The Report Committee for Andrew Walter Berning Jr.
Certifies that this is the approved version of the following report:

Verification of Successive Convexification Algorithm

APPROVED BY
SUPERVISING COMMITTEE:

Supervisor:

Maruthi R. Akella

Behçet Açıkmeşe

Verification of Successive Convexification Algorithm

by

Andrew Walter Berning Jr., B.S.

Report

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Engineering

The University of Texas at Austin

May 2016

To Mom, Dad, and Matthew

Acknowledgements

My sincerest thanks to…

… Dr. Açıkmeşe, for persistently challenging me to learn and grow academically.

… Dr. Akella, for graciously agreeing to serve on my report committee.

… Geoff, for the invariably valuable research, academic, and professional guidance.

… Miki, for being a wonderful friend and mentor.

… My mother and father, for their ceaseless love and support.

Abstract

Verification of Successive Convexification Algorithm

Andrew Walter Berning Jr., M.S.E.

The University of Texas at Austin, 2016

Supervisor: Maruthi R. Akella

In this report, I describe a technique which allows a non-convex optimal control problem to be expressed and solved in a convex manner. I then verify the resulting solution to ensure its physical feasibility and its optimality. The original, non-convex problem is the fuel-optimal powered landing problem with aerodynamic drag. The non-convexities present in this problem include mass depletion dynamics, aerodynamic drag, and free final time. Through the use of lossless convexification and successive convexification, this problem can be formulated as a series of iteratively solved convex problems that requires only a guess of a final time of flight. The solution's physical feasibility is verified through a nonlinear simulation built in Simulink, while its optimality is verified through the general nonlinear optimal control software GPOPS-II.

Table of Contents

List of Tables

List of Figures

1 Introduction

1.1 Motivation

The appeal of a reusable launch vehicle is obvious to anyone who has ever watched a rocket launch. The vast majority of structural mass—and thus cost—of the vehicle plunges back to earth after expending all of its fuel, to smash unceremoniously into the ground or ocean, ending its usefulness. The next logical step towards the goal of affordable, reliable space access involves a launch vehicle that is reusable and requires little in the way of refurbishment between missions.

The closest historical example to a reusable launch vehicle is the Space Shuttle. This design relied on the orbiting vehicle to re-enter and land horizontally, while the solid rocket boosters (SRBs) parachuted into the ocean to be recovered, refurbished, and reused. The large external fuel tank was jettisoned, broke up upon re-entry, and was not reusable. The disadvantages of this horizontal landing strategy include: the SRB descent utilized a heavy and complex system of chutes; the SRB recovery necessitated the use of specialized sea vehicles; the SRB refurbishment process was very time and resource intensive; and the external fuel tank was completely disposable.

An alternative solution that has seen a recent surge in interest and popularity is the vertical takeoff vertical landing (VTVL) solution [1-3]. A VTVL mission involves a staged launch vehicle operating traditionally up until the main engine cutoff and first stage separation. At this point, the first stage of an expendable launch system would follow a ballistic path and crash once it reached the earth's surface. A VTVL first stage follows a ballistic trajectory immediately following separation, but then the main engines re-ignite and the vehicle maneuvers itself to land softly at a predetermined landing site.

VTVL alleviates most of the concerns of the horizontal landing solution, but comes with a set of challenges of its own. The mass, propulsion, aerodynamic, and structural challenges are not trivial, but this report deals exclusively with the guidance challenges. Specifically, creating a guidance algorithm that can dictate the trajectory of the launch vehicle from the time when the engine is reignited to when the vehicle lands. There are two main guidance-related challenges that the algorithm proposed in this report seeks to address.

First is the importance of temporal proximity. The vehicle begins implementing the computed trajectory immediately after the engine is relit. Because the computation is not instantaneous, the initial conditions of the trajectory are estimated based on the vehicle's state when the computation is initialized. The potential error between this estimated state and the actual state at the time of engine relight increases with the amount of time required for the trajectory's computation. Rapid computation is desirable, then, to minimize this state error and maximize the optimality of the resultant, actual trajectory.

Second is the importance of the vehicle's divert capabilities. Space launch missions are extremely mass sensitive, and the VTVL involves reserving a fraction of the total fuel to power the vehicle during its final, landing maneuver. It is desirable, then, that this trajectory creation algorithm is posed as an optimization problem with final mass having a significant role in the cost function. Other alternatives, such as pre-computed lookup tables, will necessarily be sub-optimal and require the allocation of a greater fuel mass for this missions segment, detracting from the overall mission goals.

1.2 Project Objectives

The objective of this project was to create an algorithm that would compute an optimal, physically feasible trajectory in a predictable, reliable, and computationally swift manner.

The algorithm described in this report solves these previously-stated problems through the use of lossless [4-11] and successive [12, 13] convexification, allowing us to pose the problem in a convex manner. The final result is an algorithm that can handle non-convexities such as mass depletion dynamics, free final time, and aerodynamic drag, and can be solved using Interior Point Method solvers [16-18] with a computational speed suitable for onboard applications [19-20].

Once the algorithm had been developed, it underwent a verification process. This process involved comparing our algorithm's solutions to the solutions of a nonlinear optimizer, as well as the propagated trajectory of nonlinear simulation. In this way, we verified that our algorithm was returning an optimal and physically feasible trajectory.

1.3 Report Outline

The rest of this report is organized as follows: Section 2 outlines the formal problem formulation and the necessary convex optimization theory that was used to solve it. Section 3 briefly covers two numerical solutions to this problem. Section 4 details the nonlinear simulation that was created to verify the algorithm. Section 5 describes how our algorithm's solution compared to the solution from a nonlinear optimizer. Lastly, Section 6 draws some final conclusions.

2 Background

This section details the problem formulation and convex optimization theory necessary to construct the trajectory optimization algorithm [23].

2.1 Problem Formulation

As mentioned in the previous section, one of the goals of this project was to create an algorithm that would produce an optimal and physically feasible trajectory. In order to meet the "computationally swift" part of the goal, however, certain simplifying assumptions had to be made. This section details the problem formulation, including these assumptions, and section 4 attempts to quantify the impact of the assumptions on the resultant trajectories.

The vehicle is modelled as a point mass subject to 3DOF translational dynamics:

$$\dot{\boldsymbol{r}}(t) = \boldsymbol{v}(t)$$

$$\dot{\boldsymbol{v}}(t) = \boldsymbol{a}(t)$$

$$\boldsymbol{a}(t) = \frac{1}{m(t)}[\boldsymbol{T}(t) + \boldsymbol{D}(t)] + \boldsymbol{g}$$

Where $\boldsymbol{r}, \boldsymbol{v}$, and $\boldsymbol{a}$ are position, velocity, and acceleration, respectively. Gravity, $\boldsymbol{g}$, is assumed constant. The drag model used is for that of a sphere, where the force $\boldsymbol{D}$ acts in a direction opposite to $\boldsymbol{v}$ according to the expression:

$$\boldsymbol{D}(t) = -\frac{1}{2}\rho S_D C_D ||\boldsymbol{v}(t)||\boldsymbol{v}(t)$$

Where the coefficient of drag, C_D, drag reference area, S_D, and air pressure, ρ, are assumed to be constant. The mass depletion dynamics are assumed to vary linearly with thrust magnitude, with a correction for back-pressure losses:

$$\dot{m}(t) = -\frac{1}{I_{sp} g_0}\left|\left|\boldsymbol{T}(t)\right|\right| - \frac{P_{amb} A_{nozzle}}{I_{sp} g_0}$$

Where A_{nozzle} is the area of the rocket nozzle at the exit and $I_{sp}g_0$ is the specific impulse of the rocket motor multiplied by standard gravity. P_{amb} is the ambient air pressure, and is also assumed to be constant.

The primary term in the objective function is minimizing fuel usage, but there are also a few auxiliary terms added as a result of the successive convexification and will be explained in section 2.3.

Prescribed boundary conditions include initial and final position, initial and final velocity, and initial mass. The initial and final thrust magnitudes are left unconstrained, but they must be pointing along a prescribed unit vector. State constraints include a lower bound on mass to ensure that the trajectory does not utilize more fuel than is available. There is also a minimum and maximum thrust magnitude, a maximum thrust tilt angle, and a maximum time rate of change of thrust magnitude. These constraints will be stated more specifically in section 2.4.

Finally, in order to make the trajectory suitable for computation, it must be discretized into N segments of equal time length. Here we follow the multiple impulse transcription model, wherein each segment consists of a ballistic trajectory with an impulsive change in velocity at each time node. For the rest of this report, we will use the subscript k to refer to individual nodes.

In this formulation, there are three distinct sources of non-convexity: the minimum thrust constraint, the acceleration dynamics, and the free final time. The first is addressed in section 2.2 while the latter two are addressed in section 2.3.

2.2 Lossless Convexification

The control constraints on the minimum thrust magnitude and maximum tilt angle are non-convex constraints. In the case of the latter, it is only convex in the case where $\theta_{max} > 90°$. We address this by restricting $\theta_{max} \leq 90°$. The former is addressed by introducing a new solution variable Γ [4-11] as follows:

$$\left|\left|\boldsymbol{T}[k]\right|\right| \leq \Gamma[k]$$

$$0 \leq T_{min} \leq \Gamma[k] \leq T_{max}$$

$$\boldsymbol{\Gamma}[k]\cos(\theta_{max}) \leq \hat{\boldsymbol{e}}_u^T \boldsymbol{T}[k]$$

In can be shown for the continuous time problem that these substitutions elicit the same solution as the original problem. It has not been proven, but has been shown through numerical simulations [23] that this relaxation remains lossless for the discrete time problem.

2.3 Successive Convexification

This section details the solution strategy used to account for the non-convexities not addressed in section 2.2. The approach consists of linearizing about the trajectory from the previous iteration, and then solving the newly-convex problem. The first iteration is initialized by making some additional assumptions about the nature of the trajectory. This linearization and iteration strategy necessitates a few algorithm additions that will be discussed in this section. Namely, quadratic trust regions to ensure the trajectory doesn't change too much between iterations, and a relaxation to prevent the linearizations from producing infeasible problems.

2.3.1 Linearizations

This linearization procedure is only used for SC iterations where $i \geq 1$ ($i = 0$ is the first iteration) and is used to address the following non-linearities: the Δt term in the discretized dynamics equations, the expression for atmospheric drag, and the mass term in the expression for acceleration.

2.3.2 Trust Regions

These linearizations bring with them two major concerns, the first of which is the problem becoming unbounded or leaving the regime in which the linearization is valid. We address this by adding a weighting in the cost function that de-incentivizes the trajectory changing significantly between iterations. In an attempt to reduce the number of solution variables, we only applied this method to Δt and $\boldsymbol{T}$. The former restricts the change on the total time-of-flight does, while the latter restricts the change on the vehicle acceleration term, which subsequently trickles down into the vehicle velocity and position terms.

The specific implementation is as follows:

$$(x_i[k] - x_{i-1}[k])^T [\![(x]\!]_i[k] - x_{i-1}[k]) \leq \eta_x, \qquad i \geq 1$$

Where x is the solution variable being considered. η_x is then added to the objective function. This trust region strategy only applied to iterations $i \geq 1$.

2.3.3 Relaxation

The second problem that arises from the previously mentioned linearizations is artificial infeasibility. It is possible for no solution to exist to the linearized problem, even if the original, nonlinear problem is feasible. This can cause the algorithm to fail at an early SC iteration, before the linearizations have a chance to converge to the true solution. An example scenario might include a linearization about a Δt that is very small, creating an obviously infeasible problem.

To avoid this potential scenario, we introduce a relaxation, a_R in the acceleration dynamics term that allows the solver to utilize a very large acceleration in any direction in order to ensure that the boundary conditions of the problem can be met. Use of this artificial acceleration is then weighted heavily in the objective function. Thus, the solution algorithm utilizes this acceleration in early SC iterations where it is necessary to ensure feasibility, but usage quickly approaches zero as the SC algorithm converges. The exact implementation is as follows:

$$\| a_R[k] \| \leq \kappa_{a,R}[k]$$

The κ variables are added to the objective function in much the same way as the η variables in section 2.3.2.

2.4 Problem Summary

This section summarizes everything that has been covered in section 2 and formalizes it into the problems 1 and 2. Problem 1 covers the SOCP problem for the first SC iteration, while problem 2 is the formulation used for the remaining SC iterations. For SC iterations $i > 0$, the solution is obtained by linearizing about trajectory $i - 1$.

Problems 1 and 2 utilize the following notational conventions:

$$k_f \triangleq N - 1$$

$$t_f = k_f \Delta t$$

$$k \in [0, k_f]$$

And, for any variable x:

$$\Psi_x[k] \triangleq \left[\mathrm{x}^{\mathrm{T}}[k] \quad x^T[k+1]\right]^T, \quad k \in [0, k_f]$$

$$\delta x_i[k] \triangleq x_i[k] - x_{i-1}[k] \quad i > 0$$

Where the subscript i refers to the SC iteration.

Also, please note that the dynamics equations are discretized under the assumption that acceleration varies linearly from its value at node k to its value at node $k+1$. This is done for the purpose of improved physical accuracy when compared to other strategies, such as zero-order-hold.

Finally, we must account for the simplified problem structure that initializes the first SC iteration. Specifically, we must make guesses about the time of flight, the speed profile, and the mass profile. Thus, for the first SC iteration, we define a constant $\Delta\tau = \frac{t_{f,s}}{k_f}$ where $t_{f,s}$ is a guess of the final flight time and is an input to the algorithm. The speed profile, s, is linearly interpolated between initial and final speeds. The mass profile, μ, is linearly interpolated between m_0 and m_{dry}.

Problem 4

Objective Function:

minimize $\left(-w_{m,f} m[k_f] + w_{\kappa,\boldsymbol{a},R} \parallel \kappa_{\boldsymbol{a},R} \parallel\right)$ over $\boldsymbol{T}, \Gamma$

subject to:

Boundary Conditions:

$m[0] = m_0, \boldsymbol{r}[0] = \boldsymbol{r}_0, \boldsymbol{v}[0] = \boldsymbol{v}_0, \boldsymbol{T}[0] = \Gamma[0]\hat{\boldsymbol{n}}_0, \Gamma[0] = \Gamma_0$

$\boldsymbol{r}[k_f] = 0, \boldsymbol{v}[k_f] = 0, \boldsymbol{T}[k_f] = \Gamma[k_f]\hat{\boldsymbol{n}}_f$

Dynamics:

$$m[k+1] = m[k] - \left[\frac{\alpha}{2}(\Gamma[k] + \Gamma[k+1]) + \dot{m}_{bp}\right]\Delta\tau \qquad \kappa \in [0, k_f)$$

$$\boldsymbol{r}[k+1] = \boldsymbol{r}[k] + \boldsymbol{v}[k]\Delta\tau + \frac{1}{3}\left(\boldsymbol{a}[k] + \frac{1}{2}\boldsymbol{a}[k+1]\right)\Delta\tau^2 \qquad \kappa \in [0, k_f)$$

$$\boldsymbol{v}[k+1] = \boldsymbol{v}[k] + \tfrac{1}{2}(\boldsymbol{a}[k] + \boldsymbol{a}[k+1])\Delta\tau \qquad \kappa \in [0, k_f)$$

$$\boldsymbol{a}[k] = \frac{1}{\mu[k]}\left(\boldsymbol{T}[k] - \frac{1}{2}\rho S_D C_D s[k]\boldsymbol{v}[k]\right) + \boldsymbol{a}_R[k] + \boldsymbol{g}$$

State Constraints:

$$m_{dry} \leq m[k]$$

Control Constraints:

$$\| \boldsymbol{T}[k] \| \leq \Gamma[k]$$

$$0 \leq T_{min} \leq \Gamma[k] \leq T_{max}$$

$$\Gamma[k]\cos(\theta_{max}) \leq \hat{\boldsymbol{e}}_u^T \boldsymbol{T}[k]$$

$$\dot{T}_{min}\Delta\tau \leq \Gamma[k+1] - \Gamma[k] \leq \dot{T}_{max}\Delta\tau \qquad \kappa \in [0, k_f)$$

SC Modifications:

$$\| \boldsymbol{a}_R[k] \| \leq \kappa_{\boldsymbol{a},R}[k]$$

Problem 5

Objective Function:

minimize $\left(-w_{m,f} m[k_f] + w_{\eta,\Delta t}\eta_{\Delta t} + w_{\eta,\boldsymbol{T}} \| \eta_{\boldsymbol{T}} \| + w_{\kappa,\boldsymbol{a},R} \| \kappa_{\boldsymbol{a},R} \|\right)$ over $\boldsymbol{T}, \Gamma, \Delta t$

subject to:

Boundary Conditions:

$$m[0] = m_0, \boldsymbol{r}[0] = \boldsymbol{r}_0, \boldsymbol{v}[0] = \boldsymbol{v}_0, \boldsymbol{T}[0] = \Gamma[0]\hat{\boldsymbol{n}}_0, \Gamma[0] = \Gamma_0$$

$$\boldsymbol{r}[k_f] = 0, \boldsymbol{v}[k_f] = 0, \boldsymbol{T}[k_f] = \Gamma[k_f]\hat{\boldsymbol{n}}_f$$

Dynamics:

$$\boldsymbol{\Psi}[k] \triangleq \left[\Delta t \quad \boldsymbol{\Psi}_{\boldsymbol{m}}^T[k] \quad \boldsymbol{\Psi}_{\Gamma}^T[k] \quad \boldsymbol{\Psi}_{\boldsymbol{v}}^T[k] \quad \boldsymbol{\Psi}_{\boldsymbol{T}}^T[k] \quad \boldsymbol{\Psi}_{\boldsymbol{a},R}^T[k]\right]^T \qquad \kappa \in [0, k_f)$$

$$\boldsymbol{f}_{\boldsymbol{m}}(\boldsymbol{\Psi}[k]) \triangleq -\left[\frac{\alpha}{2}(\Gamma[k] + \Gamma[k+1]) + \dot{m}_{bp}\right]\Delta t \qquad \kappa \in [0, k_f)$$

$$\boldsymbol{f}_{\boldsymbol{r}}(\boldsymbol{\Psi}[k]) \triangleq \boldsymbol{v}[k]\Delta t + \frac{1}{3}\left(\boldsymbol{a}[k] + \frac{1}{2}\boldsymbol{a}[k+1]\right)\Delta t^2 \qquad \kappa \in [0, k_f)$$

$$\boldsymbol{f}_{\boldsymbol{v}}(\boldsymbol{\Psi}[k]) \triangleq \frac{1}{2}(a[k] + a[k+1])\Delta t \qquad \kappa \in [0, k_f)$$

$$m[k+1] = m[k] + f_m(\boldsymbol{\Psi}_{i-1}[k]) + \frac{\partial f_m}{\partial \boldsymbol{\Psi}}_{\Psi_{i-1}[k]} \delta\Psi_i[k] \qquad \kappa \in [0, k_f)$$

$$\boldsymbol{r}[k+1] = \boldsymbol{r}[k] + f_{\boldsymbol{r}}(\boldsymbol{\Psi}_{i-1}[k]) + \frac{\partial f_{\boldsymbol{r}}}{\partial \boldsymbol{\Psi}}_{\Psi_{i-1}[k]} \delta\Psi_i[k] \qquad \kappa \in [0, k_f)$$

$$\boldsymbol{v}[k+1] = \boldsymbol{v}[k] + f_{\boldsymbol{v}}(\boldsymbol{\Psi}_{i-1}[k]) + \frac{\partial f_{\boldsymbol{v}}}{\partial \boldsymbol{\Psi}}_{\Psi_{i-1}[k]} \delta\Psi_i[k] \qquad \kappa \in [0, k_f)$$

$$\boldsymbol{a}[k] = \frac{1}{m[k]}(\boldsymbol{T}[k] + \boldsymbol{D}[k]) + \boldsymbol{a}_R[k] + \boldsymbol{g}$$

State Constraints:

$$m_{dry} \leq m[k]$$

Control Constraints:

$$\| \boldsymbol{T}[k] \| \leq \Gamma[k]$$

$$0 \leq T_{min} \leq \Gamma[k] \leq T_{max}$$

$$\Gamma[k]\cos(\theta_{max}) \leq \hat{\boldsymbol{e}}_u^T \boldsymbol{T}[k]$$

$$\dot{T}_{min}\Delta\tau \leq \Gamma[k+1] - \Gamma[k] \leq \dot{T}_{max}\Delta\tau \qquad \kappa \in [0, k_f)$$

SC Modifications:

$$\delta\Delta t_i^2 \leq \eta_{\Delta t}$$

$$\delta\boldsymbol{T}_i^T[k]\delta\boldsymbol{T}_i[k] \leq \eta_{\boldsymbol{T}}[k]$$

$$\| \boldsymbol{a}_R[k] \| \leq \kappa_{\boldsymbol{a},R}[k]$$

3 Algorithm Solution

In this section I present the two solution cases that will be used for the remainder of the report. The first is a planar case, where all of the vehicle's motion occurs within the U-E plane, while the second is a full, three-dimensional case. The exact parameters and boundary conditions used for the two solution cases are summarized in tables 1 and 2.

Both problems were solved in MATLAB using CVX [17] with the SDPT3 [15] solver.

3.1 Solution 1

The specific parameters for solution 1 are given in Table 1.

Parameter	*Value*	*Units*
ρ	0.0023	slugs/ft^3
P_{amb}	2116	lbf/ft^2
g_0	32.17	ft/s^2
g	$[-g_0 \quad 0 \quad 0]^T$	ft/s^2
m_0	33,000	lbm
m_{dry}	22,000	lbm
A_{nozzle}	5	ft^2
I_{sp}	300	s
$T_{min,vac}$	22,000	lbf
$T_{max,vac}$	56,000	lbf
$\dot{T}_{min}$	22,000	lbf/s
$\dot{T}_{max}$	22,000	lbf/s
θ_{max}	15	degrees
S_D	108	ft^2
C_D	1.0	-
$t_{f,s}$	15	s
$\boldsymbol{r}_0$	$[1500 \quad 1500 \quad 0]^T$	ft
$\boldsymbol{v}_0$	$[-150 \quad 0 \quad 0]$	ft/s
$\hat{\boldsymbol{n}}_0$	$[1 \quad 0 \quad 0]^T$	-
$\hat{\boldsymbol{n}}_f$	$[1 \quad 0 \quad 0]^T$	-
N	50	-
n_{SC}	10	-
$w_{m,f}$	1.0	slugs^{-1}
$w_{\eta,\Delta t}$	1.0	s^{-1}
$w_{\eta,T}$	1.0	lbf^{-1}
$w_{\kappa,a,R}$	500	s^2/ft

Table 1: Solution 1 parameters

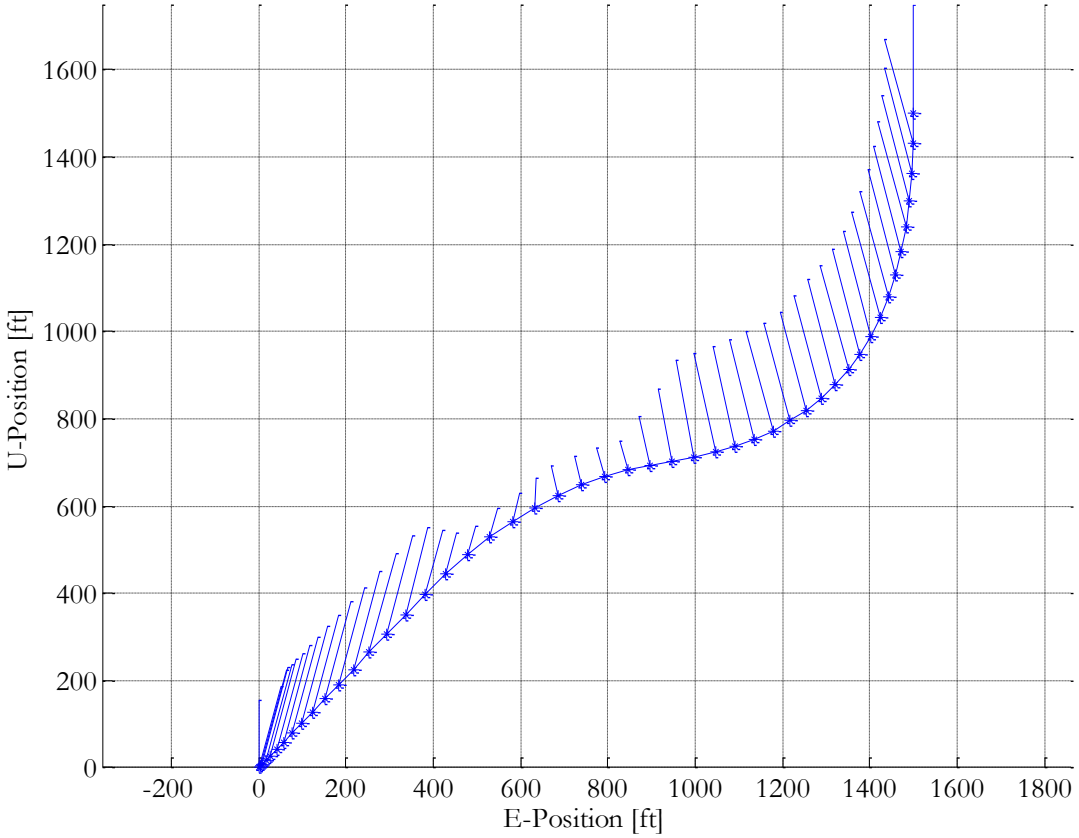

Figure 1: Plot of solution trajectory in the U-E plane. Each dot on the plot represents one of the discretized time points and are equally space temporally. The lines radiating out of the time points are the relatives thrust directions and magnitudes at that point.

This particular set of parameters in Table 1 is representative of a situation in which the vehicle is 1500 ft up and 1500 away, horizontally, from its intended landing spot. The vehicle is travelling straight down at a speed of 150 ft/s. The environmental conditions are similar to those that one might experience near sea level on Earth. Vehicle-specific parameters include a vehicle mass of 22,000 lbm laden with 11,000 lbs of fuel. It can throttle up and down at a max rate of 22,000 lbf/s and has a maximum tilt angle of 15 degrees.

The number of discretization points and SC iterations were chosen by observing that the nature of the solution did not change appreciably for numbers larger than 50 and 10, respectively.

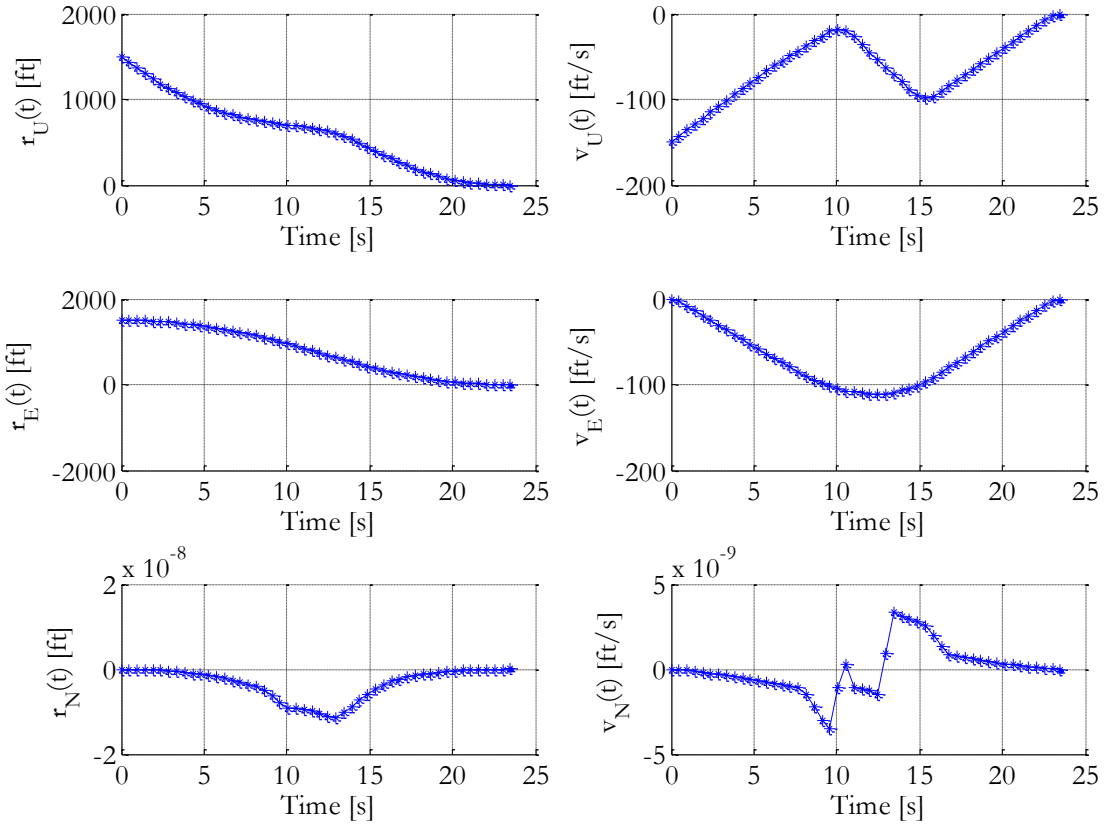

Figure 2: Position and velocity time history, broken up by components.

Figure 1 gives a nice overview of the vehicle's path to the landing pad, and shows that the trajectory ends with the position and velocity vectors equal to zero, as the boundary conditions specify.

Figure 3 truly elucidates the control that is used to generate that trajectory. Here we can see that the thrust magnitude roughly approximates the canonical "bang-off-bang" form of a minimum fuel solution. Also note at 10 and 15 seconds, where the thrust magnitude rate saturated, preventing the thrust magnitude from changing any more rapidly. The thrust starts off at full throttle and immediately tilts as far as it can to the west. This makes sense since the ratio of starting altitude to horizontal distance is unity. Halfway through the trajectory, the thrust decreases until it hits the minimum thrust magnitude constraint and coasts there for a few seconds, before tilting back east and throttling up to kill the westward velocity it has built up.

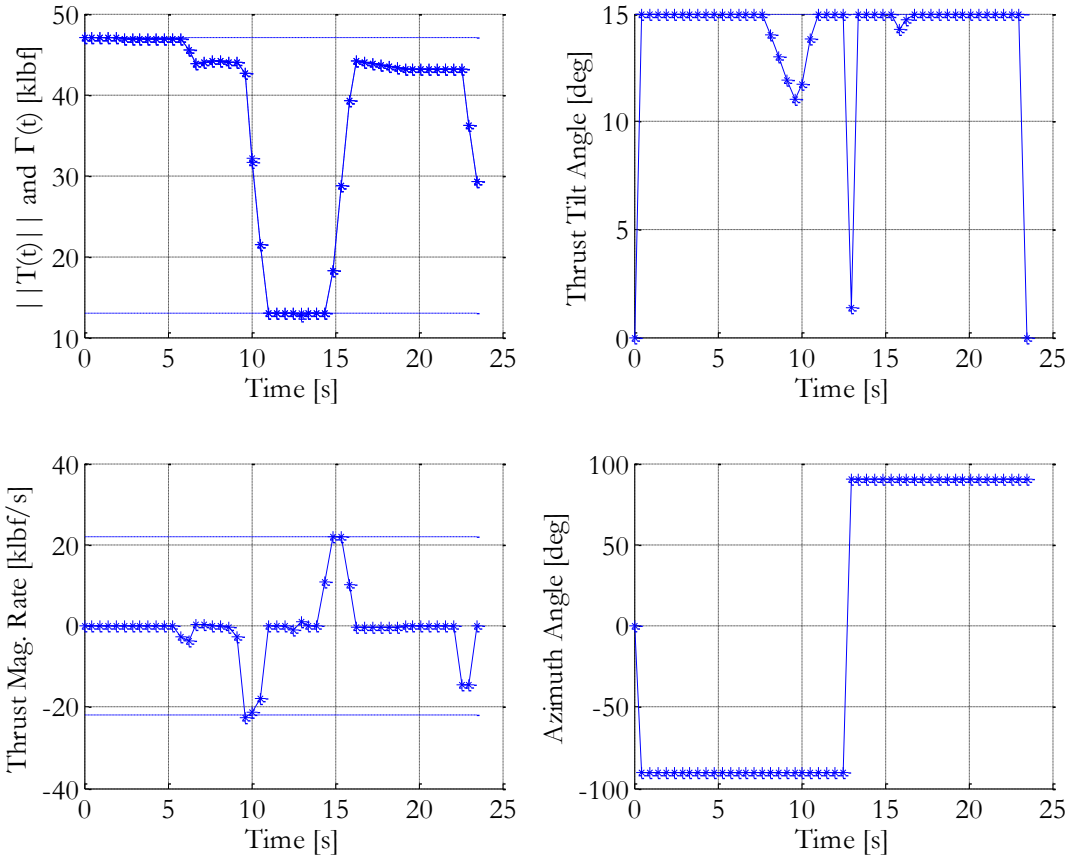

Figure 3: Thrust time history.

Every single control constraint that was imposed was utilized at some point on the trajectory. The thrust magnitude hit both the minimum and maximum, as did the thrust magnitude rate. The tilt angle was at its maximum for almost the entire time of the flight. Note in Figure 2 that the N components of position and velocity are very small because of the planar nature of the trajectory.

The mass time history in Figure 4 shows the vehicle mass decreasing as propellant is burned, but the total mass is always above the vehicle's dry mass.

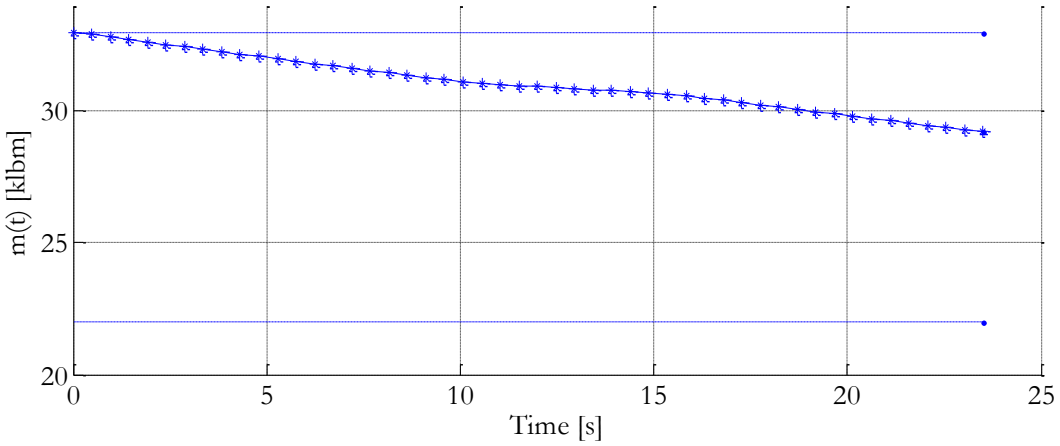

Figure 4: Mass time history. The upper and lower dashed lines represent the vehicles initial and dry mass, respectively.

3.2 Solution 2

Solution 2 is very similar to solution 1, with the exception of an out-of-plane initial velocity. This throws the resultant trajectory into the third dimension and unveils behaviors not see in the planar case.

The specific parameters for solution 2 are given in Table 2, below.

Parameter	*Value*	*Units*
ρ	0.0023	slugs/ft^3
P_{amb}	2116	lbf/ft^2
g_0	32.17	ft/s^2
g	$[-g_0 \quad 0 \quad 0]^T$	ft/s^2
m_0	33,000	lbm
m_{dry}	22,000	lbm
A_{nozzle}	5	ft^2
I_{sp}	300	s
$T_{min,vac}$	22,000	lbf
$T_{max,vac}$	56,000	lbf
$\dot{T}_{min}$	22,000	lbf/s
$\dot{T}_{max}$	22,000	lbf/s
θ_{max}	15	degrees
S_D	108	ft^2
C_D	1.0	-

Parameter	*Value*	*Units*
$t_{f,s}$	15	s
$\boldsymbol{r}_0$	$[1500 \quad 1500 \quad 0]^T$	ft
$\boldsymbol{v}_0$	$[-150 \quad 0 \quad 150]$	ft/s
$\hat{\boldsymbol{n}}_0$	$[1 \quad 0 \quad 0]^T$	-
$\hat{\boldsymbol{n}}_f$	$[1 \quad 0 \quad 0]^T$	-
N	50	-
n_{SC}	10	-
$w_{m,f}$	1.0	slugs^{-1}
$w_{\eta,\Delta t}$	1.0	s^{-1}
$w_{\eta,T}$	1.0	lbf^{-1}
$w_{\kappa,a,R}$	500	s^2/ft

Table 2: Solution 2 parameters

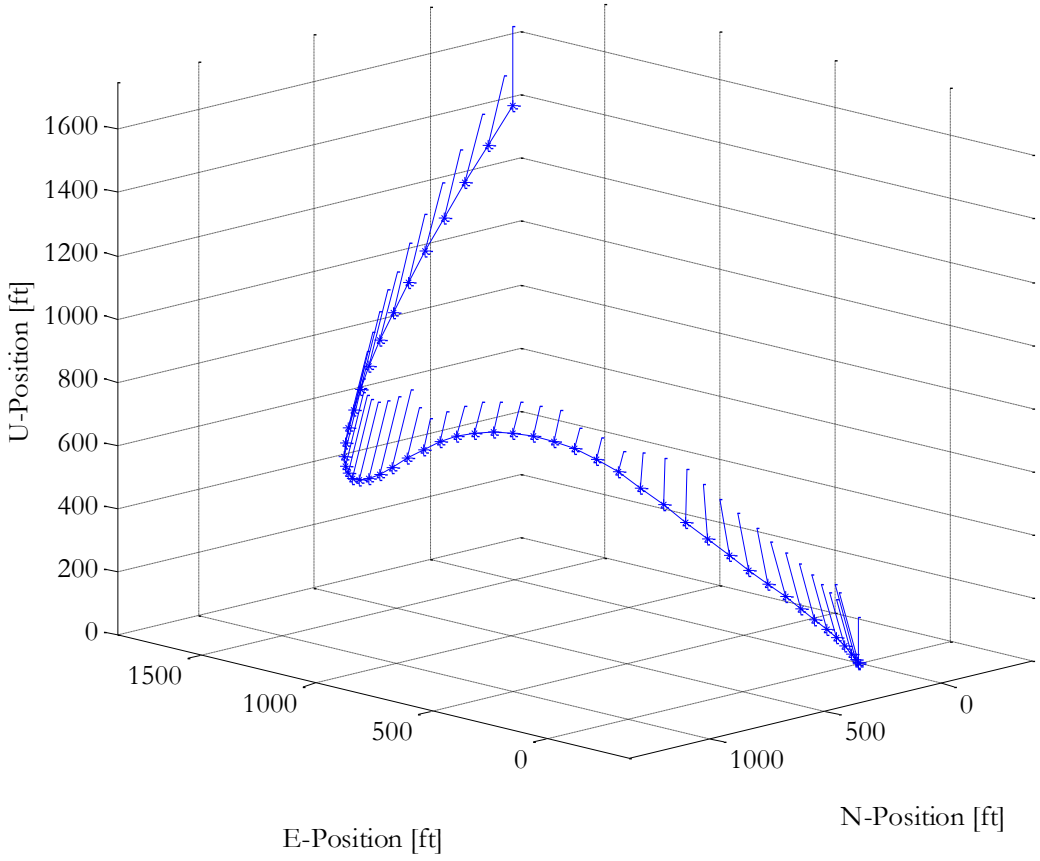

Figure 5: Plot of solution trajectory in the three dimensions.

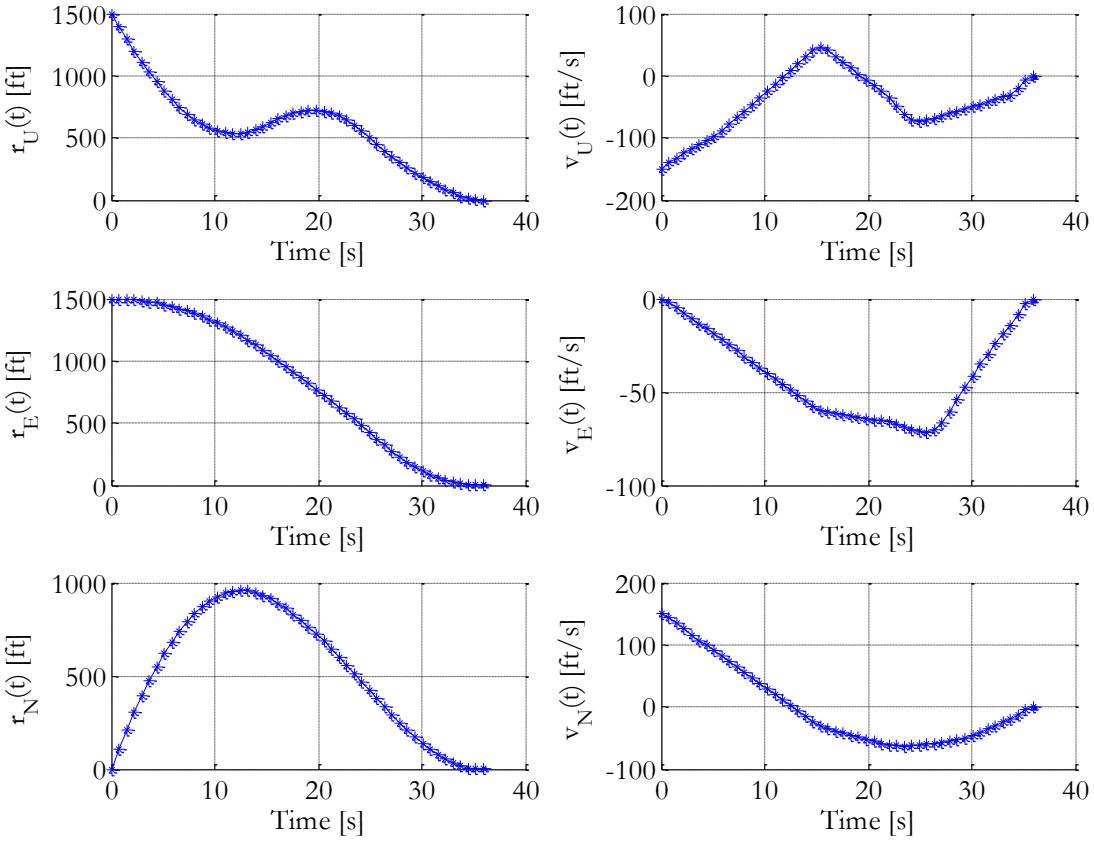

Figure 6: Position and velocity time history, broken up by components.

Solution 2 is similar to solution 1 with the exception of its out-of-plane maneuvers. It also utilizes a sort of hop during the coast phase of the bang-coast-bang thrust structure, as shown in Figure 6. This may be the fuel optimal solution, but it might not be suitable for actual flight conditions, due to excessive fuel sloshing. A simple series of altitude monotonicity constraints could eliminate this type of behavior if desired.

The initial lateral velocity is clearly evident in Figure 5, causing the thrust vectors to point south to zero out the vehicle's N position. The thrust magnitude plot in Figure 7 is not as simple as the previous, planar problem. Note that although the magnitude is initialized at full throttle setting, it drops to a lower setting within the first discretization point, indicating that starting at full throttle was not optimal.

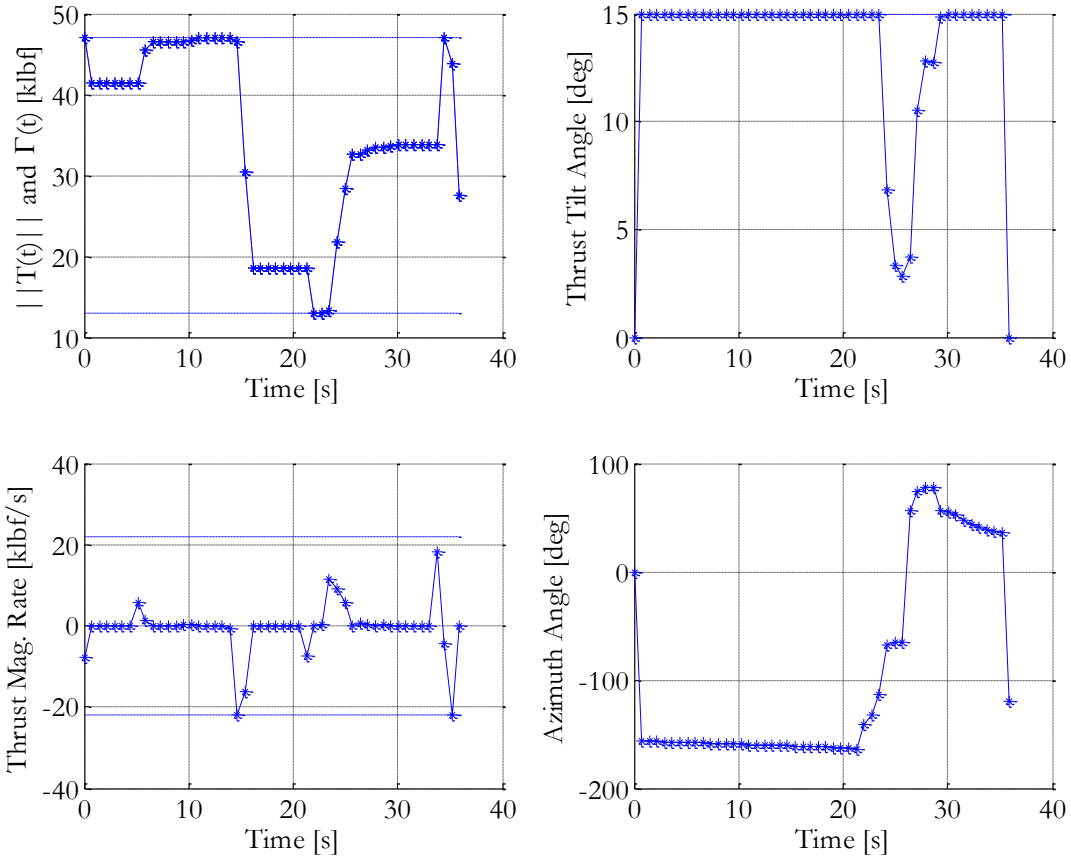

Figure 7: Thrust time history.

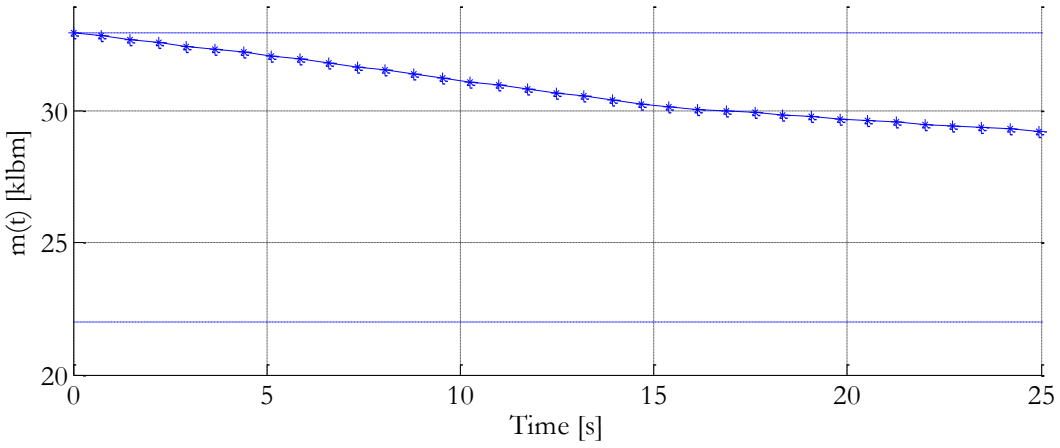

Figure 8: Mass time history for solution 2.

4 Nonlinear Simulation

In this section, I compare the solutions obtained in section 3 to a nonlinear simulation of the same thrust profile, to satisfy two project objectives. First, this will serve to verify that the solutions obtained from the convexified problem are physically feasible. Second, the use of a nonlinear simulation introduces the possibility of adding in nonlinear elements such as variable air density, coefficient of drag, and atmospheric pressure. Running the simulation with these present will give us some idea of how good our simplifying approximations are.

Section 4.1 introduces the simulation and briefly describes its formulation, and section 4.2 presents the results of the simulations.

4.1 Simulation Formulation

The nonlinear simulation that was used to verify the CVX solutions was built in Simulink. The basic outline of the model can be seen in Figure 9.

This model takes the thrust profile from the solutions in section 3, and propagates the trajectory based on that alone, subject to the expected dynamics. The important differences between Simulink and CVX are the integration scheme and the number of discretization points. This model utilizes the Dormand-Prince Runge-Kutta integration method which, with the relative tolerance set to $1E-6$, results in a number of discretization points that is between two to ten times more than the number typically used with the CVX solutions.

It its simplest form, the model's dynamics recreates the problem formulation from section 2.1 exactly. The translational dynamics follow a simple double integrator where the acceleration term is driven by gravity and a thrust control. The atmospheric pressure and density and the vehicle's coefficient of drag are all constant. The mass depletion dynamics follow the same linear model described above.

This is then built upon with simple Boolean switches to enable or disable certain nonlinear aspects of the simulation. In no particular order, these are: atmospheric thrust correction, variable coefficient of drag, atmospheric lift, and variable air density.

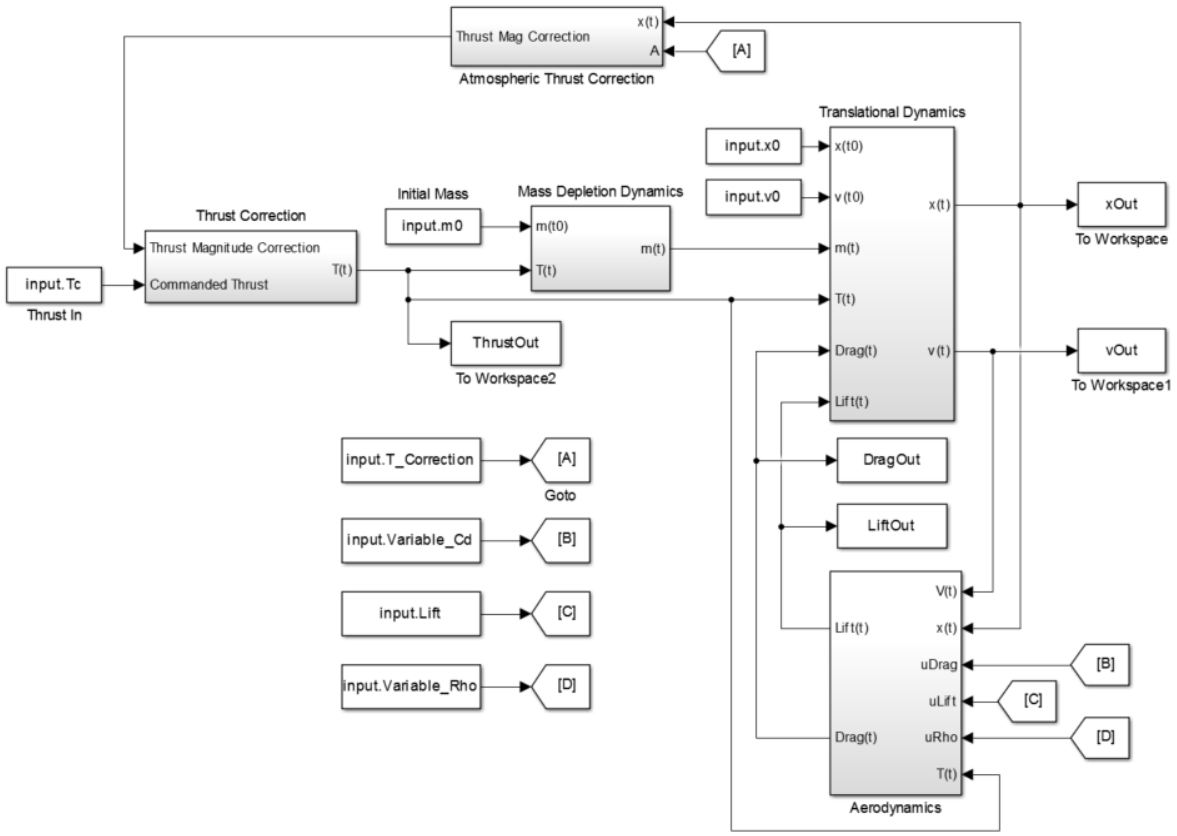

Figure 9: Simulink PDL simulation.

The atmospheric thrust correction accounts for the fact that for a given throttle setting, thrust will decrease with decreasing altitude, as atmospheric pressure increases. The corrective term in the Simulink simulation computes the air pressure for the vehicle's current altitude, multiplies it by the nozzle area, then adds or subtracts that value from the commanded thrust, depending on whether the air pressure is higher or lower than the assumed constant value, respectively.

The variable coefficient of drag correction takes into account the fact that the vehicle's coefficient of drag is dependent on the vehicle's speed. This correction replaces the usual, constant C_D with a coefficient that varies linearly according to Table 3, below.

Vehicle Speed [ft/s]	C_D
0	0.2
500	1.2

Table 3: Coefficient of drag vs vehicle speed.

The third correction accounts for the lift being created by the vehicle during the trajectory. Here, lift is being defined as the aerodynamic force normal to the velocity direction, and is expressed as:

$$\boldsymbol{L} = -\frac{1}{2}\rho S_L C_L \left(\frac{\boldsymbol{T}}{\Gamma} \cdot \frac{\boldsymbol{v}}{\| \boldsymbol{v} \|}\right)\left[\left(\frac{\boldsymbol{T}}{\Gamma} \times \boldsymbol{v}\right) \times \boldsymbol{v}\right]$$

For the purposes of this simulation, C_L is assumed to be 1.0, and the lift reference area is assumed to be equal to the drag reference area.

The final correction is variable air density, which is handled in a similar fashion to the thrust correction. The air density is calculated for the vehicle's current altitude based on a standard atmospheric model, and that value is then used for the ρ terms in lift and drag.

Finally, for purposes of comparing the Simulink results to those produced by CVX, the Fréchet distance is used as a comparison metric. For this application of discrete position time histories, this metric can be expressed as:

$$F(A,B) = \max(\| \boldsymbol{r}_A(1) - \boldsymbol{r}_B(1) \|, \| \boldsymbol{r}_A(2) - \boldsymbol{r}_B(2) \|, \dots, \| \boldsymbol{r}_A(N) - \boldsymbol{r}_B(N) \|)$$

4.2 Results

This section will start by showing the detailed results from one particular simulation: the three dimensional solution 2 from section 3.2 with the variable C_D flag set to true. Then I summarize the results from the remaining cases by using the Fréchet distance for each case in table 4, below.

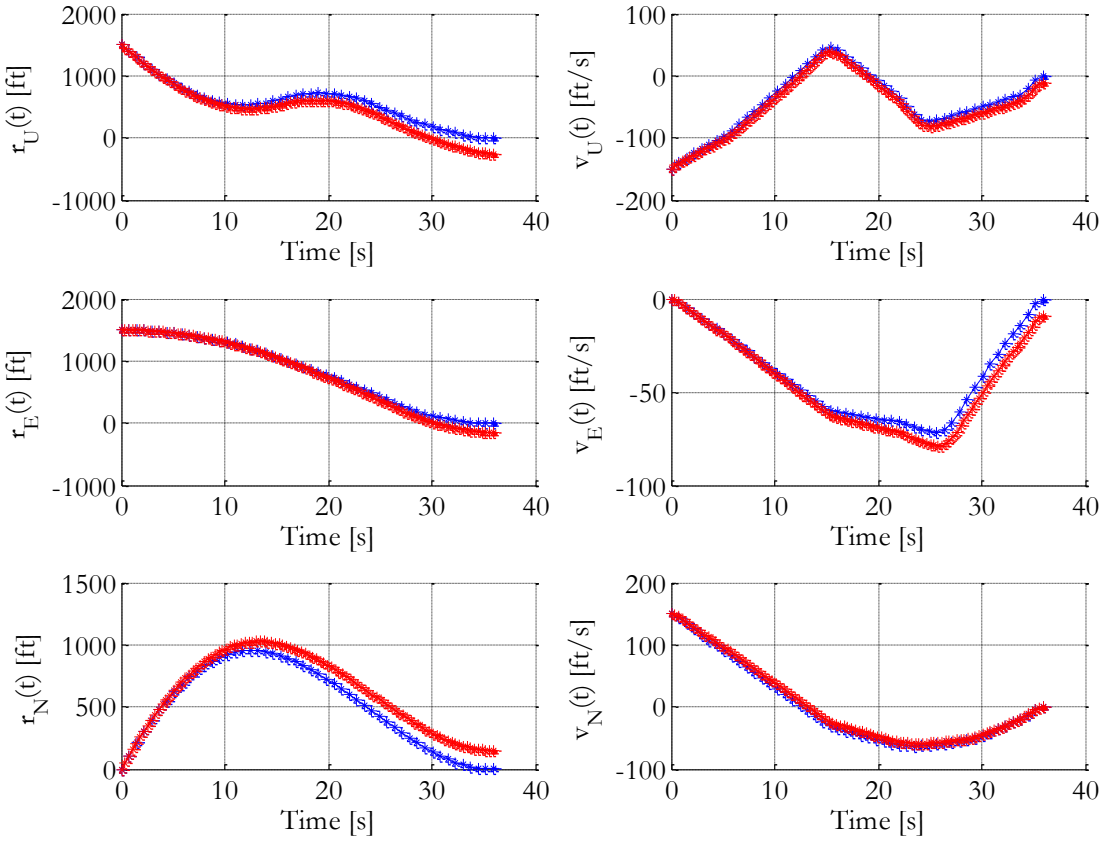

Figure 10: Position and velocity plots for CVX solution and Simulink Simulation. Variable C_D is turned on. CVX solution is in blue and the Simulink propagated trajectory is shown in red.

Note in Figures 10 and 11 that both the Up and East Simulink positions overshoot the zero position that the CVX solution successfully targeted. This makes intuitive sense as most of trajectory was spent well below 500 ft/s, so the coefficient of drag in the simulation was lower than the constant value of 1.0 used in CVX, and the vehicle was not slowed down as much, resulting in that overshoot.

Because the thrust is an input to the Simulink simulation and the mass profile is explicitly dependent on the thrust profile, the CVX-Simulink comparison of those two plots are trivially zero.

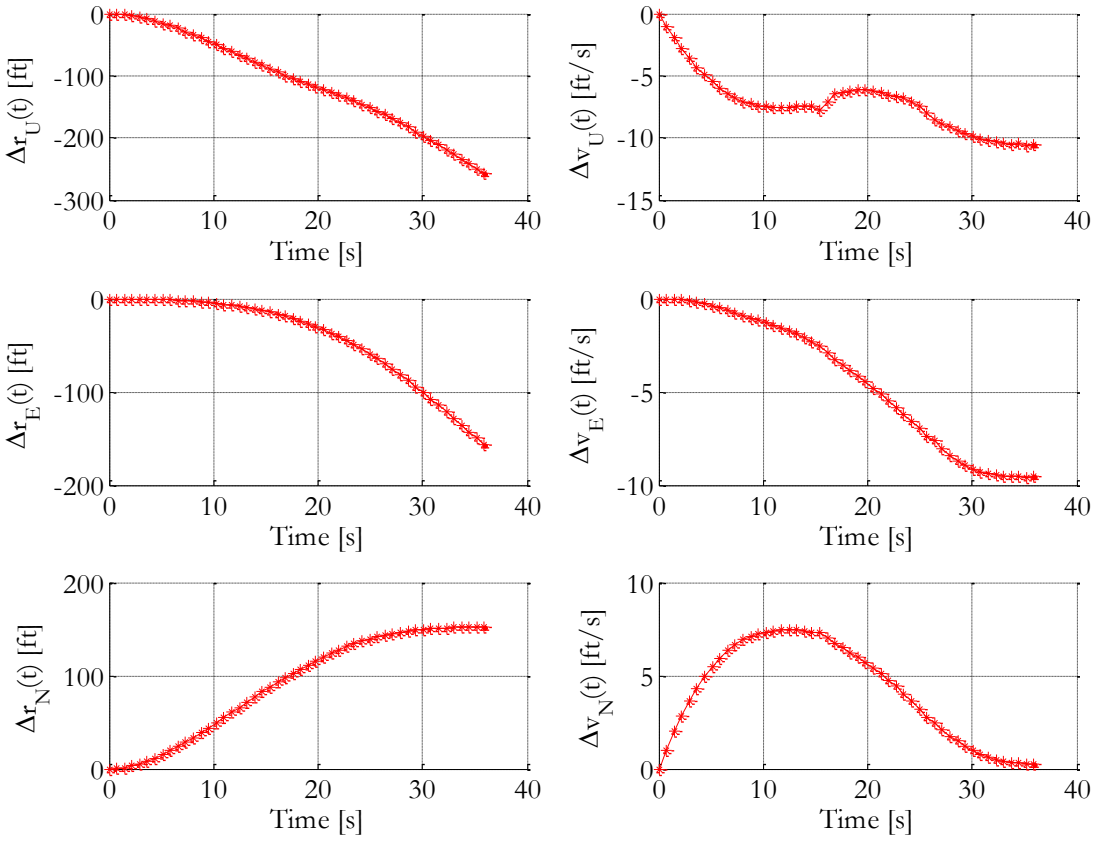

Figure 11: Relative position and velocity time histories. Variable C_D is turned on. The curves shown here are the CVX solution minus the Simulink propagation.

Fréchet Distance [ft]	*2-D Solution 1*	*3-D Solution 2*
None	0.5058	1.7850
Thrust Correction	6.8609	16.9180
Variable C_D	199.9261	337.9599
Aerodynamic Lift	136.1769	340.6984
Variable ρ	49.6554	99.5409

Table 4: Fréchet distances for selected simulation cases.

Note that the first row of Table 4, when all the nonlinearity flags are set to false, the Fréchet distances are not zero. This small difference in trajectories is due to the difference in integration schemes between the convex problem, which linearly interpolates the acceleration term between discrete time points, and the Simulink simulation, which utilizes a Runge-Kutta integration method.

The simulation was initially verified by setting both integration schemes to a simple zero-order-hold newton step, and the resulting Fréchet distances were on the order of $1E-5$ ft. The fact that these two different integration schemes only vary by a foot or two over the course of the entire trajectory reassures us that the integration strategy chosen for the original problem was a good one.

All four nonlinearities had a greater effect on solution 2. Due to that solution's 3-D nature, the time of flight is longer, so the differences in dynamics had a longer time to build up than for solution 1.

The two largest differences occurred with the variable C_D and aerodynamic lift flags. This might suggest that those two dynamic perturbations would be a good addition if one wanted to improve the fidelity of the original convex problem.

5 GPOPS-II Verification

GPOPS-II [22] is an optimal control MATLAB software for solving general nonlinear optimal control problems. Section 4 used Simulink to verify the physical feasibility of the CVX solution, now I will use GPOPS-II to verify the optimality of it.

5.1 GPOPS-II Formulation

Because GPOPS-II allows for such extremely general dynamics, constraints, and boundary conditions, I was able to formulate the problem exactly how it is stated in Section 2. All of the vehicle and environmental parameters are the same as those stated in tables 1 and 2.

5.2 Results

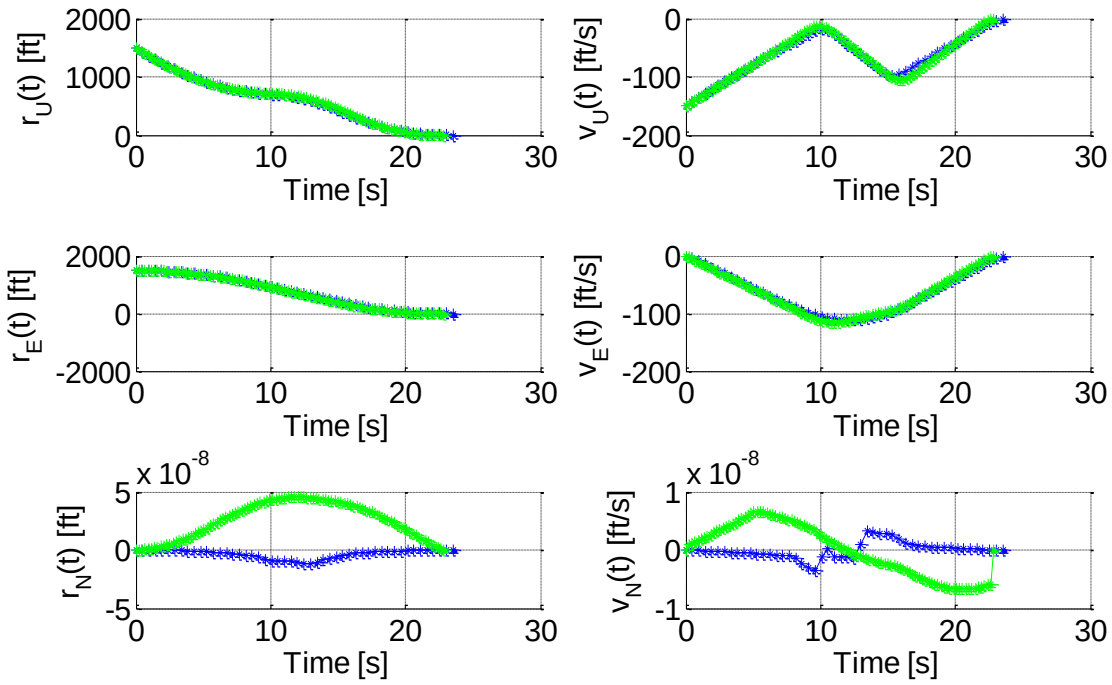

Figure 12: Position and Velocity time histories for solution 1. The CVX solution is shown in blue and GPOPS-II solution is shown in green.

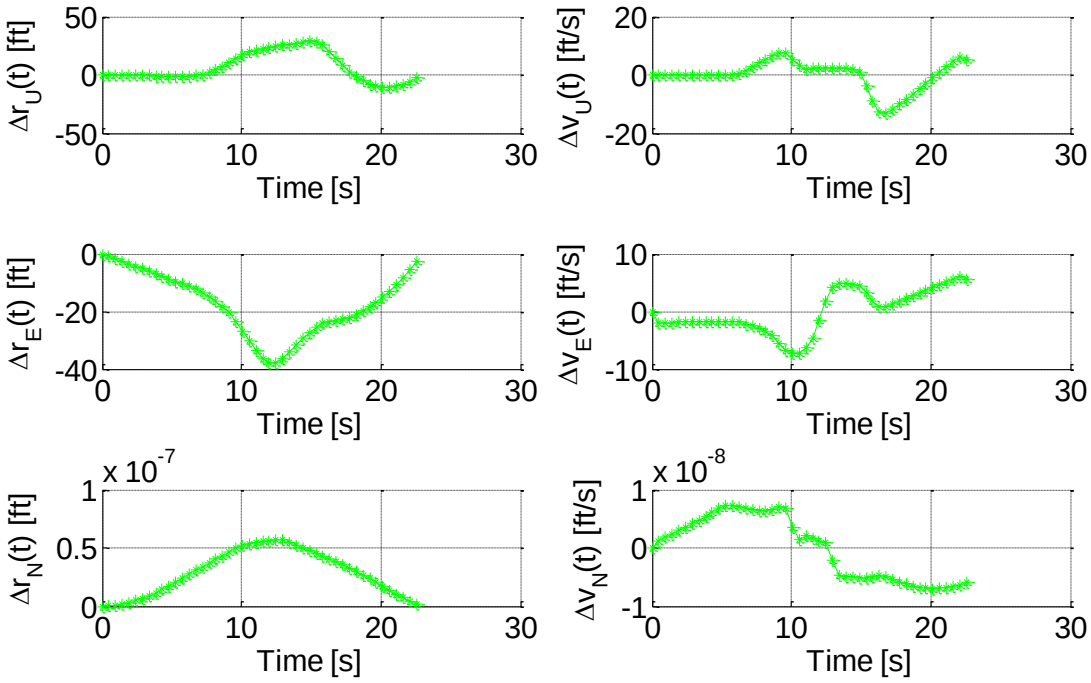

Figure 13: Relative position and velocity time histories. The curves shown here are the CVX solution minus the GPOPS-II. Notes that because final time is left unconstrained, the total time of flight for each solution is slightly different, and is scaled for comparison purposes.

Figures 11-14 show the CVX-GPOPS comparison for the first solution case described in section 3. The Fréchet distance between the two is 45 ft, and the GPOPS-II solution uses approximately 180 lbs, or 5% less fuel. The biggest difference is seen in the thrust profile, where the GPOPS-II solution exhibits a very clean bang-coast-bang structure and saturates at the max and min thrust, whereas the CVX solution does not. This discrepancy is most like due to the linearized SOCP problem not perfectly representing the general nonlinear problem.

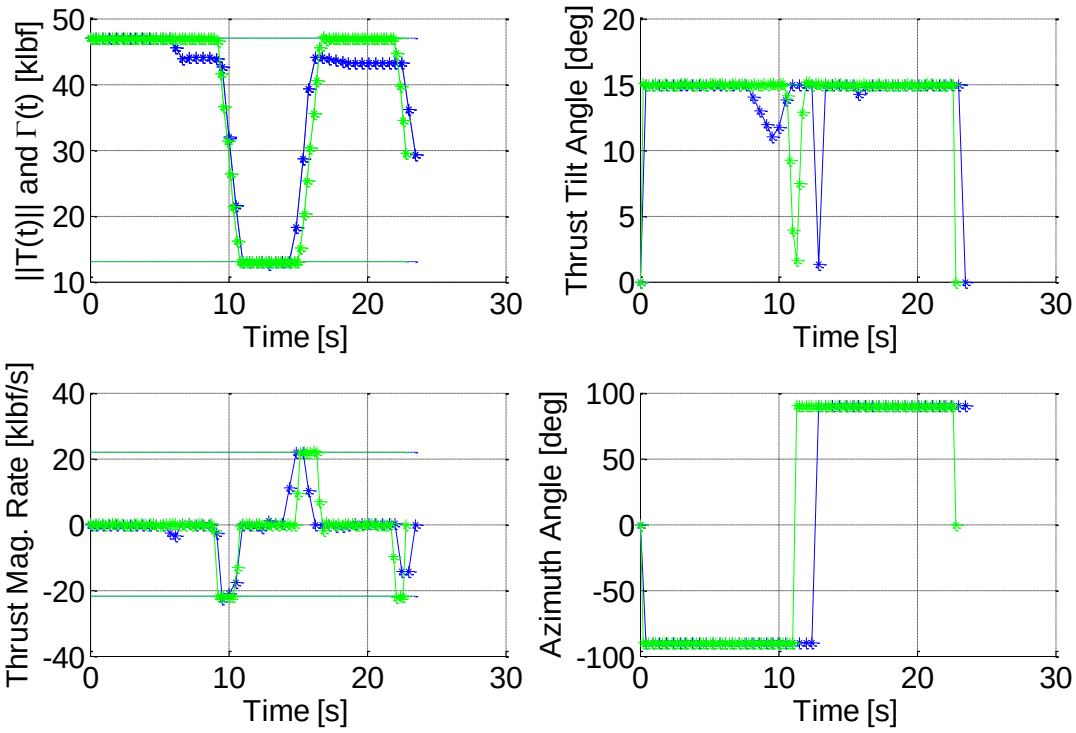

Figure 14: Thrust profile for solution 1. The CVX solution is shown in blue and GPOPS-II solution is shown in green.

6 Conclusions

In this report, I described a strategy for expressing a non-convex problem in a convex manner, then verified its physical feasibility and its optimality. The original problem included non-convex dynamics such as aerodynamic drag, as well as non-convex constraints. It physical feasibility was evaluated using a nonlinear Simulink simulation, and the two resulting trajectories were found to correspond closely. The algorithm's optimality was verified by comparing it to a solution from GPOPS-II, and it was found to have a slightly larger objective function than the actual optimal solution. This is most likely due to issues arising from the linearization of the problem.

Nomenclature

DOF	Degrees-of-Freedom
SOCP	Second-Order Cone Programming
VTVL	Vertical Takeoff, Vertical Landing
PDL	Powered Decent and Landing
SC	Successive Convexification
i	Index of SC iteration
$\hat{e}_U$	Up-pointing unit vector
m	Vehicle mass
$\boldsymbol{r}$	Vehicle position vector
$\boldsymbol{v}$	Vehicle velocity vector
$\boldsymbol{a}$	Vehicle acceleration vector
$\boldsymbol{T}$	Vehicle thrust vector
$\boldsymbol{D}$	Vehicle drag vector
$\boldsymbol{g}$	Local gravity vector
g_0	Standard gravity
ρ	Local air density
P_{amb}	Local air pressure
A_{nozzle}	Nozzle exit area of vehicle rocket motor
I_{sp}	Specific impulse of vehicle rocket motor
C_D	Vehicle coefficient of drag
C_L	Vehicle coefficient of Lift
S_D	Vehicle drag reference area
S_L	Vehicle lift reference area
θ	Vehicle tilt angle
N	Number of discretization points
n_{SC}	Number of successive convexification iterations

Bibliography

1) Ploen, S., Açıkmeşe, B., and Wolf, A., "A Comparison of Powered Descent Guidance Laws for Mars Pinpoint Landing," AIAA GNC Conference and Exhibit, Keystone, CO, 2006.
2) Wolf, A. A., Tooley, J., Ploen, S., Ivanov, M., Acikmese, B., and Gromov, K., "Performance trades for Mars pinpoint landing," Aerospace Conference, 2006 IEEE, IEEE, 2006, pp. 16–pp. 3
3) Carson, J.M., Açıkmeşe, B., Blackmore, L., and Wolf, A.A., "Capabilities of Onboard, Convex Powered-Descent Guidance Algorithms for Pinpoint and Precision Landing," Aerospace Conference, 2011 IEEE, IEEE, Big Sky, MT, 03/2011 2011.
4) Açıkmeşe, B.and Ploen, S.R., "A powered descent guidance algorithm for Mars pinpoint landing," AIAA GNC Conference and Exhibit, San Francisco, 2005.
5) Açıkmeşe, B. and Ploen, S. R., "Convex Programming Approach to Powered Descent Guidance for Mars Landing," AIAA Journal of Guidance, Control and Dynamics, Vol. 30, No. 5, 2007, pp. 1353–1366.
6) Blackmore, L., Açıkmeşe, B., and Scharf, D. P., "Minimum Landing Error Powered Descent Guidance for Mars Landing using Convex Optimization," AIAA Journal of Guidance, Control and Dynamics, Vol. 33, No. 4, 2010.
7) Açıkmeşe, B., Carson, J., and Blackmore, L., "Lossless Convexification of Non-convex Control Bound and Pointing Constraints of the Soft Landing Optimal Control Problem," IEEE Transactions on Control Systems Technology, Vol. 21, No. 6, 2013, pp. 2104–2113.
8) Açıkmeşe, B. and Blackmore, L., "Lossless Convexification of a Class of Optimal Control Problems with Non-Convex Control Constraints," Automatica, Vol. 47, No. 2, 2011, pp. 341–347.
9) Harris, M. and Açıkmeşe, B., "Lossless Convexification of Non-convex Optimal Control Problems for State Constrained Linear Systems," Automatica, Vol. 50, No. 9, 2014, pp. 2304–2311.
10) Harris, M. W. and Açıkmeşe, B., "Maximum Divert for Planetary Landing Using Convex Optimization," Journal of Optimization Theory and Applications, 2013, pp. 1–21.
11) Blackmore, L., Açıkmeşe, B., and Carson, J. M., "Lossless convexification of control constraints for a class of nonlinear optimal control problems," System and Control Letters, Vol. 61, No. 4, 2012, pp. 863–871.
12) Casoliva, J., Spacecraft Trajectory Generation by Successive Approximation for Powered Descent and Cyclers, Ph.D. thesis, University of California, Irvine, 2013.
13) Liu, X. and Lu, P., "Solving Nonconvex Optimal Control Problems by Convex Optimization," Journal of Guidance, Control, and Dynamics, Vol. 37, No. 3, 2014, pp. 750–765.
14) Boyd, S. and Vandenberghe, L., Convex Optimization, Cambridge University Press, 2004.
15) Toh, K., Todd, M., and Tutuncu, R., "SDPT3—a Matlab Software Package for Semidefinite Programming," Optimization Methods and Software, Vol. 11, No. 1, 1999, pp. 545–581.
16) Dueri, D., Zhang, J., and Açıkmeşe, B., "Automated Custom Code Generation for Embedded, Real-Time Second Order Cone Programming," 19th IFAC World Congress, 2014, pp. 1605–1612.
17) Grant, M. and Boyd, S., "CVX: Matlab Software for Disciplined Convex Programming, version 2.1," http://cvxr.com/cvx, March 2014.

18) Domahidi, A., Zgraggen, A. U., Zeilinger, M. N., Morari, M., and Jones, C. N., "Efficient interior point methods for multistage problems arising in receding horizon control," Decision and Control (CDC), 2012 IEEE 51st Annual Conference on, IEEE, 2012, pp. 668–674.
19) Açıkmeşe, B., Aung, M., Casoliva, J., Mohan, S., Johnson, A., Scharf, D., Masten, D., Scotkin, J., Wolf, A., and Regehr, M. W., "Flight Testing of Trajectories Computed by G-FOLD: Fuel Optimal Large Divert Guidance Algorithm for Planetary Landing," AAS/AIAA Spaceflight Mechanics Meeting, 2013.
20) Scharf, D. P., M. W. Regehr, D. D., Açıkmeşe, B., Vaughan, G. M., Benito, J., Ansari, H., M. Aung, A. J., Masten, D., Nietfeld, S., Casoliva, J., and Mohan, S., "ADAPT Demonstrations of Onboard Large-Divert Guidance with a Reusable Launch Vehicle," IEEE Aerospace Conference, 2014.
21) Carson, J. and Açıkmeşe, B., "A model predictive control technique with guaranteed resolvability and required thruster silent times for small-body proximity operations," Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit, 2006
22) Patterson, M. A. and Rao, A. V., "GPOPS-II: A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using hp-Adaptive Gaussian Quadrature Collocation Methods and Sparse Nonlinear Programming," ACM Transactions on Mathematical Software, Vol. 41, No. 1, Article No. 1, October 2014, pp. 1:1 - 1:37.
23) Szmuk, M., Açıkmeşe, B., and Berning, A., "Successive Convexification for Fuel-Optimal Powered Landing with Aerodynamic Drag and Non-Convex Constraints," AIAA Guidance, Navigation, and Control Conference, San Diego, CA, 2016.