Successive Convexification for the Rocket-Powered Landing Optimal Control Problem

Jake T. Sonandres, Kyle A. Sonandres
Massachusetts Institude of Technology, Cambridge, MA, 02139, USA

I. Project Motivation and Overview

In a recent test flight, SpaceX successfully demonstrated the ability to catch a 23-story tall Super Heavy rocket booster with the launch tower's large hydraulic actuating arms. The historic milestone was a critical step in achieving rapid reusability of the Starship upper stage, which can potentially enable efficient transportation of cargo into orbit, human moon landings, and eventually the first mission to Mars. To accomplish this feat, the booster must reenter the atmosphere and perform a sequence of precise commands to return to its original launch position, which requires extreme accuracy while minimizing relevant objectives such as fuel consumption or time of flight. Additionally, the control sequence must be executed autonomously, which poses a pertinent and complex problem for determining the optimal thrust commands of the gimbaled rocket engine. Ultimately, this can be formulated as a highly nonlinear and constrained optimal control problem in which decision variables include the state and control inputs.

Ideally, we wish to implement an algorithm that is able to solve this problem in real-time to enable onboard planning. This requires speed, global convergence guarantees, and the ability to self-initialize a suitable guess. Simply encoding the problem and attempting to solve with a standard nonlinear optimizer (such as IPOPT) is insufficient, as speed is highly variable, no convergence guarantees exist, and an expert initial guess is often necessary to find a solution. By deploying a specific sequential convex programming approach known as successive convexification, we can find solutions to the original problem quickly and reliably without a good initial guess. This computational project aims to tackle this problem by presenting and implementing a state-of-the-art optimization framework, which has broader applicability to the overarching class of non-convex optimal control problems with nonlinear system dynamics. In this project we implement the sequential convexification algorithm [1] from scratch using Julia and Gurobi and present the resulting optimal trajectory solutions to the 6-DoF minimum-time rocket-landing problem. All code can be found here.

II. Problem Overview

Here, we present the complete formulation of the 6-DoF minimum-time powered descent rocket landing problem. We largely follow the formulation from [2]. In this section, we outline the continuous-time and non-convex form of the problem, which will be modified in later parts. The rocket is modeled as a rigid body with three translational degrees of freedom that determine the position in three-dimensional space and three rotational degrees of freedom that define the rocket's orientation. We model a single gimbaled rocket engine which provides the commanded thrust. The overall mass depletion of the system is taken to be proportional to the commanded thrust magnitude resulting from subsequent fuel loss. We assume that the system's center of mass remains stationary and that the rocket's rotational inertia, represented by an inertia tensor, does not change. Ultimately, the problem will be subject to the nonlinear and nonconvex system dynamics and state constraints that are presented in this section.

A. Dynamics

The flight dynamics and kinematics of the 6-DoF rocket are fundamental to this project because they form the mathematical basis for accurately modeling and predicting the rocket's motion. The dynamics presented

in this section will ultimately form the constraints that ensure the system evolves in a physically realistic way. Central to this framework is the concept inertial and body-fixed reference frames, which will be used throughout the formulation. We define the following two frames:

1) $\mathcal{F}_{\mathcal{I}}$: An inertial frame fixed at the landing point, which will be taken to be the origin in this work. The fixed frame provides a reference for describing motion relative to an unchanging point in space.
2) $\mathcal{F}_{\mathcal{B}}$: A body-fixed frame centered at the rocket's center of mass. The body-fixed frame simplifies dynamics calculations by considering forces and moments in the body's local coordinate system.

Thus, we introduce a state vector $\boldsymbol{x}(t) \in \mathbb{R}^{14}$ to characterize the problem. The state vector consists of mass, $m(t) \in \mathbb{R}_+$, inertial position, $\mathbf{r}_{\mathcal{I}}(t) \in \mathbb{R}^3$, inertial velocity, $\mathbf{v}_{\mathcal{I}}(t) \in \mathbb{R}^3$, the attitude quaternion, $\mathbf{q}_{\mathcal{B}/\mathcal{I}}(t) \in \mathbb{R}^3$, and the angular velocity $\boldsymbol{\omega}(t) \in \mathbb{R}^3$. The full state vector is

$$\boldsymbol{x}(t) = \begin{bmatrix} m(t) & \mathbf{r}_{\mathcal{I}}^T(t) & \mathbf{v}_{\mathcal{I}}^T(t) & \mathbf{q}_{\mathcal{B}/\mathcal{I}}^T(t) & \boldsymbol{\omega}^T(t) \end{bmatrix}^T \tag{1}$$

We express the commanded thrust vector in $\mathcal{F}_{\mathcal{B}}$ coordinates as $\mathbf{T}_{\mathcal{B}} \in \mathbb{R}^3$. So, the control vector is

$$\boldsymbol{u}(t) = \mathbf{T}_{\mathcal{B}} \tag{2}$$

We also introduce the following constants: vacuum-specific-impulse I_{sp}, the gravitational constant g_0, and the constant gravity vector $\mathbf{g}_{\mathcal{I}}(t) \in \mathbb{R}^3$. The dependence on t remains unchanged throughout this work; therefore, we omit the explicit notation, with the understanding that all time-dependence for state and control variables is implied unless stated as constant. We next present the time derivatives of the state vector. The mass depletion dynamics are given as

$$\dot{m} = -\frac{1}{I_{\text{sp}} g_0} \|\mathbf{T}_{\mathcal{B}}\|_2. \tag{3}$$

The translational dynamics are defined as

$$\dot{\mathbf{r}_{\mathcal{I}}} = \mathbf{v}_{\mathcal{I}}, \tag{4}$$

$$\dot{\mathbf{v}_{\mathcal{I}}} = \frac{1}{m} \mathcal{F}_{\mathcal{I}} + g_{\mathcal{I}}. \tag{5}$$

The attitude of body frame $\mathcal{F}_{\mathcal{B}}$ with respect to the inertial frame $\mathcal{F}_{\mathcal{I}}$ is parameterized via the quaternion $\mathbf{q}_{\mathcal{B}/\mathcal{I}} \in \mathcal{S}^3$, where

$$\mathbf{q}_{\mathcal{B}/\mathcal{I}} = \begin{bmatrix} q_0 \; q_1 \; q_2 \; q_3 \end{bmatrix}. \tag{6}$$

Note that we follow the leading scalar convention, where q_0 is the scalar part of the quaternion. We must next introduce mathematical machinery related to quaternions and rotations in order to define their dynamics. The rotation matrix that encodes the rotation between $\mathcal{F}_{\mathcal{I}}$ and $\mathcal{F}_{\mathcal{B}}$, $C_{\mathcal{B}/\mathcal{I}}$, can be written as a function of the components of $\mathbf{q}_{\mathcal{B}/\mathcal{I}}$:

$$C_{\mathcal{B}/\mathcal{I}} = \begin{bmatrix} 1 - 2\left(q_2^2 + q_3^2\right) & 2\left(q_1 q_2 + q_0 q_3\right) & 2\left(q_1 q_3 - q_0 q_2\right) \\ 2\left(q_1 q_2 - q_0 q_3\right) & 1 - 2\left(q_1^2 + q_3^2\right) & 2\left(q_2 q_3 + q_0 q_1\right) \\ 2\left(q_1 q_3 + q_0 q_2\right) & 2\left(q_2 q_3 - q_0 q_1\right) & 1 - 2\left(q_1^2 + q_2^2\right) \end{bmatrix}. \tag{7}$$

Next we can define the skew symmetric matrices $[\boldsymbol{\xi}\times]$ and $\Omega(\boldsymbol{\xi})$ as

$$[\boldsymbol{\xi}\times] \triangleq \begin{bmatrix} 0 & -\xi_z & \xi_y \\ \xi_z & 0 & -\xi_x \\ -\xi_y & \xi_x & 0 \end{bmatrix} \quad \Omega(\boldsymbol{\xi}) \triangleq \begin{bmatrix} 0 & -\xi_x & -\xi_y & -\xi_z \\ \xi_x & 0 & \xi_z & -\xi_y \\ \xi_y & -\xi_z & 0 & \xi_x \\ \xi_z & \xi_y & -\xi_x & 0 \end{bmatrix}$$

With these operations defined, the quaternion dynamics can be written as

$$\dot{\mathbf{q}}_{\mathcal{B}/\mathcal{I}} = \frac{1}{2}\Omega(\omega_{\mathcal{B}})\mathbf{q}_{\mathcal{B}/\mathcal{I}}. \tag{8}$$

The total torque acting on the vehicle is $\mathbf{M}_{\mathcal{B}}$, which will be defined shortly. The rotational dynamics are

$$\dot{\omega} = {\mathcal{J}_{\mathcal{B}}}^{-1}\left(\mathbf{M}_{\mathcal{B}} - [\omega_{\mathcal{B}}\times]\mathcal{J}_{\mathcal{B}}\omega_{\mathcal{B}}\right), \tag{9}$$

where ${\mathcal{J}_{\mathcal{B}}}^{-1}$ is the inertia tensor of the vehicle. The last piece is to kinematically relate the force, thrust, and moment vectors used in the above dynamics calculations. The force is related to the thrust vector using the inverse of the rotation matrix

$$\mathcal{F}_{\mathcal{I}} = C_{\mathcal{I}/\mathcal{B}}\mathbf{T}_{\mathcal{B}}, \tag{10}$$

and the torque is computed using the constant moment arm length ($r_{T,\mathcal{B}}$) as

$$\mathbf{M}_{\mathcal{B}} = [r_{T,\mathcal{B}}\times]\mathbf{T}_{\mathcal{B}} \tag{11}$$

In summary, the full nonlinear problem dynamics are

$$\dot{\boldsymbol{x}} = \begin{bmatrix}\dot{m} & \dot{\mathbf{r}}_{\mathcal{I}} & \dot{\mathbf{v}}_{\mathcal{I}} & \dot{\mathbf{q}}_{\mathcal{B}/\mathcal{I}} & \dot{\omega}\end{bmatrix}^T = f(\boldsymbol{x},\boldsymbol{u}). \tag{12}$$

B. State and Control Constraints

Since the rocket engine has physical limitations such as actuation limits and thrust magnitude bounds, we must define constraints on the control vector $\mathbf{u}$. Further, we must constrain certain aspects of the state vector to restrict the solver from returning trajectories with undesirable or particularly risky characteristics.

Control constraints. An upper bound is placed on the magnitude of the commanded thrust vector, which is an inherent physical limitation of any engine. Further, switching the engine off, potentially multiple times during a short period, creates great structural demand on the rocket. We therefore impose a lower bound on the magnitude of the commanded thrust vector. Further, an upper bound is placed on the engine's gimbal angle to enforce an actuation limit. More formally, the thrust magnitude constraint is

$$0 < T_{\min} \leq \|\mathbf{T}_{\mathcal{B}}\|_2 \leq T_{\max}, \tag{13}$$

and the gimbal angle constraint is

$$\cos\delta_{\max}\|\mathbf{T}_{\mathcal{B}}\|_2 \leq \mathbf{e}_3\mathbf{T}_{\mathcal{B}} \quad \text{for } \delta_{\max} \in (0, 90). \tag{14}$$

Note that the thrust upper bound and gimbal angle are naturally convex constraints, while the thrust lower bound is not.

State constraints. First, the vehicle mass must remain above the dry mass, m_{dry}. This can be represented via the convex constraint

$$m \geq m_{\text{dry}}. \tag{15}$$

The angular rotation rate is limited to $\omega_{\max}$ prevent excessive rotational velocities using the convex constraint

$$\|\omega_{\mathcal{B}}\|_2 \leq \omega_{\max}. \tag{16}$$

Next, the vehicle's trajectory is restricted to lie within an upward-facing glide-slope cone. The angle the cone makes with the horizontal is $\gamma_{\text{gs}} \in [0, 90)$. The convex constraint is

$$\mathbf{e_3}\mathbf{r}_{\mathcal{I}} \geq \tan\gamma_{\text{gs}}\|H_{12}^T\mathbf{r}\|_2, \tag{17}$$

where

$$H_{12} = [\mathbf{e_1}\ \mathbf{e_2}]\,. \tag{18}$$

The tilt-angle of the vehicle is defined as the angle between the Z-axes of $\mathcal{F}_\mathcal{I}$ and $\mathcal{F}_\mathcal{B}$. This can be imposed via the third and fourth members of the quaternion in a convex constraint as

$$\cos\theta_{\max} \leq 1 - 2(q_2^2 + q_3^2) \tag{19}$$

Boundary Conditions. This problem formulation does not require specific boundary conditions, thus this is largely an individual design choice. We choose to constrain the entire initial state vector and to constrain every member of the final state vector besides mass.

C. Nonconvex Formulation

The full nonconvex continuous optimization problem can now be defined simply. We wish to minimize the final time subject to the above constraints, system dynamics, and boundary conditions.

$$\begin{aligned}
\underset{x,u,t_f}{\text{Minimize}} \quad & t_f \\
\text{Subject to} \quad & \dot{x}(t) = f(x(t), u(t)), \\
& g_i(x(t), u(t)) \leq 0, \quad i \in \mathcal{I}_{\text{nonconvex}}, \\
& h_j(x(t), u(t)) \leq 0, \quad j \in \mathcal{I}_{\text{convex}}, \\
& \mathbf{x}(0) = x_0, \quad \mathbf{x}(t_f) = x_f,
\end{aligned}$$

III. Convex Formulation

The problem formulation in the previous section is the problem we intend to solve. However, it is non-convex, free-final-time, and difficult to solve. In this section, we reformulate the problem into a convex fixed-final time subproblem that is much faster to solve. We will iteratively solve these subproblems until convergence is achieved and we have a solution to the original non-convex problem.

A. Dynamics and Constraint Linearization

The full 6-DoF rocket dynamics impose a highly nonlinear constraint on the original problem. Further, the problem is continuous and therefore of infinite dimension. To convexify the dynamics constraint and reduce the problem to finite dimension, we *linearize* and *discretize* the dynamics. This process is math heavy and we attempt here to succinctly explain the process, so we encourage the reader to visit the more detailed explanation in Ref. [2]. We begin by nondimentionalizing time, which will cast the free-final-time problem into a fixed-final-time problem. Let the normalized trajectory time be $\tau \in [0, 1]$. We define the time dilation between t and τ as

$$\sigma = \left(\frac{d\tau}{dt}\right)^{-1}. \tag{20}$$

The linearized system dynamics with respect to normalized time can be defined as

$$\mathbf{x}'(\tau) = A(\tau)\mathbf{x}(\tau) + B(\tau)\mathbf{u}(\tau) + S(\tau)\sigma + \mathbf{z}(\tau). \tag{21}$$

Define a reference state and control trajectory as $\hat{\mathbf{x}}(\tau), \hat{\mathbf{u}}(\tau)$ and dilation coefficient as $\hat{\sigma}$. The linearization matrices above are defined as the Jacobians w.r.t. the state/ control vectors evaluated along the current reference trajectories:

$$A(\tau) \triangleq \hat{\sigma} \cdot \frac{\partial}{\partial \mathbf{x}} f(\mathbf{x}, \mathbf{u}) \Big|_{\hat{\mathbf{x}}(\tau), \hat{\mathbf{u}}(\tau)} \tag{22}$$

$$B(\tau) \triangleq \hat{\sigma} \cdot \frac{\partial}{\partial \mathbf{u}} f(\mathbf{x}, \mathbf{u}) \Big|_{\hat{\mathbf{x}}(\tau), \hat{\mathbf{u}}(\tau)} \tag{23}$$

$$S(\tau) \triangleq f(\hat{\mathbf{x}}(\tau), \hat{\mathbf{u}}(\tau)) \tag{24}$$

$$\mathbf{z}(\tau) \triangleq -A(\tau)\hat{\mathbf{x}}(\tau) - B(\tau)\hat{\mathbf{u}}(\tau) \tag{25}$$

The linearization is now complete, and we proceed to reducing the problem into a finite-dimensional discrete problem. Further, we need to know how the state evolves from time k to time $k+1$ in order to encode the constraint in the optimization problem. We discretize the trajectory into K evenly spaced discretization points and assume a first-order-hold on the control over each time step. The state evolution is given as

$$\mathbf{x}_{k+1} = \bar{A}_k \mathbf{x}_k + \bar{B}_k \mathbf{u}_k + \bar{C}_k \mathbf{u}_{k+1} + \bar{S}_k \sigma + \bar{\mathbf{z}}_k, \tag{26}$$

where the discretization matrices are functions of the state transition matrix $\Phi_A(\tau_{k+1}, \tau_k)$, which describes the zero-input evolution from x_k to x_{k+1}

$$\bar{A}_k \triangleq \Phi_A(\tau_{k+1}, \tau_k), \tag{27}$$

$$\bar{B}_k \triangleq \int_{\tau_k}^{\tau_{k+1}} \Phi_A(\tau_{k+1}, \xi) B(\xi) \alpha_k(\xi) d\xi, \tag{28}$$

$$\bar{C}_k \triangleq \int_{\tau_k}^{\tau_{k+1}} \Phi_A(\tau_{k+1}, \xi) B(\xi) \beta_k(\xi) d\xi, \tag{29}$$

$$\bar{S}_k \triangleq \int_{\tau_k}^{\tau_{k+1}} \Phi_A(\tau_{k+1}, \xi) S(\xi) d\xi, \tag{30}$$

$$\bar{\mathbf{z}}_k \triangleq \int_{\tau_k}^{\tau_{k+1}} \Phi_A(\tau_{k+1}, \xi) \mathbf{z}(\xi) d\xi, \tag{31}$$

where $\alpha_k(\tau) = (\tau_{k+1} - \tau)/(\tau_{k+1} - \tau_k)$ and $\beta_k(\tau) = (\tau - \tau_k)/(\tau_{k+1} - \tau_k)$. Lastly, the state transition matrix is computed by solving the differential equation

$$\frac{d}{d\tau}\Phi_A(\tau, \tau_k) = A(\tau)\Phi_A(\tau, \tau_k), \quad \Phi_A(\tau_k, \tau_k) = I. \tag{32}$$

B. Artificial Infeasibility and Unboundedness

The linearization and discretization process introduces two well-known challenges that must be compensated for in the successive convexification framework. First, any linearization loses accuracy as one moves away from the linearization point. Further, the linearization can render the solution unbounded below (convex cost can be driven to negative infinity). This is known as *artificial unboundedness*. To account for this, we require that the solution of the convex subproblem stay sufficiently close to the reference by enforcing a hard *trust region* constraint. That is, we define $\delta x = x - \hat{x},\ \delta u = u - \hat{u},\ \delta\sigma = \sigma - \hat{\sigma}$. In the subproblem, we require that the sum of 2-norms of these quantities remain within a specified trust region radius η

$$\|\delta x\|_2 + \|\delta u\|_2 + \|\delta\sigma\|_2 \leq \eta \tag{33}$$

Next, the linearization of the constraints may lead to an empty solution set, even if the original problem is feasible. This is known as *artificial infeasibility*. To remedy this, we add unconstrained, but penalized, slack

variables for each linearized constraint known as *virtual control* terms. The virtual control term associated with the dynamics is ν, and the virtual control term associated with the control lower bound is s. We augment the objective function with the norms of these variables and a large weight in order to drive the terms to zero

$$w_\nu \|\nu\|_1 + w_s \|s\|_1. \tag{34}$$

C. Complete Convex Subproblem Formulation

Variables and sets:

- i: number of states $i = 1, \cdots, 14$
- j: number of controls $i = 1, \cdots, 3$
- k: number of discretized timesteps $k = 1, \cdots, 30$
- $x_{i,k}$: state variable i at timestep k for $i = 1, \cdots, 14$, $k = 1, \cdots, 30$
- $u_{j,k}$: control variable j at timestep k for $j = 1, \cdots, 3$, $k = 1, \cdots, 30$
- $\nu_{i,k}$: virtual control dynamic relaxation variable $i = 1, \cdots, 14$, $k = 1, \cdots, 30$
- σ: nondimentionalized time
- s_k: virtual control thrust lower bound variable $k = 1, \cdots, 30$

Objective. We wish to minimize the sum of the final time and the norm of virtual control terms

$$\min w_\sigma \sigma + w_\nu \|\nu\|_1 + w_s \|s\|_1$$

Constraints. The evolution of the state variable over time must follow the linearized, discretized state dynamics. This convexifies the dynamics constraint

$$x_{i,k+1} = \bar{A}_k x_{i,k} + \bar{B}_k u_{j,k} + \bar{C}_k u_{j,k+1} + \bar{S}_k \sigma + \bar{z}_k + \nu_{i,k}. \tag{35}$$

The other nonconvex constraint from the original problem is the thrust lower bound. This can be linearized and implemented as

$$T_{\min} - \frac{\hat{u}_{1:3,k}^T}{\|\hat{u}_{1:3,k}\|_2} u_{1:3,k} \leq s_k \quad \forall k. \tag{36}$$

The rest of the constraints were originally convex, so they directly follow from section II.B. They are rewritten here to use the implementation notation. First, the mass must stay above the dry mass

$$x_{1,k} \geq m_{\text{dry}} \quad \forall k. \tag{37}$$

Next, the vehicle must remain in the glide-slope cone

$$\tan \gamma_{\text{gs}} \|x_{2:3,k}\|_2 \leq x_{4,k} \quad \forall k. \tag{38}$$

The vehicle must also avoid excessive tilt angles

$$1 - 2\|x_{10:11,k}\|_2^2 \geq \cos \theta_{\max} \quad \forall k. \tag{39}$$

Next, we impose an angular velocity upper bound as

$$\|x_{12:14,k}\|_2 \leq \omega_{\max} \quad \forall k. \tag{40}$$

The thrust upper bound is given as

$$\|u_{1:3,k}\|_2 \leq T_{\max} \quad \forall k, \tag{41}$$

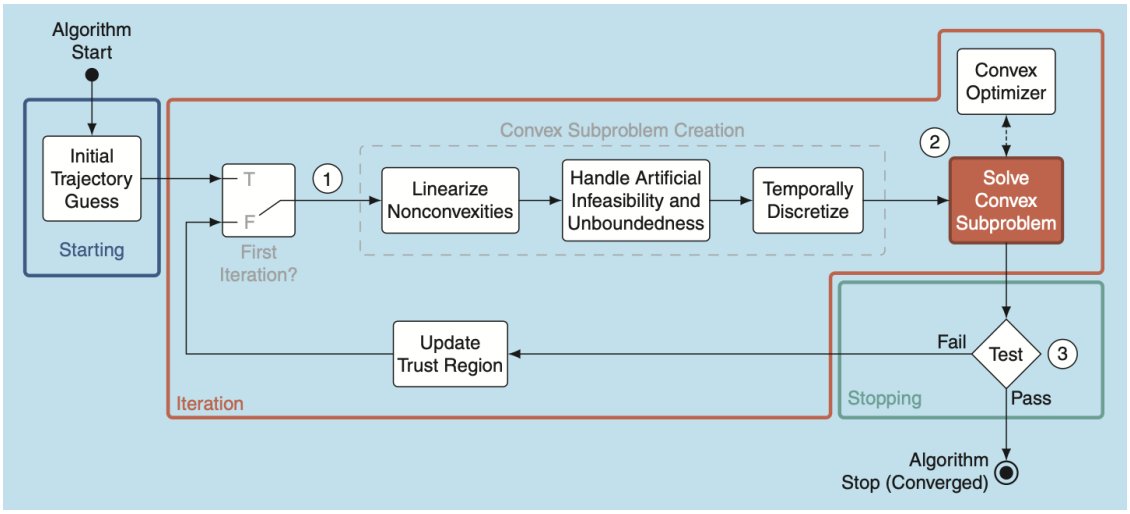

Fig. 1 **Typical Sequential Convex Programming framework [3]. The algorithm begins with an initial trajectory guess. After the first iteration, the output from the previous iteration is used as the current trajectory guess. The global problem is converted to a convex, quick to solve discrete subproblem. The convex subproblem is solved and the stopping criteria is checked. If convergence is not achieved, the trust regions are updated and the process repeats.**

and the gimbal angle constraint is

$$\|u_{1:2,k}\|_2 \leq 2\tan\delta_{\max} u_{3,k} \quad \forall k. \tag{42}$$

Lastly, we impose the boundary conditions as before, constraining all but the final mass

$$x_{i,0} = x_{i,0}^0 \quad \forall i, \tag{43}$$

$$x_{i,k} = x_{i,k}^0 \quad \forall i > 1. \tag{44}$$

The convex subproblem is now completely defined. We will iteratively form and solve these subproblems until we have a solution to the original nonconvex constrainted optimization problem.

IV. Successive Convexification

This section outlines the sequential convex programming (SCP) approach used to iteratively solve the non-convex optimization problem. The implementation used in this project largely follows the generic SCP framework presented in Fig. 1. We begin with initialization, in which an initial guess is generated by linearly interpolating between the initial and final states. Since final mass is unconstrained, we interpolate from the initial mass to the dry mass to generate the guess. The control guess is initialized as a constant thrust vector with magnitude $(T_{\max} + T_{\min})/2$ along the Z-axis. With an initial reference trajectory, we move into convex subproblem creation, where the problem is linearized and discretized according to the procedure defined in section III. The subproblem is next solved, yielding the optimal trajectory. The convergence criteria is checked, which is a measure of how large the virtual control terms are as well as how much the trajectory solution has deviated from the previous iteration. If convergence is not detected, the reference trajectory is updated as the solution found in the current iteration. The trust regions are updated and we move on to the next iteration. This is repeated until convergence or until the maximum number of iterations is reached.

V. Results

In this section, we present results of our Julia implementation of the successive convexification algorithm for the 6-DoF rocket landing problem. We follow a framework similar to that which was established throughout

the term, in which we frequently used Julia's JuMP and Gurobi to solve complicated optimization problems. The main decision variables were the state vector and control input outlined in section II.A. To numerically compute the Jacobians of the dynamics from equations (22) and (23), we implemented an automatic differentiation procedure with Julia's ForwardDiff library. The ODE in equation (32) was numerically solved using the DifferentialEquations package. We leveraged Gurobi's well-known and highly-optimized nonlinear convex solvers by passing the nonlinear constraints into the model as second-order cone constraints (SOCCs) after executing the convexification process given by section III. We added the necessary quantities to a JuMP model which was ultimately solved with the Gurobi optimizer for each convex subproblem.

We will explore two representative cases in this section. The first aims to present a preliminary validation for the performance of our algorithm. Though both cases will be performed in three-dimensional space, the first case will attempt to restrict the rocket's trajectory to a single plane. Additionally, the initial conditions will be such that we can reasonably predict the form of the optimal trajectory. This section will mainly focus on the second case, as it will introduce complexities via non-trivial initial conditions and more challenging system constraints.

A. Case 1

This simulation attempts to restrict the rocket to motion only in the $x - z$ plane through the careful selection of initial conditions. This simulation was created for the purpose of validating the implementation of the algorithm since the resulting trajectory is expected to be relatively simple to interpret, analyze, and sanity-check. Thus, the rocket is given an initial position on the x-axis with an arbitrary z position: $\mathbf{r}_0 = (200, 0, 500)\,[\mathrm{m}]$. The rocket is also instantiated with an initial linear velocity in the plane and towards the origin (landing site): $\mathbf{v}_0 = (-70, 0, -100)\,[\mathrm{m/s}]$. A small initial angular velocity is provided such that it rotates clockwise in the plane of interest. We expect the optimized trajectory to exhibit a smooth descent towards the origin with no out-of-plane motion or control inputs. Of course, we also expect the rocket to adhere to all constraints. We give the rocket freedom to adopt large tilt angles, $\theta_{max} = 45^\circ$ and physically reasonable gimbal angles, $\delta_{max} = 15^\circ$.

The results of the successive convexification algorithm for the planar case are shown in figure 2. As expected, we observe smoothness in the dynamics, which can be seen from the position and velocity evolution. The rocket performs an initial maneuver to align itself with the descent direction. It then applies the maximum thrust while aligned in this direction for an efficient reduction in its kinetic energy as it tracks toward the origin and its required final velocity state. Interestingly, the rocket saturates both the tilt angle constraint and the gimbal angle constraint for different portions of the trajectory, which is consistent with the intuition we built in the linear optimization module for optimal solutions, though it is not always necessary in nonlinear problems. Building on this preliminary validation, we proceed to examine the more interesting non-planar case.

B. Case 2

This case attempts to push the limits of trajectory control and design by presenting non-trivial out-of-plane initialization in position, translational velocity, and angular velocity. As such, this case is more representative of a real-world scenario. Initially, the vehicle is positioned at $\mathbf{r}_0 = (-100, 150, 500)\,[\mathrm{m}]$ with a velocity $\mathbf{v}_0 = (-50, 55, -100)\,[\mathrm{m/s}]$. In contrast to the planar case, we decrease the maximum tilt angle to $\theta_{max} = 35^\circ$ and increase the maximum gimbal angle to $\delta_{max} = 25^\circ$ with the intention of giving the system more freedom to exhibit extreme control inputs while limiting the vehicle to a more upright position in general. This poses a unique trade-off between allowing for greater control authority to handle the non-trivial initial conditions and maintaining overall stability and adherence to stricter attitude constraints, which challenges the algorithm to find a feasible and efficient solution.

The resulting trajectory displays many interesting characteristics that reveal key insights into the design of

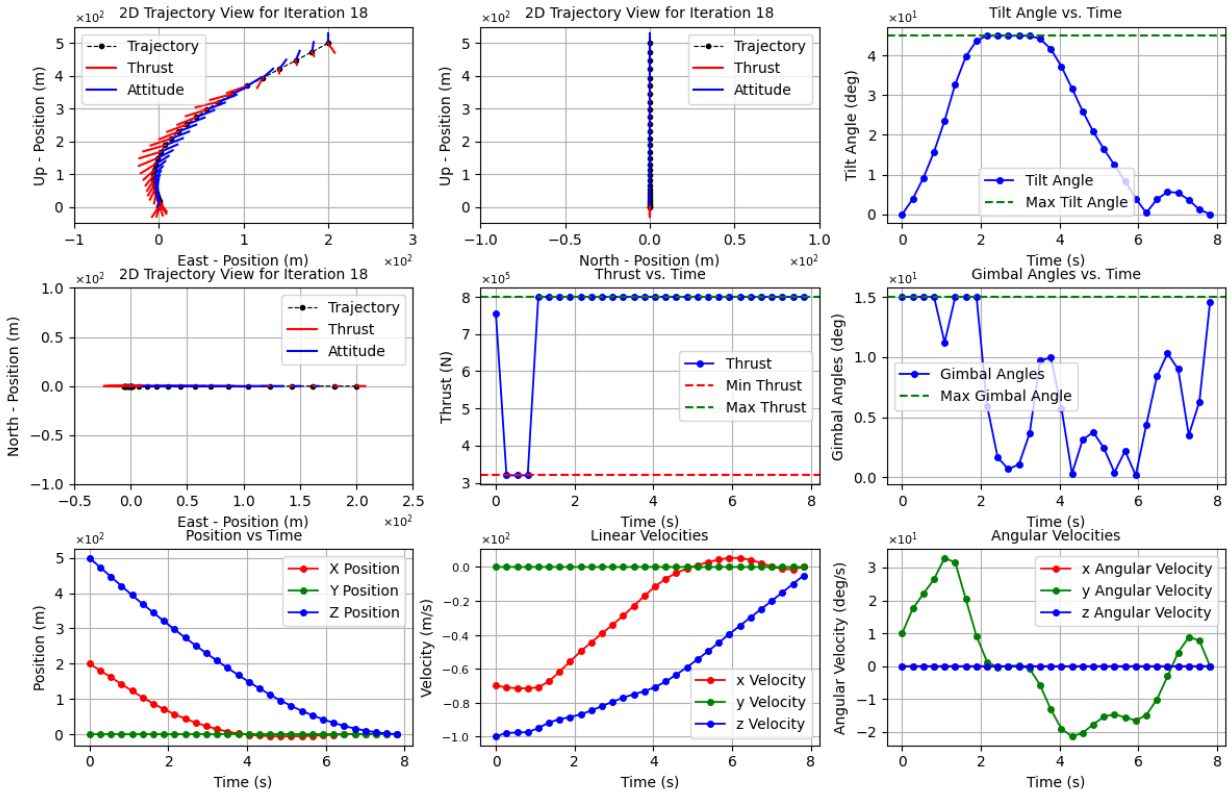

Fig. 2 **Comprehensive visualization of the 6-DoF rocket trajectory optimization results for the planar example. The top row presents 2D trajectory views along the East-Up and North-Up planes, with overlaid thrust and attitude vectors, and a plot of the tilt angle over time. The middle row includes the North-East 2D trajectory, thrust magnitude over time, and gimbal angles over time. The bottom row illustrates the time evolution of position, linear velocities, and angular velocities.**

an optimal trajectory that are not immediately obvious. The minimum time of the determined trajectory is approximately 11 seconds, and it is achieved by two distinct maneuvers; first, the rocket provides an initial "kick" by applying the maximum allowable thrust in conjunction with the maximum gimbal angle, which leads to an increase in angular velocity such that the rocket can align itself in a near-linear descent direction moving towards the landing site. This allows the rocket to slow itself down while making adequate progress toward the final position, which can be seen as a steady-state phase when the rocket is at its maximum tilt angle, maximum thrust, and zero gimbal angle. When the rocket enters the vicinity of the origin, it performs its second maneuver, where it again applies the maximum gimbal angle to induce a rotation into the final orientation. Perhaps the main takeaway is the bang-bang nature of the control profile, which is seen in the prevalence of applying *either* the minimum *or* maximum value in both the commanded thrust and gimbal angle. This finding is consistent with many other state-of-the-art control algorithms across various applications, including those who have applied a similar optimization framework to orbital transfers [4], quad-rotor motion problems with rigid obstacles [5], and the Mars fuel-optimal powered descent landing problem [6].

VI. Conclusion and Future Work

This work demonstrates how we can leverage advanced optimization algorithms like the successive convexification framework to solve optimal control problems that are nonlinear and nonconvex in general. We

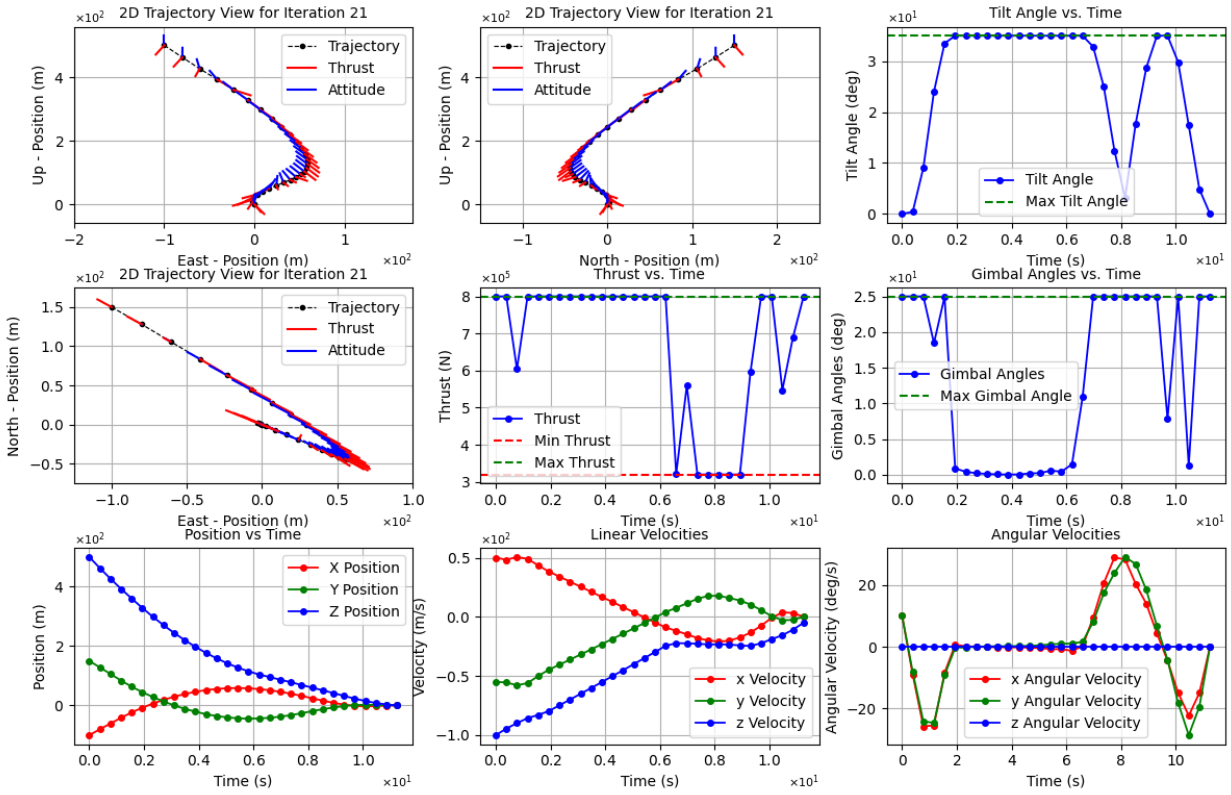

Fig. 3 **Comprehensive visualization of the 6-DoF rocket trajectory optimization results for the non-planar example. The top row presents 2D trajectory views along the East-Up and North-Up planes, with overlaid thrust and attitude vectors, and a plot of the tilt angle over time. The middle row includes the North-East 2D trajectory, thrust magnitude over time, and gimbal angles over time. The bottom row illustrates the time evolution of position, linear velocities, and angular velocities.**

have shown the ability to compute trajectories and control inputs that would be impossible to derive manually. From a practical standpoint, our Julia implementation was able to achieve iteration times on the order of tenths of seconds, which is exceptional when considering the difficulty of the original formulation and the desire to apply such methods to real-time trajectory planning decisions made by on-board computers. One way in which we would like to extend this work in the future is to incorporate robust optimization techniques for uncertainty quantification of initial conditions, environmental disturbances, or system dynamics. To this end, we would be able to show the ability to dynamically respond to system perturbations by re-planning when uncertainty is realized. This is an active area of development for companies like SpaceX, and it enables incredible feats like the one we were able to witness recently with the Super Heavy booster landing.

References

[1] Mao, Y., Szmuk, M., and Açıkmeşe, B., "Successive Convexification of Non-Convex Optimal Control Problems and Its Convergence Properties," *Proceedings of the 2016 IEEE 55th Conference on Decision and Control (CDC)*, IEEE, Las Vegas, NV, 2016, pp. 3636–3641.

[2] Szmuk, M., and Açıkmeşe, B., "Successive Convexification for 6-DoF Mars Rocket Powered Landing with Free-Final-Time," *2018 AIAA Guidance, Navigation, and Control Conference*, American Institute of Aeronautics and Astronautics, Kissimmee, FL, 2018. https://doi.org/10.2514/6.2018-0617, paper 2018-0617.

[3] Malyuta, D., Reynolds, T. P., Szmuk, M., Lew, T., Bonalli, R., Pavone, M., and Açıkmeşe, B., "Convex Optimization for Trajectory Generation: A Tutorial on Generating Dynamically Feasible Trajectories Reliably and Efficiently," *IEEE Control Systems Magazine*, Vol. 42, No. 5, 2022, pp. 40–113. https://doi.org/10.1109/MCS.2022.3187542.

[4] 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. 750–757. https://doi.org/10.2514/1.62110.

[5] Mao, Y., Szmuk, M., Xu, X., and Açıkmeşe, B., "Successive Convexification: A Superlinearly Convergent Algorithm for Non-Convex Optimal Control Problems," *arXiv*, Vol. arXiv:1804.06539v2, 2019. https://arxiv.org/abs/1804.06539v2.

[6] Szmuk, M., Eren, U., and Açıkmeşe, B., "Successive Convexification for Mars 6-DoF Powered Descent Landing Guidance," *AIAA Guidance, Navigation, and Control Conference*, American Institute of Aeronautics and Astronautics, Grapevine, Texas, 2017. https://doi.org/10.2514/6.2017-1500, URL https://arc.aiaa.org/doi/10.2514/6.2017-1500.