Optimal Control Problem for Minimum-Fuel Rocket Vertical Landing

Robin Loh

Abstract—**In this project I formulate and solve a minimum-fuel rocket vertical landing optimal control problem. This problem was adapted from a paper which solves a minimum-fuel powered descent guidance problem using a point-mass model. Constraints were added to, and adapted from, this problem which resulted in a continuous time OCP which models the rocket as a rectangle with boosters on one end. This problem was then discretized and solved. The results display the *max-min-max* thrust pattern which is typical of such minimum-fuel trajectories.**

Nomenclature

$\mathbf{e_i}$	vector of zeroes with unity at index i
$\mathbf{g}$	vector containing constant acceleration due to gravity in each spatial dimension
h	height of rocket rectangle i.e. length of longer side
$m(t)$	mass of rocket at time t
m_{dry}	dry mass of rocket
m_N	terminal mass of rocket
m_{wet}	wet mass of rocket
N	number of discrete time steps
$\mathbf{q}_i$	constant control vector on time interval i
$\mathbf{r}(t)$	center of mass position at time t
$\dot{\mathbf{r}}(t)$	center of mass velocity at time t
$\ddot{\mathbf{r}}(t)$	center of mass acceleration at time t
$\mathbf{r}_N$	final position of center of mass
$\dot{\mathbf{r}}_N$	final velocity of center of mass
$\mathbf{r}_0$	initial position of center of mass
$\dot{\mathbf{r}}_0$	initial velocity of center of mass
$\mathbf{s}_i$	control vector at ith temporal node
t	time
t_f	final time
t_k	time at kth time step
$\mathbf{U}(t)$	control vector at time t
w	width of rocket rectangle i.e. length of shorter side
$\mathbf{X}(t)$	state vector at time t
α	constant of proportionality between thrust magnitude and mass consumption
Δt	length of time step for discretization
ρ	upper bound on thrust magnitude
θ	angle between rocket rectangle and vertical direction
$\dot{\theta}$	angular velocity of rocket rectangle
$\ddot{\theta}$	angular acceleration of rocket rectangle
θ_N	final value for θ
$\dot{\theta}_N$	final value for $\dot{\theta}$
θ_0	initial value for θ
$\dot{\theta}_0$	initial value for $\dot{\theta}$
$\mathbf{0}$	vector consisting of all zeroes
$\lvert\lvert\cdot\rvert\rvert$	two-norm of a vector

I Introduction

From communications satellites to scientific instruments to human beings, the demand for delivering payloads into space has grown rampantly over the last several decades. Unsurprisingly, so has the importance of cheapening the price of launching rockets. One way in which the company SpaceX has significantly reduced the price of delivering these payloads is by making their rockets reusable through vertical landing capabilities. In this project, I approach the vertical landing problem through the lens of optimal control. More specifically, I formulate this problem in 2 dimensions as a continuous time optimal control problem and transform it into a finite-dimensional NLP. Then I use the Python interface of Casadi [1] to solve this discretized problem with IPOPT [2].

The problem formulation took inspiration from a paper [3] which solves a minimum-fuel powered descent guidance problem using a point-mass model. The problem formulated in that paper, however, differs from the version which I have used here in that I have relaxed a constraint to make the search space for the controls convex, whereas the main contribution of [3] is a method of ”lossless convexification” for this problem i.e. a convex version is formulated whose solutions also give solutions for the nonconvex version. The adaptation presented here does not exhibit this property: it is a mere simplification to serve as a starting point for the vertical landing model. Once this simplified point-mass problem was solved, I added variables and constraints to model the vertical landing problem in 2 dimensions, and solved this NLP in much the same way as the first.

The rest of the report is organized as follows: Sec. II details the formulation of the continuous time optimal control problem for rocket vertical landing beginning with the simplified convex version of the point-mass model described above, while Sec. III describes the discretization of the vertical landing problem and the resulting NLP. Sec. IV presents and discusses some simulation results, while Sec. V summarizes the conclusions of this report and discusses future work to better model the problem.

II Continuous Time Formulation

In this section, I present the formulation of the rocket vertical landing problem, which found its beginnings in [3], where a point-mass model is presented for powered descent guidance. This model is nonconvex due to a nonzero lower bound on the thrust magnitude, which arises from the complications of thruster reignition in practice. The method of ”lossless convexification" and proof thereof contributed by [3] is very interesting but is not the focus of this project. Here I have simply relaxed this lower bound on the thrust magnitude to remove the nonconvexity. This is not a lossless convexification as the solutions do in fact exploit the zero-thrust capability.

Here is the relaxed version of the point-mass powered descent guidance problem:

Problem 1:

$$
\begin{aligned}
\min_{t_f, \mathbf{U}(\cdot)} \quad & -m(t_f) && \text{(1a)}\\
\text{s.t.} \quad & \ddot{\mathbf{r}}(t) = \mathbf{g} + \mathbf{U}(t)/m(t), && \text{(1b)}\\
& \dot{m}(t) = -\alpha||\mathbf{U}(t)||, && \text{(1c)}\\
& 0 \le ||\mathbf{U}(t)|| \le \rho, && \text{(1d)}\\
& \mathbf{e}_1^T \mathbf{r}(t) \ge 0, && \text{(1e)}\\
& m_{\text{dry}} \le m(t) \le m_{\text{wet}}, && \text{(1f)}\\
& m(0) = m_{\text{wet}}, && \text{(1g)}\\
& \mathbf{r}(0) = \mathbf{r}_0, && \text{(1h)}\\
& \dot{\mathbf{r}}(0) = \dot{\mathbf{r}}_0, && \text{(1i)}\\
& \mathbf{r}(t_f) = \dot{\mathbf{r}}(t_f) = \mathbf{0} && \text{(1j)}
\end{aligned}
$$

where:

- $\mathbf{U}(t) \in \mathbb{R}^3$ is the thrust profile acting on the point-mass
- $\mathbf{r}(t) \in \mathbb{R}^3$ is the position of the point-mass
- t_f is the final time, i.e., $t \in [0, t_f]$

Further descriptions of the variables may be found in the Nomenclature section before Sec. I. Equations (1b-1c) define the dynamics of the point-mass. Equation (1d) defines the constraints on the thrust magnitude, which is where the lower bound has been relaxed, and Eq. (1e) constrains the trajectory of the point-mass to always be above the ground. Here and throughout this report, I adopt the convention that the vertical direction is $\mathbf{e}_1$, and the first coordinate of vectors which refer to space ($\mathbf{r}, \dot{\mathbf{r}}, \ddot{\mathbf{r}}, \mathbf{g}$ etc.) refers to this direction. Equation (1f) enforces that the mass must always be between the initial and starting values, while Eq. (1g-1i) set the initial values. Equation (1j) sets the landing target position and velocity.

II-A Vertical Landing Problem Formulation

While in [3], the point-mass model is used to model trajectories in $\mathbb{R}^3$, the vertical landing problem we wish to formulate in this section should be strictly 2 dimensional, where the rocket is modeled by a rectangle with uniform mass distribution at all times $t \in [0, t_f]$, so that the center of mass is always the center of the rectangle. Figure 1 depicts the rectangle along with the control scheme.

II-A1 Control Definition

The description of the controls is as follows[1]:

$$\mathbf{U} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

[1]Note that in Figure 1, there are vectors $\mathbf{u}_1$ and $\mathbf{u}_2$, but the actual controls consist only of the magnitudes of these vectors. A state variable θ is later introduced which accounts for the direction of these magnitudes.

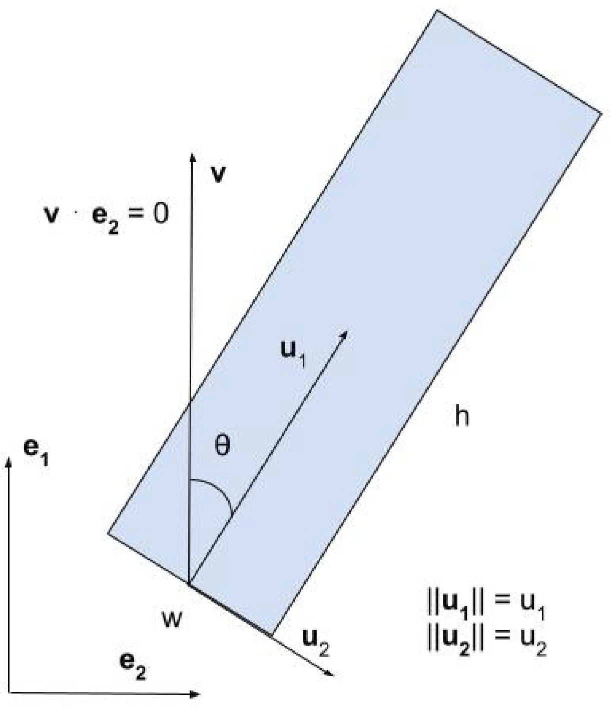

Fig. 1: Diagram which shows the controls as well as the state variable θ in relation to the unit vectors $\mathbf{e}_1$ and $\mathbf{e}_2$. In this diagram, $0 \le \theta \le \frac{\pi}{2}$, i.e. $\theta = 0$ implies that $\mathbf{u}_1$ is in the direction of $\mathbf{e}_1$.

where:

- $u_1 \in \mathbb{R}$ represents the magnitude of the force in Newtons through the center of mass of the rectangle, in the direction in which the rocket points. This models the effect of a "main" thruster which should do the heavy lifting of reducing the velocity of the center of mass to zero.
- $u_2 \in \mathbb{R}$ represents the magnitude of the force in Newtons applied perpendicularly to the long side of the rectangle, directly at the base where the "main" thruster would be located. This models the effect of having two thrusters on either side of the base (at the corners) which may be used to control the direction in which the rocket points.

Since the thrusters may only push and not pull, u_1 must be constrained to exclude the possibility of the main engine "sucking" the rocket backwards:

$$u_1 \ge 0$$

Such a constraint is not imposed on u_2 because $u_2 < 0$ implies a force pushing the base "to the left" (if the rocket faces upward as in Figure 1). This is part of our desired functionality.

Note the convenience of defining the controls in this way: Equation (1c) still applies to this model because $||\mathbf{U}(t)||$ is still the net magnitude of the thrust, even though $\mathbf{U}(t)$ has very different implications for the dynamics of each model.

II-A2 Complete State

A fundamental difference between modeling the lander as a point-mass versus a rectangular mass is that the rectangular mass has an orientation at all times t. We will account for this with the variable θ, which represents the angle the rocket makes with $\mathbf{e}_1$, the unit vector pointing straight up. The convention adopted here is as follows:

$\theta = 0$ implies that the rocket points in the positive $\mathbf{e}_1$ direction, while $\theta = \frac{\pi}{2}$ implies that the rocket faces directly in the positive $\mathbf{e}_2$ direction.

The following constraint says that the landing should be vertical and have zero angular velocity:

$$\theta(t_f) = \dot{\theta}(t_f) = 0$$

With θ and its time-derivative, $\dot{\theta}$, we are ready to formulate the generic state vector of the model:

$$\mathbf{X} = \begin{bmatrix} \mathbf{r} \\ \dot{\mathbf{r}} \\ \theta \\ \dot{\theta} \\ m \end{bmatrix}$$

where $\mathbf{r}, \dot{\mathbf{r}} \in \mathbb{R}^2$, and $\theta, \dot{\theta}, m \in \mathbb{R}$.

II-A3 Dynamics Formulation

With the complete state and control vectors in hand, we are ready to formulate the dynamics of the system:

$$f(\mathbf{X}, \mathbf{U}) = \dot{\mathbf{X}} = \begin{bmatrix} \dot{\mathbf{r}} \\ \ddot{\mathbf{r}} \\ \dot{\theta} \\ \ddot{\theta} \\ \dot{m} \end{bmatrix}$$

Since $\dot{\mathbf{r}}$ and $\dot{\theta}$ may be found directly in $\mathbf{X}$, and $\dot{m}$ in this context is readily defined by Equation (1c) (see end of Sec. II-A1), all that is left to define for these dynamics are $\ddot{\mathbf{r}}$ and $\ddot{\theta}$.

In Figure 1, note that only $\mathbf{u_1}$ influences the dynamics of the center of mass at any time t. In this way, $\mathbf{u}_1$ functions much like the thrust profile does in Equation (1b) for the point-mass model, i.e. $\mathbf{u}_1$ is the force on the center of mass. So, we may simply substitute this in for $\mathbf{U}$ in Equation (1b) to obtain $\ddot{\mathbf{r}}$ for the rectangle-rocket model:

$$\ddot{\mathbf{r}} = \mathbf{g} + \mathbf{u}_1/m$$

where

$$\mathbf{u}_1 = \begin{bmatrix} u_1 \cos\theta \\ u_1 \sin\theta \end{bmatrix}$$

For $\ddot{\theta}$, the angular acceleration, we may use the fact that the magnitude of the torque on the axis of rotation which goes through the center of mass of the rectangle, perpendicular to the page, T, may be expressed in two ways:

$$T = Fd = I\ddot{\theta}$$

where F is the magnitude of the force applied perpendicularly to the lever arm at a distance d from the axis of rotation, and I is the rotational moment of inertia [4].

For this model, $F = u_2$ and $d = h/2$. For the rotational moment of inertia, we will treat the rectangle in Figure 1 as a thin rectangular plate rotating about the axis perpendicular to the page, passing through the center of mass, which yields the following expression [5]:

$$I = \frac{m}{12}(h^2 + w^2)$$

Therefore:

$$\ddot{\theta} = \frac{u_2 h}{(\frac{m}{6}(h^2 + w^2))}$$

II-A4 Vertical Landing Problem Statement

Now we use the definitions of this section (II-A) to state the minimum-fuel vertical landing problem:

Problem 2:

$$\begin{aligned}
\min_{t_f, \mathbf{U}(\cdot)} \quad & -m(t_f) && \text{(2a)} \\
\text{s.t.} \quad & \ddot{\mathbf{r}}(t) = \mathbf{g} + \mathbf{u}_1(t)/m(t), && \text{(2b)} \\
& \ddot{\theta}(t) = \frac{u_2(t)h}{(\frac{m(t)}{6}(h^2 + w^2))}, && \text{(2c)} \\
& \dot{m}(t) = -\alpha||\mathbf{U}(t)||, && \text{(2d)} \\
& 0 \leq ||\mathbf{U}(t)|| \leq \rho, && \text{(2e)} \\
& u_1 \geq 0, && \text{(2f)} \\
& \mathbf{e}_1^T \mathbf{r}(t) \geq 0, && \text{(2g)} \\
& m_{\text{dry}} \leq m(t) \leq m_{\text{wet}}, && \text{(2h)} \\
& \mathbf{r}(0) = \mathbf{r}_0, && \text{(2i)} \\
& \dot{\mathbf{r}}(0) = \dot{\mathbf{r}}_0, && \text{(2j)} \\
& \theta(0) = \theta_0, && \text{(2k)} \\
& \dot{\theta}(0) = \dot{\theta}_0, && \text{(2l)} \\
& m(0) = m_{\text{wet}}, && \text{(2m)} \\
& \mathbf{r}(t_f) = \dot{\mathbf{r}}(t_f) = \mathbf{0}, && \text{(2n)} \\
& \theta(t_f) = \dot{\theta}(t_f) = 0 && \text{(2o)}
\end{aligned}$$

III Discretization

In this section we apply a discretization to Problem 2 which transforms this infinite-dimensional optimization problem into a finite-dimensional NLP. This is done by the method of Direct Multiple Shooting [6]. Note that this same method may be followed for Problem 1.

We begin by discretizing the time domain into N equal time intervals, the edges of which we will refer to as temporal nodes. For any time interval $[0, t_f]$, the temporal nodes are given as

$$t_k = k\Delta t, \quad k = 0, \ldots, N$$

where $N\Delta t = t_f$. Next we use this time grid to define a piece-wise constant control discretization:

$$\mathbf{U}(t) = \mathbf{q}_i, \quad t \in [t_i, t_{i+1}]$$

where $i = 0, \dots, N-1$. Using these discrete control inputs, we may discretize the state at each temporal node as follows:

$$\mathbf{X}(t_{i+1}) = \mathbf{s}_{i+1} = F_{\text{RK4}}(\mathbf{s}_i, \mathbf{q}_i)$$

where:

- $\mathbf{s}_0 = \mathbf{X}(0)$
- F_{RK4} returns the output of one step of an RK4 integrator using the dynamics function f defined in Sec. II-A3 and the time step Δt defined in this section

Now we collect variables into larger vectors to make the discretized NLP statement more succinct.

For the decision variables:

$$\hat{\mathbf{s}} = \begin{bmatrix} \mathbf{s}_0 \\ \vdots \\ \mathbf{s}_N \end{bmatrix} \qquad \hat{\mathbf{q}} = \begin{bmatrix} \mathbf{q}_0 \\ \vdots \\ \mathbf{q}_{N-1} \end{bmatrix}$$

For defining initial and terminal values:

$$\mathbf{X}_k = \begin{bmatrix} \mathbf{r}_k \\ \dot{\mathbf{r}}_k \\ \theta_k \\ \dot{\theta}_k \\ m_k \end{bmatrix}, \quad k = 0, \dots, N$$

We also introduce an indexing notation with which to reference individual values within each state or control vector. For brevity, we define it for a generic state vector $\mathbf{s}_k \in \hat{\mathbf{s}}$, but it works analogously for control vectors $\mathbf{q}_i \in \hat{\mathbf{q}}$:

$$\mathbf{s}_{k,m} = \text{the } m\text{th element of } \mathbf{s}_k, \text{ indexed from } 1$$

For example, the terminal mass m_N is $\mathbf{s}_{N,5}$, and the full position vector $\mathbf{r}$ at temporal node i is $\mathbf{s}_{i,1}$.

Now we are ready to state the discretized NLP for the minimum-fuel rocket vertical landing problem. Note that the dynamics constraints are implicit in F_{RK4}.

Problem 3:

$$\begin{aligned}
\min_{\hat{\mathbf{s}}, \hat{\mathbf{q}}} \quad & -\mathbf{s}_{N,5} && && \text{(4a)} \\
\text{s.t.} \quad & 0 \le ||\mathbf{q}_i|| \le \rho && i = 0, \dots, N-1, && \text{(4b)} \\
& \mathbf{q}_{i,1} \ge 0 && i = 0, \dots, N-1, && \text{(4c)} \\
& \mathbf{e}_1^T \mathbf{s}_{k,1} \ge 0 && k = 0, \dots, N, && \text{(4d)} \\
& m_{\text{dry}} \le \mathbf{s}_{k,5} \le m_{\text{wet}} && k = 0, \dots, N, && \text{(4e)} \\
& \mathbf{s}_0 = \mathbf{X}_0, && && \text{(4f)} \\
& \mathbf{s}_N = \mathbf{X}_N, && && \text{(4g)} \\
& \mathbf{s}_{i+1} = F_{\text{RK4}}(\mathbf{s}_i, \mathbf{q}_i) && i = 0, \dots, N-1 && \text{(4h)}
\end{aligned}$$

IV Results

In this section, we present and discuss the results of a converged solution to Problem 3 which was obtained by using Casadi's [1] Python interface to solve the discretized minimum-fuel rocket vertical landing problem with IPOPT [2]. We also discuss practical techniques which were used to help the solver converge.

Note that the first step of the discretization in Sec. (III) makes reference to t_f, the time it takes to land. This is not known prior to running the simulations, i.e. the duration of the problem is free. To surmount this hurdle in practice, we have implemented the method of time-scaling from [7]. For more information on this technique, we refer the reader there.

Here we define the parameters. We use the actual length and diameter of SpaceX's Falcon 9 first-stage for the length and width of our rectangle [8]: $h = 41.2$ meters, $w = 3.7$ meters. The number of control intervals is $N = 200$ intervals.

The rest of the parameters are as follows:

- $\mathbf{g} = [-3.7114, 0]^T \ \text{m}\,\text{s}^{-2}$
- $\alpha = 4.53 \times 10^{-4}\,\text{s}\,\text{m}^{-1}$
- $\rho = 13\,260\,\text{N}$
- $m_{\text{wet}} = 1905\,\text{kg}$
- $m_{\text{dry}} = 1505\,\text{kg}$
- $\mathbf{r}_0 = [1500, 2000] \ \text{m}$
- $\dot{\mathbf{r}}_0 = [0, 0] \ \text{m}\,\text{s}^{-1}$
- $\theta_0 = -\pi/2$ rad
- $\dot{\theta}_0 = 0$ rad/s
- $\theta_N = 0$ rad
- $\dot{\theta}_N = 0$ rad/s

To help the solver converge, we constrain θ to keep the rocket from pointing downward:

$$-\pi/2 \le \theta \le \pi/2$$

That is, we add the following constraint to Problem 3:

$$-\pi/2 \le \mathbf{s}_{k,3} \le \pi/2, \quad k = 0, \dots, N$$

Even with a linearly interpolated initial guess between all initial and final values, this NLP would exceed the maximum number of iterations of 600, and the plot of the mass would show an increase, even though such behavior should be impossible due to Equation (2d). So, we also add the following constraint to Problem 3 to help force the mass to decrease:

$$\mathbf{s}_{i,5} > \mathbf{s}_{i+1,5}, \quad i = 0, \dots, N-1$$

With this constraint added, the NLP converges. Values from the solution are given in Figure 2.

We can see from the mass plot that no thrust is being applied between roughly 10 and 30 seconds, and that the slopes during the other time intervals are roughly equal, suggesting a *max-min-max* thrust profile which is characteristic of minimum-fuel trajectories [3]. Indeed, the thrust plot for just the main thruster (whose magnitude of thrust is u_1) displays this behavior, only deviating from the max thrust on the nonzero intervals so that the second thruster (magnitude of u_2) may enact adjustments to the rocket angle (as shown in Figure 3).

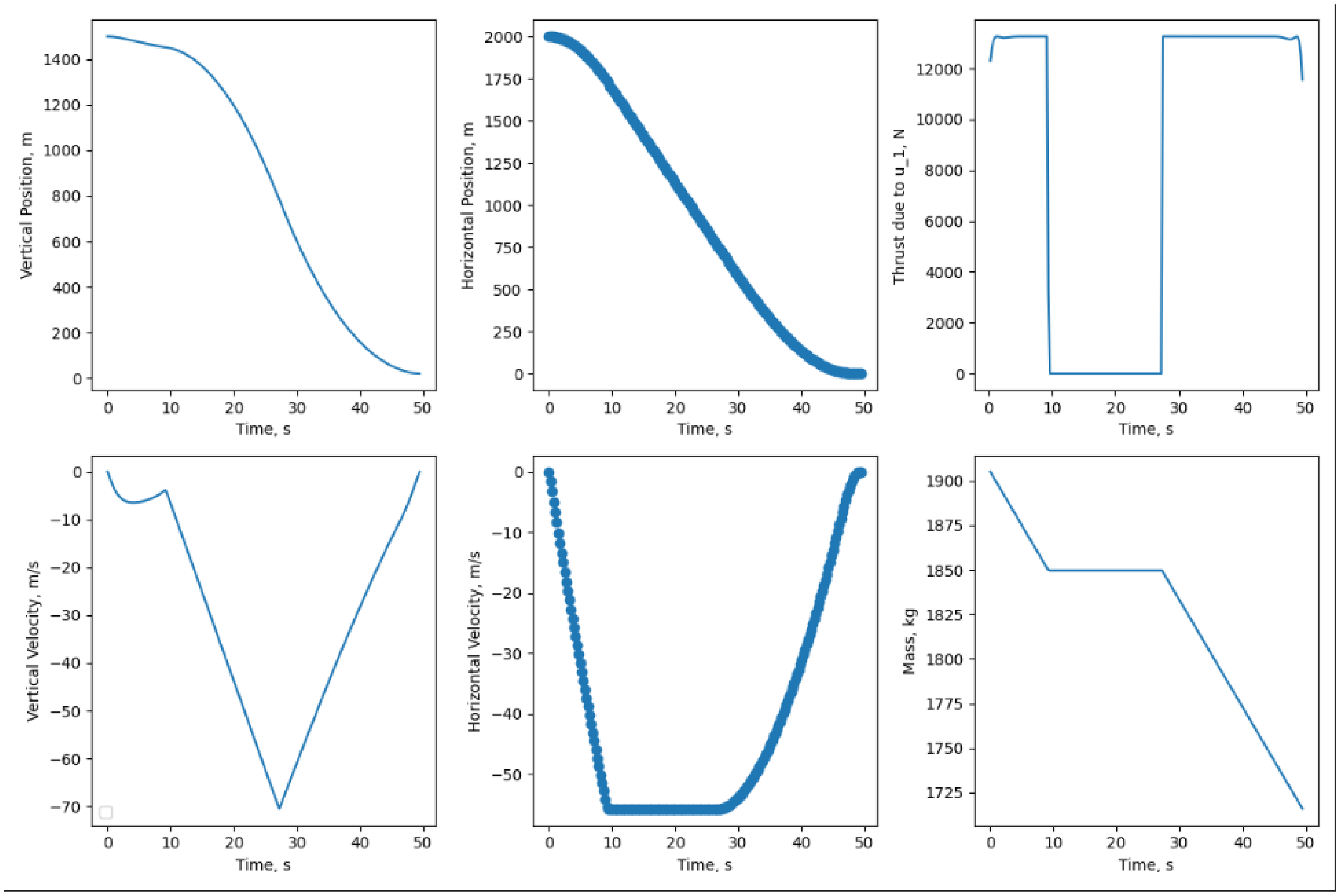

Fig. 2: Position, velocity, mass, and main engine data.

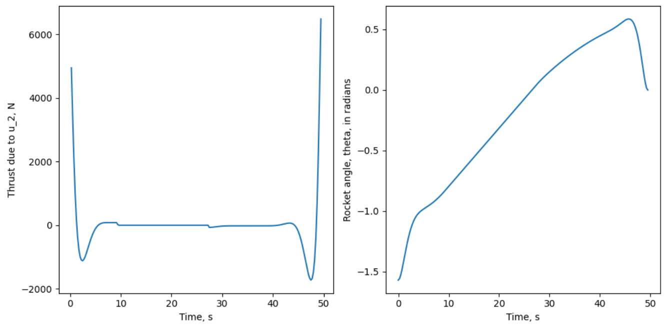

Fig. 3: Side engine and rocket angle data.

V Conclusion

In this report, I have approached the minimum-fuel rocket vertical landing problem through the lens of optimal control. This was done by adapting a point-mass model which was developed to find minimum-fuel trajectories in [3]. The adaptation succeeded in modeling the vertical landing problem in 2 dimensions by viewing the rocket as a rectangle with two sets of thrusters. Then, this continuous time optimal control problem was discretized using Direct Multiple Shooting [6]. The NLP model was then implemented in Casadi [1] and the parameters of the model were given the values in Sec. IV. This NLP was then solved using IPOPT [2], and some aspects of this solution were discussed in Sec. IV.

Getting solutions to converge for real-world situations proved to be very difficult with this model, and represents an avenue for future work. A real-world situation would be, for example, modeling a so-called ”boostback burn” of the Falcon 9. This is when, after separating from the second stage, the first stage turns around and lands back near the launchpad. Since wind-resistance was absent from this model and accounts for much of the deceleration of such a situation in the real-world, it was difficult to find parameters in this model which would render such a trajectory feasible.

Acknowledgments

I would like to express my sincere gratitude to the following individuals for their invaluable help and support during the completion of this project:

- Jakob Harzer for providing guidance and feedback throughout the project.
- Akash John Subash and Sourish Pramanik for suggesting tips to help the NLP's converge.
- Armin Nurkanović and Prof. Diehl for providing an engaging course in Numerical Optimal Control.

I am deeply grateful for their contributions, which have significantly enriched the outcome of this work.

References

[1] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl. CasADi – A software framework for nonlinear optimization and optimal control. Mathematical Programming Computation, 11(1):1– 36, 2019.

[2] A. Wachter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical Programming, 106(1):25–57, 2006.

[3] Açıkmeşe, Behçet & Polen, S.. (2007). Convex programming approach to powered descent for mars landing. J. Guidance Control Dyn.. 44. 310-322.

[4] "Torque." Wikipedia, 22 July 2023, en.wikipedia.org/wiki/Torque.

[5] "Moment of Inertia of Rectangular Plate." AmesWeb, amesweb.info/inertia/moment-of-inertia-of-rectangular-plate.aspx. Accessed 24 July 2023.

[6] Gros, Sebastian & Diehl, Moritz. Direct Multiple Shooting. *Numerical Optimal Control (Draft)*, 235-238. April 27, 2022.

[7] Gros, Sebastian & Diehl, Moritz. Problem reformulation (Time-scaling). *Numerical Optimal Control (Draft)*, 162-163. April 27, 2022.

[8] "Falcon 9," SpaceFlight Insider. https://www.spaceflightinsider.com/hangar/falcon-9/