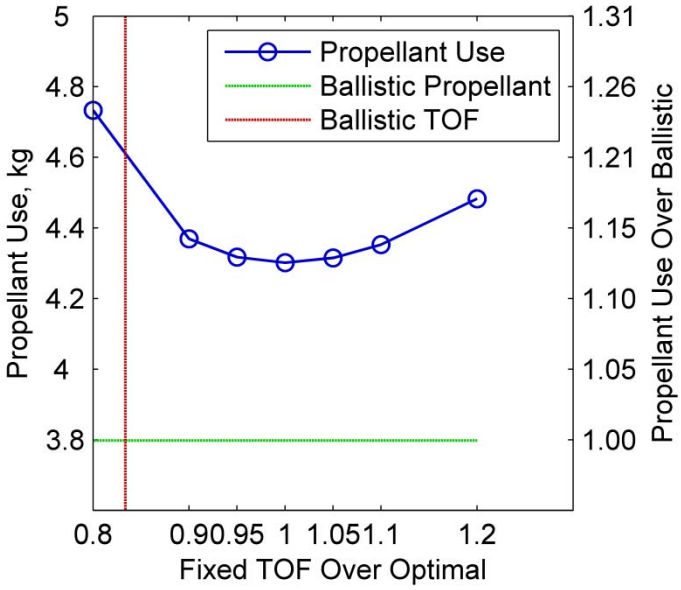

Figure 5.17: Effect of Increasing TOF on Fitness

The ballistic TOF is shown as a red vertical line in Figure 5.17 to show that while it is possible to find finite maneuvers that take less time than the ballistic TOF, there is a large propellant penalty.

5.9 Nominal VTVL Solution

While a spacecraft with a gimbaled main engine such as Craft B in Figure 2.2 may be able to take off and land at an angle, fixed engine spacecraft such as Craft C are limited to vertical takeoff, vertical landing. This is expressed through fixing the thrust angle at the boundaries to be zero. Additionally, a pitch rate constraint of $\dot{\theta}_{max} = 20°/second$ was added to account for real spacecraft having MOIs. Lastly, the acceptable lower bounds on the altitude were shifted to 2 meters for all nodes except at the boundary conditions as discussed in Section 5.6.

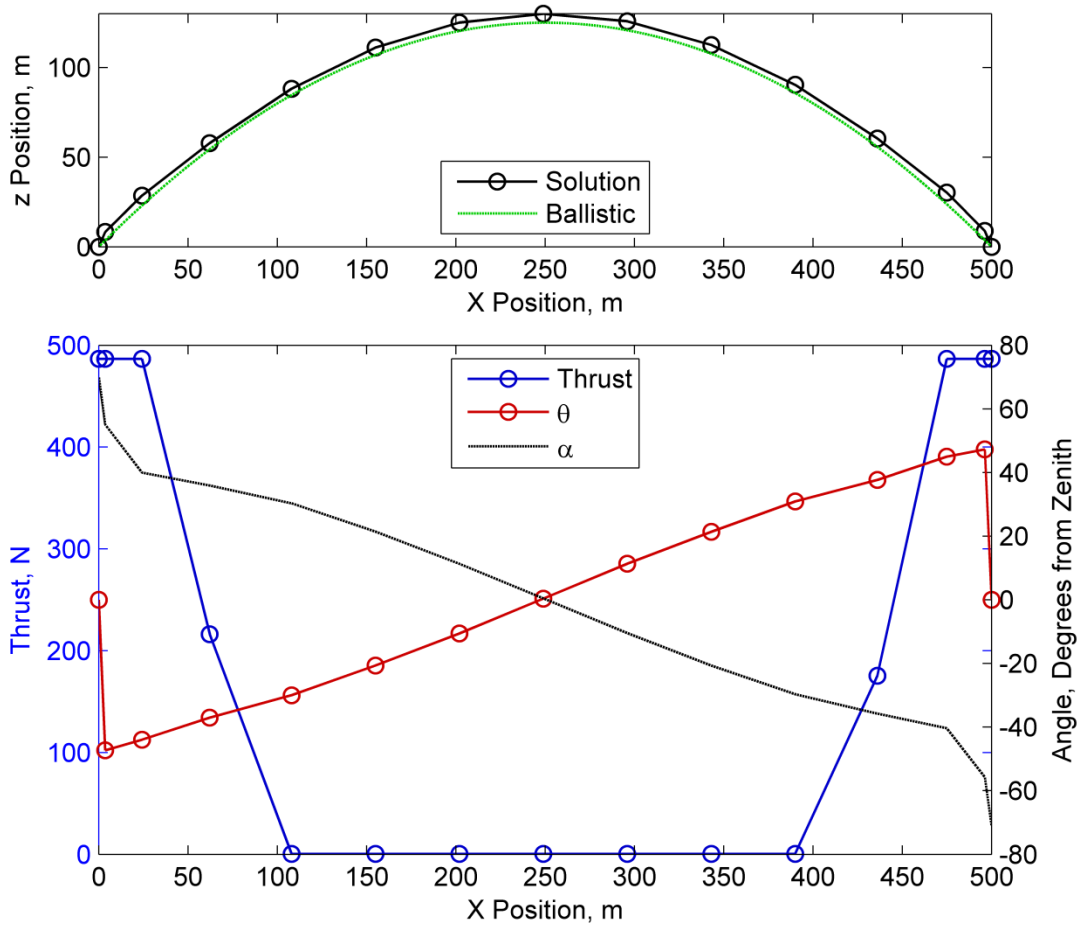

Figure 5.18: Nominal VTVL Trajectory and Control Law

The peak height reached with VTVL in Figure 5.18 is slightly higher than the ballistic trajectory. The control variables are plotted against the X Position, which distorts the first and last nodes.

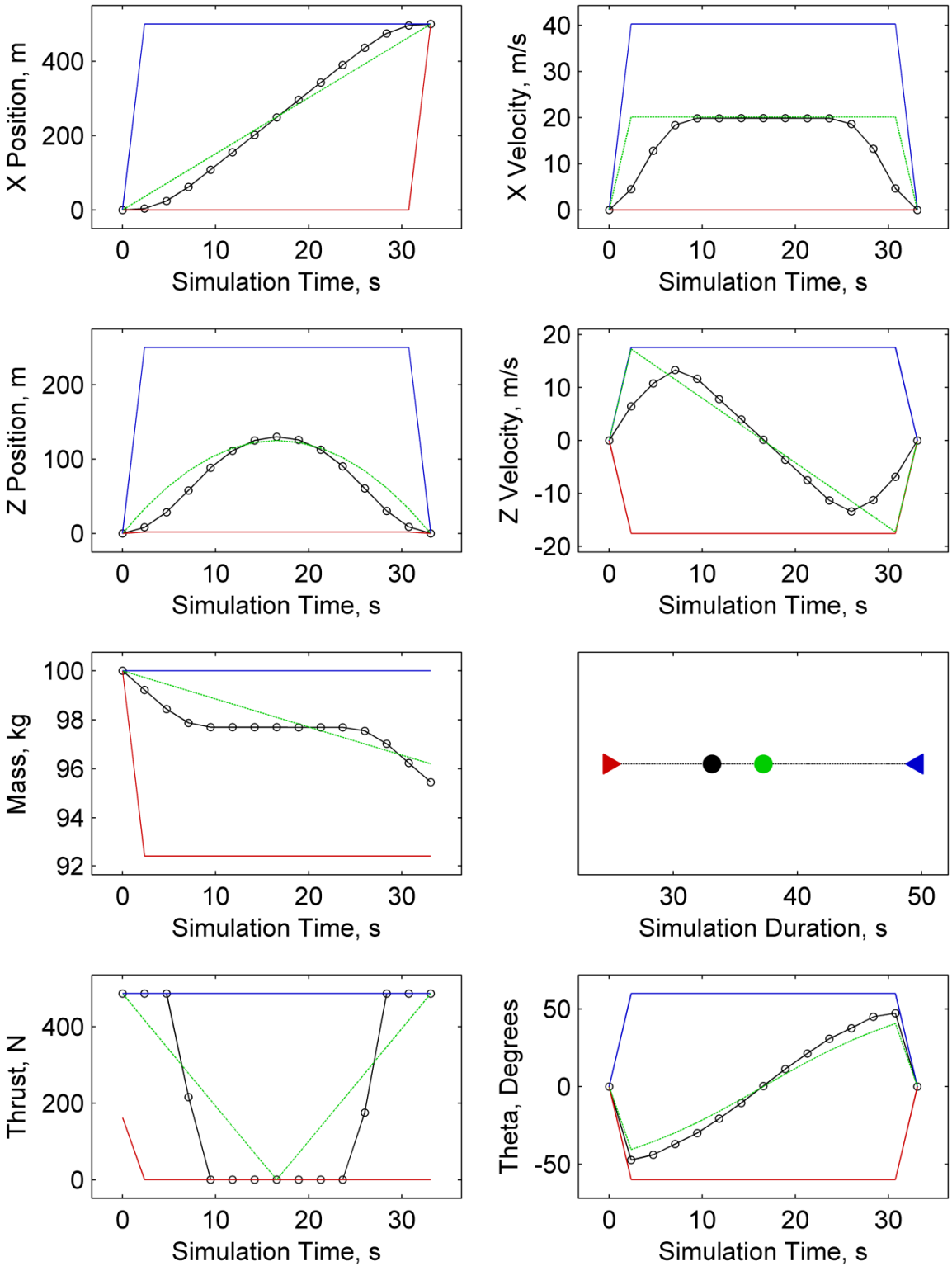

Figure 5.19: VTVL Bounded Search Space with Guess and Final Trajectories

It is useful to compare Figure 5.19 with Figure 5.2. Here, the upper and lower bounds on θ are constrained to enforce the VTVL condition as per Eq. (4.35). The optimal TOF is slightly longer and maximum height slightly higher since the spacecraft cannot start to accelerate along the optimal unrestricted initial flight path as show in Figure 5.10, and instead must vertically liftoff and gradually start turning to accelerate horizontally. The additional gravity losses explain why more propellant is needed for the VTVL trajectory as compared to the unconstrained take off angle solution.

5.10 VTVL Max Thrust Angular Rate Inequality Constraint Enforcement

If only the initial thrust angle is constrained for the VTVL trajectory, the solver will return a solution where θ changes very rapidly. An inequality constraint in the form of Eq. (4.36) where $\dot{\theta}_{max} = \mathbf{20}°/second$ was included as a rough model to account for real spacecraft attitude rate constraints.

This type of constraint is not visualized through simple global upper and lower bounds on the search space. Instead, the control law for θ and its piecewise derivative are shown in Figure 5.20. Note that there are $n - 1$ inequality constraints, one for each trajectory segment τ as shown in Figure 4.1.

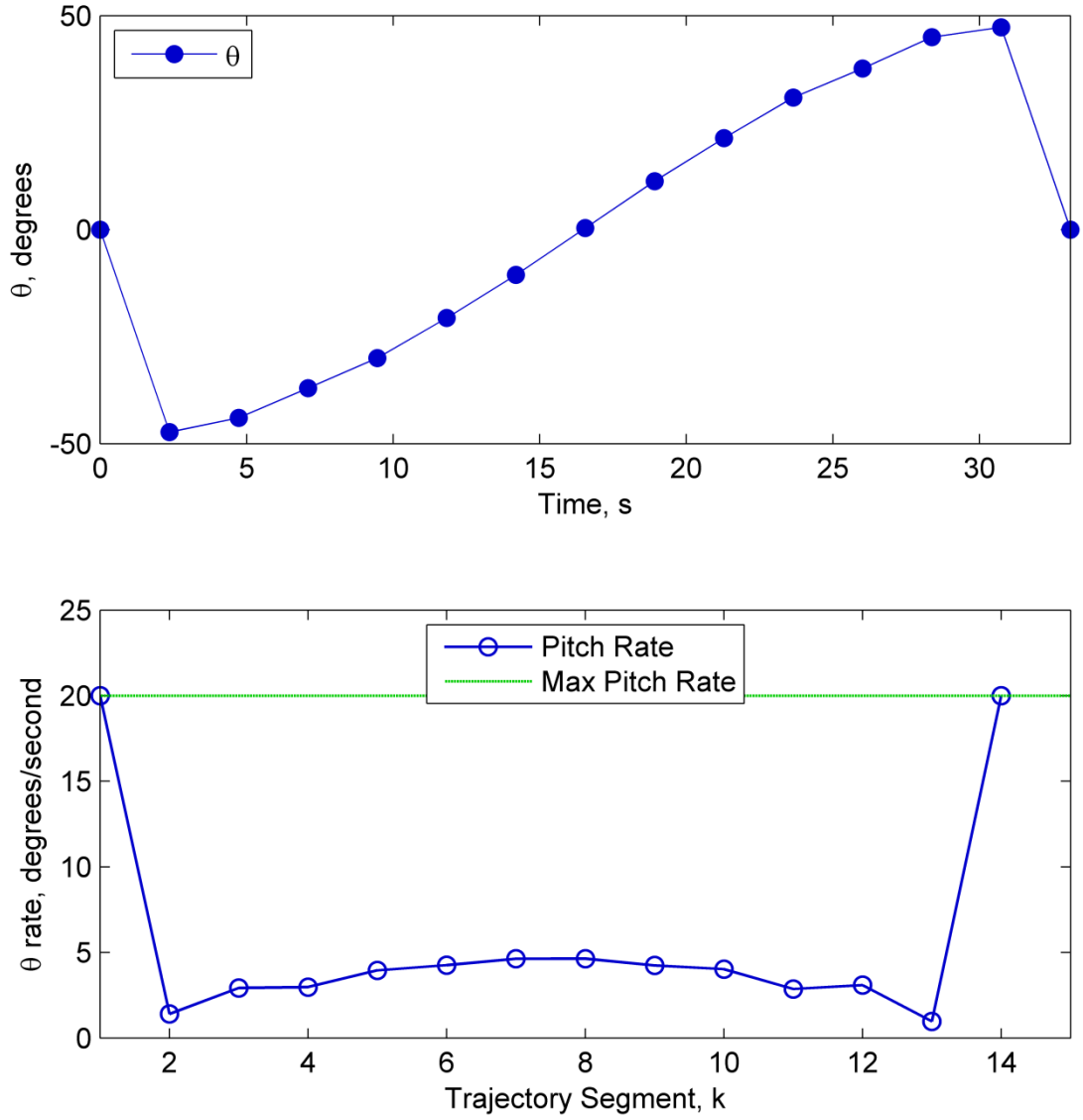

Figure 5.20: VTVL $\dot{\theta}_{max}$ Inequality Constraint Enforcement $\dot{\theta}_{max} = 20°/second$

5.11 VTVL with Ceiling (Top Hat Trajectory)

For a variety of mission profiles or spacecraft limitations, it may be desirable to limit the maximum allowing height. For example, a spacecraft's range altimeter may only be sufficiently accurate within a specific distance. Alternatively, the spacecraft may be recording video or be utilizing scientific instruments while translating.

The parameterized tophat ceiling height was set as simply

$$zPos_{Upper,Tophat} = \sqrt{xPos_{final}} \tag{5.2}$$

As opposed to setting "floors," (minimum altitudes), altitude ceilings can be readily enforced through global simple upper and lower bounds, as increasing the number of discretization nodes does not required a minimum altitude to be reached by the second or penultimate node.

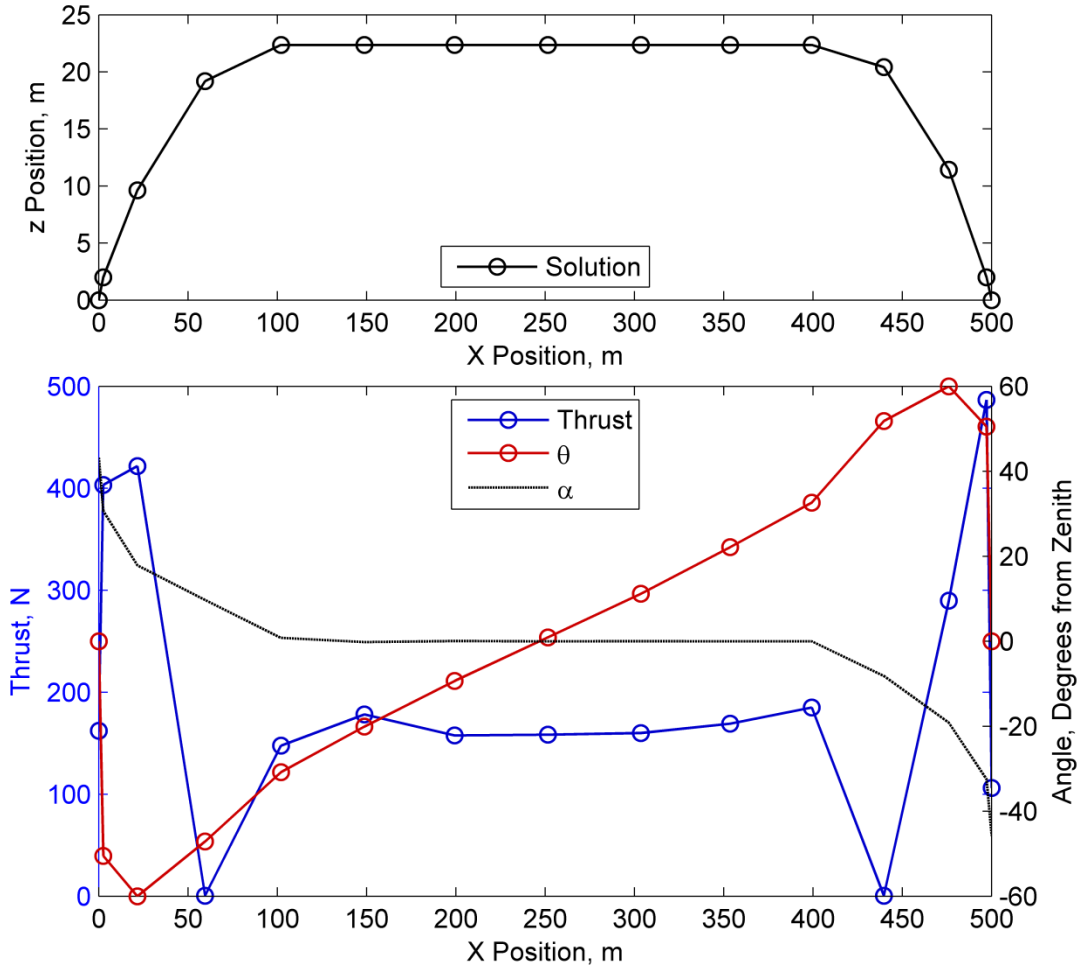

Figure 5.21: Nominal VTVL + Ceiling Profile and Control Law

Looking at the XZ trajectory profile and control law in Figure 5.21, the finite "burn-coast-burn" equivalent no longer holds. The spacecraft cannot coast because of the altitude limits and must use additional thrust to essentially hover at the ceiling altitude.

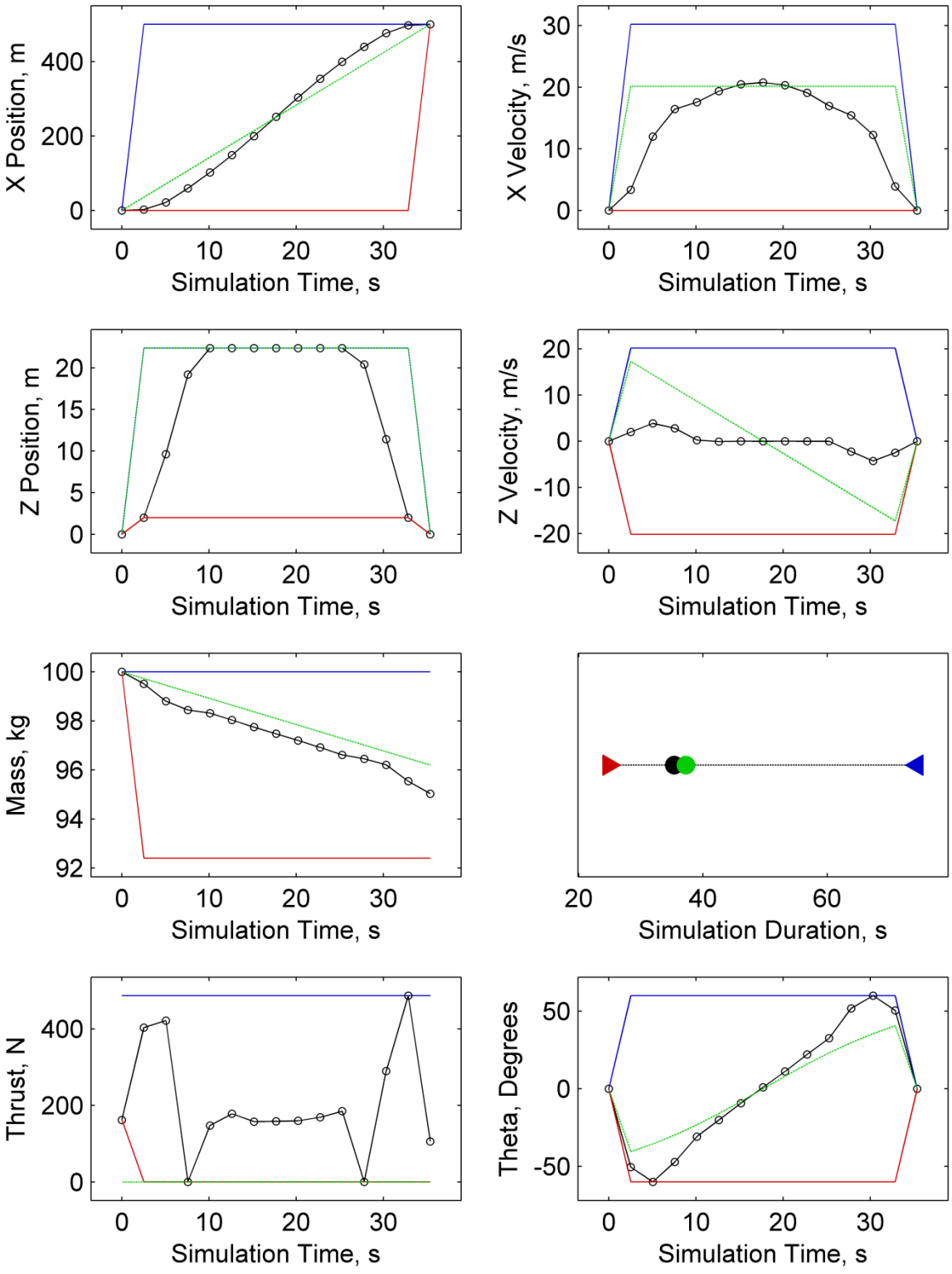

Figure 5.22: Bounds for VTVL + Ceiling (Top Hat)

This is referred to a "Top hat" trajectory because of the shape, and most closely resembles the trajectories flown during the DC-X program[5]. Flying this sort of trajectory might be done because of the relative ease of developing closed-loop flight software for it. However, this ease comes at a cost. Looking through the graphs in Figure 5.22, the propellant mass required and TOF is greater than the unconstrained and simple VTVL trajectories.

5.12 Effect of Changing Specific Impulsive

Using the nominal, VTVL, and top hat parameters, the effect of changing the propellant specific impulse was varied through $[150: 75: 450]$ seconds. All of the data points and best fit curves with $R^2 > 0.99$ are shown in Figure 5.23. Since all of the best fit curves are roughly proportional to the inverse of the specific impulse, once the required propellant mass is known for one potential trajectory in particular, the propellant required with a different propellant can be estimated by a simple ratio:

$$Required\ Propellant\ Mass_{Isp1} * \frac{Isp2}{Isp1} \sim Required\ Propellant\ Mass_{Isp2} \qquad (5.3)$$

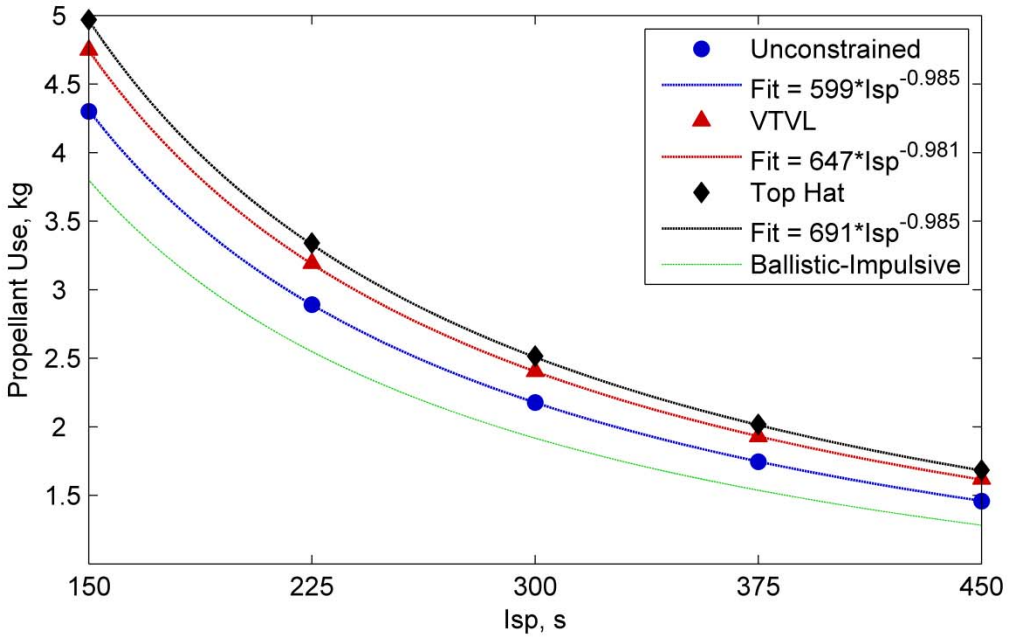

Figure 5.23: Effect of changing Isp on Propellant Use for Unconstrained, VTVL, and Top Hat Trajectories

5.13 Propellant Required for Various Hopping Distances on the Moon and Mars

The required propellant to perform each type of maneuver was determined for a range of hopping distances ranging from 100 meters to 10 km for the Moon and Mars as shown in Figure 5.24 and Figure 5.25. Since the spacecraft original mass was 100 kg, the required propellant use is equivalent to the propellant mass ratio.

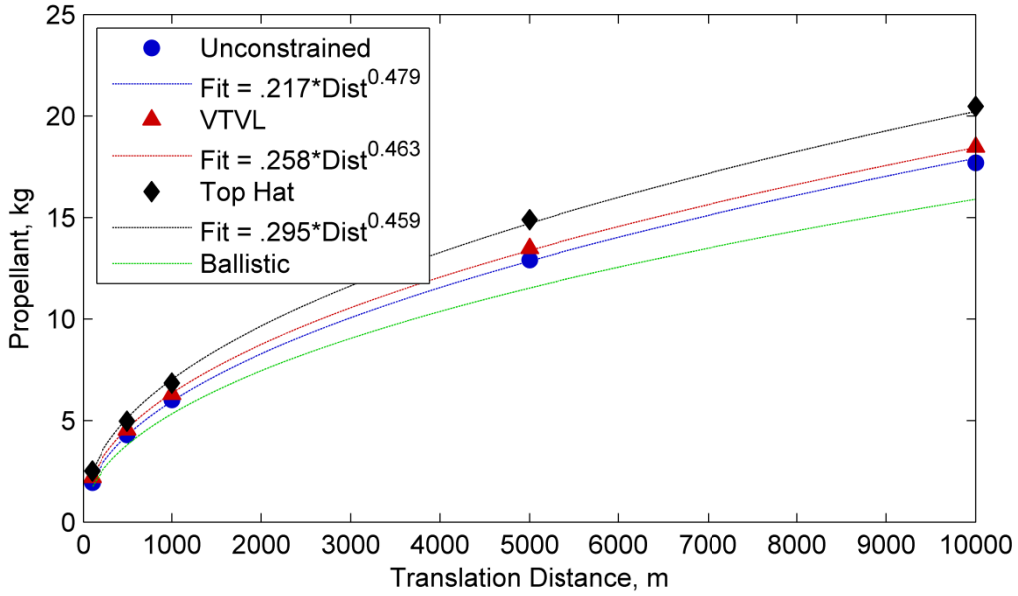

Figure 5.24: Propellant Required for Hops of Various Distances and Trajectory Profiles on the Moon

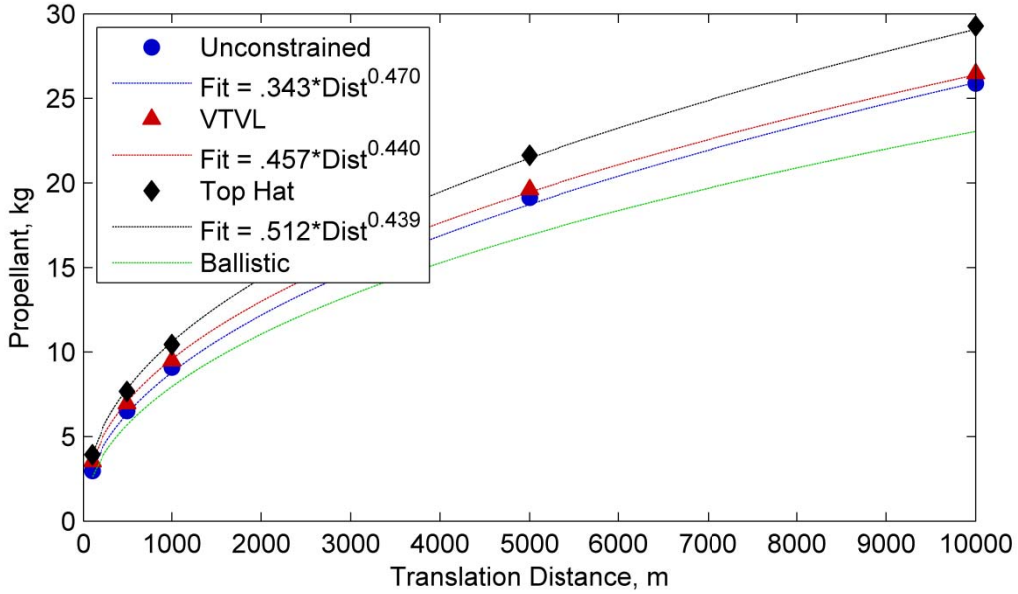

Figure 5.25: Propellant Required for Hops of Various Distances and Trajectory Profiles on Mars

Chapter 6

Conclusions

6.1 Direct Optimization of VTVL Trajectories

The ballistic-impulsive "burn-coast-burn" method found in literature to estimate the propellant required to perform a desired VTVL translation maneuver was found to be inadequate and ultimately too inaccurate to be useful for mission planning. Any real trajectories will necessarily require additional propellant to compensate for gravity losses, and various spacecraft propulsion system architectures place additional constraints on the allowable solution space.

Nonlinear Programming as a direct optimization method is very well suited to handle the problem dynamics and constraints of a VTVL maneuver. Parameterized global upper and lower bounds on all of the state, control, and TOF variables allowed straightforward implementation of initial and final boundary conditions, and were useful in increasing the solver performance by framing a search space without distorting the optimal trajectories found. Equality constraints developed via Direct Collocation were successfully used to enforce feasibility and inequality constraints created to enforce spacecraft potential limitations or additional path constraints.

In conclusion, a robust, direct trajectory optimization method was successfully developed to model VTVL spacecraft dynamics. The required propellant mass to perform a range of possible hopping maneuvers across a variety of mission and spacecraft parameters was explored. The trajectories found were asymmetric in varying degrees due to propellant loss over the course of the flights. The work performed within can provide a higher fidelity model for future mission planners in deciding whether to utilize VTVL as a spacecraft mobility method.

6.2 Recommendations for Future Work

While the VTVL trajectory optimization model developed can be a useful initial tool for space mission planning, further work could extend the robustness and usefulness.

- Using a separate NLPV for each node time value and allowing the node distribution to shift as needed. This should result in the greatest fidelity for a given number of nodes.
- Additional boundary conditions to better model realistic takeoff and landing behaviors.
- Developing additional inequality constraints to handle X position dependent path constraints instead of relying on simple bounds to enforce minimum altitudes.
- Updating the coordinate system and state equations with a two-body gravity model and/or extending into the third dimension.
- Including additional state variables to model blow-down versus pressure regulated systems such as propellant tank pressures. This could enable modeling the effective real-time specific impulse.
- Including attitude control systems and modeling a dynamic Moment of Inertia as the propellant is expended from the craft instead of making a point-mass approximation.
- Explore translation maneuvers where the secondary landing is significantly lower or higher in altitude than the takeoff trajectory. This could potentially be used to make quick hops into or out of craters such as Shackleton near the lunar South Pole.
- Include the ability to model gaps in the throttle ranges of the propulsion system. This could be done as readily as overriding the commanded thrust value to zero in the state equations if the value is within the non-throttleable range.
- The state equations could be modified so that the spacecraft cannot move until the T/W ratio is greater than unity to account for the spacecraft weight on the surface.

Appendix A

Notes on Direct Collocation

For clarity for any future readers trying to implement Direct Collocation:

- Direct Collocation enables the creation of the equality constraint shown in Eq. (4.23), but creating the cubic polynomials and solving for their coefficients is not needed to implement the equality constraint in code. This is why they are referred to as fictitious functions in Section 4.8.
- In Eq.(4.14) the coefficients for only a single polynomial function are shown corresponding to a single state variable function. In reality, separate polynomial functions could be created for each state variable function along each trajectory segment τ_k, for a total of $[5 \times (n-1)]$ cubic polynomials and $[5 \times (n-1) \times 4]$ coefficents. This is not required for the reason listed above.
- When deriving Eq. (4.21) by substituting the relations found in (4.19) into (4.20), the Δt_k term comes from the implicit $\Delta\tau$ in the cubic polynomial's derivative, i.e. $\dot{\boldsymbol{\sigma}}(\tau) = \frac{d\boldsymbol{\sigma}(\tau)}{d\tau}$
- The original method pioneered by Hargraves and Paris[20] did not use Simpson-Hermite integration but rather constructed a *defect equality constraint* by comparing the value of Eq. (4.22) to the derivative of Eq. (4.14) evaluated at $\tau = 0.5$ (the midpoint).
- Direct Collocation is distinct from Nonlinear Programming. Other methods exist for developing feasibility equality constraints.

Appendix B

Time as a Fixed or Free Variable, and Node Distribution

Temporal discretization of an OCP presents an interesting challenge. If the simulation duration is fixed for a specific maneuver, the spacing between any two nodes does not have to be constant Δt, i.e. the time vector $\mathbf{t}$ in Eq. (4.1) does not have to be linearly distributed. Increased performance may potentially be gained through discretization methods that shift the relative density of nodes to locations of the trajectory where the greatest rate of change in state and control variables is found. Shifting the node spacing usually requires some prior knowledge of the likely regions where increasing the node density would be most beneficial, and conversely, where decreasing the density would matter least.

A potential alternative is the method of Chebyshev-Gauss-Lobatto (CGL) which distributes node points along the interval $\varphi \in [-1{:}\,1]$ according to the formula

$$\varphi_k = -\cos\left(\frac{\pi * (\mathbf{k}-1)}{n-1}\right) \; for \; k = \; [1{:}\,n] \tag{B.1}$$

where $\mathbf{k}$ is a vector of the node indices and n is the number of nodes as per

$$\mathbf{k} = 1,2,\ldots,n \tag{B.2}$$

The actual node times are mapped between the initial and final time according to

$$t_k = \left(\frac{1}{2}\right) * \left[\varphi_k * \left(t_{final} - t_{\text{intial}}\right) + \left(t_{\text{initial}} + t_{final}\right)\right] for \; k = \; [1{:}\,n] \tag{B.3}$$

The CGL method has a greater density of points near the beginning and end of the time interval, at the expense of a decreased number of points towards the middle. Figure B.1 shows an example CGL distribution of 41 nodes from the time span [0:40] against a linear distribution. Note that only the time span of [0:20] is shown since both the linear and CGL distribution are symmetric.

In this example, in the range of (0:3] there are only three nodes in the linear distribution, but seven in the CGL. Conversely in the range of [17:20) there is one node in the CGL distribution, but three in the linear distribution. If the state variables rapidly change in the beginning and end of the flight, CGL may give better performance for a reduced number of nodes.

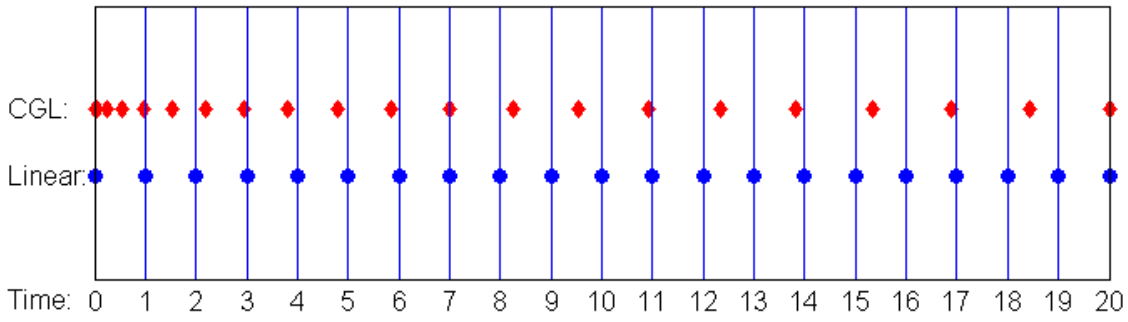

Figure B.1: CGL versus Linear Node Distribution for Half of Time Array

Alternatively, even for a fixed TOF the node times do not need to be fixed. For example, for an initial linear distribution of 11 nodes from $[0: 10]$ seconds, there would initially be a node at each second, i.e. $t_1 = 0\ s, t_2 = 1\ s, \cdots t_{11} = 10\ s$. While the initial and final node times would need to be fixed if the TOF was fixed, nodes $2: 10$ could be allowed to shift to automatically find the optimal distribution of nodes. If implemented successfully, this could significantly increase algorithm performance in terms of a reduced number of function evaluations (NFEs).

References

1 Way, D. W., Powell, R. W., Chen, A., Steltzner, A. D., Martin, A. M. S., Burkhart, P. D., and Mendeck, G. F., "Mars Science Laboratory: Entry , Descent , and Landing System Performance," *IEEE Aerospace Conference*, Big Sky, MT: 2006, p. No. 1467.

2 Yoshimitsu, T., Kubota, T., Adachi, T., and Kuroda, Y., "Advanced robotic system of hopping rovers for small solar system bodies," *International Symposium on Artificial Intelligence, Robotics and Automation in Space*, 2012, pp. 3–7.

3 Thurman, S. W., "Surveyor Spacecraft Automatic Landing System," *27th Annual Advances in the Astronautical Sciences Guidance and Control Conference*, Breckenridge, Colorado: 2004, p. 10.

4 Ball, A. J., Garry, J. R. C., Lorenz, R. D., and Kerzhanovich, V. V., *Planetary Landers and Entry Probes*, New York: Cambridge University Press, 2007.

5 "The Delta Clipper Experimental: Flight Testing Archive" Available: http://www.hq.nasa.gov/office/pao/History/x-33/dc-xa.htm Accessed: 2014 July 14.

6 Ibrahim, A., "Lunar Lion Receives new H202 Engines," *GLXP* Available: http://www.googlelunarxprize.org/teams/penn-state-lunar-lion-team/blog/lunar-lion-receives-new-h202-engines Accessed: 2014 July 14.

7 Mohon, L., "Overview: Robotic Lander" Available: http://www.nasa.gov/mission_pages/lunarquest/robotic/#.U5FTJ_mwKbM Accessed: 2014 June 6.

8 "About Morpheus" Available: http://morpheuslander.jsc.nasa.gov/about/ Accessed: 2014 June 6.

9 Xie, F., Yang, Y., Chang, S., and Wang, Y., "Improving strategies on PSO for suborbit launch vehicle trajectory optimization," *The Fourth International Workshop on Advanced Computational Intelligence*, Oct. 2011, pp. 113–119.

10 "About Blue Origin" Available: http://www.blueorigin.com/about 2014 Accessed: 2014 July 13.

11 Clark, S., "Grasshopper flight captured in breathtaking video," *Spaceflightnow.com* Available: http://www.spaceflightnow.com/news/n1310/14grasshopper/#.U8L-7PmzG-Y Accessed: 2014 July 13.

12 Clark, S., "SpaceX achieves controlled landing of Falcon 9 first stage," *Spaceflightnow.com* Available: http://spaceflightnow.com/falcon9/009/140419reusability/#.U8L_jfmzG-Y Accessed: 2014 July 13.

13 Meditch, J. S., "On the Problem of Optimal Thrust Programming For a Lunar Soft Landing," *IEEE Transactions on Automatic Control*, 1964, pp. 477–484.

14 Cherne, J. M., *Mechanical Design of the Lunar Module Descent Engine*, Redondo Beach, CA: .

15 Newton, I., *Mathematical Principles of Natural Philosophy*, London, UK: 1686.

16 Bryson, A. E. J., and Ho, Y.-C., *Applied Optimal Control*, Taylor & Francis, 1975.

17 Betts, J. T., "Survey of Numerical Methods for Trajectory Optimization," *Journal of Guidance, Control, and Dynamics*, vol. 21, Mar. 1998, pp. 193–207.

18 Eagle, D., "A MATLAB Script that Demonstrates Aerospace Trajectory Optimization Using Direct Transcription and Collocation" Available: http://www.mathworks.com/matlabcentral/fileexchange/38936-aerospace-trajectory-optimization-using-direct-transcription-and-collocation Accessed: 2013 July 17.

19 Dickmanns, E. D., and Well, K. H., "Approximate Solution of Optimal Control Problems Using Third-Order Hermite Polynomial Functions," *Proceedings of the IFIP Technical Conference*, London, UK: Springer-Verlag, 1974, pp. 158–166.

20 Hargraves, C. R., and Paris, S. W., "Direct Trajectory Optimization Using Nonlinear Programming and Collocation," *Journal of Guidance, Control, and Dynamics*, vol. 10, 1987, pp. 338–342.

21 Enright, P. J., and Conway, B. A., "Discrete Approximations to Optimal Trajectories Using Direct Transcription and Nonlinear Programming," *Journal of Guidance, Control, and Dynamics*, vol. 15, 1992.

22 Geiger, B., Horn, J., DeLullo, A., Niessner, A., and Long, L., "Optimal Path Planning of UAVs Using Direct Collocation with Nonlinear Programming," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Reston, Virginia: American Institute of Aeronautics and Astronautics, 2006, p. 6199.

23 Geiger, B., "Unmanned Aerial Vehicle Trajectory Planning with Direct Methods," Ph.D. Thesis, The Department of Aerospace Engineering, The Pennsylvania State University, 2009.