Fast and Resilient Optimization-based Control with Spacecraft Applications

Purnanand Elango

A dissertation

submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

University of Washington

2024

Reading Committee:

Behçet Açıkmeşe, Chair

Mehran Mesbahi

Michael Szmuk

Program Authorized to Offer Degree:

William E. Boeing Department of Aeronautics & Astronautics

University of Washington

Abstract

Fast and Resilient Optimization-based Control
with Spacecraft Applications

Purnanand Elango

Chair of the Supervisory Committee:
Professor Behçet Açıkmeşe
William E. Boeing Department of Aeronautics & Astronautics

Trajectory optimization, which forms a key component of modern guidance, navigation, and control (GNC) systems, occupies the middle layer within a three-layer hierarchy, where the top layer is mission planning (or high-level decision making) and the bottom layer consists of low-level computation in the form of general-purpose solvers for optimization problems and feedback controllers. In this dissertation, we develop fast and resilient optimization-based solution methods that cover the full stack—spanning the three layers and tightly integrating them—while remaining anchored to the middle layer. First, we develop a convergence-guaranteed, real-time-capable solution method for nonconvex trajectory optimization, based on sequential convex programming (SCP) and isoperimetric constraint reformulation, that ensures continuous-time path constraint satisfaction. Next, we customize a first-order conic optimization algorithm, by exploiting the sparsity structure of trajectory optimization problems, to obtain a real-time-capable solver implementation that is suitable for embedded applications. Consequently, the overhead of a parser layer between the middle and bottom layers is eliminated. Finally, we embed a class of mission requirements (to ensure resilience to unmodeled uncertainties

and contingencies) within a trajectory optimization problem, which can then be solved using the solution methods that we developed. We demonstrate the suite of solution methods via numerical examples based on real-world optimal control applications: precision six-degree-of-freedom (6-DoF) rocket landing, quadrotor motion planning, dynamic obstacle avoidance, and three-degree-of-freedom (3-DoF) rocket landing with lossless convexification.

Furthermore, we develop optimization-based closed-loop control strategies specialized to spacecraft applications in the cislunar space, such as station keeping, autonomous optical navigation, and passively-safe rendezvous on the near rectilinear halo orbit (NRHO). Again, the motivation is to design solution methods that combine two layers of the hierarchy—(middle) feedforward trajectory optimization and (bottom) feedback control. We demonstrate that exploiting properties of the dynamic behavior of the system and its environment is useful for designing lightweight algorithms (suitable for onboard deployment) that can deliver the required performance.

To my parents

Contents

List of Figures

List of Tables

ACKNOWLEDGEMENTS

At face value, the worth of my Ph.D. might seem to lie solely in the contents of this dissertation. However, I believe the true value of this Ph.D. stems from the sum total of my collaborations, interactions, and learning experiences with brilliant and inspiring individuals over the past several years. It is to them that I owe my memorable five-year journey at UW.

I am deeply grateful to my advisor, Prof. Behçet Açıkmeşe, for the opportunity to join ACL. When I applied to the Ph.D. program at UW Aeronautics & Astronautics, I was drawn to the controls and optimization research but was unaware of Behçet's work and his research group, ACL. Fortunately, Behçet reached out to interview me shortly after I submitted my application. Thank you, Behçet, for granting me the freedom to explore various directions and evolve a theme for my dissertation at my own pace. Working closely with you on research problems has been exhilarating. I thoroughly enjoyed our meetings where we would "think together," as you often put it, and delve into problems or ideas in great detail. My work has greatly benefitted from your sharp insights and creative solutions. I have been told that I have "cracked the code" to getting you deeply invested in a research problem, but I believe I have just been very lucky.

I want to thank Prof. Mehran Mesbahi for his comprehensive and critical feedback on my research as part of my dissertation committee. Beyond our many insightful conversations over the years, I had the pleasure of taking his abstract linear algebra course, which was enriched by his fascinating anecdotes. Mehran is by far one of the best instructors I have encountered at UW.

I am grateful to Miki Szmuk, whom I first met at SciTech 2019. Miki took the time to chat with me about his experience at ACL, which convinced me that ACL was the place to pursue research on optimization-based control and to be part of a group with a strong

collaborative spirit. Miki has had a significant impact on my Ph.D. research. The trajectory optimization algorithms he developed have been a source of inspiration for my work, and he also facilitated an internship at Amazon Prime Air, where I tested my research on a cutting-edge aerospace application. Working with Miki and Taylor at Prime Air felt surreal and was a dream come true. Thank you, Miki, for your confidence in me and for being a part of my dissertation committee, which has been a great honor.

I would like to thank Yue Yu for playing a critical role in helping me lay the foundation of my Ph.D. research. He took me under his wing and introduced me to modern first-order optimization algorithms. I feel privileged to have been part of PIPG's development from the very beginning. Moreover, his mentorship taught me how to formulate research questions, scope problems, maintain high standards for quality work, and, most importantly, adopt the mindset needed to deliver results on time.

Thank you to Danylo Malyuta, whose steadfast discipline, work ethic, and technical brilliance have been a constant source of motivation and inspiration. Above all, Danylo exemplifies what it means to be a good-natured human being, always offering the kindest and most thoughtful words.

I want to thank Abhi Kamath, with whom I've had a highly enjoyable and successful collaboration since 2021. Abhi's remarkable creativity, perseverance, and willingness to go above and beyond made our collaboration a definitive part of my Ph.D. journey—I'm honored to have been part of it. Besides being an amazing collaborator, Abhi is a great friend, always ready to help. I'm glad I introduced you to sushi sooner rather than later, and we have had a good Omakase run in 2024.

Thank you to Burak Sarsılmaz. Collaborating with Burak was another major highlight of my Ph.D. His mentorship taught me how to analyze ideas with rigor and clarity, and how to be vigilant for corner cases while tying together the pieces of a framework.

I would like to thank Taewan Kim. It has been a pleasure to collaborate with you and witness the amazing strides you have made in funnel synthesis research. I appreciated your balanced, well-rounded comments on a variety of topics during our informal discussions, and I have valued having you as my sounding board.

I want to thank Skye Mceowen. Our collaboration over the years has deepened my understanding and appreciation for the numerics of SCP. You have been a crucial member of ACL, helping to sustain and carry forward the SCP camaraderie.

I want to thank Avishai Weiss from MERL for our fruitful collaboration over the past three years, which began with a summer internship in 2021 and led to a job opportunity in 2024. I am very excited about working with you and also want to extend my gratitude to Stefano Di Cairano, Abraham Vinod, and Karl Berntorp at MERL for their help and advice over the years.

I am grateful to John Carson and Gavin Mendeck at NASA Johnson Space Center for their continued collaboration and support of my trajectory optimization research, including their efforts in arranging my visit to JSC.

A special thank you to Prof. Rekha Thomas for her advice and kind follow-up during my first quarter at UW. Prof. Thomas is an excellent teacher who genuinely cares about her students and presents elegant, easy-to-grasp geometric intuition for abstract concepts whenever possible. I also want to thank Prof. Steve Brunton for his initial service on my committee and for his excellent YouTube lectures on control theory and numerical approximations. I appreciate Prof. Sawyer Fuller for agreeing to step in as the GSR of my committee at the last minute. Thanks to Profs. Dima Drusvyatskiy and Sasha Aravkin for their courses on convex analysis and optimization algorithms, which opened doors for me to understand convergence guarantees and the properties of modern first-order algorithms—a timely knowledge for my collaboration with Yue on PIPG.

I am also grateful for a couple of memorable events during my Ph.D. First, my collaboration with Yue and Danylo on the spacecraft control survey paper for the Annual Reviews in Control was an intense learning opportunity. I consider myself fortunate to have participated in such an important project just a few months into my Ph.D. It was incredible to witness Danylo and Yue execute the plan to write a high-stakes survey paper from start to finish within three months. Second, my SciTech hat-trick from 2022-2024. I'm glad I could be there for the eventful ACL dinner with Behçet, where he regaled us with stories from his Ph.D. and JPL days. Abhi and Skye, I'm sure you will agree that this is an

important rite of passage for every ACL member.

I also want to thank the members of ACL. Dayou and Samet, without your diligent efforts, a significant part of my dissertation would not have been possible. Your collaboration was invaluable. Govind, it was refreshing and inspiring to witness your unmatched energy and enthusiasm for learning. Sam, I enjoyed working with you on the numerical implementations of DDTO—I'm excited to see where you take it. I also want to thank Sarah Li, Sean Rice, Kazuya Echigo, Chris Hayner, Ben Chung, Natalia Pavlasek, Aman Tiwary, Jason Zhou, Avi Mittal, and Fabio Spada for the many interesting conversations and feedback on my work over the years.

My interest in trajectory optimization and control systems research began during my undergraduate studies at IIT Madras, where Prof. Ranjith Mohan introduced me to a fascinating albatross-inspired aircraft maneuver called dynamic soaring for a semester project. What started as a small side project turned into a years-long research collaboration where we explored spectral methods and Floquet analysis. The results of this collaboration paved the way for my Ph.D. opportunity at UW in no small part. I want to thank the members of his research group, RAFT Lab, between 2017 and 2019—Sandeep, Cibin, Salini, Ramanujam, and Navneeth—for their advice and many interesting conversations.

I also want to express my gratitude to Profs. Shankar Narasimhan, Arun Tangirala, and Raghunathan Rengaswamy at IIT Madras. The tools for statistical modeling and analysis that I learned in their courses have continued to be invaluable throughout my Ph.D.

Finally, I am deeply grateful to my family: my parents, Yamini and Elango, my brother Priyanth, and my in-laws, for their unwavering love, support, and belief in me. My parents moved mountains to provide me with the best opportunities, and I will forever be indebted to them. To my wife Joen, thank you for standing by me through the highs and lows of my Ph.D.; I can't wait to embark on our adventure together on the East Coast. A special thanks to my cats, Lei and Lulu, who contributed to this dissertation in their own unique way (literally, with a stray keystroke or two).

Chapter 1

INTRODUCTION

Trajectory optimization forms an important part of modern guidance, navigation, and control (GNC) systems, wherein, it is used to generate reference trajectories for onboard use and also in offline design and analysis. Trajectory optimization is a part of a three-level hierarchy (shown in Figure 1.1), where the top layer is mission planning (or high-level decision making), the middle layer is trajectory optimization (formulated as constrained nonlinear optimal control problems), and the bottom layer consists of low-level computation in the form of feedback controllers and off-the-shelf solvers for optimization problems. The top layer plans for multiple vehicles and maneuvers or a mission consisting of a

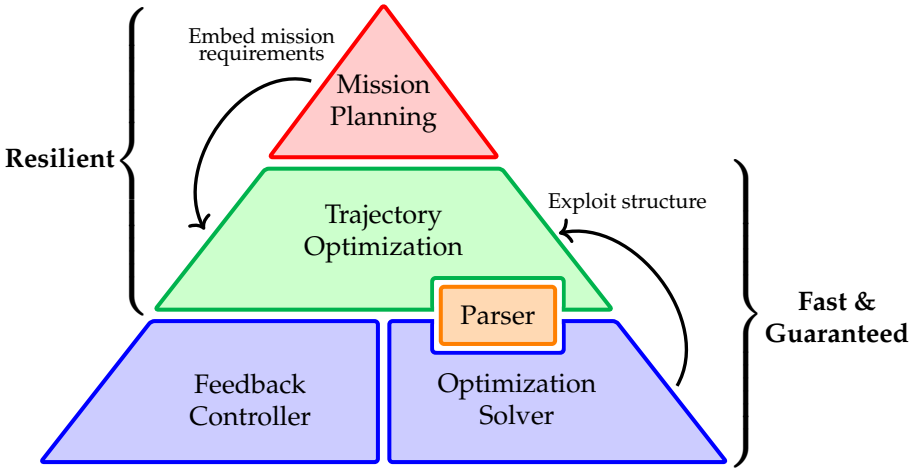


Figure 1.1: The hierarchy of mission planning, trajectory optimization, feedback controllers, and optimization solvers.

sequence of maneuvers (see for e.g., Amazon Prime Air's drone delivery CONOPS [2, Fig. 3]). The middle layer formulates the problem of generating continuous-time dynamically feasible maneuvers for a vehicle while satisfying continuous-time path constraints and boundary conditions [3, Eq. 1]. Unlike the top layer, this layer adopts a more fine-grained

view of the dynamic model of the vehicle and its environment, with an emphasis on constraint satisfaction. Finally, the bottom layer consists of two parts: (i) modular algorithms [4, 5, 6, 7, 8, 9] and solvers [10, 11, 12, 13] which numerically solve the optimization problem formulated in the above layers; and (ii) feedback controllers that treat the solution from the above layers as a feedforward reference signal. Furthermore, a parser layer (such as YALMIP [14], CVX [15], or CVXPY [16]) is required between the middle layer and off-the-shelf optimization solvers to transform the trajectory optimization problem to a canonical form [17, Sec. 5.1]. Figure 1.2 shows the placement of the three layers within a typical control system architecture. Note that the hierarchy we consider is similar to the one traditionally analyzed for control systems (also referred to as layered control architectures [18, Fig. 2]).

Autonomy, the ability to operate without intervention or assistance from humans, has become a highly desirable trait in modern robotics, due to its necessity for large-scale deployment of systems in highly-constrained environments with performance requirements [19]. Optimization-based methods are a natural choice for enabling autonomy. Human operators do not need to handcraft maneuvers by relying on years of domain expertise; they can simply state "what" needs to happen in the form of an optimization problem, without worrying about "how" the task is completed. Therefore, the ability to formulate and solve optimization problems that encompass a large class of mission and operational scenarios reliably would enable rapid prototyping, certifiably-safe deployment, and cost-efficient maintenance [20, Sec. 1].

Tight integration of the three layers is necessary to fully harness the capabilities of modern optimization algorithms in terms of modelling flexibility, convergence guarantees and real-time performance. For e.g., majority of the commercial-off-the-shelf (COTS) solvers for nonlinear programming require twice differentiability of constraint and objective functions (e.g., SNOPT [21] and Ipopt [12]), which can be unduly restrictive for optimization problems formulated in certain domains (e.g., hybrid systems), necessitating specialized optimization solvers. Also, certain applications (e.g., precision rocket landing) demand customized real-time-capable solvers with small code footprint for use onboard the vehicle.

On a similar vein, stronger coupling between the top and middle layers is beneficial. Typically, since mission planning adopts a coarse view of vehicle dynamic model, it is likely that the plan it generates is not simultaneously feasible in continuous-time with respect to the dynamic model and the path constraints. So, such artificial infeasibility can be avoided by embedding mission requirements into the trajectory optimization formulation whenever possible. Doing so would allow us to leverage state-of-the-art algorithms for convex and nonconvex optimization to generate dynamically feasible, path-constrained maneuvers that satisfy all mission requirements.

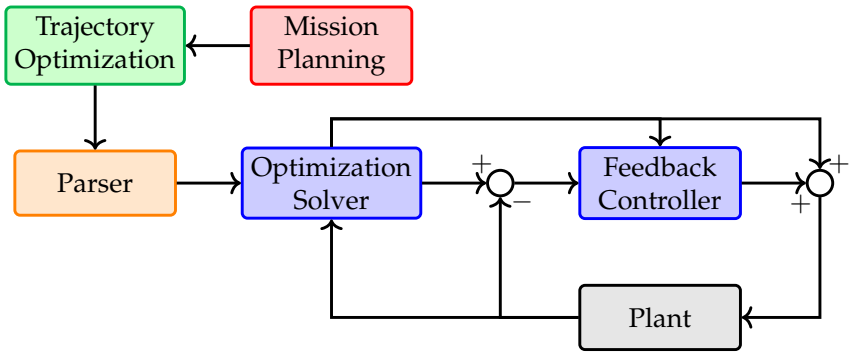


Figure 1.2: The three-layer hierarchy is placed within the context of a control system architecture.

In this dissertation, we develop fast and resilient optimization-based solution methods that cover the three layers (i.e., the full stack) and tightly integrate them, while remaining anchored to the middle layer. First, we develop a general-purpose, convergence-guaranteed, real-time-capable solution method for nonconvex trajectory optimization that ensures continuous-time constraint satisfaction [22]. Next, we customize a first-order conic optimization algorithm, by exploiting the sparsity structure of trajectory optimization problems, to obtain a real-time-capable solver implementation that is suitable for embedded applications [23, 24, 25]. Consequently, the overhead of a parser layer between the middle and bottom layers is eliminated. Finally, we embed a class of mission requirements (to ensure resilience to unmodeled uncertainties and contingencies) within a trajectory optimization problem [26, 27], which can then be solved using the solution methods that we developed.

Besides the aforementioned methods, we develop optimization-based closed-loop con-

trol strategies specialized to spacecraft applications in the cislunar space. The motivation is again to design solution methods that combine two layers of the hierarchy—(middle) feedforward trajectory optimization and (bottom) feedback control. Exploiting properties of the dynamic behavior of the system and its environments is useful for designing lightweight algorithms (suitable for onboard deployment) which can deliver the required performance. Some examples of properties that can be exploited for algorithm design are as follows: (i) geometric symmetries and time scales within cislunar orbits [28, 29]; (ii) empirical observations about the zone of validity of dynamical system linearization [30]; and (iii) rate of growth of open-loop covariance and the statistics of the relative range measurements during spacecraft rendezvous [31].

Methods and frameworks for unifying the three layers in Figure 1.1 have appeared in the recent literature. Some notable examples include funnel synthesis [32, 33, 34, 35], system level synthesis [36], layered control architectures [18], unified trajectory generation and tracking [37, 38], FaSTrack [39], and desensitized optimal control [40, 41]. However, to the best of the author's knowledge, none of these methods provide: (i) continuous-time constraint satisfaction; (ii) a factorization-free, library-free, first-order conic optimization solver which avoids sparse linear algebra and is suitable for embedded applications; (iii) a real-time-capable, convergence-guaranteed, sequential convex programming (SCP) algorithm for trajectory optimization; and (iv) resilience to unmodeled uncertainties and contingencies (which is distinct from what robust and stochastic control methods can ensure). This dissertation bridges the gap in the literature by developing solution methods that possess these attributes, which are crucial for safe, reliable and performant operation of autonomous systems.

1.1 Contributions and Thesis Outline

In this section, we describe the specific contributions of this dissertation in the subsequent chapters. The existing literature relevant to each contribution is surveyed in the respective chapter.

Chapter 2. Nonconvex trajectory optimization [22]:

In this work, we develop continuous-time successive convexification (CT-SCVX), a real-time-capable solution method for constrained trajectory optimization, with continuous-time constraint satisfaction and guaranteed convergence. The proposed solution framework only relies on first-order information, and it combines several key methods to solve a large class of nonlinear optimal control problems: (i) exterior penalty-based reformulation of the path constraints; (ii) generalized time-dilation; (iii) multiple-shooting discretization; (iv) ℓ_1-exact penalization of the nonconvex constraints; and (v) the prox-linear method, a sequential convex programming (SCP) algorithm for convex-composite minimization. The proposed reformulation of the path constraints enables continuous-time constraint satisfaction even on sparse temporal discretization grids and obviates the need for mesh-refinement heuristics. Through the prox-linear method, we guarantee that: (i) CT-SCVX converges to stationary points of the penalized problem; (ii) the converged stationary points that are feasible for the discretized and control-parameterized optimal control problem are also Karush-Kuhn-Tucker (KKT) points. Furthermore, we specialize this property to global minimizers of convex optimal control problems and obtain stronger convergence results by exploiting convexity. In addition to theoretical analysis, we demonstrate the effectiveness and real-time performance of CT-SCVX by means of numerical examples from real-world optimal control applications: dynamic obstacle avoidance, and 3-degree-of-freedom (3-DoF) and 6-DoF autonomous rocket landing.

Chapter 3. Customized conic optimization solver [23, 25, 42, 43]:
We develop a real-time-capable customized implementation of the proportional-integral projected gradient PIPG method for solving (standalone) convex trajectory optimization problems and the subproblems within SCP-based methods for nonconvex trajectory optimization. The customization of PIPG to exploit the trajectory optimization structure allows library-free, factorization-free, and easily-verifiable solver implementation for real-time and embedded applications. It achieves automatic primal and dual infeasibility detection, optimal global convergence rates in theory, and orders-of-magnitudes faster computation in practice. It exploits the sparsity structure of conic constraints via parallel matrix operations and the geometric structure of constraint sets via efficient projections. Furthermore, it relies only on simple linear algebra operations such as matrix-vector multiplication and

vector addition, and thus lends to not only easy verification and validation, but also very small code footprint, making it suitable for onboard implementation on computationally-constrained hardware. Unlike most off-the-shelf methods, PIPG allows easy warm-starts and avoids the overheard of cumbersome canonical transformation of standard conic programs. We demonstrate the performance of the customized solver on numerical examples based on two optimal control applications: powered-descent guidance and oscillating masses.

Chapter 4. Deferred decision trajectory optimization [27, 26]:
This work proposes DDTO—deferred decision trajectory optimization—a framework for trajectory generation with the embedded mission requirement of ensuring robustness to unmodeled uncertainties and contingencies. The key idea is to ensure that a collection of candidate targets are reachable for as long as possible while satisfying constraints, which provides time to quantify the uncertainties. We propose optimization-based constrained reachability formulations and construct equivalent cardinality minimization problems, which then inform the design of computationally tractable and efficient solution methods that leverage state-of-the-art convex solvers and SCP algorithms. The goal of establishing the equivalence between constrained reachability and cardinality minimization is to provide theoretically sound underpinnings for the proposed solution methods. We demonstrate the solution methods on real-world optimal control applications based on quadrotor motion planning.

Chapter 5. Closed-loop methods for cislunar applications:
We develop optimization-based closed-loop control strategies specialized to spacecraft applications in the cislunar space.

> ***Section 5.1. Local eigenmotion control for station keeping [44]:***
> The upcoming deployment of the Lunar Orbital Platform-Gateway (LOP-G) on a near rectilinear halo orbit (NRHO) calls for reliable, low-cost strategies for station keeping and relative motion tailor-made for NRHO. This work proposes a control approach which harnesses the eigenvectors of state transition matrices (STM) associated with a high-fidelity NRHO solution in the ephemeris model to design long-term

station keeping and bounded relative motion. The proposed method effectively utilizes the natural motion of the spacecraft so that control actions are infrequent and fuel efficient. The performance of the proposed approach is demonstrated via simulations with a state estimator that uses simulated measurements from the Deep Space Network.

Section 5.2. Sequential linearization station keeping with OPNAV [45]:
Station-keeping approaches relying on an autonomous navigation system that does not require communications with Earth are particularly important for ensuring safety and reliability. This work presents a targeting approach for NRHO station keeping based on sequential linearization and evaluates its performance in a closed-loop simulation with a state estimator that receives position measurements from horizon-based optical navigation (OPNAV). Simulation results indicate an annual station-keeping cost (delta-v) of about 1.14 m s^{-1} for the proposed OPNAV-based station keeping, which is competitive compared to existing closed-loop station-keeping methods.

Section 5.3. Spacecraft rendezvous with continuous-time passive safety [46]:
We propose a solution method for long-range, passively-safe rendezvous of a spacecraft to the LOP-G. We reformulate the continuous-time passive-safety constraint as an isoperimetric constraint and augment it to the system dynamics to eliminate the commonly-encountered inter-sample constraint violation in direct methods. Consequently, computationally expensive mesh-refinement heuristics are not necessary: the proposed approach can generate a high-fidelity solution with coarse discretization grids. Furthermore, we impose chance constraints to incorporate uncertainties encountered in a closed-loop execution, and reduce its conservativeness by jointly synthesizing a feedback controller that inhibits the growth of the state covariance by using the statistics of relative range and range-rate measurements. We solve the constraint-reformulated, feedback-augmented optimal control problem for rendezvous through an approach based on the CT-SCVX framework. The solution method is demonstrated using a realistic numerical example based on a three-phase

rendezvous to the Gateway while ensuring safety.

Finally, we conclude with directions for future research in Chapter 6.

The work presented in this dissertation was funded by the Office of Naval Research Grant N00014-20-1-228, the Air Force Office of Scientific Research Grant A9550-20-1-0053, and the NASA Grant NNX17AH02A.

Chapter 2

NONCONVEX TRAJECTORY OPTIMIZATION

2.1 Introduction

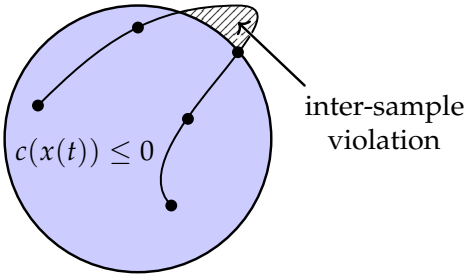


Figure 2.1: Direct methods for trajectory optimization impose path constraints (such as $c(x(t)) \leq 0$) at finitely-many time nodes (black dots), invariably causing inter-sample violation (shaded region) in the state trajectory x obtained through integration of the dynamical system with the control input solution. The proposed framework mitigates this phenomenon.

State-of-the-art trajectory optimization methods do not yet check all the boxes with regard to desirable features: continuous-time constraint satisfaction, real-time performance, convergence guarantees, and numerical robustness [20]. Among these, continuous-time feasibility, which refers to the feasibility of the state trajectory and control input with respect to the system dynamics *and* path constraints in continuous-time, is especially challenging due to the infinite-dimensional nature of optimal control problems. However, it is essential for meeting safety and performance requirements, which is a prerequisite for the deployment of autonomous systems such as space exploration vehicles, reusable rockets, dexterous robotic manipulators, etc [3]. Aerospace applications, in particular, trade optimality for robustness [47].

Trajectory optimization methods can be broadly classified into *indirect* and *direct methods* [48, Sec. 4.3]. Indirect methods solve the optimality conditions of an optimal control problem (based on the Pontryagin's maximum principle (PMP) [49, Chap. 4]), which can

be challenging for nonlinear dynamics subject to nonconvex path constraints. Such solution methods usually take an *optimize-then-discretize* approach, and can only solve a restricted class of problems. On the other hand, direct methods, which take the *discretize-then-optimize* approach, are capable of handling general optimal control problems and are more reliable numerically (they are less sensitive to initialization). In rare circumstances, PMP can reveal convexification approaches that direct methods can exploit [50, 51]. However, since discretization is an unavoidable step in direct methods, the resulting solutions invariably suffer from so-called inter-sample constraint violations [52] (see Figure 2.1).

Given their flexibility and generality, our focus is on designing a direct method with desirable features. Existing direct methods compromise continuous-time feasibility by either—(i) parameterizing the state trajectory with piecewise polynomials and imposing the system dynamics and path constraints at finitely-many time nodes (e.g., direct collocation [48, Sec. 4.5], [53]); or (ii) disregarding the conversion of continuous-time problem descriptions to discrete-time ones (e.g., TrajOpt [54], PANOC [4], CALIPSO [55]). Meanwhile, some approaches ensure continuous-time feasibility for a restricted class of dynamical systems [56, 57, 52], while some provide convergence guarantees for a restricted class of path constraints [58, 59, 60, 61]. Furthermore, a majority of the existing software for direct methods, such as MISER [62], DIRCOL [63], PROPT [64], and GPOPS-II [65], transcribe optimal control problems to nonlinear programs (NLP) and call standard solvers, such as SNOPT [21] and IPOPT [12], which require twice-differentiable objective and constraint functions, and are unsuitable for real-time, embedded applications. In contrast, software such as `acados` [66] and ACADO [67] interface with custom sequential quadratic programming (SQP)- and interior-point method (IPM)-based NLP solvers that exploit the structure of the underlying problem. However, they do not enforce continuous-time feasibility.

Sequential convex programming (SCP) algorithms for trajectory optimization have received attention in the recent years [68, 3] as a competitive alternative to SQP- and IPM-based NLP algorithms, especially since SCP is a multiplier-free method and does not require second-order information. In trajectory optimization, computing the second-order sensitivities of

nonlinear dynamics (which form a part of the Hessian of the Lagrangian within SQP and IPMs) is expensive [69, Sec. 10.5]. Recent work has explored convergence guarantees for SCP, in the context of both direct methods for trajectory optimization [60, 70, 71, 72, 73, 74] and general nonconvex optimization [75, 76, 77, 5]. SCP-based direct methods, without theoretical guarantees, have been applied to a wide range of robotics and aerospace applications, with domain-specific heuristics that ensure effective practical performance [54, 78, 79, 80]. Widespread adoption of SCP for performance- and safety-critical applications, however, will necessitate the development of a general framework with rigorous analysis of its capabilities.

We propose continuous-time successive convexification (CT-SCVX), a direct solution method for constrained trajectory optimization, with continuous-time feasibility and guaranteed convergence. The proposed SCP-based framework combines several key methods to solve a large class of nonlinear optimal control problems: (i) exterior penalty-based reformulation of the path constraints; (ii) generalized time-dilation; (iii) multiple-shooting discretization; (iv) ℓ_1-exact penalization of the nonconvex constraints; and (v) the prox-linear method, a convergence-guaranteed SCP algorithm for convex-composite minimization.

The proposed reformulation of path constraints involves integrating the continuous-time constraint violation using a smooth exterior penalty, which is transformed into an auxiliary dynamical system with boundary conditions. The reformulation combined with multiple-shooting discretization [81] enables continuous-time feasibility even on sparse discretization grids, and obviates the need for mesh-refinement heuristics. While such constraint reformulations have appeared in the optimal control literature since the 1960s [82, 83, 84] with specific choices of penalty functions [85, 86] and including the integration of residuals [87], the full extent of their capabilities for enabling SCP-based convergence-guaranteed, continuous-time-feasible trajectory optimization have not been explored, to the best of the authors' knowledge. We address the consequences of the reformulation and provide a rigorous quantification of the extent of continuous-time constraint satisfaction.

Through the prox-linear method [5, 88], we guarantee that: (i) CT-SCVX converges to

stationary points of the ℓ_1-penalized problem; (ii) the converged solutions that are feasible for the discretized and control-parameterized optimal control problem are also Karush-Kuhn-Tucker (KKT) points. Furthermore, we highlight the specialization of this property to global minimizers of convex optimal control problems, wherein the reformulated path constraints cannot be represented in the canonical conic form required by existing convex optimization solvers. Note that CT-SCVX is distinct from the trust-region-based SCVX algorithm [70, 73] and its variant [74], which consider a discrete-time formulation and impose restrictive assumptions to guarantee convergence.

In addition to theoretical analysis, we demonstrate the effectiveness of CT-SCVX by means of numerical examples based on real-world optimal control applications: two nonconvex problems—dynamic obstacle avoidance and six-degree-of-freedom (6-DoF) rocket landing, and one convex problem—3-DoF rocket landing using lossless convexification. We also demonstrate the real-time capability of CT-SCVX on the nonconvex examples considered, through a C codebase generated using SCVXGEN, an in-house-developed general-purpose real-time trajectory optimization software with customized code-generation support. CT-SCVX was recently demonstrated for applications ranging from GPU-accelerated trajectory optimization for 6-DoF rocket landing [43] and nonlinear model predictive control (NMPC) for obstacle avoidance [89], to trajectory optimization for 6-DoF aircraft approach and landing [90].

2.2 Problem Formulation

This section describes the transformation of a path-constrained, free-final-time optimal control problem to a fixed-final-time optimal control problem through generalized time-dilation and constraint reformulation. The constraint reformulation involves the conversion of path constraints into a two-point boundary value problem for an auxiliary dynamical system.

2.2.1 Notation

We adopt the following notation in the remainder of the discussion. The set of real numbers is denoted by $\mathbb{R}$, the set of nonnegative real numbers by $\mathbb{R}_+$, the set of real $n \times m$ matrices by $\mathbb{R}^{n\times m}$, and the set of real $n \times 1$ vectors by $\mathbb{R}^n$. The concatenation of vectors $v \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$ is denoted by $(v, u) \in \mathbb{R}^{n+m}$, the concatenation of matrices $A \in \mathbb{R}^{l\times m}$ and $B \in \mathbb{R}^{l\times n}$ by $[A\ B] \in \mathbb{R}^{l\times(m+n)}$, the Cartesian product of sets C and D by $C \times D$, and the Kronecker product by $\otimes$. The vector of ones in $\mathbb{R}^n$ is denoted by 1_n, the identity matrix in $\mathbb{R}^{n\times n}$ by I_n, and the matrix of zeros in $\mathbb{R}^{n\times m}$ by $0_{n\times m}$. Whenever the subscript is omitted, the size is inferred from context. For any scalar v, we define $|v|_+ = \max\{0, v\}$. The operations $|\square|_+$, $|\square|$, $\square^2$ (and their compositions) apply elementwise for a vector. The Euclidean norm of a vector v is denoted by $\|v\|$. The indicator function of a convex set D is denoted by $\tilde{I}_D$ (see [91, E.g. 3.1]), and its normal cone at x by $\mathcal{N}_D(x)$ (see [92, Def. 5.2.3]). The subdifferential of a function f, evaluated at x, is a set denoted by $\partial f(x)$, and its members are called subgradients. The gradient of a differentiable function $g : \mathbb{R}^n \to \mathbb{R}^l$ with respect to $z \in \mathbb{R}^m$, evaluated at $x \in \mathbb{R}^n$, is denoted by $\nabla_z g(x) \in \mathbb{R}^{l\times m}$, where the elements of z can include a subset or superset of the arguments of g (irrespective of the order), and the subscript z is omitted if it coincides with the list of arguments of g. The (sub)gradient of a scalar-valued function at a point is defined to be a row vector, and the (sub)gradient of a vector-valued function is a matrix consisting of the (sub)gradients of its scalar-valued elements along the rows. In particular, the partial derivatives of function $(u, v) \mapsto h(u, v)$, evaluated at $\bar{u}, \bar{v}$, are denoted by $\nabla_u h(\bar{u}, \bar{v})$, $\nabla_v h(\bar{u}, \bar{v})$, respectively (with the argument inferred from context whenever omitted). Furthermore, the notation $\mathrm{d}\square(\varsigma)/\mathrm{d}\varsigma$ represents the derivative with respect to a scalar variable.

2.2.2 Optimal Control Problem

We consider a class of *free-final-time* optimal control problems for nonlinear dynamical systems with nonconvex constraints on the state and input, given by

$$\begin{aligned}
\underset{x,u,t_\mathrm{f}}{\text{minimize}} \quad & L(t_\mathrm{f}, x(t_\mathrm{f})) && \text{(2.1a)}\\
\text{subject to} \quad & \dot{x}(t) = f(t, x(t), u(t)) \ \text{ a.e. } t \in [t_\mathrm{i}, t_\mathrm{f}] && \text{(2.1b)}\\
& g(t, x(t), u(t)) \leq 0_{n_g} \quad \text{a.e. } t \in [t_\mathrm{i}, t_\mathrm{f}] && \text{(2.1c)}\\
& h(t, x(t), u(t)) = 0_{n_h} \quad \text{a.e. } t \in [t_\mathrm{i}, t_\mathrm{f}] && \text{(2.1d)}\\
& P(t_\mathrm{i}, x(t_\mathrm{i}), t_\mathrm{f}, x(t_\mathrm{f})) \leq 0_{n_P} && \text{(2.1e)}\\
& Q(t_\mathrm{i}, x(t_\mathrm{i}), t_\mathrm{f}, x(t_\mathrm{f})) = 0_{n_Q} && \text{(2.1f)}
\end{aligned}$$

where the derivative with respect to time $t \in [t_\mathrm{i}, t_\mathrm{f}]$ in (2.1b) is denoted by $\dot{\square} = \mathrm{d}\square(t)/\mathrm{d}t$. We assume that the terminal state cost function $L : \mathbb{R}_+ \times \mathbb{R}^{n_x} \to \mathbb{R}$, the dynamics function $f : \mathbb{R}_+ \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$, the path constraint functions $g : \mathbb{R}_+ \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_g}$, $h : \mathbb{R}_+ \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_h}$, and the boundary condition constraint functions $P : \mathbb{R}_+ \times \mathbb{R}^{n_x} \times \mathbb{R}_+ \times \mathbb{R}^{n_x} \to \mathbb{R}^{n_P}$, $Q : \mathbb{R}_+ \times \mathbb{R}^{n_x} \times \mathbb{R}_+ \times \mathbb{R}^{n_x} \to \mathbb{R}^{n_Q}$, are continuously differentiable. The inequality and equality constraints in (2.1c)-(2.1f) are interpreted elementwise. The initial time $t_\mathrm{i} \in \mathbb{R}_+$ is fixed, while the final time $t_\mathrm{f} \in \mathbb{R}_+$ is a free (decision) variable. For simplicity, we omit parameters [3, Eq. 1] of the system dynamics and constraints in (2.1), which can be handled with minimal modifications to the subsequent development. Note that the (2.1) can be a *nonconvex trajectory optimization* problem due to either: (i) nonlinear dynamics, i.e., f is a nonlinear function; (ii) final time being free; (iii) nonlinear functions h and Q; or (iv) nonconvex functions g and P.

We assume that the *control input* $u : [t_\mathrm{i}, t_\mathrm{f}] \to \mathbb{R}^{n_u}$ is a *piecewise continuous function*. Since f is continuously differentiable, this ensures the existence and uniqueness of an absolutely continuous function $x : [t_\mathrm{i}, t_\mathrm{f}] \to \mathbb{R}^{n_x}$, the *state trajectory*, which satisfies (2.1b) almost everywhere and the integral equation: $x(t) = x_\mathrm{i} + \int_{t_\mathrm{i}}^{t} f(\gamma, x(\gamma), u(\gamma))\mathrm{d}\gamma$, where $x(t_\mathrm{i}) = x_\mathrm{i}$ is the initial state. Note that existence and uniqueness of the state trajectory

can be ensured with weaker assumptions; we refer to [49, Sec. 3.3.1] and [93, Sec. II.3] for detailed discussions. We also ensure boundedness of the state trajectory via Gronwall's Lemma [94, Chap. 4, Prop. 1.4] using the following assumption.

Assumption 1. *For any compact* $\mathbb{U} \subset \mathbb{R}^{n_u}$ *and* $t_\mathrm{f} > t_\mathrm{i} \geq 0$, *there exist positive* ϑ *and* ϱ *such that* $\|f(t,x,u)\| \leq \vartheta\|x\| + \varrho$ *for all* $(t,x,u) \in [t_\mathrm{i}, t_\mathrm{f}] \times \mathbb{R}^{n_x} \times \mathbb{U}$.

Note that Assumption 1 may not always hold. Consider $f(t,x,u) = x^2$. For any positive ϑ and ϱ, when x becomes sufficiently large, Assumption 1 is invalid. Nonetheless, for physical systems, one can define a compact set $\mathbb{X} \subset \mathbb{R}^{n_x}$, containing all physically meaningful states. Any state not contained in $\mathbb{X}$ can be regarded as infeasible. Consequently, we create a modified dynamics function so that it satisfies Assumption 1 and is consistent with the original dynamics function on $\mathbb{X}$. To be precise, for any open set $\mathbb{W}$ containing $\mathbb{X}$, there exists a smooth function $\omega : \mathbb{R}^{n_x} \to \mathbb{R}$ such that $\omega(x) = 1$ for all $x \in \mathbb{X}$ and the support of ω is compact and contained in $\mathbb{W}$, as shown in [95, Thm. 8.18]. Therefore, we can replace function f in (2.1b) with ωf, which is compactly supported in $\mathbb{R}^{n_x}$, ensuring Assumption 1. Note that the inclusion of ω is only for the purpose of proofs of results discussed in subsequent sections; it is not necessary in a practical implementation.

2.2.3 Constraint Reformulation

To reformulate path constraints (2.1c) and (2.1d), we consider continuous *exterior penalty functions*, given by

$$q_i : \mathbb{R} \to \mathbb{R}_+, \quad i = 1, \ldots, n_g, \tag{2.2a}$$

$$q_i(z) \begin{cases} = 0 & \text{if } z \leq 0, \\ > 0 & \text{otherwise}, \end{cases} \tag{2.2b}$$

$$p_j : \mathbb{R} \to \mathbb{R}_+, \quad j = 1, \ldots, n_h, \tag{2.2c}$$

$$p_j(z) \begin{cases} = 0 & \text{if } z = 0, \\ > 0 & \text{otherwise}, \end{cases} \tag{2.2d}$$

and the penalty function $\Lambda : \mathbb{R}_+ \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}_+$, given by

$$\Lambda(t, x, u) = \sum_{i=1}^{n_g} q_i(g_i(t, x, u)) + \sum_{j=1}^{n_h} p_j(h_j(t, x, u)). \tag{2.3}$$

Note that g_i and h_j, for $i = 1, \ldots, n_g$ and $j = 1, \ldots, n_h$, denote the scalar-valued elements of functions g and h, respectively. It is straightforward to show that $t \mapsto \Lambda(t, x(t), u(t))$ is piecewise continuous over $[t_\mathrm{i}, t_\mathrm{f}]$ when x is a state trajectory and u is the corresponding piecewise continuous control input.

Lemma 1. *Path constraints* (2.1c) *and* (2.1d) *are satisfied by* $x(t)$ *and* $u(t)$ a.e. $t \in [t_\mathrm{i}, t_\mathrm{f}]$ *iff*

$$\int_{t_\mathrm{i}}^{t_\mathrm{f}} \Lambda(t, x(t), u(t)) \mathrm{d}t = 0. \tag{2.4}$$

Proof. For each $t \in [t_\mathrm{i}, t_\mathrm{f}]$, $\Lambda(t, x(t), u(t))$ is nonnegative since $q_i(g_i(t, x(t), u(t)))$ and $p_j(h_j(t, x(t), u(t)))$ are nonnegative. Since $t \mapsto \Lambda(t, x(t), u(t))$ is piecewise continuous, (2.4) holds iff $\Lambda(t, x(t), u(t)) = 0$ a.e. $t \in [t_\mathrm{i}, t_\mathrm{f}]$. Then $q_i(g_i(t, x(t), u(t))) = 0$ and $p_j(h_j(t, x(t), u(t))) = 0$ a.e. $t \in [t_\mathrm{i}, t_\mathrm{f}]$, which is equivalent to $g_i(t, x(t), u(t)) \leq 0$ and $h_j(t, x(t), u(t)) = 0$ a.e. $t \in [t_\mathrm{i}, t_\mathrm{f}]$. □

The integral representation of the path constraints is also known as the isoperimetric constraint [96]. Next, we augment the system dynamics with an auxiliary state to accumulate the constraint violations, and impose periodic boundary conditions on it to ensure continuous-time constraint satisfaction.

Corollary 1. *The differential-algebraic system*

$$\dot{x}(t) = f(t, x(t), u(t)), \tag{2.5a}$$

$$g(t, x(t), u(t)) \leq 0, \tag{2.5b}$$

$$h(t, x(t), u(t)) = 0, \tag{2.5c}$$

is satisfied by x *and* u *a.e. in* $[t_\mathrm{i}, t_\mathrm{f}]$ *iff* x *and* u *solve the boundary value problem on* $[t_\mathrm{i}, t_\mathrm{f}]$

$$\dot{x}(t) = f(t, x(t), u(t)), \tag{2.6a}$$

$$\dot{y}(t) = \Lambda(t, x(t), u(t)), \tag{2.6b}$$

$$y(0) = y(t_f). \tag{2.6c}$$

Proof. Suppose that x and u satisfy (2.1b), (2.1c), and (2.1d) almost everywhere in $[t_i, t_f]$. Since $t \mapsto \Lambda(t, x(t), u(t))$ is piecewise continuous over $[t_i, t_f]$, there exists a unique state trajectory $y : [t_i, t_f] \to \mathbb{R}$ which satisfies (2.6b) almost everywhere in $[t_i, t_f]$ with initial condition $y(t_i) = 0$. Then $y(t_f) = 0$ iff (2.4) holds. Therefore, from Lemma (1), the path constraints (2.1c) and (2.1d) are satisfied almost everywhere in $[t_i, t_f]$.

For the other direction, we let x, y, and u be the solution to (2.6). Then (2.6c) implies that (2.4) holds. We obtain the desired result from Lemma 1. □

Remark 1. *Since a gradient-based solution method will be adopted, the exterior penalty functions q_i and p_j, for $i = 1, \ldots, n_g$ and $j = 1, \ldots, n_h$, must be continuously differentiable in order to compute first-order sensitivities [97, Sec. 4.2] of (2.6b). For e.g.,*

$$q_i(z) = |z|_+^2, \; p_j(z) = z^2, \tag{2.7}$$

for $z \in \mathbb{R}$, are valid choices. Then Λ can be compactly written as

$$\Lambda(t, x, u) = 1_{n_g}^\top |g(t, x, u)|_+^2 + 1_{n_h}^\top h(t, x, u)^2. \tag{2.8}$$

Remark 2. *The partial derivatives of Λ are given by*

$$\nabla_\square \Lambda = \sum_{i=1}^{n_g} \nabla q_i \nabla_\square g_i + \sum_{i=1}^{n_g} \nabla p_j \nabla_\square h_j, \tag{2.9}$$

where $\square = t$, x, and u. A consequence of the continuous differentiability of q_i and p_j, and their construction in (2.2), is that their derivatives in (2.9) are zero wherever $g_i \leq 0$ and $h_j = 0$.

Remark 3. *The penalty function Λ can be defined more generally as a vector-valued function of the form*

$$\Lambda(t, x, u) = M \begin{bmatrix} q(g(t, x, u)) \\ p(h(t, x, u)) \end{bmatrix}, \tag{2.10}$$

for any $t \in \mathbb{R}_+$, $x \in \mathbb{R}^{n_x}$, *and* $u \in \mathbb{R}^{n_u}$, *where matrix* $M \in \mathbb{R}^{n_y \times (n_g + n_h)}$, *which we refer to as the mixing matrix, has nonnegative entries, and functions* $q : \mathbb{R}^{n_g} \to \mathbb{R}^{n_g}$ *and* $p : \mathbb{R}^{n_h} \to \mathbb{R}^{n_h}$ *are defined as*

$$q((z_g^1, \ldots, z_g^{n_g})) = (q_1(z_g^1), \ldots, q_{n_g}(z_g^{n_g})) \tag{2.11a}$$

$$p((z_h^1, \ldots, z_h^{n_h})) = (p_1(z_h^1), \ldots, p_{n_h}(z_h^{n_h})) \tag{2.11b}$$

for any $z_g^i \in \mathbb{R}$ *and* $z_h^j \in \mathbb{R}$, *with* $i = 1, \ldots, n_g$ *and* $j = 1, \ldots, n_h$. *The mixing matrix has at most* $n_g + n_h$ *rows, at most one positive entry per column, and exactly* $n_g + n_h$ *positive entries in total. We obtain* (2.3) *and a scalar-valued state* y *in* (2.6b) *by choosing* $M = 1_{n_g + n_h}^\top$. *In general, choosing* (2.10) *will lead to a vector-valued state* $y \in \mathbb{R}^{n_y}$ *for quantifying the total constraint violation. While* (2.3) *is suitable for most cases, choosing* (2.10) *for certain applications can be numerically more reliable. In the remainder of the discussion, we adopt the definition in* (2.3). *However, the results also apply to the general representation in* (2.10).

2.2.4 Generalized Time-Dilation

Time-dilation is a transformation technique for posing a free-final-time optimal control problem as an equivalent fixed-final-time problem [79, Sec. III.A.1]. Let x be a state trajectory for (2.1b) over $[t_\mathrm{i}, t_\mathrm{f}]$ obtained with control input u and some initial condition. We define a strictly increasing, continuously differentiable mapping $t : [0, 1] \to \mathbb{R}_+$, with boundary conditions $t(0) = t_\mathrm{i}$, $t(1) = t_\mathrm{f}$, and treat the derivative of the map

$$s(\tau) = \frac{\mathrm{d}t(\tau)}{\mathrm{d}\tau} = \overset{\circ}{t}(\tau),$$

for $\tau \in [0, 1]$, as an additional control input, which we refer to as the *dilation factor*. Note that the derivative with respect to $\tau \in [0, 1]$ is denoted by $\overset{\circ}{\square} = \mathrm{d}\square(\tau)/\mathrm{d}\tau$. Previously, dilation factors were either treated as a single parameter decision variable [79, Sec. III.A.1] or as multiple discrete parameter decision variables [42, Sec. 2.1], [98]. The approach proposed herein generalizes them by treating dilation factor as a continuous-time control input. We

treat t as an additional state, and define *augmented control input* $\tilde{u}$ and *augmented state* $\tilde{x}$ as

$$\tilde{u}(\tau) = \big(u(t(\tau)),\, s(\tau)\big) \in \mathbb{R}^{n_u+1}, \tag{2.12a}$$

$$\tilde{x}(\tau) = \big(x(t(\tau)),\, y(t(\tau)),\, t(\tau)\big) \in \mathbb{R}^{n_x+2}, \tag{2.12b}$$

for $\tau \in [0,1]$. Then we obtain the *augmented system*, given by

$$\mathring{\tilde{x}}(\tau) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ 1 \end{bmatrix} \frac{\mathrm{d}t(\tau)}{\mathrm{d}\tau} = F(\tilde{x}(\tau), \tilde{u}(\tau)), \tag{2.13}$$

where

$$F(\tilde{x}(\tau), \tilde{u}(\tau)) = \begin{bmatrix} f(t(\tau), x(t(\tau)), u(t(\tau))) \\ \Lambda(t(\tau), x(t(\tau)), u(t(\tau))) \\ 1 \end{bmatrix} s(\tau). \tag{2.14}$$

The partial derivatives of F, which are continuous (see Remark 1), are given by

$$\nabla_{\tilde{x}} F(\tilde{x}, \tilde{u}) = s \begin{bmatrix} \nabla_x f(t,x,u) & 0_{n_x} & \nabla_t f(t,x,u) \\ \nabla_x \Lambda(t,x,u) & 0 & \nabla_t \Lambda(t,x,u) \\ & 0_{1\times n_{\tilde{x}}} & \end{bmatrix}, \tag{2.15a}$$

$$\nabla_{\tilde{u}} F(\tilde{x}, \tilde{u}) = \begin{bmatrix} s\nabla_u f(t,x,u) & f(t,x,u) \\ s\nabla_u \Lambda(t,x,u) & \Lambda(t,x,u) \\ 0_{1\times n_u} & 1 \end{bmatrix}, \tag{2.15b}$$

for any $x \in \mathbb{R}^{n_x}$, $t \in \mathbb{R}_+$, $u \in \mathbb{R}^{n_u}$, and $s \in \mathbb{R}_+$. Note that choosing (2.8) yields

$$\nabla_\square \Lambda = 2|g|_+^\top \nabla_\square g + 2h^\top \nabla_\square h,$$

for $\square = t$, x, and u.

In the subsequent development, we define $n_{\tilde{x}} = n_x + 2$ and use matrices ${}^xE$, ${}^yE$, and

${}^{t}E$ to select the first n_x elements (x), the penultimate element (y), and the last element (t), respectively, of $\tilde{x}$. Similarly, matrices ${}^{u}E$ and ${}^{s}E$ select the first n_u elements (u) and the last element (s), respectively, of $\tilde{u}$. If the system dynamics, constraint, and objective functions are time-invariant, then time is not included as a state, and if the final time is fixed, then generalized time-dilation can be skipped. Henceforth, we drop the qualifier "generalized" for brevity.

2.2.5 Reformulated Optimal Control Problem

Constraint reformulation and time-dilation transforms (2.1) into the following optimal control problem

$$\underset{\tilde{x},\,\tilde{u}}{\text{minimize}}\ \tilde{L}(\tilde{x}(1)) \tag{2.16a}$$

$$\text{subject to}\ \mathring{\tilde{x}}(\tau) = F(\tilde{x}(\tau), \tilde{u}(\tau)) \qquad \text{a.e. } \tau \in [0,1] \tag{2.16b}$$

$${}^{s}E\,\tilde{u}(\tau) > 0 \qquad \tau \in [0,1] \tag{2.16c}$$

$${}^{y}E\,(\tilde{x}(1) - \tilde{x}(0)) = 0 \tag{2.16d}$$

$$\tilde{P}(\tilde{x}(0), \tilde{x}(1)) \leq 0 \tag{2.16e}$$

$$\tilde{Q}(t_{\mathrm{i}}, \tilde{x}(0), \tilde{x}(1)) = 0 \tag{2.16f}$$

where,

$$\tilde{L}(\tilde{x}) = L({}^{t}E\,\tilde{x}, {}^{x}E\,\tilde{x}), \tag{2.17a}$$

$$\tilde{P}(\tilde{x}, \tilde{x}') = P({}^{t}E\,\tilde{x}, {}^{x}E\,\tilde{x}, {}^{t}E\,\tilde{x}', {}^{x}E\,\tilde{x}'), \tag{2.17b}$$

$$\tilde{Q}(t_{\mathrm{i}}, \tilde{x}, \tilde{x}') = \begin{bmatrix} Q(t_{\mathrm{i}}, {}^{x}E\,\tilde{x}, {}^{t}E\,\tilde{x}', {}^{x}E\,\tilde{x}') \\ {}^{t}E\,\tilde{x} - t_{\mathrm{i}} \end{bmatrix}, \tag{2.17c}$$

for any $\tilde{x}, \tilde{x}' \in \mathbb{R}^{n_{\tilde{x}}}$. Since time is treated as a state variable, its initial condition needs to be specified through $\tilde{Q}$. This is not needed in (2.1) because (constant) t_{i} is explicitly passed to functions P and Q. Furthermore, positivity of the dilation factor is ensured using (2.16c). Observe that, as a consequence of the constraint reformulation, the *path constraints are not*

explicitly imposed in (2.16); they are instead embedded within (2.16b) and (2.16d).

2.3 Parameterization and Discretization

This section describes the transformation of the (infinite-dimensional) reformulated optimal control problem (2.16) into a (finite-dimensional) nonconvex optimization problem through *parameterization* of the augmented control input, and *discretization* of (2.16b) over the interval $[0,1]$. In addition, after discretization, we introduce a relaxation to the constraint reformulation step to avoid trivial loss of constraint qualification, and analyze the consequences of the relaxation.

We parameterize the augmented control input through $\tilde{v} : [0,1] \to \mathbb{R}^{n_{\tilde{u}}}$, defined as

$$\tilde{v}(\tau) = \left(\Gamma_u(\tau) \otimes \begin{bmatrix} I_{n_u} \\ 0_{1 \times n_u} \end{bmatrix} \right) U + \begin{bmatrix} 0_{n_u \times N_s} \\ \Gamma_s(\tau) \end{bmatrix} S, \tag{2.18}$$

for $\tau \in [0,1]$, and $\Gamma_\square = \begin{bmatrix} \Gamma_\square^1 & \dots & \Gamma_\square^{N_\square} \end{bmatrix}$, for $\square = u, s$. For each $k = 1, \dots, N_\square$, $\Gamma_\square^k : [0,1] \to \mathbb{R}$ is a polynomial basis function. The expression in (2.18) can also be written compactly as $\tilde{v}(\tau) = \big((\Gamma_u(\tau) \otimes I_{n_u}) U, \Gamma_s(\tau) S \big)$. Decision variables $U \in \mathbb{R}^{n_u N_u}$ and $S \in \mathbb{R}^{N_s}$ are coefficients for parameterizing the control input and dilation factor.

After parameterizing the augmented control input, we discretize (2.16b) via its integral form over a finite grid of size N in $[0,1]$, denoted by $0 = \tau_1 < \dots < \tau_N = 1$. We treat the augmented states $\tilde{x}_k$ at the node points τ_k also as decision variables, and denote them compactly as $\tilde{X} = (\tilde{x}_1, \dots, \tilde{x}_N) \in \mathbb{R}^{n_{\tilde{x}} N}$. Next, for each $k = 1, \dots, N-1$, we define $F_k : \mathbb{R}^{n_{\tilde{x}}} \times \mathbb{R}^{n_{\tilde{x}}} \times \mathbb{R}^{n_u N_u} \times \mathbb{R}^{N_s} \to \mathbb{R}^{n_{\tilde{x}}}$ as

$$\begin{aligned} &(\tilde{x}_{k+1}, \tilde{x}_k, U, S) \mapsto \\ &\qquad \tilde{x}_{k+1} - \tilde{x}_k - \int_{\tau_k}^{\tau_{k+1}} F\big(\tilde{x}^k(\tau), \tilde{v}(\tau)\big) \mathrm{d}\tau, \end{aligned} \tag{2.19}$$

where the augmented state trajectory $\tilde{x}^k$ satisfies (2.16b) almost everywhere on $[\tau_k, \tau_{k+1}]$ with augmented control input $\tilde{v}$ and initial condition $\tilde{x}_k$. Then the discretization of (2.16b),

together with (2.16d), yields the constraints

$$F_k(\tilde{x}_{k+1}, \tilde{x}_k, U, S) = 0, \tag{2.20a}$$

$${}^{y}E\,(\tilde{x}_{k+1} - \tilde{x}_k) = 0, \tag{2.20b}$$

for $k = 1, \ldots, N-1$. The discretization in (2.20a), commonly referred to as *multiple-shooting*, is exact because there is no approximation involved in switching to the integral representation of the differential equation (2.16b), and it is numerically more stable compared to its precursor, single-shooting [99, Sec. VI.A.3].

Definition 1 ***(Continuous-Time Feasibility).*** *The triplet $\tilde{X}$, U, and S is said to be continuous-time feasible if it satisfies* (2.20).

A continuous-time feasible triplet: $\tilde{X}$, U, and S corresponds to an augmented state trajectory generated by integration of (2.16b) over $[0, 1]$ with augmented control input $\tilde{v}$ and initial condition $\tilde{x}_1$, such that it passes through the node values $\tilde{x}_k$ at τ_k, for $k = 2, \ldots, N$. Furthermore, due to (2.20b), the resulting state trajectory and control input satisfy the path constraints almost everywhere in $[t_\mathrm{i}, t_\mathrm{f}]$.

2.3.1 Constraint Qualification and Relaxation

An issue with directly imposing (2.20) is that it violates *linear independence constraint qualification* (LICQ) [100, Def. 12.4]. For each point in $\mathbb{R}^{n_{\tilde{x}}N} \times \mathbb{R}^{n_u N_u} \times \mathbb{R}^{N_s}$, we can determine the active set [100, Def. 12.1] corresponding to the constraints in (2.20). LICQ is said to hold at a point if the gradients of the constraints in the corresponding active set are linearly independent. The following result shows that LICQ is violated at all points feasible with respect to (2.20).

Lemma 2. *If $\tilde{X}$, U, and S are feasible with respect to* (2.20a) *and* (2.20b)*, then for $k = 1, \ldots, N-1$*

$${}^{y}E\,\nabla_{\tilde{x}_{k+1}} F_k(\tilde{x}_{k+1}, \tilde{x}_k, U, S) = [0_{1\times n_x}\ 1\ 0], \tag{2.21a}$$

$${}^{y}E\,\nabla_{\tilde{x}_k} F_k(\tilde{x}_{k+1}, \tilde{x}_k, U, S) = [0_{1\times n_x}\ -1\ 0], \tag{2.21b}$$

$$ {}^{y}E\,\nabla_U F_k(\tilde{x}_{k+1}, \tilde{x}_k, U, S) = 0_{1\times n_u N_u}, \tag{2.21c} $$

$$ {}^{y}E\,\nabla_S F_k(\tilde{x}_{k+1}, \tilde{x}_k, U, S) = 0_{1\times N_s}. \tag{2.21d} $$

Proof. Suppose $\tilde{X}$, U, and S satisfy (2.20a) and (2.20b). For each $k = 1, \ldots, N-1$, let $\tilde{x}^k$ be the augmented state trajectory on $[\tau_k, \tau_{k+1}]$ with augmented control input $\tilde{v}$ and initial condition $\tilde{x}_k$. Then due to Lemma 1, path constraints (2.1c) and (2.1d) are satisfied almost everywhere on the time interval $[{}^{t}E\,\tilde{x}_k, {}^{t}E\,\tilde{x}_{k+1}]$. Recall that F defined in (2.14) is continuously differentiable and that $\tilde{v}$ is piecewise continuous. This implies that

$$ \tau \mapsto \nabla_{\tilde{x}} F(\tilde{x}^k(\tau), \tilde{v}(\tau)),\ \tau \mapsto \nabla_{\tilde{u}} F(\tilde{x}^k(\tau), \tilde{v}(\tau)), $$

are piecewise continuous functions. Therefore, they are integrable and the partial derivatives of F_k are well-defined. Elements of the penultimate row of $\nabla_{\tilde{x}} F$ and $\nabla_{\tilde{u}} F$, shown in (2.15), are zeros almost everywhere on $[\tau_k, \tau_{k+1}]$ when evaluated on $\tilde{x}^k$ and $\tilde{v}$. This is because all elements of this row contain factors of the derivatives of q_i or p_j, which are zeros due to continuity (see Remark 1). Hence, only the partial derivatives of F_k with respect to $\tilde{x}_k$ and $\tilde{x}_{k+1}$ contain nonzero elements in that row, as shown in (2.21a) and (2.21b). □

Using Lemma 2, for any $\tilde{X}$, U, and S satisfying (2.20a) and (2.20b), the gradient of the left-hand-side of (2.20b), and the penultimate row of the gradient of F_k (shown in (2.21)) are identical. Hence, all feasible solutions of an optimization problem having both (2.20a) and (2.20b) as constraints will not satisfy LICQ. However, CT-SCVX relies on exact penalization of the nonconvex constraints of (2.22), which means that a constraint qualification such as LICQ is required for relating a local minimizer of (2.22) to that of the penalized problem with a finite penalty weight [101, Thm. 4.4]. So, we remedy the pathological scenario due to (2.20) by relaxing (2.20b) to an inequality with a positive constant ϵ. Then (2.16) transforms under augmented control input parameterization (2.18) and discretization (2.20) as follows

$$ \underset{\tilde{X}, U, S}{\text{minimize}} \quad \tilde{L}(\tilde{x}_N) \tag{2.22a} $$

$$\text{subject to} \quad F_k(\tilde{x}_{k+1}, \tilde{x}_k, U, S) = 0 \tag{2.22b}$$

$$ {}^{y}E\,(\tilde{x}_{k+1} - \tilde{x}_k) \leq \epsilon \tag{2.22c}$$

$$k = 1, \ldots, N-1$$

$$U \in \mathcal{U},\ S \in \mathcal{S} \tag{2.22d}$$

$$\tilde{P}(\tilde{x}_1, \tilde{x}_N) \leq 0,\ \tilde{Q}(t_{\mathrm{i}}, \tilde{x}_1, \tilde{x}_N) = 0 \tag{2.22e}$$

where $\mathcal{U} \subset \mathbb{R}^{n_u N_u}$ and $\mathcal{S} \subset \mathbb{R}^{N_s}$ are compact convex sets. A key step in any gradient-based solution method for (2.22) is to compute the partial derivatives of F_k, as described in Appendix A. We adopt the so-called variational method [98, Sec. 3.2], [97, Sec. 4.2], also referred to as inverse-free exact discretization [42, Sec. 2.3].

Remark 4. *If $\tilde{X}$, U, and S satisfy (2.22b) and (2.22c), then LICQ will not be trivially violated by the active set associated with (2.22c) and the penultimate row of (2.22b). We infer this as follows. If (2.22c) holds with strict inequality for all $k = 1, \ldots, N-1$, then the result follows immediately. However, if for some $k = 1, \ldots, N-1$, (2.22c) holds with equality, then*

$$\int_{\tau_k}^{\tau_{k+1}} {}^{s}E\,\tilde{v}(\tau)\Lambda\big({}^{t}E\,\tilde{x}(\tau), {}^{x}E\,\tilde{x}^k(\tau), {}^{u}E\,\tilde{v}(\tau)\big)\mathrm{d}\tau = \epsilon, \tag{2.23}$$

where $\tilde{x}^k$ is the augmented state trajectory with augmented control input $\tilde{v}$ and initial condition $\tilde{x}_k$. Due to piecewise continuity of the integrand in (2.23) with respect to $\tau \in [\tau_k, \tau_{k+1}]$, it implies that there exists an interval $\mathcal{I}_k \subset [\tau_k, \tau_{k+1}]$ where the integrand of (2.23) is positive, i.e., some of the path constraints are violated on the segments of $\tilde{x}^k$ and $\tilde{v}$ corresponding to interval $\mathcal{I}_k$. Therefore, unlike the case in Lemma 2, the partial derivatives of Λ evaluated on those segments of $\tilde{x}^k$ and $\tilde{v}$ will not trivially be zeros.

Remark 5. *Besides addressing the LICQ issue, relaxation (2.22c) is essential for enabling exact penalization. The combination of (2.20a) and (2.20b) implies the following constraint for each $k = 1, \ldots, N-1$*

$$ {}^{y}E\,(\tilde{x}_{k+1} - \tilde{x}_k - F_k(\tilde{x}_{k+1}, \tilde{x}_k, U, S)) = 0. \tag{2.24}$$

The effect of a nonsmooth exterior penalty function for penalizing (2.24) is suppressed due to the

positivity and continuous differentiability of Λ, thereby destroying the exactness of the penalty term (see discussion in [100, p. 513]). Therefore, the relaxation is crucial for recovering a KKT point of (2.22) through a stationary point of the penalized problem with finite penalty weight.

Remark 6. *The parameterization (2.18) for the augmented control input is general enough to subsume several control input parameterizations used in practice, such as zero- and first-order-hold, and pseudospectral polynomials in orthogonal collocation (see Appendix C). We distinguish between the parameterizations for the control input and dilation factor because the former can be represented with a piecewise polynomial as long as U is elementwise bounded (via set $\mathcal{U}$), whereas, the latter must be bounded and positive due to (2.16c) (via set $\mathcal{S}$). Whenever the choice of parameterization permits (e.g., with zero- or first-order-hold), the convex control constraints in (2.1c) and (2.1d) could be encoded with $\mathcal{U}$, i.e., the constraints are imposed only at the node points (inter-sample constraint satisfaction is guaranteed). Furthermore, requiring the augmented control input to lie in a compact set via (2.22d) is necessary for deriving a pointwise constraint violation bound, given the ϵ-relaxation in (2.22c).*

2.3.2 Pointwise Constraint Violation Bound

Given $\epsilon > 0$ and a solution for (2.22), we can construct a state trajectory x, a control input u, and a grid of size N, given by $t_{\mathrm{i}} = t_1 < \ldots < t_N = t_{\mathrm{f}}$. For each $k = 1, \ldots, N$, the time instant corresponding to τ_k is t_k, and for each $k = 1, \ldots, N-1$, the length of interval $[t_k, t_{k+1}]$ is Δt_k, with a lower bound $\Delta t_{\min}$. Next, for each $i = 1, \ldots, n_g$, $j = 1 \ldots, n_h$, and $k = 1, \ldots, N-1$, define

$$\hat{g}_i^k(\epsilon) = \operatorname*{ess\,sup}_{t \in [t_k, t_{k+1}]} |g_i(t, x(t), u(t))|_+, \tag{2.25a}$$

$$\hat{h}_j^k(\epsilon) = \operatorname*{ess\,sup}_{t \in [t_k, t_{k+1}]} |h_j(t, x(t), u(t))|, \tag{2.25b}$$

which quantify the maximum measurable violation of the path constraints in $[t_k, t_{k+1}]$. With the choice of exterior penalty functions in (2.7) we can then establish an upper bound for $\hat{g}_i^k(\epsilon)$ and $\hat{h}_j^k(\epsilon)$ as follows.

Theorem 1. *For $i = 1, \ldots, n_g$, $j = 1, \ldots, n_h$, and $k = 1, \ldots, N-1$, there exist ω_{g_i} and ω_{h_j} such that*

$$\epsilon \leq \omega_{\square}^2 \frac{\Delta t_{\min}^3}{4} \implies \hat{\square}^k(\epsilon) \leq \delta_{\square}(\epsilon) = (4\epsilon\omega_{\square})^{\frac{1}{3}}, \tag{2.26}$$

where $\square = g_i, h_j$.

Proof. See Appendix B. While the proof assumes (2.7), other valid exterior penalty functions can be handled in a similar manner. □

The bound in (2.26) is consistent because $\delta_{\square}(\epsilon)$ is strictly monotonic for $\epsilon \in \mathbb{R}_+$ and $\delta_{\square}(\epsilon) \to 0$ as $\epsilon \to 0$, which then ensures that $\hat{\square}^k(\epsilon)$ is right continuous at $\epsilon = 0$, where $\square = g_i, h_j$. We can use these bounds to select ϵ values that are numerically significant yet physically insignificant for the underlying dynamical system and constraints.

Remark 7. *The pointwise bound for the inequality path constraints specifies the amount of constraint tightening required for the solutions obtained with the relaxation (2.22c) to satisfy the constraints with no pointwise violation, i.e.,*

$$\begin{aligned} &\int_{t_k}^{t_{k+1}} |g_i(t, x(t), u(t)) + \delta_{g_i}(\epsilon)|_+^2 \mathrm{d}t \leq \epsilon, \\ \implies &\; g_i(t, x(t), u(t)) \leq 0 \quad \text{a.e. } t \in [t_k, t_{k+1}]. \end{aligned}$$

Remark 8. *The estimate of ω_{g_i} and ω_{h_j} can be conservative whenever the bound on $f(t, x(t), u(t))$ is not tight, leading to a conservative bound in (2.26). Then a solution to (2.22) can be iteratively refined until the pointwise violations are within a desired tolerance. We can successively solve (2.22) by reducing ϵ by a constant factor and warm-starting with the previous solution. This refinement process will terminate since $\delta_{\square}$ is strictly monotonic and consistent.*

The key distinction between the approach in Remark 8 and the mesh-refinement techniques for alleviating inter-sample constraint violation [102, 103] is that the former *does not add more node points to the discretization grid, i.e., N remains fixed.* Most direct methods verify the extent of inter-sample constraint violation a posteriori. In contrast, CT-SCvx specifies the extent of allowable inter-sample constraint violation within the optimization problem.

2.4 Prox-Linear Method for Numerical Optimization

CT-SCvx applies an SCP algorithm called the prox-linear method [76, 5] to an unconstrained minimization problem, obtained by transforming (2.22) via ℓ_1-exact penalization of (2.22b) and (2.22e) [100, Eq. 17.22].

2.4.1 Exact Penalization

We can construct an unconstrained minimization problem from the constrained problem (2.22) by penalizing the constraints, where the penalty weights are tunable hyperparameters. The penalty functions are said to be *exact* whenever the stationary points and local minimizers of the unconstrained problem, with a finite penalty weight, can capture the KKT points and local minimizers of the constrained problem. A widely-used class of exact penalty functions are the ℓ_1 penalty functions [100, Eq. 17.22].

Standard results on exact penalization [101, 104], [100, Chap. 17] for general constrained optimization appeared close to three decades ago, and are widely used. However, they don't directly apply for CT-SCvx since the construction of (2.22) is atypical, in that it possesses convex constraints: (2.22c), (2.22d) (represented with convex sets), and nonconvex constraints: (2.22b), (2.22e) (represented with functions). While transforming (2.22) into an unconstrained minimization, we subject only the nonconvex constraints to exact penalization, while using indicator functions to represent the convex constraints within the prox-linear method. The standard exact penalization results cannot be directly applied to this hybrid case. To the best of the authors' knowledge, an exposition on the modifications necessary to handle the hybrid case is not available in the literature and hence it is presented in this section.

Given $\gamma > 0$, function $\Theta : \mathbb{R}^{n_z} \to \mathbb{R}$ is defined as

$$\begin{aligned}\Theta_\gamma(z) = \tilde{L}(\tilde{x}_N) + \tilde{I}_{\mathcal{Z}}(z) \\ + \gamma 1^\top |\tilde{P}(\tilde{x}_1, \tilde{x}_N)|_+ + \gamma \|\tilde{Q}(t_{\mathrm{i}}, \tilde{x}_1, \tilde{x}_N)\|_1\end{aligned} \tag{2.27}$$

$$+\gamma \sum_{k=1}^{N-1} \|F_k(\tilde{x}_{k+1}, \tilde{x}_k, U, S)\|_1$$

where $n_z = n_{\tilde{x}} N + n_u N_u + N_s$, $z = (\tilde{X}, U, S)$, $\tilde{X} = (\tilde{x}_1, \ldots, \tilde{x}_N)$, and $\mathcal{Z} = \tilde{\mathcal{X}} \times \mathcal{U} \times \mathcal{S}$ is a closed convex set with

$$\tilde{\mathcal{X}} = \left\{ \begin{array}{l|l} (\tilde{x}_1, \ldots, \tilde{x}_N) & {}^{y}E\,(\tilde{x}_{k+1} - \tilde{x}_k) \leq \epsilon, \\ \in \mathbb{R}^{n_{\tilde{x}} N} & \tilde{x}_k \in \mathbb{R}^{n_{\tilde{x}}},\ k = 1, \ldots, N-1 \end{array} \right\}.$$

The ℓ_1-penalization of constraints (2.22b) and (2.22e) involves penalty functions $\square \mapsto |\square|_+$ and $\square \mapsto \|\square\|_1$, respectively.

Our goal then is to compute a local minimizer of Θ_γ where the penalty terms are zero. We start by seeking a stationary point of Θ_γ, since local minimizers are stationary points. Determining stationary points is a relatively simpler task than directly attempting to solve (2.22). We can rewrite the indicator function for $\mathcal{Z}$ as explicit convex constraints while computing a stationary point of Θ_γ via SCP. A stationary point $z^\star$ satisfies $0 \in \partial\Theta_\gamma(z^\star)$, i.e., there exists $\mu \in \mathbb{R}^{n_z}$ and $\lambda \in \mathbb{R}^{n_\lambda}$ denoted by

$$\lambda = (\lambda_1^{\tilde{P}}, \ldots, \lambda_{n_P}^{\tilde{P}}, \lambda_1^{\tilde{Q}}, \ldots, \lambda_{n_Q+1}^{\tilde{Q}}, \ldots, \lambda_1^{F_1}, \ldots, \lambda_{n_{\tilde{x}}}^{F_{N-1}}),$$

with $n_\lambda = n_P + n_Q + n_{\tilde{x}}(N-1) + 1$, such that

$$\begin{aligned} \lambda_i^{\tilde{P}} &\in \partial |\tilde{P}_i(\tilde{x}_1^\star, \tilde{x}_N^\star)|_+, && i = 1, \ldots, n_P, \\ \lambda_i^{\tilde{Q}} &\in \partial \left|\tilde{Q}_i(t_{\mathrm{i}}, \tilde{x}_1^\star, \tilde{x}_N^\star)\right|, && i = 1, \ldots, n_Q + 1, \\ \lambda_i^{F_k} &\in \partial \left|F_{ki}(\tilde{x}_{k+1}^\star, \tilde{x}_k^\star, U^\star, S^\star)\right|, && k = 1, \ldots, N-1, \\ & && i = 1, \ldots, n_{\tilde{x}}, \end{aligned}$$

$$\begin{aligned} 0_{1 \times n_z} = \nabla_z L\,(\tilde{x}_N^\star) + \mu^\top \\ + \gamma \sum_{i=1}^{n_P} \lambda_i^{\tilde{P}} \nabla_z \tilde{P}_i\,(\tilde{x}_1^\star, \tilde{x}_N^\star) \end{aligned} \tag{2.28}$$

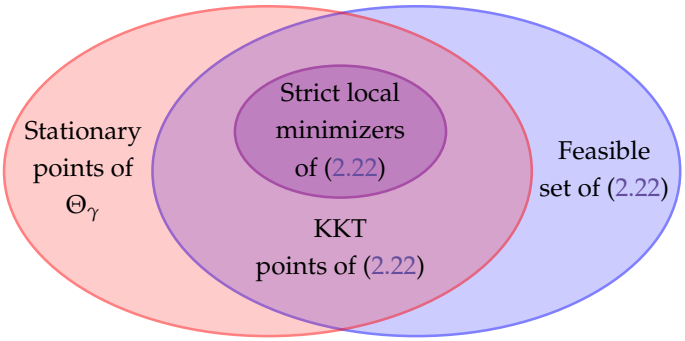


Figure 2.2: Exact penalization ensures that, for a large enough finite γ, all stationary points of Θ_γ that are feasible with respect to (2.22) are KKT points of (2.22). Furthermore, all strict local minimizers of (2.22) are stationary points of Θ_γ for a large enough γ.

$$+\gamma \sum_{i=1}^{n_Q+1} \lambda_i^{\tilde{Q}} \nabla_z \tilde{Q}_i(t_\mathrm{i}, \tilde{x}_1^\star, \tilde{x}_N^\star)$$
$$+\gamma \sum_{k=1}^{N-1} \sum_{i=1}^{n_{\tilde{x}}} \lambda_i^{F_k} \nabla_z F_{ki}(\tilde{x}_{k+1}^\star, \tilde{x}_k^\star, U^\star, S^\star),$$

where $z^\star = (\tilde{X}^\star, U^\star, S^\star)$ with $\tilde{X}^\star = (\tilde{x}_1^\star, \ldots, \tilde{x}_N^\star)$. The scalar-valued functions $\tilde{P}_i$, $\tilde{Q}_i$, and F_{ki} are elements of $\tilde{P}$, $\tilde{Q}$, and F_k, respectively, indexed by i. The gradients of $\tilde{P}_i$, $\tilde{Q}_i$, and F_{ki} are with respect to $z = (\tilde{x}_1, \ldots, \tilde{x}_N, U, S)$. The convex subdifferential of $\square \mapsto |\square|_+$ and $\square \mapsto |\square|$ are

$$\partial |w|_+ = \begin{cases} \{1\} & w > 0, \\ [0,1] & w = 0, \\ \{0\} & w < 0, \end{cases} \tag{2.29a}$$

$$\partial |w| = \begin{cases} \{1\} & w > 0, \\ [-1,1] & w = 0, \\ \{-1\} & w < 0, \end{cases} \tag{2.29b}$$

for any $w \in \mathbb{R}$. We refer to [88, Sec. 3.2] and the references therein, for the definition and properties of the subdifferential of a nonconvex function. Next, we need the following lemma about the normal cone of a convex set.

Lemma 3. *Suppose that $z^\star \in \mathbb{R}^n$, convex set $\mathcal{C} = \{z \in \mathbb{R}^n \,|\, \check{\Omega}_i(z) \leq 0,\ i = 1, \ldots, N_\mathcal{C}\}$ is defined with continuously differentiable convex functions $\check{\Omega}_i : \mathbb{R}^n \to \mathbb{R}$, and convex set*

$\mathcal{D} = \{z \in \mathbb{R}^n \mid a_j^\top z + b_j = 0,\ j = 1, \ldots, N_\mathcal{D}\}$ is defined with $a_j \in \mathbb{R}^n$ and $b_j \in \mathbb{R}$. Assume that: (i) the collection of vectors $\nabla \check{\Omega}_i(z^\star)$, for $i \in I_\mathcal{C}(z^\star)$, and a_j, for $j = 1, \ldots, N_\mathcal{D}$, are linearly independent, where $I_\mathcal{C}(z^\star) = \{i \mid \check{\Omega}_i(z^\star) = 0\}$ is an index set; and (ii) the intersection of $\mathcal{D}$ and the interior of $\mathcal{C}$ is nonempty. Then we have

$$\mathcal{N}_\mathcal{C}(z^\star) = \left\{ \sum_{i \in I_\mathcal{C}} \lambda_i \nabla \check{\Omega}_i(z^\star)^\top \,\middle|\, \lambda_i \in \mathbb{R}_+ \right\}, \tag{2.30a}$$

$$\mathcal{N}_\mathcal{D}(z^\star) = \left\{ \sum_{j=1}^{N_\mathcal{D}} \lambda_j a_j \,\middle|\, \lambda_j \in \mathbb{R} \right\}, \tag{2.30b}$$

$$\mathcal{N}_{\mathcal{C}\cap\mathcal{D}}(z^\star) = \mathcal{N}_\mathcal{C}(z^\star) + \mathcal{N}_\mathcal{D}(z^\star). \tag{2.30c}$$

where the set addition in (2.30c) is a Minkowski sum.

Proof. The construction of $\mathcal{N}_\mathcal{C}(z^\star)$ and $\mathcal{N}_\mathcal{D}(z^\star)$ follow directly from [105, Thm. 10.39, Cor. 10.44, Thm. 10.45], and the expression for $\mathcal{N}_{\mathcal{C}\cap\mathcal{D}}(z^\star)$ is derived from [106, E.g. 3.5] and [105, Thm. 4.10], where we use the fact that $\partial \tilde{I}_\square(z) = \mathcal{N}_\square(z)$ for any $z \in \mathbb{R}^n$, with $\square = \mathcal{C}, \mathcal{D}$. □

Remark 9. *Given a convex optimization problem with a feasible set described by $\mathcal{C} \cap \mathcal{D}$ in Lemma 3, the corresponding assumptions are regularity conditions similar to LICQ and strong Slater's assumption [92, Def. 2.3.1], which are typically needed for a well-posed optimization problem. We require the assumptions in Lemma 3 on set $\mathcal{Z}$ since the proposed SCP approach involves a sequence of convex problems with $\mathcal{Z}$ as the feasible set.*

Then we have the following result relating the stationary points of (2.27) and KKT points of (2.22).

Theorem 2. *Suppose that $\mathcal{Z}$ and $z^\star$ satisfy the assumptions of Lemma 3. If $z^\star$ is a stationary point of Θ_γ and is feasible with respect to (2.22), then $z^\star$ is a KKT point of (2.22). Conversely, if $z^\star$ is a KKT point of (2.22) and γ is sufficiently large, then $z^\star$ is also a stationary point of Θ_γ.*

Proof. Suppose that $z^\star$ is feasible with respect to (2.22), where $z^\star = (\tilde{X}^\star, U^\star, S^\star)$ with $\tilde{X}^\star = (\tilde{x}_1^\star, \ldots, \tilde{x}_N^\star)$. Note that $z^\star$ is a KKT point of (2.22) if and only if there exists $\mu \in \mathcal{N}_\mathcal{Z}(z^\star)$

(using Lemma 3) and $\lambda = (\lambda_1, \ldots, \lambda_{n_\lambda}) \in \mathbb{R}^{n_\lambda}$ such that

$$\nabla_z L\left(\tilde{x}_N^\star\right) + \mu^\top + \lambda^\top \begin{bmatrix} \nabla_z \tilde{P}(\tilde{x}_1^\star, \tilde{x}_N^\star) \\ \nabla_z \tilde{Q}(t_{\mathrm{i}}, \tilde{x}_1^\star, \tilde{x}_N^\star) \\ \nabla_z F_1(\tilde{x}_2^\star, \tilde{x}_1^\star, U^\star, S^\star) \\ \vdots \\ \nabla_z F_{N-1}(\tilde{x}_N^\star, \tilde{x}_{N-1}^\star, U^\star, S^\star) \end{bmatrix} = 0_{1\times n_z}, \tag{2.31a}$$

$$\lambda_i \tilde{P}_i(\tilde{x}_1^\star, \tilde{x}_N^\star) = 0, \tag{2.31b}$$

for $i = 1, \ldots, n_P$. Similar to (2.28), the gradients of $\tilde{P}$, $\tilde{Q}$, and F_k in (2.31a) are with respect to $z = (\tilde{x}_1, \ldots, \tilde{x}_N, U, S)$. If $z^\star$ is a stationary point of Θ_γ, then there exists $\mu \in \mathcal{N}_{\mathcal{Z}}(z^\star)$ and $\hat{\lambda} = (\hat{\lambda}_1, \ldots, \hat{\lambda}_{n_\lambda}) \in \mathbb{R}^{n_\lambda}$ such that (2.31a) holds, where

$$\begin{aligned} \frac{1}{\gamma}\hat{\lambda} \in \partial|\tilde{P}_1|_+ \times \ldots \times \partial|\tilde{P}_{n_P}|_+ \times \partial|\tilde{Q}_1| \times \ldots \\ \times \partial|\tilde{Q}_{n_Q+1}| \times \partial|F_{11}| \times \ldots \times \partial|F_{(N-1)n_{\tilde{x}}}|. \end{aligned} \tag{2.32}$$

The arguments of $\tilde{P}_i$, $\tilde{Q}_i$, and F_{ki}, which are elements of $z^\star$, are omitted in (2.32) for brevity. Observe that $\hat{\lambda}$ satisfies (2.31b) due to the feasibility of $z^\star$ for (2.22) and due to the definition of $\partial|\square|_+$ in (2.29). Therefore, $z^\star$ is a KKT point of (2.22).

Conversely, if $z^\star$ is a KKT point of (2.22), then there exists $\mu \in \mathcal{N}_{\mathcal{Z}}(z^\star)$ and $\lambda = (\lambda_1, \ldots, \lambda_{n_\lambda})$ such that (2.31a) and (2.31b) hold. Note that $z^\star$ is feasible with respect to (2.22) and suppose that $\gamma > \|\lambda\|_\infty$. Then we obtain $\lambda_{n_P+i}/\gamma \in \partial|\tilde{Q}_i(t_{\mathrm{i}}, \tilde{x}_1^\star, \tilde{x}_N^\star)| = [-1, 1]$, for $i = 1, \ldots, n_Q + 1$, and $\lambda_{n_P+n_Q+n_{\tilde{x}}(k-1)+i+2}/\gamma \in \partial|F_{ki}(\tilde{x}_{k+1}^\star, \tilde{x}_k^\star, U^\star, S^\star)| = [-1, 1]$, for $k = 1, \ldots, N-1$, $i = 1, \ldots, n_{\tilde{x}}$. Furthermore, for $i = 1, \ldots, n_P$, we have the following. If $\tilde{P}_i(\tilde{x}_1^\star, \tilde{x}_N^\star) < 0$, then $\lambda_i/\gamma \in \partial|\tilde{P}_i(\tilde{x}_1^\star, \tilde{x}_N^\star)|_+ = \{0\}$. Alternatively, if $\tilde{P}_i(\tilde{x}_1^\star, \tilde{x}_N^\star) = 0$, then $0 \leq \lambda_i \leq \gamma$, i.e., $\lambda_i/\gamma \in \partial|\tilde{P}_i(\tilde{x}_1^\star, \tilde{x}_N^\star)|_+ = [0, 1]$. Therefore, due to (2.31a) and (2.32), $z^\star$ is a stationary point of Θ_γ for a large enough γ. □

Next we relate the local minimizers of (2.22) to the stationary points and local minimiz-

ers of Θ_γ.

Theorem 3. *Suppose LICQ holds at $z^\star$ and it is a strict local minimizer of (2.22). Then there exists $\bar{\gamma} > 0$ such that for all $\gamma \geq \bar{\gamma}$, $z^\star$ is a local minimizer of Θ_γ. Moreover, there exist constants $\tilde{\gamma} > 0$ and $\tilde{\sigma} > 0$ such that, for all $\gamma \geq \tilde{\gamma}$, if z is a stationary point of Θ_γ satisfying $\|z - z^\star\| \leq \tilde{\sigma}$, then z is feasible with respect to (2.22).*

Proof. Consider a modification of Θ_γ, denoted by $\tilde{\Theta}_\gamma$, where convex constraints due to $\mathcal{Z}$ are also ℓ_1-penalized instead of using the indicator function. The first statement follows from [101, Thm. 4.4] if we first replace Θ_γ with $\tilde{\Theta}_\gamma$, and note that $\tilde{\Theta}_\gamma(z) \leq \Theta_\gamma(z)$, with equality iff $z \in \mathcal{Z}$. The last statement follows from [104, Prop. 3.1, 3.3]. □

Stronger versions of the exact penalization result also exist. For instance, it can be shown under certain assumptions that, given a local minimizer of (2.22), there exists a neighborhood around it where the stationary points of Θ_γ are also local minimizers of (2.22). We refer the reader to [107] for further details.

2.4.2 Prox-Linear Method

We use the prox-linear method [5, Sec. 5] developed for convex-composite minimization to compute stationary points of Θ_γ. The prox-linear method is applicable when Θ_γ can be fit into the following template

$$\Theta_\gamma(z) = J(z) + H(G(z)), \tag{2.33}$$

where J is a proper closed convex function, H is an α-Lipschitz continuous convex function, and G is nonconvex and continuously differentiable with a β-Lipschitz continuous gradient. The prox-linear method finds a stationary point $z^\star$ of Θ_γ by iteratively minimizing a convex approximation for Θ_γ given by

$$\begin{aligned}\Theta_\gamma^\rho(z; z_k) = J(z) + H\big(G(z_k) + \nabla G(z_k)(z - z_k)\big) \\ + \frac{1}{2\rho}\|z - z_k\|^2,\end{aligned} \tag{2.34}$$

where z_k is the current iterate, with $k \geq 1$, and ρ determines the proximal term weight. The nonconvex term $H(G(z))$ is convexified by linearizing G at z_k. We denote the unique minimizer [91, Sec. 9.1.2] of the strongly convex function $z \mapsto \Theta_\gamma^\rho(z; z_k)$ as z_{k+1}. The iterative minimization of Θ_γ^ρ amounts to solving a sequence of convex subproblems, which is numerically very efficient and reliable. The convergence of this iterative approach is quantified using the prox-gradient mapping

$$\mathcal{G}_\rho(z) = \frac{1}{\rho}\big(z - \operatorname*{argmin}_{z'} \Theta_\gamma^\rho(z'; z)\big).$$

This is because $\mathcal{G}_\rho(z) = 0$ iff z is a stationary point of Θ_γ [88, Sec. 3.3]. Observe that Θ_γ defined in (2.27) fits the template of (2.33): the terminal cost function $\tilde{L}$ and the exact penalties for $\tilde{P}$, $\tilde{Q}$, and F_k are represented by the composition $H \circ G$, and the indicator function $\tilde{I}_{\mathcal{Z}}$ is represented by J. More precisely, for any $z = (\tilde{x}_1, \ldots, \tilde{x}_N, U, S) \in \mathbb{R}^{n_z}$, we choose

$$J(z) = \tilde{I}_{\mathcal{Z}}(z), \tag{2.35a}$$

$$H(\zeta) = \zeta_{\tilde{L}} + 1^\top |\zeta_{\tilde{P}}|_+ + \|(\zeta_{\tilde{Q}}, \zeta_F)\|_1, \tag{2.35b}$$

$$\begin{aligned} G(z) = \big(&\tilde{L}(\tilde{x}_N),\ \gamma\tilde{P}(\tilde{x}_1, \tilde{x}_N),\ \gamma\tilde{Q}(t_\mathrm{i}, \tilde{x}_1, \tilde{x}_N), \\ &F_1(\tilde{x}_2, \tilde{x}_1, U, S), \ldots, F_{N-1}(\tilde{x}_N, \tilde{x}_{N-1}, U, S)\big), \end{aligned} \tag{2.35c}$$

where $\zeta = (\zeta_{\tilde{L}}, \zeta_{\tilde{P}}, \zeta_{\tilde{Q}}, \zeta_F)$, with $\zeta_{\tilde{L}} \in \mathbb{R}$, $\zeta_{\tilde{P}} \in \mathbb{R}^{n_P}$, $\zeta_{\tilde{Q}} \in \mathbb{R}^{n_Q+1}$, and $\zeta_F \in \mathbb{R}^{n_{\tilde{x}}(N-1)}$. While minimizing (2.34), we translate $\tilde{I}_{\mathcal{Z}}$ into explicit convex constraints that can be handled by convex optimization solvers such as ECOS [10] and PIPG [24].

Lemma 4. *Under Assumption 1, the gradients of $\tilde{L}$, $\tilde{P}$, $\tilde{Q}$, and F_k are bounded on on $\mathcal{Z}$.*

Proof. Note that functions $\tilde{L}$, $\tilde{P}$, $\tilde{Q}$, and F_k are continuously differentiable, and the domain of Θ_γ is $\mathcal{Z}$. From Assumption 1 and [94, Chap. 4, Prop. 1.4], state trajectories generated with a bounded control input and dilation factor (due to (2.22d)) are bounded. Boundedness of the dilation factor also ensures that t_f is bounded. Then there exists a constant, β, that bounds the norm of the gradients of $\tilde{L}$, $\tilde{P}$, $\tilde{Q}$, and F_k on $\mathcal{Z}$. □

Figure 2.3 describes CT-SCVX and Algorithm 1 shows the prox-linear method. The inputs

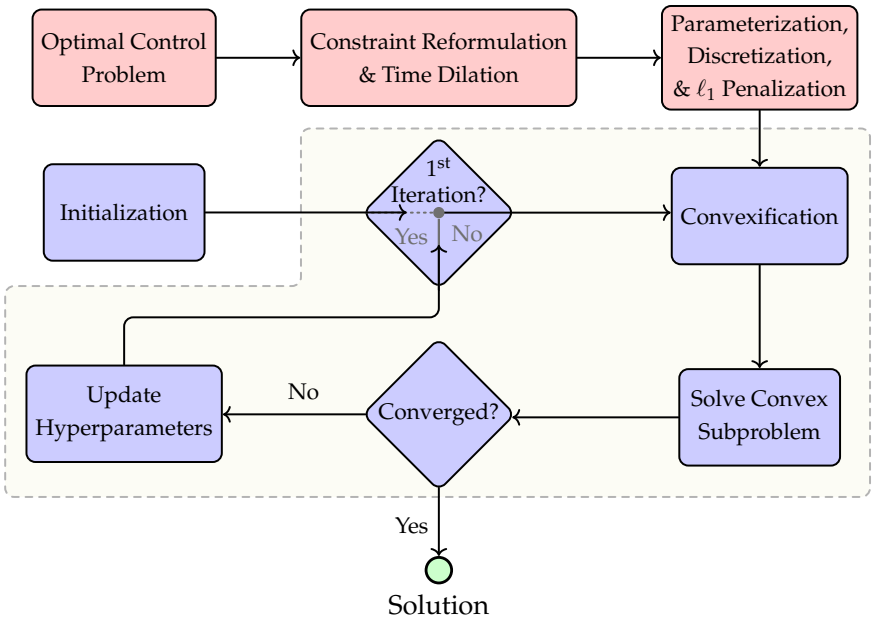


Figure 2.3: The proposed successive convexification framework: CT-SCVX. The ▢ blocks show the construction of (2.22) and (2.27), the ▢ blocks show the components of the prox-linear method, and the ▢ block demarcates the iterative parts.

to the algorithm are the maximum number of iterations, $k_{\max}$, the termination tolerance, ε, and the proximal term weight, ρ. Note that the "Convexification" block in Figure 2.3 constructs the subproblem based on (2.34), which forms a convex approximation of the composition $H \circ G$. Since the convex approximation is non-affine based on our choice of H, we call this step "Convexification" rather than "Linearization". Furthermore, this step computes the gradient of the discretized dynamics (2.20a), as described in Section A.

Algorithm 1 Prox-linear Method

Input: $k_{\max}$, ε, ρ
Initialize: z_1
  $k \leftarrow 1$
  while $k \leq k_{\max}$ **and** $\left\|\mathcal{G}_\rho(z_1)\right\| > \varepsilon$ **do**
    $z_{k+1} \leftarrow \underset{z}{\operatorname{argmin}}\ \Theta_\gamma^\rho(z; z_k)$
    $k \leftarrow k + 1$
  end while
Output: z_k

We have the following result about the monotonic decrease of Θ_γ for a sufficiently small ρ.

Theorem 4 *(Lemma 5.1 of [5])*. *At iteration k of Algorithm 1, the following holds*

$$\Theta_\gamma(z_k) \geq \Theta_\gamma(z_{k+1}) + \frac{\rho}{2}(2 - \alpha\beta\rho)\|\mathcal{G}_\rho(z_k)\|^2.$$

Therefore, $\Theta_\gamma(z_k)$ is monotonically decreasing if $\rho \leq 1/\alpha\beta$.

The magnitude of $\mathcal{G}_\rho(z_k)$ is both a practical and rigorous measurement to assess if z_k is close to a stationary point (see [5, Thm. 5.3] for details). We then have the following finite-stop corollary.

Corollary 2. *Suppose Θ_γ is bounded below by $\Theta_\gamma^\star$ and $\rho \leq 1/\alpha\beta$. For a given tolerance ε, prox-linear method achieves*

$$\|\mathcal{G}_\rho(z_k)\|^2 < \varepsilon \quad \textit{for } k \leq \frac{2\alpha\beta(\Theta_\gamma(z_1) - \Theta_\gamma^\star)}{\varepsilon}.$$

Remark 10. *In practice, β can be computed by either implementing a line-search, as described in [5, Alg. 1], or by adopting the adaptive weight update strategy in [77, Alg. 2.2], where the equivalence between the trust-region and prox-linear methods is also discussed. The "Update Hyperparameters" block in Figure 2.3 represents the use of such update techniques. The implementation of the prox-linear method can be further enhanced using the acceleration scheme described in [88, Sec. 7].*

Note that the upper bound for ρ in Theorem 4 is a sufficient condition for convergence to a stationary point. In general, if the prox-linear method converges with arbitrary user-specified weights ρ and γ, then the converged solution is guaranteed to be a stationary point, due to [5, Thm. 5.3]. In summary, the prox-linear method enables global convergence of CT-SCVX to a stationary point of Θ_γ (Theorem 4), and if the stationary point is feasible with respect to (2.22), then it is a KKT point (Theorem 2). Furthermore, each strict local minimizer of (2.22) with LICQ satisfied is a local minimizer of Θ_γ for a large-enough γ (Theorem 3).

Remark 11. *The prox-linear method is susceptible to the crawling phenomenon exhibited by SCP algorithms [108], wherein the iterates make vanishingly small progress towards a stationary point when they are not close to one. This phenomenon can happen when γ is chosen to be very large, which decreases the upper bound on ρ for ensuring monotonic decrease of Θ_γ. Then the lower*

bound $\frac{1}{2\alpha\beta}\|\mathcal{G}_{\frac{1}{\alpha\beta}}(z_k)\|$ *for* $\Theta_\gamma(z_k) - \Theta_\gamma(z_{k+1})$ *becomes very small even if* $\mathcal{G}_{\frac{1}{\alpha\beta}}(z_k)$ *is not small. The crawling phenomenon could be remedied by scaling the decision variables and the constraint functions so that their orders of magnitude are similar.*

Remark 12. *The penalized trust region (PTR) SCP algorithm in [79, 109] solves a sequence convex subproblems similar to the ones generated by the prox-linear method. The PTR algorithm also applies a class of time-dilation, discretization, and parameterization operations to the optimal control problem (2.1), albeit in an order that is different from what we propose. The resulting sequence of convex subproblems that PTR solves can fit within the template of (2.33) after minor modifications. The weights of the proximal and exact penalty terms are heuristically selected to obtain good convergence behavior.*

2.5 Exploiting Convexity

Next we consider a fixed-final-time optimal control problem (2.1) where the terminal cost function L is convex in the terminal state, the dynamical system (2.1b) is a linear nonhomogeneous ordinary differential equation, given by

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) + w(t), \tag{2.36}$$

for $t \in [t_\mathrm{i}, t_\mathrm{f}]$, and the constraint functions in (2.1c), (2.1d), (2.1e), and (2.1f) are jointly convex in all arguments (except time). In particular, constraint functions h and Q are affine in state and control input. Furthermore, we deviate from the Mayer form by explicitly specifying a running cost term in the objective with a continuously differentiable function $Y : \mathbb{R}_+ \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ that is jointly convex in the state and control input. Embedding the running cost into the dynamics, as mentioned in Section 2.2.2, would turn the dynamical system nonlinear and hence destroy the convexity of the optimal control problem.The resulting optimal control problem is now in the Bolza form [49, Sec. 3.3.2].

To obtain a finite-dimensional optimization problem similar to (2.22), we first parameterize the control input and discretize (2.36). Time-dilation is not necessary in the fixed-final-time setting, and constraint reformulation is postponed to the end with a minor

modification to exploit convexity of the constraints. We parameterize the control input similar to (2.18), as $\nu(t) = (\Gamma_u(t) \otimes I_{n_u})U$, for $t \in [t_\mathrm{i}, t_\mathrm{f}]$. To discretize (2.36), we form a grid of size N in $[t_\mathrm{i}, t_\mathrm{f}]$ denoted by $t_\mathrm{i} = t_1 < \ldots < t_N = t_\mathrm{f}$, and treat the states x_k at node points t_k as decision variables. Then for each $k = 1, \ldots, N-1$, we exactly discretize (2.36) via the integral form into

$$x^k(t) = \Phi_x(t, t_k)x_k + \Phi_u(t, t_k)U + \phi(t, t_k), \tag{2.37}$$

for $t \in [t_k, t_{k+1}]$, where the state trajectory x^k with control input ν and initial condition x_k satisfy (2.36) a.e. $t \in [t_k, t_{k+1}]$. Moreover, $t \mapsto \Phi_x(t, t_k)$, $t \mapsto \Phi_u(t, t_k)$, and $t \mapsto \phi(t, t_k)$ are computed as the solution to an initial value problem[1] similar to (A.1), and we denote their terminal values as $A_k = \Phi_x(t_{k+1}, t_k)$, $B_k = \Phi_u(t_{k+1}, t_k)$, and $w_k = \phi(t_{k+1}, t_k)$. Next, the path constraints (2.1c) and (2.1d) are reformulated as follows. For each $k = 1, \ldots, N-1$, define $\tilde{g}_k : \mathbb{R}_+ \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u N_u} \to \mathbb{R}^{n_g}$ and $\tilde{h}_k : \mathbb{R}_+ \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u N_u} \to \mathbb{R}^{n_h}$ by substituting in the control parameterization and the discretized form (2.37) into g and h, as follows

$$\begin{aligned} &\tilde{\square}_k(t, x_k, U) = \\ &\square\big(t, \Phi_x(t, t_k)x_k + \Phi_u(t, t_k)U + \phi(t, t_k), \nu(t)\big), \end{aligned} \tag{2.38}$$

where $\square = g, h$. Note that each scalar-valued element of $\tilde{g}_k$ and $\tilde{h}_k$ is jointly convex in x_k and U. Then for each $k = 1, \ldots, N-1$, we obtain

$$\begin{aligned} &\tilde{g}_k(t, x_k, U) \leq 0,\ \tilde{h}_k(t, x_k, U) = 0,\ \forall\, t \in [t_k, t_{k+1}], \\ \iff\ &\xi_k(x_k, U) \leq 0, \end{aligned} \tag{2.39}$$

where $\xi_k : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u N_u} \to \mathbb{R}_+$ is defined as

$$\xi_k(x_k, U) = \int_{t_k}^{t_{k+1}} 1^\top |\tilde{g}_k(t, x_k, U)|_+^2 + 1^\top \tilde{h}_k(t, x_k, U)^2 \mathrm{d}t$$

[1] Note that, unlike in (A.1), superscript "k" is absent from Φ_x, Φ_u, and ϕ since they are independent of the state trajectory x^k.

Similar to (2.22c), we need to relax (2.39) with $\epsilon > 0$ in order to facilitate exact penalization.

Lemma 5. *ξ_k is a convex function.*

Proof. Observe that $\square \mapsto \max\{0, \square\}^2$ is a convex, non-decreasing function. Then the composition $1^\top |\tilde{g}_k|_+^2$ is a convex function [91, Sec. 3.2.4]. Furthermore, $1^\top \tilde{h}_k^2$ is a sum of convex quadratics of its arguments since $\tilde{h}_k$ is an affine function. □

For each $k = 1, \ldots, N-1$, the running cost over $[t_k, t_{k+1}]$ can be compactly expressed with function $\Upsilon_k : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u N_u} \to \mathbb{R}$ by substituting in the control parameterization and the discretized form (2.37) into Υ, as follows

$$\Upsilon_k(x_k, U) = \int_{t_k}^{t_{k+1}} \Upsilon(t, x^k(t), \nu(t)) \mathrm{d}t. \tag{2.40}$$

Then the convex optimal control problem subject to parameterization, discretization, and reformulation (2.39) of path constraints with ϵ-relaxation is given by

$$\begin{aligned}
\underset{X,U}{\text{minimize}} \quad & L(t_\mathrm{f}, x_N) + \sum_{k=1}^{N-1} \Upsilon_k(x_k, U) && \text{(2.41a)}\\
\text{subject to} \quad & x_{k+1} = A_k x_k + B_k U + w_k && \text{(2.41b)}\\
& \xi_k(x_k, U) \leq \epsilon && \text{(2.41c)}\\
& k = 1, \ldots, N-1 &&\\
& U \in \mathcal{U} && \text{(2.41d)}\\
& P(t_\mathrm{i}, x_1, t_\mathrm{f}, x_N) \leq 0 && \text{(2.41e)}\\
& Q(t_\mathrm{i}, x_1, t_\mathrm{f}, x_N) = 0 && \text{(2.41f)}
\end{aligned}$$

where $X = (x_1, \ldots, x_N) \in \mathbb{R}^{n_x N}$ and the compact convex set $\mathcal{U}$ plays a role similar to that in Remark 6. Although $\xi_k(x_k, U) \leq \epsilon$ is a convex constraint, we do not possess any structural or geometric information needed to classify it as a canonical conic constraint supported in the existing convex optimization solvers; we only have access to first-order oracles for Υ_k and ξ_k, that provide the function value and its gradient at a query point. Consequently, we adopt the prox-linear method to solve (2.41) after exactly penalizing

(2.41c).

We assume that representation (2.41b)-(2.41f) for the feasible set of (2.41) satisfies the strong Slater's assumption [92, Def. 2.3.1], which is necessary and sufficient for the set of Lagrange multipliers associated with a minimizer of (2.41) to be nonempty, compact, and convex [92, Thm. 2.3.2]. We will show that the convexity of (2.41) guarantees global convergence of the prox-linear method to the set of minimizers.

2.5.1 Exact Penalization & Prox-Linear Method

Similar to (2.27), for a given $\gamma > 0$, consider the convex function $\Theta_\gamma : \mathbb{R}^{n_x N + n_u N_u} \to \mathbb{R}$ defined by

$$\begin{aligned}\Theta_\gamma(z) = L(x_N) + \sum_{k=1}^{N-1} Y_k(x_k, U) \\ + \tilde{I}_{\mathcal{Z}}(z) + \gamma \sum_{k=1}^{N-1} |\xi_k(x_k, U) - \epsilon|_+\end{aligned} \tag{2.42}$$

where $z = (X, U)$ and $\mathcal{Z} = \mathcal{X} \times \mathcal{U}$ with

$$\mathcal{X} = \left\{ X \in \mathbb{R}^{n_x N} \;\middle|\; \begin{array}{l} X = (x_1, \ldots, x_N),\ x_k \in \mathbb{R}^{n_x} \\ x_{k+1} = A_k x_k + B_k U + w_k \\ 1 \le k \le N-1 \\ P(t_\mathrm{i}, x_1, t_\mathrm{f}, x_N) = 0 \\ Q(t_\mathrm{i}, x_1, t_\mathrm{f}, x_N) \le 0 \end{array} \right\},$$

is the convex set encoding all state constraints except for (2.41c), which is subject to ℓ_1 penalization. Due to the convexity of (2.41), we can invoke a stronger exact penalization result than those in Section 2.5.1.

Theorem 5. *Suppose strong Slater's assumption holds for the constraints in* (2.41)*. The set of stationary points of* (2.42) *coincides with the set of minimizers of* (2.41) *when* γ *is larger than the largest magnitude Lagrange multiplier associated with the minimizers of* (2.41)*.*

Proof. The result follows from [92, Cor. 3.2.3, Thm. 3.2.4]. Note that strong Slater's

assumption ensures that strong duality holds. □

Similar to Lemma 4, we can show that the gradients of Y_k and ξ_k are bounded on $\mathcal{Z}$. Then the prox-linear method applies to (2.42) and shows the following property.

Theorem 6. *Suppose strong Slater's assumption holds for the constraints in* (2.41) *and* $\rho \leq 1/\alpha\beta$. *Then Algorithm 1 globally converges arbitrarily close to a solution of* (2.41) *in a finite number of iterations.*

Proof. The monotonic decrease of (2.42) guaranteed by Theorem 4 when $\rho \leq 1/\alpha\beta$, together with Corollary 2 and [5, Thm. 5.3], implies that Algorithm 1 will get arbitrarily close to a stationary point in a finite number of iterations. Since stationary points of (2.42) are identical to the minimizers of (2.41), due to Theorem 5, we have that Algorithm 1 will get arbitrarily close to a minimizer of (2.41) in a finite number of iterations. □

Even though prox-linear method will globally converge to a minimizer of (2.41), providing a good initial point z_1 will dramatically improve its practical convergence behavior. A solution to the closely related convex optimization problem, where all path constraints are explicitly imposed as conic constraints at the discretization nodes, will serve as a good initialization. Computing such a solution is very efficient in practice with existing convex solvers.

2.6 Numerical Results

We provide a numerical demonstration of CT-SCVX with three examples: dynamic obstacle avoidance, six-degree-of-freedom (6-DoF) rocket landing, and 3-DoF rocket landing with lossless convexification. The first two examples are nonconvex problems while the third one is convex. The results here are generated with code in the following repository.

`https://github.com/UW-ACL/ct-scvx`

Appendices D.1.1 through D.1.3 describe the problem formulation (in the form of (2.1)) and the parameter values for each of the examples. The convex subproblems of prox-linear method are quadratic programs (QP), which are solved using the PIPG algorithm [24]. The proximal term weight ρ is determined using a line search and the exact penalty weight γ is determined by gradually increasing it by factors of 10 (see the discussion in [3, p. 78]). The

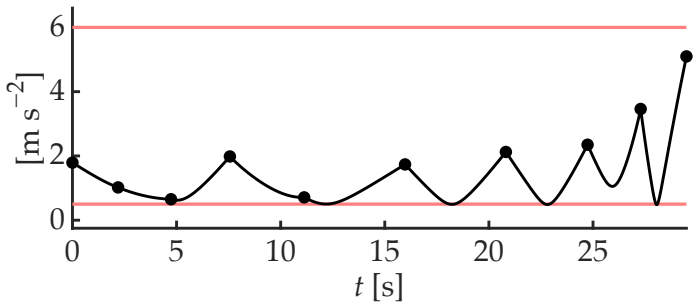
initial guess z_1 for the prox-linear method consists of a linear interpolation between the initial and final augmented states, and a constant profile for the augmented control input. The nodes of the discretization grid in $[0, 1]$ for free-final-time problems, and in $[t_\mathrm{i}, t_\mathrm{f}]$ for fixed-final-time problems, are uniformly spaced. To ensure reliable numerical performance, all decision variables and constraint functions g_i and h_j are scaled [109, Sec. IV.C.2] to similar orders of magnitude (see [48, Sec. 4.8] for a discussion on scaling). We compare solutions from CT-SCVX with those from an implementation where path constraints are not reformulated (i.e., they are only imposed at the discretization time nodes), which we refer to as the *node-only* approach. We also demonstrate the real-time capability of CT-SCVX using a C implementation generated by SCVXGEN, a general-purpose SCP-based trajectory optimization software with customized code generation, in tandem with ECOS [10].

We adopt the following convention in the plots: black for CT-SCVX and blue for node-only approach. Dots denote the solution at discretization nodes, and lines denote the continuous-time control input and state trajectory (obtained by integrating (2.1b)). Red lines denote the constraint bounds, and the inter-sample violation in the node-only solutions is highlighted with magnified inset.

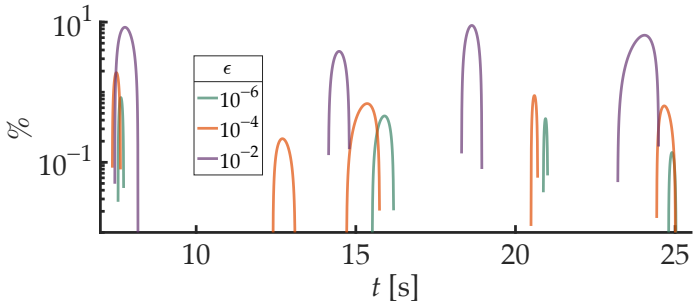
2.6.1 Dynamic Obstacle Avoidance

We consider two scenarios: one where the obstacles exhibit periodic motion and the other where they are static. Figure 2.4a shows the magnitude of control input in the case with dynamic obstacles obtained with CT-SCVX (an animation of the position trajectory is provided in the code repository). The corresponding node-only solution is not shown in Figure 2.4a, since the resulting state trajectory is physically meaningless with the same discretization grid due to aliasing in the obstacle motion. The time variation in the obstacle positions necessitates a denser discretization grid in the node-only approach. As a result, however, the number of distinct obstacle avoidance constraints imposed grows with the discretization grid size and the number of obstacles, which is computationally expensive. On the other hand, the number of constraints imposed in CT-SCVX only depends on the discretization grid size. Figure 2.4b shows the percentage of violation in the obstacle

avoidance constraint by the state trajectories obtained from CT-SCVX for different values of ϵ. As long as the constraint functions are well-scaled, a state trajectory with physically insignificant continuous-time constraint violation can be obtained by picking ϵ to be several orders of magnitude greater than machine precision, which is essential for reliable numerical performance. In practice, picking ϵ close to 10^{-4} ensures that the continuous-time constraint violation does not exceed 1%.



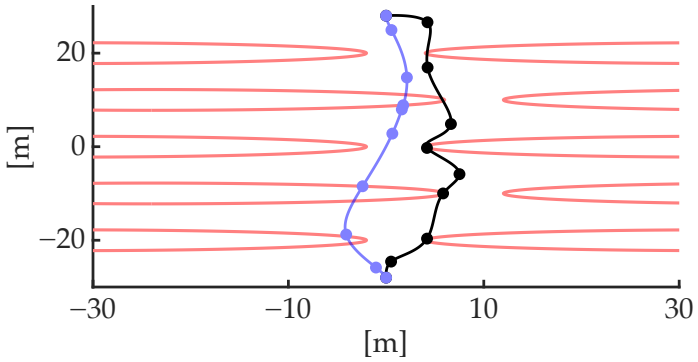
(a) Control input magnitude



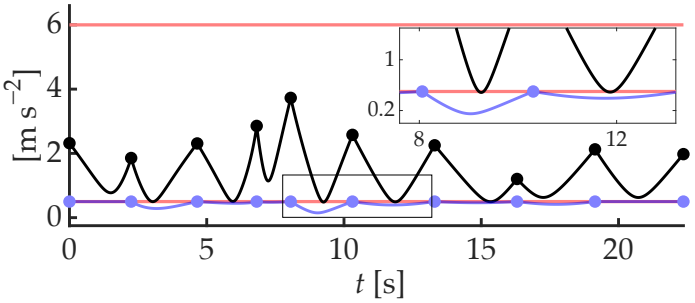
(b) Constraint violation

Figure 2.4: Dynamic obstacle avoidance

To compare CT-SCVX and the node-only approach, we consider static obstacles: Figure 2.5a shows the position and Figure 2.5b shows the control input magnitude. Note that the control input magnitude in the node-only solution is smaller than the lower bound (strictly smaller between the discretization nodes). As a result, the terminal state cost with the node-only approach, 5.06, is significantly smaller than that from CT-SCVX, 47.91—the node-only approach optimizes the cost at the expense of inter-sample constraint violation.



(a) Position



(b) Control input magnitude

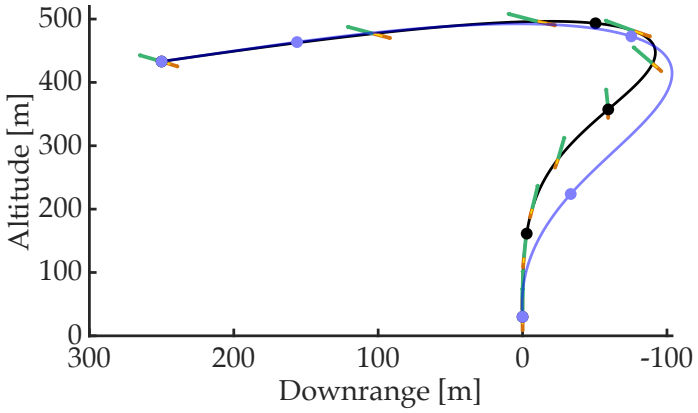
Figure 2.5: Static obstacle avoidance

2.6.2 6-DoF Rocket Landing

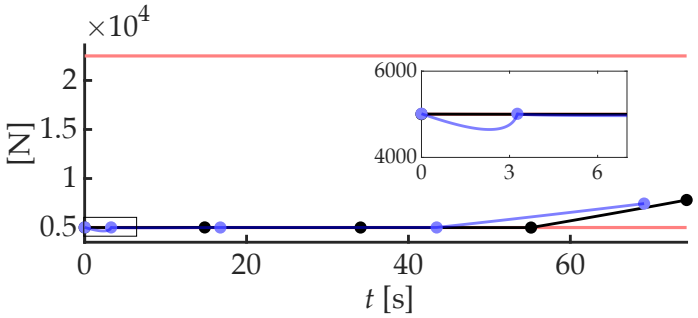
Figure 2.6a shows the position of the 6-DoF rocket along with the attitude of its body-axis (in green) and the thrust vector (as an orange-yellow plume). Figure 2.6b shows the thrust magnitude and Figure 2.6c shows the tilt of the body axis—both of which show inter-sample violation with the node-only approach. Similar to the obstacle avoidance example, the terminal state cost with the node-only approach, 170.1 kg, is smaller than that from CT-SCVX, 180.5 kg.

2.6.3 3-DoF Rocket Landing (Lossless Convexification)

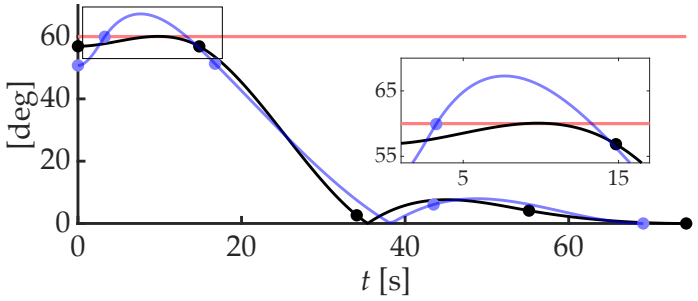
This example demonstrates the application of CT-SCVX for solving a convex optimal control problem. The node-only approach, in this case, amounts to solving a single convex problem where all constraints are imposed via canonical convex cones in a convex optimization solver; we use ECOS [10]. The node-only solution is also used to warm-start CT-SCVX, which



(a) Position



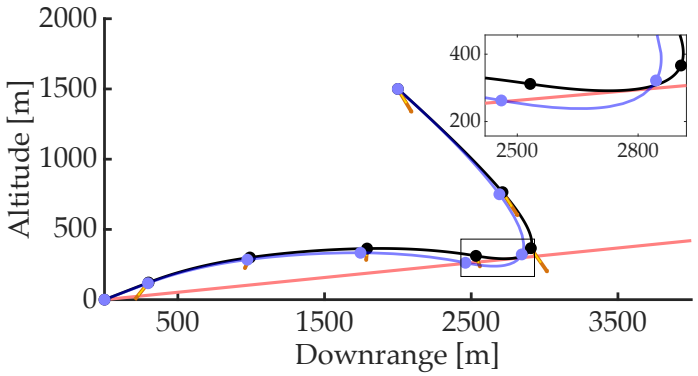
(b) Thrust magnitude



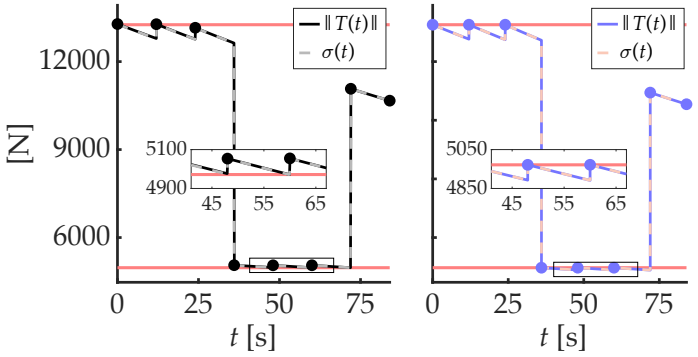
(c) Tilt angle

Figure 2.6: 6-DoF rocket landing

then "cleans up" the inter-sample constraint violations. As with the previous examples, we see that the trajectory cost (in this case, fuel consumption) with the node-only approach, 351 kg, is smaller than that from CT-SCVX, 352.4 kg. Figure 2.7a shows the position of the point-mass rocket along with the thrust vector (as an orange-yellow plume). The glideslope constraint shows inter-sample violation with the node-only approach. Figure 2.7b confirms that lossless convexification holds for the solutions from both methods—CT-SCVX on the left and node-only on the right. The thrust lower bound constraint shows inter-sample



(a) Position



(b) Thrust magnitude

Figure 2.7: 3-DoF rocket landing with lossless convexification

violation with the node-only approach, since a zero-order-hold (ZOH) parameterization is used for the mass-normalized thrust.

The node-only approach requires a dense discretization grid to ensure validity of the lossless convexification result [51], which was proven for the continuous-time problem (prior to discretization). CT-SCVX, in contrast, provides a valid solution despite using a sparse discretization grid. Future work will examine whether the continuous-time lossless convexification result can apply directly to (2.41).

2.6.4 Real-Time Performance

We provide real-time performance statistics for solving the two nonconvex examples in Table 2.1. These results were obtained by executing C code generated using the SCVXGEN software. The binary executables are provided in the code repository. We note that these

Table 2.1: The mean solve-time and standard deviation (Std.) over 1000 solves, along with the number of iterations the prox-linear method takes to converge (Iters.).

Problem	Solve-time	Std.	Iters.
Dyn. obstacle avoidance	59.4 ms	1.4 ms	23
6-DoF rocket landing	42.5 ms	1.3 ms	14

solve-times are on the same order-of-magnitude as the solve-times reported in the literature for real-time powered-descent guidance [110, 42, 25] and real-time quadrotor trajectory optimization with moving obstacles [111], thus demonstrating that CT-SCVX is amenable to online/embedded applications.

2.7 Conclusions

We propose continuous-time successive convexification (CT-SCVX), a real-time-capable trajectory optimization framework that guarantees convergence and enables continuous-time constraint satisfaction. The framework combines: (i) exterior penalty-based reformulation of path constraints; (ii) generalized time-dilation; (iii) multiple-shooting discretization; (iv) ℓ_1-exact penalization; and (v) the prox-linear method, a convergence-guaranteed se-

quential convex programming algorithm for convex-composite minimization. When the optimal control problem is convex, we highlight the stronger convergence property of the solution method.

Chapter 3

CUSTOMIZED CONIC OPTIMIZATION SOLVER

Real-time trajectory optimization focuses on the simultaneous computation and execution of dynamically feasible trajectories for robots while considering operational constraints and environmental factors. Convex optimization-based methods, with their guaranteed convergence to optimal solutions, have emerged as an appealing choice for real-time applications. One key aspect of convex optimization is the representation of a large class of problems as conic optimization problems. A conic optimization problem, shown by (3.1), is characterized by a linear or quadratic objective function and a feasible set formed by intersecting a convex cone $\mathbb{K}$ with an affine space [112, Chap. 3], [113, Eq. 1]. Constraints with special geometric structure (such as constraint sets with simple projection operation) can be represented with $\mathbb{D}$ [24, Eq. 1]. In the context of trajectory optimization, most constraints can be represented using cones and sets for which projection is efficient.

$$\underset{z}{\text{minimize}} \quad \frac{1}{2} z^\top P z + q^\top z \tag{3.1a}$$

$$\text{subject to} \quad Hz - h \in \mathbb{K},\ z \in \mathbb{D} \tag{3.1b}$$

Popular solution methods for conic optimization typically fall into two categories: first-order methods, which use gradient information, and second-order methods, which also incorporate Hessian information. First-order algorithms have become particularly favored for real-time applications due to their low-complexity operations and suitability for small-footprint, library-free code implementations that are ideal for embedded systems. The recent decades have witnessed significant progress in the mathematical tools used to enhance the capabilities and determine the limits of first-order methods.

In this chapter, we discuss the specialization (also referred to as customization) of the

recently developed first-order algorithm, proportional integral projected gradient method [114, 9, 24], for convex and nonconvex trajectory optimization problems (with CT-SCVX). We emphasize the algorithm's features that enable efficient, low-footprint implementation and provide numerical demonstrations.

3.1 Customization of PIPG

Trajectory optimization problems possess a special sparsity structure, in which most constraints on decision variables are temporally separated, with the dynamic feasibility constraint being the exception as it couples decision variables at successive time instants. The PIPG algorithm has been developed with the ability to take advantage of this unique structure through a process called customization, which ultimately results in high-performance software implementations specifically tailored to trajectory optimization problems.

The PIPG algorithm relies on simple operations such as vector addition, matrix and vector multiplication, and dot products, enabling it to exploit the unique structure of trajectory optimization problems through devectorization. Devectorization refers to decomposing optimization problem parameters and variables into smaller segments, where constraints on each segment, involving state and control inputs at a particular time instant, are mostly decoupled from constraints on the other segments. Prior to devectorization, parameters like the conic constraint matrix and the objective function Hessian are large sparse objects with minimal fill-in, with the decision vector typically comprising hundreds of entries. However, after devectorization, the algorithm works with several small dimensional dense vectors and matrices, which is very desirable for high-performance software implementation.

Meanwhile, high-performance sparse linear algebra often demands careful design and implementation of data structures for storing large sparse matrices and vectors, as well as special, a highly target-specific implementation. In contrast, a devectorized implementation involves small dimensional dense objects, which have optimized linear algebra implementations like BLAS and LAPACK. Moreover, devectorization is particularly advantageous for writing software implementations that avoid dynamic memory allocation, are

cache friendly, and can be parallelized. This concept of tailoring a solver to exploit the structure of a specific problem class, such as trajectory optimization through devectorization, is referred to as customization.

In this section, we describe how an extension of PIPG with infeasibility detection, called xPIPG, can be customized for convex trajectory optimization, resulting in an efficient implementation. Furthermore, we discuss heuristics that can improve the practical performance of the solver, making it a powerful tool for solving trajectory optimization problems.

Consider the following discrete-time convex optimal control problem

$$\begin{aligned}
\underset{x_t,\,u_t}{\text{minimize}} \quad & \sum_{t=0}^{N} \frac{1}{2} x_t^\top Q_t x_t + q_t^\top x_t + \frac{1}{2} u_t^\top R_t u_t + r_t^\top u_t, && && \text{(3.2a)}\\
\text{subject to} \quad & x_{t+1} = A_t x_t + B_t^- u_t + B_{t+1}^+ u_{t+1} + c_t, && t \in [0\!:\!N-1], && \text{(3.2b)}\\
& x_t \in \mathbb{D}_t^x, \;\; u_t \in \mathbb{D}_t^u, && t \in [0\!:\!N], && \text{(3.2c)}\\
& F_t^0 x_t + G_t^0 u_t + g_t^0 = 0, && t \in [0\!:\!N], && \text{(3.2d)}\\
& F_t^1 x_t + G_t^1 u_t + g_t^1 \geq 0, && t \in [0\!:\!N], && \text{(3.2e)}
\end{aligned}$$

where $A_t \in \mathbb{R}^{n_x}$, $u_t \in \mathbb{R}^{n_u}$ and $c_t \in \mathbb{R}^{n_x}$, for $t \in [0\!:\!N]$, define the linear system dynamics. For simplicity of discussion, we will assume that there are no system parameters. Sets D_t^x and D_t^u, for $t \in [0\!:\!N]$, are easy to project onto with simple, closed-form expression for the projection. Constraint parameters $F_t^j \in \mathbb{R}^{n_t^j \times n_x}$, $G_t^j \in \mathbb{R}^{n_t^j \times n_u}$, $g_t^j \in \mathbb{R}^{n_t^j}$, for $t \in [0\!:\!N]$, $j = 0, 1$, determine linear equality and inequality constraints on state and control input at each time. The constraints in (3.2c)-(3.2e) determine all the physical and operational constraints on the system state and input. The boundary conditions on state and control input are accounted in $\mathbb{D}_1^x, \mathbb{D}_N^x, \mathbb{D}_1^u$, and $\mathbb{D}_N^u$. To track a known state reference x_t^{ref} and/or a control reference u_t^{ref}, choose $q_t = -2x_t^{\text{ref}}$ and $r_t = -2u_t^{\text{ref}}$. The convex subproblem in CT-SCvx can be modelled using (3.2). In fact, SCP methods themselves can be further customized while using PIPG as the subproblem solver. The SECO framework for SCP developed in [25, 42] is one such example.

By defining the following quantities, we can represent the optimal control problem (3.2)

in terms of the general conic optimization problem (3.1)

$$z = \left(x_0,\, x_1 \cdots,\, x_N,\, u_0,\, u_1,\, \cdots, u_N\right), \tag{3.3a}$$

$$P = \text{blkdiag}\left(Q_0, \ldots, Q_N, R_0, \ldots, R_N\right), \tag{3.3b}$$

$$q = \left(q_0,\, \cdots,\, q_N,\, r_0,\, \cdots,\, r_N\right), \tag{3.3c}$$

$$H = \left[\begin{array}{ccccccc|ccccccc}
A_0 & -I & 0 & \cdots & & & 0 & B_0^- & B_1^+ & 0 & \cdots & & & 0 \\
0 & A_1 & -I & & & & \vdots & 0 & B_1^- & B_2^+ & & & & \vdots \\
\vdots & & & \ddots & & & & \vdots & & & \ddots & & & \\
0 & \cdots & & & A_{N-1} & -I & & 0 & \cdots & & & & B_{N-1}^- & B_N^+ \\
\hline
F_0^0 & 0 & & \cdots & & & 0 & G_0^0 & 0 & & \cdots & & & 0 \\
0 & F_1^0 & & & & & \vdots & 0 & G_1^0 & & & & & \vdots \\
\vdots & & & \ddots & & & & \vdots & & & \ddots & & & \\
0 & \cdots & & & & F_N^0 & & 0 & \cdots & & & & & G_N^0 \\
\hline
F_0^1 & 0 & & \cdots & & & 0 & G_0^1 & 0 & & \cdots & & & 0 \\
0 & F_1^1 & & & & & \vdots & 0 & G_1^1 & & & & & \vdots \\
\vdots & & & \ddots & & & & \vdots & & & \ddots & & & \\
0 & \cdots & & & & F_N^1 & & 0 & \cdots & & & & & G_N^1
\end{array}\right], \tag{3.3d}$$

$$h = -\left(c_0, \ldots, c_{N-1}, g_0^0, \cdots, g_N^0, g_0^1, \cdots, g_N^1\right), \tag{3.3e}$$

$$\mathbb{D} = \mathbb{D}_0^x \times \cdots \times \mathbb{D}_N^x \times \mathbb{D}_0^u \times \cdots \times \mathbb{D}_N^u, \tag{3.3f}$$

$$\mathbb{K} = 0_{n_x N} \times 0_{n_0^0} \times \cdots \times 0_{n_N^0} \times \mathbb{R}_+^{n_0^1} \times \cdots \times \mathbb{R}_+^{n_N^1}. \tag{3.3g}$$

Note that $\mathbb{K}$ could also consider second-order cone and positive semidefinite cone constraints on the state and control input. We exclude specification of such constraints in (3.2) for simplicity. Algorithm 2 provides the customization of xPIPG for solving (3.2). Apart from the problem parameters in (3.2), the input to Algorithm 2 include the xPIPG step-sizes α, β and the maximum iteration count $k_{\max}$. Instead of terminating after $k_{\max}$ iterations, we could also use a termination criteria based on constraint violation or relative change in the primal and dual iterates. The notation $a_{0:M}$ denotes the collection of vectors a_t, for $t = [0\!:\!M]$, arranged into a 2D array. Note that ϕ_t, θ_t, ψ_t in Algorithm 2 are the dual

variables associated with the constraints in $\mathbb{K}$. In a fully self-contained implementation we also need to compute the maximum singular value H (which is required for computing step-sizes α and β) without performing any matrix factorization. This is achieved by the devectorized power iteration method, shown in Algorithm 3. The maximum and minimum eigenvalues of P are straightforward to compute due to its block diagonal structure in typical trajectory optimization problems.

A key step in implementing PIPG is to compute the projection onto cone $\mathbb{K}^\circ$ and set $\mathbb{D}$. These projections can be computed efficiently for the following reasons. First, projections onto many common closed convex cones and sets can be computed using simple formulas, see [115, Chp. 29] for some popular examples. Second, let $\mathbb{D}_1 \subset \mathbb{R}^{n_1}$ and $\mathbb{D}_2 \subset \mathbb{R}^{n_2}$ be closed convex sets, $x_1 \in \mathbb{R}^{n_1}$ and $x_2 \in \mathbb{R}^{n_2}$. Then one can verify the following:

$$\Pi_{\mathbb{D}_1 \times \mathbb{D}_2} \left[\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right] = \begin{bmatrix} \Pi_{\mathbb{D}_1}[x_1] \\ \Pi_{\mathbb{D}_2}[x_2] \end{bmatrix}.$$

Therefore, projections onto sets that are Cartesian products of sets with simple projection formulas also admit simple formulas.

3.2 Numerical Examples

This section demonstrates the efficiency of PIPG on two numerical optimal control applications.

3.2.1 Powered-Descent Guidance

This section compares the performance of an efficient C implementation of customized PIPG with that of some state-of-the-art optimization solvers–including of ECOS, GUROBI, MOSEK, CPLEX, and CONEPROG–for solving [23, Eq. 3] to obtain minimum-fuel and reference-tracking solutions. All simulations are run on a 2020 iMac with a 3.1 GHz 6-Core Intel Core i5 processor and 16 GB RAM. The numerical values of the parameters of the problem, inspired from [116] for a Mars landing scenario, are shown in [23, Tab. 3]. The problem

Algorithm 2 Customized xPIPG

Input: $\alpha, \beta, \rho, k_{\max}$

▷ Initialize: $\tilde{x}_{0:N}, \tilde{u}_{0:N}, \tilde{\phi}_{0:N-1}, \tilde{\theta}_{0:N}, \tilde{\psi}_{0:N}$

for $k \in [1 : k_{\max} - 1]$ **do**

▷ Primal update

$x_0 \leftarrow \Pi_{\mathbb{D}_0^x}\left[\tilde{x}_0 - \alpha\left(Q_0\tilde{x}_0 + q_0 + A_0^\top\tilde{\phi}_0 + {F_0^0}^\top\tilde{\theta}_0 + {F_0^1}^\top\tilde{\psi}_0\right)\right]$

$u_0 \leftarrow \Pi_{\mathbb{D}_0^u}\left[\tilde{u}_0 - \alpha\left(R_0\tilde{u}_0 + r_0 + {B_0^-}^\top\tilde{\phi}_0 + {G_0^0}^\top\tilde{\theta}_0 + {G_0^1}^\top\tilde{\psi}_0\right)\right]$

for $t \in [1 : N-1]$ **do**

$x_t \leftarrow \Pi_{\mathbb{D}_t^x}\left[\tilde{x}_t - \alpha\left(Q_t\tilde{x}_t + q_t + A_t^\top\tilde{\phi}_t - \tilde{\phi}_{t-1} + {F_t^0}^\top\tilde{\theta}_t + {F_t^1}^\top\tilde{\psi}_t\right)\right]$

$u_t \leftarrow \Pi_{\mathbb{D}_t^u}\left[\tilde{u}_t - \alpha\left(R_t\tilde{u}_t + r_t + {B_t^-}^\top\tilde{\phi}_t + {B_t^+}^\top\tilde{\phi}_{t-1} + {G_t^0}^\top\tilde{\theta}_t + {G_t^1}^\top\tilde{\psi}_t\right)\right]$

end for

$x_N \leftarrow \Pi_{\mathbb{D}_N^x}\left[\tilde{x}_N - \alpha\left(Q_N\tilde{x}_N + q_N - \tilde{\phi}_{N-1} + {F_N^0}^\top\tilde{\theta}_N + {F_N^1}^\top\tilde{\psi}_N\right)\right]$

$u_N \leftarrow \Pi_{\mathbb{D}_N^u}\left[\tilde{u}_N - \alpha\left(R_N\tilde{u}_N + r_N + {B_N^+}^\top\tilde{\phi}_{N-1} + {G_N^0}^\top\tilde{\theta}_N + {G_N^1}^\top\tilde{\psi}_N\right)\right]$

▷ Dual update

for $t \in [0 : N-1]$ **do**

$\phi_t \leftarrow \tilde{\phi}_t + \beta(-2x_{t+1} + \tilde{x}_{t+1} + A_t(2x_t - \tilde{x}_t) + B_t^-(2u_t - \tilde{u}_t) + B_{t+1}^+(2u_{t+1} - \tilde{u}_{t+1}) + c_t)$

$\theta_t \leftarrow \tilde{\theta}_t + \beta\left(F_t^0(2x_t - \tilde{x}_t) + G_t^0(2u_t - \tilde{u}_t) + g_t^0\right)$

$\psi_t \leftarrow \min\left\{\tilde{\psi}_t + \beta\left(F_t^1(2x_t - \tilde{x}_t) + G_t^1(2u_t - \tilde{u}_t) + g_t^1\right), 0\right\}$

end for

$\theta_N \leftarrow \tilde{\theta}_N + \beta\left(F_N^0(2x_N - \tilde{x}_N) + G_N^0(2u_N - \tilde{u}_N) + g_N^0\right)$

$\psi_N \leftarrow \min\left\{\tilde{\psi}_N + \beta\left(F_N^1(2x_N - \tilde{x}_N) + G_N^1(2u_N - \tilde{u}_N) + g_N^1\right), 0\right\}$

▷ Extrapolation

$\tilde{x}_{0:N} \leftarrow (1-\rho)\tilde{x}_{0:N} + \rho x_{0:N}$

$\tilde{u}_{0:N} \leftarrow (1-\rho)\tilde{u}_{0:N} + \rho u_{0:N}$

$\tilde{\phi}_{0:N-1} \leftarrow (1-\rho)\tilde{\phi}_{0:N-1} + \rho\phi_{0:N-1}$

$\tilde{\theta}_{0:N} \leftarrow (1-\rho)\tilde{\theta}_{0:N} + \rho\theta_{0:N}$

$\tilde{\psi}_{0:N} \leftarrow (1-\rho)\tilde{\psi}_{0:N} + \rho\psi_{0:N}$

end for

Output: $x_{0:N}, u_{0:N}, \phi_{0:N-1}, \theta_{0:N}, \psi_{0:N}$

Algorithm 3 Power iteration for computing maximum singular value of H

Input: ϵ ▷ *Termination tolerance*

▷ Initialize: σ, σ', $x_{0:N}$, $u_{0:N}$, $\phi_{0:N-1}$, $\theta_{0:N}$, $\psi_{0:N}$

for $|\sigma - \sigma'| \geq \epsilon$ **do**

$\sigma' \leftarrow \sigma$

$\sigma \leftarrow \sqrt{\sum_{t=0}^{N} \|(x_t, u_t)\|_2^2}$

for $t \in [0:N]$ **do**

$x_t \leftarrow x_t / \sigma$

$u_t \leftarrow u_t / \sigma$

end for

for $t \in [0:N-1]$ **do**

$\phi_t \leftarrow -x_{t+1} + A_t x_t + B_t^- u_t + B_{t+1}^+ u_{t+1} + c_t$

$\theta_t \leftarrow F_t^0 x_t + G_t^0 u_t + g_t^0$

$\psi_t \leftarrow F_t^1 x_t + G_t^1 u_t + g_t^1$

end for

$\theta_N \leftarrow F_N^0 x_N + G_N^0 u_N + g_N^0$

$\psi_N \leftarrow F_N^1 x_N + G_N^1 u_N + g_N^1$

$x_0 \leftarrow A_0^\top \phi_0 + {F_0^0}^\top \theta_0 + {F_0^1}^\top \psi_0$

$u_0 \leftarrow {B_0^-}^\top \phi_0 + {G_0^0}^\top \theta_0 + {G_0^1}^\top \psi_0$

for $t \in [1:N-1]$ **do**

$x_t \leftarrow A_t^\top \phi_t - \phi_{t-1} + {F_t^0}^\top \theta_t + {F_t^1}^\top \psi_t$

$u_t \leftarrow {B_t^-}^\top \phi_t + {B_t^+}^\top \phi_{t-1} + {G_t^0}^\top \theta_t + {G_t^1}^\top \psi_t$

end for

$x_N \leftarrow -\phi_{N-1} + {F_N^0}^\top \theta_N + {F_N^1}^\top \psi_N$

$u_N \leftarrow {B_N^+}^\top \phi_{N-1} + {G_N^0}^\top \theta_N + {G_N^1}^\top \psi_N$

end for

Output: σ

parameters are scaled before the solve step to ensure that the decision variables are on the same order of magnitude. Scaling the problem parameters is a well-known technique to improve solver performance [3]. The termination criteria is set such that solvers terminate when the solution is within 0.03% of a high-accuracy *ground-truth* solution computed using GUROBI.

All solvers except for PIPG require a nontrivial parsing operation that converts the original optimal control problem using the data in [23, Tab. 3] into the parameters associated with a conic optimization problem in the homogeneous self-dual embedding form. We use YALMIP, a convex optimization modeling tool in MATLAB, to carry out the parse step. The solvers accept the parsed parameters from YALMIP to solve the problem and return a solution (if one exists), which is then transformed by YALMIP into the solution variables of the original problem. On the other hand, PIPG doesn't require YALMIP for the parse step since it is customized to incorporate the structure of the original problem which if of the form (3.2).

Table 3.1 and Table 3.2 summarize the execution-time–the sum of solve-time and parse-time–of different solvers for both the minimum-fuel and reference-tracking problems across three horizon lengths, averaged over 50 simulation runs with random initialization. Figure 3.1 compares the solve- and parse-times for these problems. The average execution-time of PIPG is significantly smaller than that of the other solvers for both the minimum-fuel and reference-tracking problems. Furthermore, it is worthwhile to note that the C code generated for uncustomized PIPG [23, Alg. 1] also performs well with the use of sparse linear algebra; it solves both the minimum-fuel and reference-tracking problems in tens of milliseconds.

The reference-tracking problem has a strongly convex objective function, which enables PIPG to converge at the fastest rate possible for a first-order solver. Since PIPG uses variable step sizes given in [114] for solving this problem, it is restarted to improve its practical convergence rate. Restarting it at 1200 iterations is observed to be optimal and a total of 3000 iterations are required for convergence. Restarting is a popular heuristic for improving the practical convergence rates of primal-dual methods [117, 118]. The minimum-fuel

problem, on the other hand, does not have a strongly convex cost function. So, PIPG uses constant step sizes as given in [114], and is guaranteed to converge at a slightly slower rate. In practice, however, problem scaling parameters can be appropriately tuned to obtain fast execution-times. Convergence is achieved in 7500 iterations in our case.

Note that the difference in fuel consumption between the reference-tracking and minimum-fuel solutions is $\sim$ 9 kg, which is less than 3% of the initial fuel in the vehicle. So, for a minor loss in fuel efficiency, using the reference-tracking solution computed using PIPG instead can lead to dramatic gains in real-time performance (owing to the $\sim$ 4 ms execution-time of PIPG). The solution from PIPG and the ground-truth solution from GUROBI (with $N = 20$) for the reference tracking problem are shown in Figure 3.2. The estimated solution is lossless, i.e. $\|u_t\|_2 = \sigma_t \quad \forall\, 0 \leq t \leq N$.

Table 3.1: Reference-tracking problem execution-time [ms]

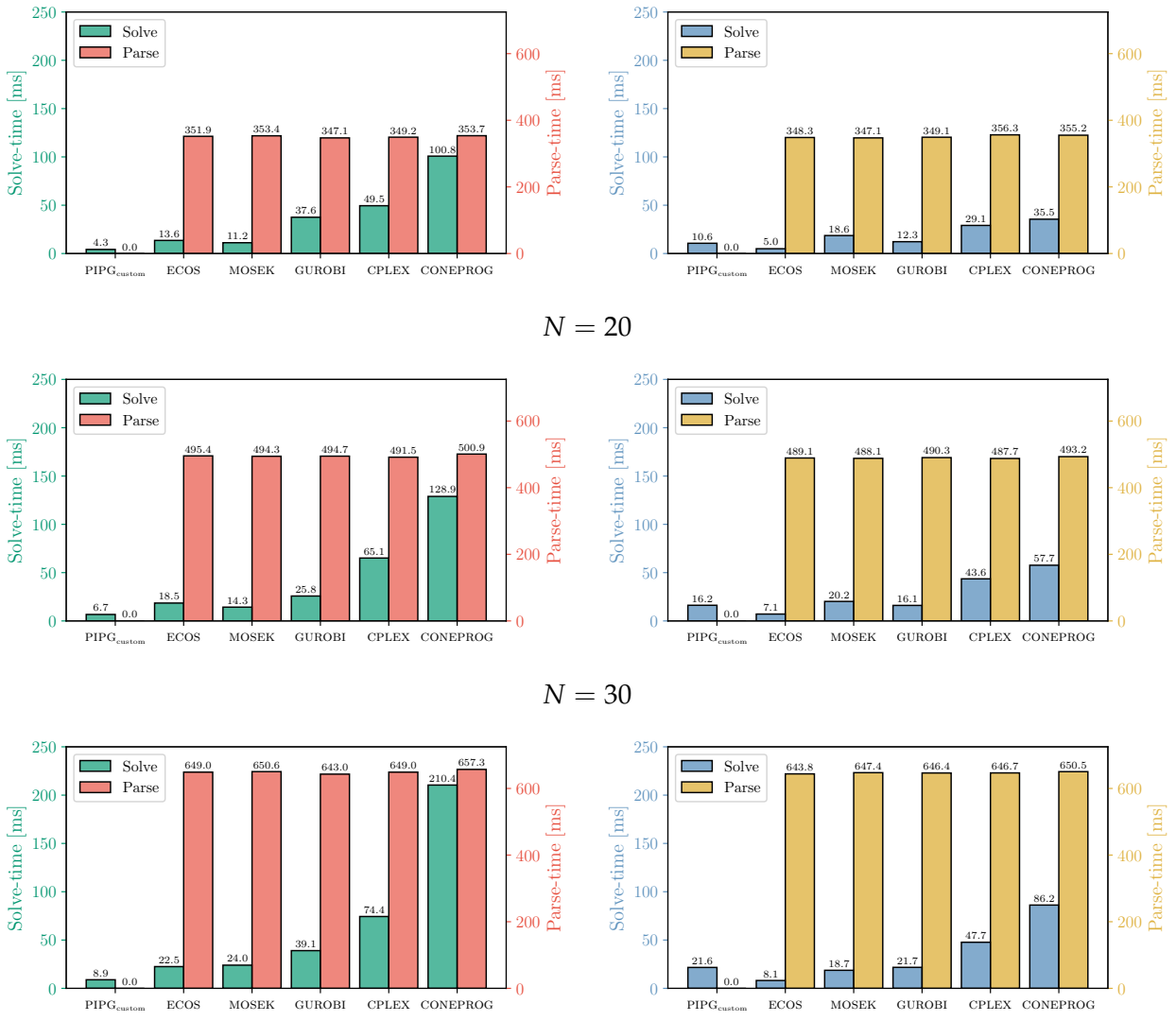
M	**PIPG**	ECOS	MOSEK	GUROBI	CPLEX	CONEPROG
20	**4.25**	365.43	364.57	384.64	398.72	454.49
30	**6.71**	513.96	508.59	520.45	556.64	629.78
40	**8.90**	671.49	674.55	682.05	723.32	867.70

Table 3.2: Minimum-fuel problem execution-time [ms]

M	**PIPG**	ECOS	MOSEK	GUROBI	CPLEX	CONEPROG
20	**10.61**	353.22	365.72	361.35	385.35	390.69
30	**16.22**	496.20	508.27	506.38	531.25	550.83
40	**21.64**	651.88	666.09	668.19	694.39	736.68

3.2.2 Oscillating Masses

In this section we demonstrate the real-time capability of XPIPG along with infeasibility detection. To this end, we consider the two-point boundary-value optimal control problem of a mechanical system composed of l ($l \in \mathbb{N}$) oscillating masses [119, 6, 114]; see Figure



(a) Reference-tracking problem

(b) Minimum-fuel problem

Figure 3.1: Benchmarking test results for PIPG against several state-of-the-art industrial-grade solvers, with solve- and parse-times averaged over 50 runs.

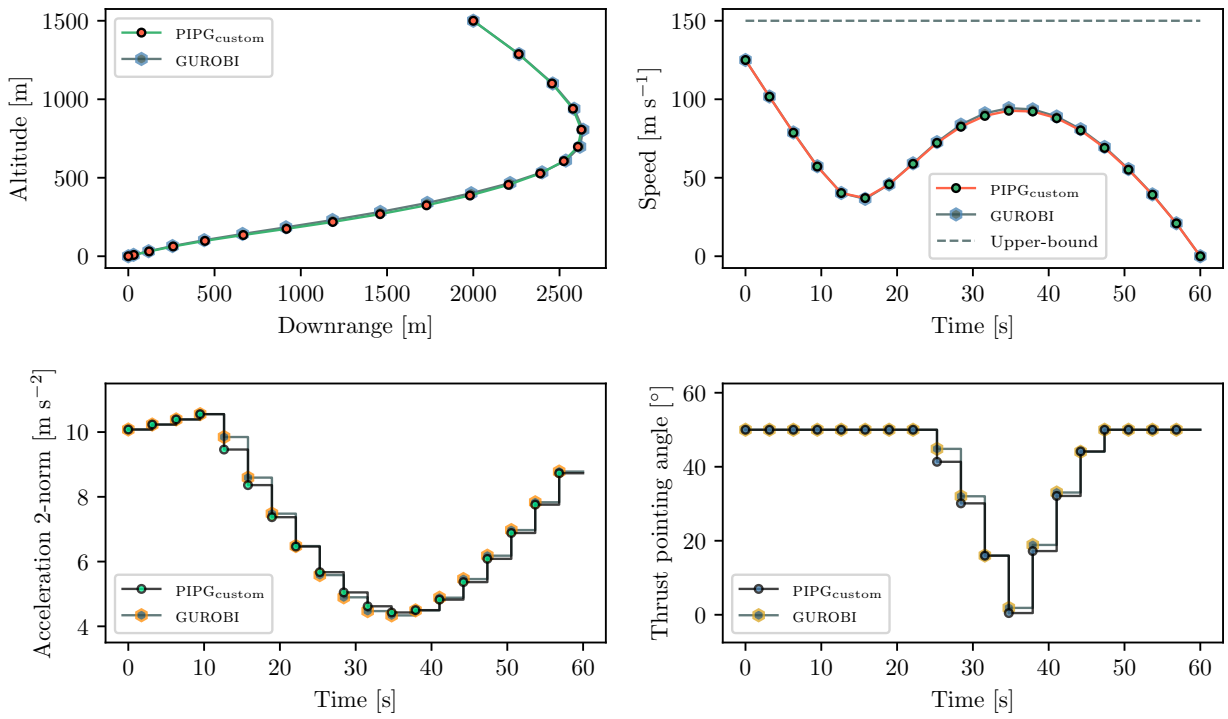


Figure 3.2: Generated PDG trajectories for $N = 20$.

3.3 for an illustration. The discrete-time linear system is described by

$$
\begin{aligned}
&A_t = \exp\left(\Delta \begin{bmatrix} 0_{l,l} & I_l \\ -L & 0_{l,l} \end{bmatrix}\right), \\
&B_t = \int_0^{\Delta} \exp\left(s \begin{bmatrix} 0_{l,l} & I_l \\ -L & 0_{l,l} \end{bmatrix}\right) \mathrm{d}s \begin{bmatrix} 0_{l,l} \\ I_l \end{bmatrix}, \\
&Q_t = I_{2l}, \;\; q_t = 0_{2l}, \;\; R_t = I_l, \;\; r_t = 0_l, \\
&\mathbb{X} = \{x \in \mathbb{R}^{2l} \,|\, \|x\|_\infty \leq \varrho_x\}, \;\; \mathbb{U} = \{u \in \mathbb{R}^{l} \,|\, \|u\|_\infty \leq \varrho_u\},
\end{aligned}
$$

where $\hat{x}_0$ is the initial state, $\Delta = 0.1$ is the sampling period of the dynamics, $L \in \mathbb{R}^{l \times l}$ is a tridiagonal matrix: its diagonal entries are 2's, its subdiagonal and superdiagonal entries are -1's. We apply Algorithm 2 (equipped with a C implementation) to instances

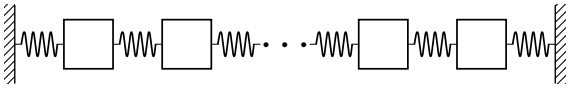
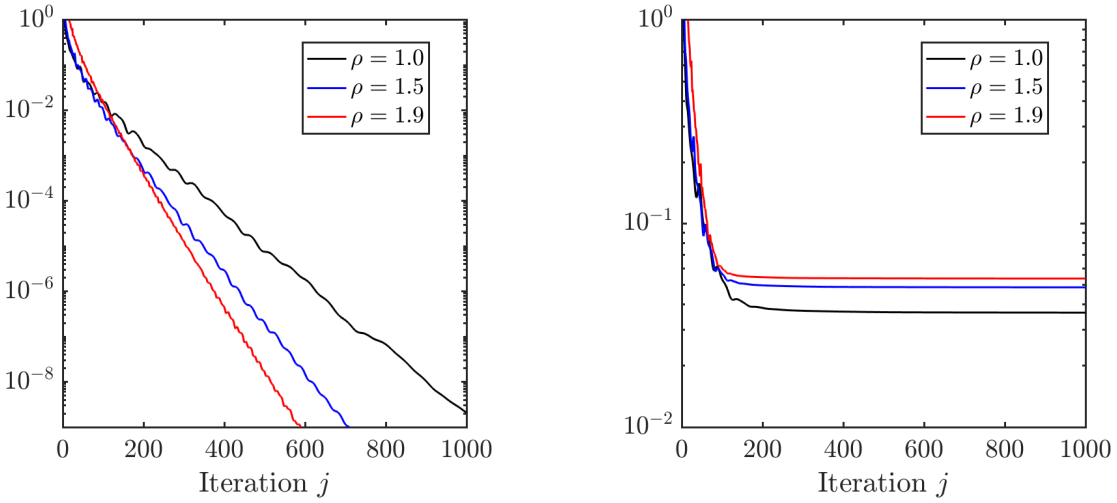


Figure 3.3: The oscillating masses system

of (3.2) where $\varrho_x = 1$, $\varrho_u = 0.5$, $\tau = 20$ and $l \in \{32, 64, 128, 256\}$; furthermore, we sample

the value of $\hat{x}_0$ from a normal distribution with mean $(\gamma 1_l, 0_l)$ and standard deviation 0.05, where $\gamma = 0.1$ for feasible instances and $\gamma = 0.8$ for infeasible instances of (3.2), respectively. Figure 3.4 shows the convergence of xPIPG applied to an problem instance where $l = 32$. xPIPG converges about twice as fast as PIPG; the latter is equivalent to xPIPG with $\rho = 1$. Furthermore, any value within the interval $[1.5, 1.9]$ provides a good choices for ρ. These observations agree with those made for the Douglas-Rachford splitting methods [8]. Furthermore, we compare the performance of Algorithm 2 (with $\rho = 1.6$) against those



(a) Feasible.

(b) Infeasible.

Figure 3.4: The asymptotic convergence of $\frac{1}{\beta\rho}\|w^{j+1} - w^j\|$ in Algorithm 2 when applied to an feasible (left) and infeasible (right) instance of (3.2).

of other state-of-the-art solvers, including OSQP [120], SCS [113, 121], ECOS [10], and MOSEK [11]. We terminate Algorithm 2 and all the other solvers when the following condition is met for a feasible problem[1]:

$$\max\left\{\|Hz^j - h\|_\infty, \|z^j - \min(z_u, \max(z_l, z^j))\|_\infty\right\} < \epsilon_{\text{fea}}, \tag{3.4}$$

where $z_u = (\hat{x}_0, 1_{2l(N-1)}\varrho_x, 0_{2l}, 1_{lN}\varrho_u)$, $z_u = (\hat{x}_0, -1_{2l(N-1)}\varrho_x, 0_{2l}, -1_{lN}\varrho_u)$.

The above condition implies that z^j satisfies the constraints up to a tolerance of ϵ_{fea}. For

[1]For each of the other solvers, we set their optimality tolerance to the largest value for (3.4) to hold, since we empirically observed that the constraint violation decreases monotonically with the optimality tolerance.

an infeasible problem, we terminate Algorithm 2 when the following condition holds:

$$\inf_{z \in \mathbb{D}} \langle Hz - h, w^{j+1} - w^j \rangle + \epsilon_{\text{inf}} > 0. \tag{3.5}$$

Notice that since $\mathbb{K} = \{0_{2Nl}\}$, (3.5) implies that [24, Eq. 2] holds up to a tolerance of ϵ_{inf}; furthermore, the infimum in (3.5) has a closed form solution since set $\mathbb{D}$ is a box. For the other solvers, we set their corresponding infeasibility certificate tolerance value to ϵ_{inf}. Table 3.3 shows the computation time of different solvers, where the best computation time

Table 3.3: Comparison of the computation time [ms] of different solvers for feasible and infeasible instances of optimization (3.2).

(a) Feasible problems, averaged over 100 random instances.

ϵ_{fea}	l	OSQP	SCS	ECOS	MOSEK	xPIPG
10^{-4}	32	11.6	11.3	40.0	23.2	**10.7**
	64	55.4	47.6	214.3	83.1	**20.0**
	128	333.8	305.9	1400.6	133.9	**68.5**
	256	2477.7	2287.6	13674.0	269.7	**251.2**
10^{-8}	32	**19.3**	20.0	61.2	27.5	23.4
	64	84.3	81.7	277.6	93.3	**46.6**
	128	505.4	467.8	1703.8	**169.9**	189.2
	256	3443.0	3232.7	19030.6	**337.4**	616.0

(b) Infeasible problems, averaged over 100 random instances.

ϵ_{inf}	l	OSQP	SCS	ECOS	MOSEK	xPIPG
10^{-4}	32	20.5	24.6	44.9	23.9	**7.9**
	64	87.4	112.5	200.5	93.6	**15.8**
	128	670.8	703.2	1375.8	154.2	**63.0**
	256	4309.7	5146.5	13471.5	321.3	**209.0**
10^{-8}	32	20.2	27.6	44.8	30.8	**7.9**
	64	86.3	111.9	198.8	108.5	**15.5**
	128	654.1	758.2	1357.7	192.2	**63.2**
	256	4367.7	5202.7	13611.8	385.9	**211.7**

in each row is highlighted. We can see that Algorithm 2 has a clear advantage against open-source solvers–including OSQP, SCS, and ECOS–especially for large-scale and infeasible problems. The overall performance of Algorithm 2 is comparable to the highly optimized commercial solver MOSEK.

Chapter 4

DEFERRED DECISION TRAJECTORY OPTIMIZATION

4.1 Introduction

The reference signals generated by the feedforward block of a control system typically solve a constrained optimal control problem. It is desirable to have the system response tightly track the reference signals in closed-loop with a feedback controller so that the constraint satisfaction and performance requirements guaranteed by the reference signal continue to hold. However, for nominal performance of the complete closed-loop system, uncertainties need to be accounted for, both within feedforward and feedback blocks. Depending on the nature of the uncertainties, techniques from robust and stochastic control may not be readily applicable. The former deals with uncertainties which are known to belong to a certain class or a set, while the latter handles uncertainties that are known to possess a probability distribution. However, in many real-world circumstances, it is not possible to quantify or model all sources of uncertainties and contingencies a priori.

So, how do we reduce the likelihood of mission failure or off-nominal performance in such cases? How do we embed within a trajectory optimization problem the mission requirement of ensuring resilience to unmodeled uncertainties? The proposed work addresses this question for the problem of generating a constrained trajectory for a vehicle from a known initial state to a desired target set or state. One approach to staying immune to unmodeled uncertainties and contingencies (i.e., "unknown" unknowns) is by considering multiple candidate targets. We focus our attention on the feedforward block and ask the question: how can the reference signal help with the identification of the best (most reliable) target? In the real-world setting, it is reasonable to assume that the vehicle can acquire new actionable information as it moves towards the candidate targets.

So, deferring the decision to commit to/select a target provides time to learn (acquire more information) about the uncertainties. In a closed-loop feedback system, for example, the perception and measurement updates provide new information. Ensuring that the candidate targets are reachable for as long as possible enables the vehicle to acquire new information for as long as possible, which then informs the best choice of target, where "best" could mean safest or cost efficient. Note that we refer to the ability to *recover* (i.e., prevent mission failure) in spite of the effect of uncertainties and contingencies as *resilience*, which is the notion established in the literature [122].

The example of soft landing of a spacecraft on an unknown planetary terrain highlights the usefulness of deferring decision. The goal is to land a spacecraft at one of landing sites from a collection of candidates (see Figure 4.1). We only possess coarse-grained information about the terrain and need to wait until the spacecraft nears the terrain to gather high-resolution information. It is straightforward to imagine a situation where the spacecraft chooses a landing site prematurely, but determines that it is not viable after getting too close, at which point recovery or divert might be impossible. Therefore, deferring decision is useful in this scenario for keeping a collection of landing sites reachable for a period of time while high-resolution terrain information is gathered to determine which candidate landing sites are viable. Note that interplanetary communication has large latency which makes it infeasible to control the spacecraft in real-time with an Earth-based operator.

We present DDTO—deferred decision trajectory optimization, a fully-deterministic framework for formulating constrained trajectory generation in the presence of such unmodeled uncertainties and contingencies. First, we propose constrained-reachability-based formulations for DDTO—not with intention of directly solving the resulting optimization, but with the goal of discovering/analyzing the solution structure. Constrained reachable sets offer the most natural way to describe the notion of deferring decision. While they are intractable to compute for nonlinear systems with state dimension greater than four [123, Fig. 2], they are still useful building blocks for analyzing the problem at hand. Next, we show that the proposed constrained-reachability formulations have equivalent representations as cardinality minimization problems. The cardinality minimization problems are

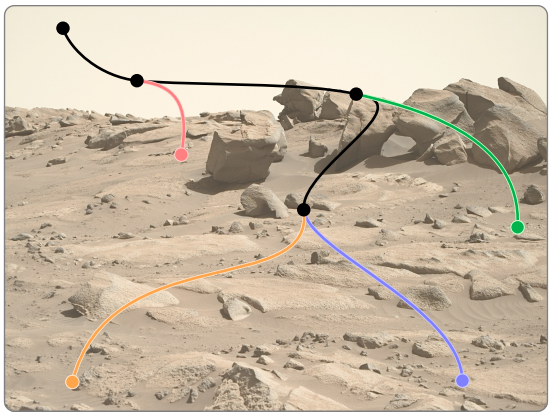


Figure 4.1: A Mars landing example where deferring decision is useful. The black trajectory segments keep a collection of candidate landing sites reachable (colored nodes). Each black node serves as a decision point beyond which reachability to one of the landing sites is lost. While the spacecraft follows the black segment it can learn more about the terrain to determine the most viable landing site. The background image (taken by the Perseverance rover in December 2023 [1]) shows examples of a priori unknown irregularities on the Martian surface which can potentially make landing sites infeasible.

again not meant to be directly solvable, their purpose here is to illuminate the structure of an optimal solution. With this knowledge we can design approximate, effective and efficient solution methods that are specialized to trajectory optimization problems. We do not explore well-established, general-purpose solution methods for general cardinality minimization problem, such as compressed sensing [124], sparse recovery methods [125].

We propose two solution methods that mimic the inferred optimal solution structure: one for discrete-time affine systems subject to convex constraints, where we leverage quasiconvex optimization, and another for continuous-time nonlinear systems subject to nonconvex constraints, where we use the recently developed sequential convex programming (SCP) framework for nonconvex trajectory generation [22]. The convex case solution method was earlier proposed as a heuristic approach for DDTO in [26], which was followed by another variant, ADAPTIVE-DDTO, demonstrated within a closed-loop simulation framework called hazard-aware landing optimization (HALO), designed for precision landing on unknown planetary terrain [126] with vision-based real-time perception.

We infer that the optimal solution for DDTO has a tree-like structure with trunk and branch trajectories (see Figure 4.1). Hence, graph-based techniques such as sampling based-planners [127] might seem naturally suited for modeling and computing such struc-

tures. However, we adopt optimization-based modeling to place emphasis on constraint satisfaction and dynamic feasibility, which allows us to harness a variety of specialized techniques such as multiple-shooting and time-dilation [22], and state-of-the-art convex optimization-based solution methods which efficient and scalable [10, 24]. Furthermore, it is a natural concern that deferring decision (which is essentially procrastination) could come at the expense of some trajectory cost. It is undesirable for the vehicle to consume all of the onboard power-resource or fuel in its pursuit of "buying more time". In the proposed formulation and solution methods, we explicitly call out the importance of imposing constraints on the cumulative cost of a trajectory. Further, note that DDTO is *not* meant as an alternative or a replacement for methods that model uncertainties. In fact, whenever possible, known models of uncertainties should be incorporated into the trajectory optimization formulation to reduce conservativeness and to meet the desired performance requirements.

4.1.1 Related Work

Ideas related to deferred decision making have received attention in the robotics and motion planning literature. An approach for air traffic congestion resolution which uses deferred decisions [128] demonstrated that conserving future flexibility can help manage the risk of uncertain predictions. Deferred decisions can counter the accumulation of errors in INS-based navigation [129] by simultaneously allowing multiple potential trajectories to exist and selecting the best one to represent the robots's path.

Enhancements to model predictive control (MPC) to anticipate future uncertainties have been developed. A feedback min-max MPC approach was introduced in [130], where a family of control sequences is optimized at each time, each one corresponding to a different disturbance profile. Consequently, this approach avoids the likely feasibility problems that result from the use of other min-max formulations that optimize a single control profile over all possible future disturbance realizations. However, the knowledge of the type and source of the uncertainties is still necessary for this approach. The optimal cost design for MPC in [131] demonstrates that delaying the decision until later implicitly accounts for

the fact that the agent will get more information in the future and be able to make a better decision, which was possible by purposefully choosing a new MPC cost which is different from the planned trajectory cost. The branching MPC framework in [132] considers a finite set of policies to represent the continuous spectrum of reactive behaviors of an uncontrolled agent. This approach bears similarity to the notion of interaction-awareness in [133].

Methods developed for contingency planning and abort guidance are also related to deferred decisions since they accomplish a similar goal. Anticipating contingencies that might take place instead of simply reacting to them is beneficial and safer. An approach for computing abort trajectories along with the nominal is provided in [134]. The vehicle can divert to a predetermined safe region using a precomputed abort trajectory if any off-nominal effects are detected. Similar contingency planning measures are proposed for autonomous driving applications in [135, 136].

Besides the robotics applications, deferred decision making has been incorporated in floorplanning algorithms for VLSI [137], in data classification tasks [138], and in active fault diagnosis [139].

Finally, research from psychology shows that deferring decision is a likely outcome when there many options to choose from [140], i.e., contrary to the principle of value maximization, choices can introduce conflict. Moreover, studies from organizational psychology have determined that procrastination plays a key role in improving human decision-making [141]. The ability to defer the decision about choosing a goal can improve the performance and reliability of human-in-the-loop (HIL) robotic systems.

Notation

The set of real numbers is denoted by $\mathbb{R}$, the set of nonnegative real numbers by $\mathbb{R}_+$, the set of n-dimensional real vectors by $\mathbb{R}^n$, and the set of $m \times n$ matrices by $\mathbb{R}^{m \times n}$. The set of integers between and including integers a and b (with $a \leq b$) is denoted by $[a\!:\!b]$, and the cardinality of a set C by $|C|$. The vectors of zeros in $\mathbb{R}^n$ is denoted by 0_n, and the identity

matrix in $\mathbb{R}^{n\times n}$ by I_n. The concatenation of vectors $v \in \mathbb{R}^n$ and $w \in \mathbb{R}^m$ is denoted by $(v, w) \in \mathbb{R}^{n+m}$. The function $x \mapsto \|x\|_\blacklozenge$ indicates whether a vector x is non-zero, i.e.,

$$\|x\|_\blacklozenge \triangleq \begin{cases} 1 & \text{if } x \neq 0, \\ 0 & \text{otherwise.} \end{cases}$$

4.2 Preliminaries

Consider a time-invariant, discrete-time nonlinear dynamical system

$$x_{k+1} = f(x_k, u_k), \quad k \geq 1, \tag{4.1}$$

where index k corresponds to time, $x_k \in \mathbb{R}^{n_x}$ is the state, and $u_k \in \mathbb{R}^{n_u}$ is the control input. The state and control input are required to lie in sets $\mathbb{X} \subset \mathbb{R}^{n_x}$ and $\mathbb{U} \subset \mathbb{R}^{n_u}$, respectively. We are interested in a feasible trajectory of length N, denoted by: $x_k \in \mathbb{X}$, for $k \in [1\!:\!N]$, which is generated with a feasible control input sequence: $u_k \in \mathbb{U}$, for $k \in [1\!:\!N-1]$. The trajectory starts from the initial state $z^0 \in \mathbb{X}$ and terminates at one of the n targets $\mathcal{Z}^j \in \mathbb{X}$, for $j \in [1\!:\!n]$.

The space of feasible trajectories and control input sequences for (4.1) can be characterized with constrained forward and backward reachable sets. The set of all terminal states of the feasible trajectories of length $M+1$ with initial state $z \in \mathbb{X}$ is denoted by $\mathcal{F}_M(z)$.

Definition 2. *Given $M \geq 1$, the M-step constrained forward reachable set of $z \in \mathbb{X}$ is given by*

$$\mathcal{F}_M(z) \triangleq \left\{ x_{M+1} \;\middle|\; \begin{array}{l} u_1, \ldots, u_M \in \mathbb{U} \\ x_{k+1} = f(x_k, u_k),\ k \in [1\!:\!M] \\ x_1 = z,\ x_k \in \mathbb{X},\ k \in [2\!:\!M+1] \end{array} \right\}.$$

Furthermore, $\mathcal{F}_0(z) \triangleq \{z\}$.

The set of all initial states of the feasible trajectories of length $M+1$ that terminates at $\mathcal{Z} \subset \mathbb{X}$ is denoted by $\mathcal{B}_M(z)$.

Definition 3. *Given $M \geq 1$, the M-step constrained backward reachable set of $\mathcal{Z} \subset \mathbb{X}$ is given by*

$$\mathcal{B}_M(\mathcal{Z}) \triangleq \left\{ x_1 \;\middle|\; \begin{array}{l} u_1, \ldots, u_M \in \mathbb{U} \\ x_{k+1} = f(x_k, u_k),\ k \in [1\!:\!M] \\ x_k \in \mathbb{X},\ k \in [1\!:\!M],\ x_{M+1} \in \mathcal{Z} \end{array} \right\}.$$

Furthermore, $\mathcal{B}_0(\mathcal{Z}) \triangleq \mathcal{Z}$.

We have the following result about a sequence of states selected from the constrained forward reachable sets.

Lemma 6. *Let x_k, for $k \in [1\!:\!N]$, be a sequence of states. Then, the following hold for each $k \in [2\!:\!N]$*

1. *If $x_k \in \mathcal{F}_1(x_{k-1})$ with $x_1 = z^0$, then $\mathcal{F}_1(x_{k-1}) \subseteq \mathcal{F}_{k-1}(z^0)$.*
2. *If $x_{k-1} \in \mathcal{B}_1(\{x_k\})$ with $\{x_N\} = \mathcal{Z}^j$, for some $j \in [1\!:\!n]$, then $\mathcal{B}_1(\{x_k\}) \subseteq \mathcal{B}_{N-k+1}(\mathcal{Z}^j)$.*

Proof. 1. Let $x_1 = z^0$ and $x_k \in \mathcal{F}_1(x_{k-1})$, for $k \in [2\!:\!N]$. There exist $u_k \in \mathbb{U}$, for $k \in [1\!:\!N-1]$, such that $x_{k+1} = f(x_k, u_k) \in \mathbb{X}$, for $k \in [1\!:\!N-1]$. Then, from Definition 2, $x_k \in \mathcal{F}_{k-1}(z^0)$, for $k \in [1\!:\!N]$. Since the choice of $x_k \in \mathcal{F}_1(x_{k-1})$ is arbitrary, it follows that $\mathcal{F}_1(x_{k-1}) \subseteq \mathcal{F}_{k-1}(z^0)$, for $k \in [2\!:\!N]$.

2. Given $j \in [1\!:\!n]$, let $\{x_N\} = \mathcal{Z}^j$ and $x_{k-1} \in \mathcal{B}_1(\{x_k\})$, for $k \in [2\!:\!N]$. There exist $u_k \in \mathbb{U}$, for $k \in [1\!:\!N-1]$, such that $x_{k+1} = f(x_k, u_k) \in \mathbb{X}$, for $k \in [1\!:\!N-1]$. Then, from Definition 3, $x_k \in \mathcal{B}_{N-k}(\mathcal{Z}^j)$, for $k \in [1\!:\!N]$. Since the choice of $x_{k-1} \in \mathcal{B}_1(\{x_k\})$ is arbitrary, it follows that $\mathcal{B}_1(x_k) \subseteq \mathcal{B}_{N-k+1}(\mathcal{Z}^j)$, for $k \in [2\!:\!N]$. □

Remark 13. *As a consequence of Lemma 6, if $x_k \in \mathcal{F}_{k-1}(z^0)$, for $k \in [1\!:\!N]$, then it does not imply that the sequence of states x_k, for $k \in [1\!:\!N]$, form a dynamically feasible trajectory, i.e., (4.1) need not be satisfied. In other words,*

$$x_k \in \mathcal{F}_{k-1}(z^0) \implies x_k \in \mathcal{F}_1(x_{k-1}), \tag{4.2}$$

for $k \in [1\!:\!N]$. The observation above can be intuitively inferred from the example of a vehicle with finite onboard fuel capacity. The fuel capacity constraint is equivalent to a pointwise-in-

time constraint on the cumulative fuel consumption. Given $1 < k_1 < k_2 < N$*, consider* $x_{k_1} \in \mathcal{F}_{k_1-1}(z^0)$ *and* $x_{k_2} \in \mathcal{F}_{k_2-1}(z^0)$ *such that all of the fuel is required to reach* x_{k_1} *in* k_1 *steps. Then it is impossible to reach* x_{k_2} *from* x_{k_1} *in* $k_2 - k_1$ *steps without violating the fuel capacity constraint.*

Given any $J \subseteq [1\!:\!n]$ and $k \in [1\!:\!N]$, the intersection of the $(N-k)$-step constrained backward reachable sets of targets $\mathcal{Z}^j$, for $j \in J$, forms a key building block for assessing reachability to targets from a given trajectory.

Definition 4. *Given* $J \subseteq [1\!:\!n]$ *and* $k \in [1\!:\!N]$*, define*

$$\mathcal{B}^J_{N-k} \triangleq \bigcap_{j \in J} \mathcal{B}_{N-k}(\mathcal{Z}^j).$$

For brevity in the subsequent discussion, we say "targets in J" instead of "targets $\mathcal{Z}^j$, for $j \in J$", for any $J \subseteq [1\!:\!n]$. Another key construction associated with each target is that of a k-reach set which characterizes states lying on a feasible trajectory of length N to a target.

Definition 5 ***(k-reach set).*** *Given* $j \in [1\!:\!n]$ *and* $k \in [1\!:\!N]$*, define*

$$\mathcal{R}^j_k \triangleq \mathcal{F}_{k-1}(z^0) \bigcap \mathcal{B}_{N-k}(\mathcal{Z}^j).$$

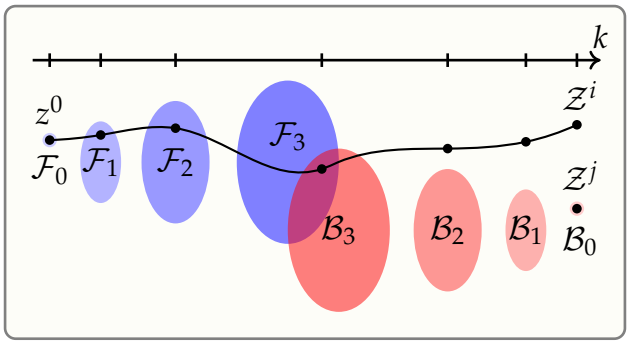


Figure 4.2: A trajectory of length $N = 7$ from z^0 to $\mathcal{Z}^i$ passing through $\mathcal{R}^j_k$ at $k = 4$. The arguments of $\mathcal{F}_k$ and $\mathcal{B}_k$, for $k \in [0\!:\!3]$, are are omitted for brevity.

Corollary 3. *Any feasible trajectory of length* N *from* z^0 *to* $\mathcal{Z}^j$ *intersects* $\mathcal{R}^j_k$ *at time* k*.*

Proof. Let $x_1, \ldots, x_N$, with $x_1 = z^0$ and $x_N \in \mathcal{Z}^j$, be a feasible trajectory of length N

generated by the control input sequence: u_k, for $k \in [1\!:\!N-1]$. Then, $x_k \in \mathcal{F}_{k-1}(z^0)$ and $x_k \in \mathcal{B}_{N-k}(\mathcal{Z}^j)$, for $k \in [1\!:\!N]$, from Definitions 2 and 3, respectively. □

Definitions 4 and 5 are useful for representing the intersection of k-reach sets of targets in J.

Definition 6. *Given $J \subseteq [1\!:\!n]$ and $k \in [1\!:\!N]$, the intersection of k-reach sets of targets in J is defined as*

$$\mathcal{R}_k^J \triangleq \bigcap_{j \in J} \mathcal{R}_k^j = \mathcal{F}_{k-1}(z^0) \bigcap \mathcal{B}_{N-k}^J.$$

Definition 7. *Given $k \in [1\!:\!N]$, the collection of all subsets of targets for which the intersection of k-reach sets is non-empty is given by*

$$\Lambda_k \triangleq \{\, J \subseteq [1\!:\!n] \mid \mathcal{R}_k^J \neq \varnothing \,\}.$$

DDTO seeks trajectories, from initial state to targets, that maximize the size of the member of Λ_k selected for each $k \in [1\!:\!N]$, in a cumulative sense.

Lemma 7. *For any $J \subseteq [1\!:\!n]$, the following holds:*

1. *If there exists $\overline{k} \in [1\!:\!N]$ such that $\mathcal{R}_{\overline{k}}^J \neq \varnothing$, then $\mathcal{R}_k^J \neq \varnothing$ for each $k \in [1\!:\!\overline{k}]$.*
2. *If there exists $\underline{k} \in [1\!:\!N]$ such that $\mathcal{R}_{\underline{k}}^J = \varnothing$, then $\mathcal{R}_k^J = \varnothing$ for each $k \in [\underline{k}\!:\!N]$.*

Proof. 1. Given $J \subseteq [1\!:\!n]$, let $\overline{z} \in \mathcal{R}_{\overline{k}}^J$ for some $\overline{k} \in [2\!:\!N-1]$, and let $x_1, \ldots, x_{\overline{k}-1}, \overline{z}$, $x_{\overline{k}+1}^j, \ldots, x_N^j$ denote a feasible trajectory from z^0 to $\mathcal{Z}^j$ for some $j \in J$. Since $\overline{z} \in \mathcal{R}_{\overline{k}}^J$, for each $i \in J$, there exists $x_{\overline{k}+1}^i, \ldots, x_N^i$ such that $x_1, \ldots, x_{\overline{k}-1}, \overline{z}, x_{\overline{k}+1}^i, \ldots, x_N^i$ is a feasible trajectory from z^0 to $\mathcal{Z}^i$. Hence, we have that $x_k \in \mathcal{R}_k^J$ for each $k \in [1\!:\!\overline{k}-1]$.

2. Let $\mathcal{R}_{\underline{k}}^J = \varnothing$ for some $\underline{k} \in [1\!:\!N-2]$. Assume that there exists $k' \in [\underline{k}+1\!:\!N-1]$ such that $\overline{z} \in \mathcal{R}_{k'}^J$. Then, for each $i \in J$, there exists a feasible trajectory of length N from z^0 to $\mathcal{Z}^i$ of the form $x_1, \ldots, x_{k'-1}, \overline{z}, x_{k'+1}^i, \ldots, x_N^i$. Therefore, $x_k \in \mathcal{R}_k^J$ for each $k \in [1\!:\!k']$, which contradicts $\mathcal{R}_{\underline{k}}^J = \varnothing$. □

Remark 14. *The definitions of the constrained forward and backward reachable sets only consider pointwise (in time) constraints on the state and control input. Constraints based on the cumulative*

value of a function of state or control input evaluated for the entire trajectory can also be considered in Definitions 2 and 3, by appending the system dynamics with additional states. This approach turns a cumulative constraint on the trajectory and control input sequence into a terminal constraint on the additional state. For instance, the following constraint

$$\sum_{k=1}^{N} l(x_k, u_k) \leq l_{\max}, \tag{4.3}$$

where l is a stage cost function and $l_{\max} \in \mathbb{R}$ is an upper bound, is transformed by augmenting the original system (4.1) as follows

$$\tilde{x}_{k+1} = \begin{bmatrix} x_{k+1} \\ \theta_{k+1} \end{bmatrix} = \tilde{f}(\tilde{x}_k, u_k) \triangleq \begin{bmatrix} f(x_k, u_k) \\ \theta_k + l(x_k, u_k) \end{bmatrix}. \tag{4.4}$$

Suppose $\mathcal{Z}$ is the original target. By requiring an augmented state trajectory $\tilde{x}_k$, for $k \in [1\!:\!N]$, to reach the augmented target $\tilde{\mathcal{Z}} \triangleq \mathcal{Z} \times \{\theta \in \mathbb{R} \,|\, \theta \leq l_{\max}\}$ with initial condition $\theta_1 = 0$, we can satisfy (4.3).

It is natural that the goal of deferring decision (or procrastination) can be at odds with the goal of managing the onboard power and fuel resource: long duration maneuvers that keep targets reachable will consume more fuel. Hence, it is necessary to impose constraints based on the cumulative cost of a trajectory.

Remark 15. *The trajectories to each of the targets can have different final time (even if they have the same lengths) via the time-dilation approach, which is described in Appendix H*

The framework, analyses and solution methods presented in this paper can be extended to time-varying dynamical systems with minor modifications. The goal of DDTO is to generate trajectories that ensure reachability to maximum number of targets for as long as possible. We explore two different modeling approaches:

1. *Constrained reachability:* generate a trajectory that stays within the intersection of k-reach sets of a collection of targets.

2. *Cardinality minimization:* minimize the number of non-zeros in the pairwise difference

between trajectories at each time instant.

Assumption 2. *The dynamical system (4.1), initial state z^0, targets: $\mathcal{Z}^j$, for $j \in [0:n]$, and constraint sets $\mathbb{X}$ and $\mathbb{U}$, are chosen such that:*

- $\mathcal{R}_k^j \neq \varnothing$, *for each* $j \in [1:n]$ *and* $k \in [1:N]$.
- $\mathcal{R}_N^{[1:n]} \neq \varnothing$.

4.3 Constrained Reachability

We seek a trajectory of length N to one of the targets, and for each $k \in [1:N]$, we consider $(N-k)$-step constrained reachability to other targets from the state at time k on the trajectory.

4.3.1 Maximize duration of reachability to a collection of targets

Definition 8 ***(Branch time).*** *Given $J \subseteq [1:n]$, the branch time k^J is the latest time until which targets in J are reachable from a trajectory of length N starting from z^0, i.e.,*

$$k^J \triangleq \max\{\, k \in [1:N] \mid \mathcal{R}_k^J \neq \varnothing \,\}.$$

Since $\mathcal{R}_k^J \neq \varnothing$ iff $J \in \Lambda_k$, the branch time can also be defined as: $k^J \triangleq \max\{\, k \in [1:N] \mid J \in \Lambda_k\}$.

Corollary 4. *Trajectories of length N to targets in J cannot simultaneously intersect at any time later than the branch time.*

4.3.2 Maximize reachable targets from trajectory to particular target

Given $i \in [1:n]$, a solution to (4.5) consists of a trajectory to target i which maximizes the number of reachable targets at each time instant, i.e., the trajectory selects the largest

possible member of Λ_k, for each $k \in [1\!:\!N-1]$.

$$\max_{x_k, J_k} \sum_{t=1}^{N-1} |J_k| \tag{4.5a}$$

$$\text{s.t.} \quad x_k \in \mathcal{F}_1(x_{k-1}) \bigcap \mathcal{B}^{J_k}_{N-k}, \qquad k \in [2\!:\!N] \tag{4.5b}$$

$$J_k \in \Lambda_k, \qquad k \in [1\!:\!N] \tag{4.5c}$$

$$x_1 = z^0,\ J_N = \{i\} \tag{4.5d}$$

4.3.3 Maximize reachable targets from trajectory to any target

A solution to (4.6) consists of a trajectory to a target in $[1\!:\!n]$ which maximizes number of reachable targets at each time instant, i.e., the trajectory will terminate at a target that allows it to select the largest possible member of Λ_k, for each $k \in [1\!:\!N-1]$.

$$\max_{x_k, J_k} \sum_{k=1}^{N-1} |J_k| \tag{4.6a}$$

$$\text{s.t.} \quad x_k \in \mathcal{F}_1(x_{k-1}) \bigcap \mathcal{B}^{J_k}_{N-k}, \qquad k \in [2\!:\!N] \tag{4.6b}$$

$$J_k \in \Lambda_k, \qquad k \in [1\!:\!N] \tag{4.6c}$$

$$x_1 = z^0,\ |J_N| = 1 \tag{4.6d}$$

Remark 16. *We obtain* (4.6) *from* (4.5) *by treating* $i \in [1\!:\!n]$ *in* (4.5d) *as a decision variable, i.e., the choice of target is unspecified in* (4.6)*. The boundary conditions* (4.6d) *ensure that a feasible trajectory for* (4.6) *will terminate at any one of the targets in* $[1\!:\!n]$ *at* $k = N$*.*

Lemma 8 ***(Monotonicity).*** *The sequence of sets* J_k*, for* $k \in [1\!:\!N]$*, which form a solution to* (4.6) *must satisfy* $J_k \subseteq J_{k-1}$*, for* $k \in [2\!:\!N]$*.*

Proof. Let x_k, J_k, for $k \in [1\!:\!N]$, be a solution to (4.6). Suppose that $J_k \setminus J_{k-1} \neq \varnothing$ for some $k \in [2\!:\!N]$. Pick $j \in J_k \setminus J_{k-1}$. Then, x_k belongs to $\mathcal{R}^j_k$ from Lemma 6 and (4.6b). As a

consequence, from Lemma 7, x_{k-1} must be an element of $\mathcal{R}^j_{k-1}$. Therefore, J_{k-1} can be enlarged to include j. This contradicts the initial assumption that J_{k-1} forms a solution to (4.6). □

The monotonicity property described in Lemma 8 is also satisfied by solutions of (4.5). Further, a sequence of sets J_k, for $[1\!:\!N]$, which form a solution to (4.5) satisfy $i \in J_k$, for $k \in [1\!:\!N]$.

Definition 9 ***(Branch time/point).*** *Let x_k, J_k, for $k \in [1\!:\!N]$, be a solution to either (4.5) or (4.6). The latest time target $j \in [1\!:\!n]$ is reachable is a branch time, given by*

$$k^j \triangleq \max\{\, k \,|\, j \in J_k \,\}, \tag{4.7}$$

and the corresponding state, x_{k^j}, is a called a branch point.

Corollary 5. *Let x_k, J_k, for $k \in [1\!:\!N]$, be a solution to (4.5) or (4.6), with branch times given by (4.7). Since $x_{k^j} \in \mathcal{R}^j_{k^j}$, for each $j \in [1\!:\!n]$, there exists a trajectory of length $N - k^j + 1$ from x_{k^j} to z^j, denoted by x^j_k, for $k \in [k^j\!:\!N]$. Also, if $k^j > 1$, let $x^j_k \triangleq x_k$, for $k \in [1\!:\!k^j - 1]$. Then x^j_k, for $k \in [1\!:\!N]$, is a feasible trajectory to target j.*

The trajectories constructed in Corollary 5 are optimal in the sense of cardinality minimization, which we will explore in a subsequent section.

Remark 17. *Constraints (4.5b) and (4.6b), which ensure dynamic feasibility, cannot alternately be represented using 1-step constrained backward reachable sets. More precisely, a constraint such as*

$$x_k \in \mathcal{B}_1(\{x_{k+1}\}) \bigcap \mathcal{F}_{k-1}(z^0), \quad k \in [1\!:\!N-1],$$

along with a boundary condition $\{x_N\} = \mathcal{Z}^j$, for some $j \in [1\!:\!n]$, will result in a naïve single-target trajectory optimization problem which disregards reachability to other targets.

Remark 18. *If the implication in (4.2) is true, then (4.6) simplifies to*

$$\max_{J_k} \quad \sum_{k=1}^{N-1} |J_k| \tag{4.8a}$$

$$\text{s.t.} \quad J_k \in \Lambda_k, \qquad\qquad k \in [1\!:\!N] \tag{4.8b}$$

$$|J_N| = 1 \tag{4.8c}$$

where the trajectory x_k, for $k \in [1:N]$, is no longer an explicit decision variable. Then a solution to (4.8) selects the largest cardinality member of Λ_k, for each $k \in [1:N]$, and monotonicity (as described in Lemma 8) is not guaranteed to hold.

Remark 19. *A consequence of Remark 13 is that, a feasible trajectory to target j, for any $j \in [1:n]$, that selects J_k as the largest member of Λ_k which contains j, for each $k \in [1:N-1]$, might not exist.*

4.3.4 Prioritization

Let $\lambda^k \in [1:n]$, for $k \in [1:n]$, represent a priority order for the targets, with λ^1 having the highest priority and λ^n the least. Define a sequence of target sets using the prioritization as follows

$$J^{k+1} \triangleq J^k \setminus \{\lambda^{n-k+1}\},$$

for $k \in [1:n-1]$, with $J^1 = [1:n]$, and collect them into $\mathcal{J} \triangleq \{J^1, \ldots, J^n\}$.

A solution to (4.9) consists of a trajectory to target λ^1 which maximizes the number of reachable targets at each time instant while maintaining the target prioritization.

$$\max_{x_k, J_k} \quad \sum_{k=1}^{N-1} |J_k| \tag{4.9a}$$

$$\text{s.t.} \quad x_k \in \mathcal{F}_1(x_{k-1}) \bigcap \mathcal{B}_{N-k}^{J_k}, \qquad k \in [2:N] \tag{4.9b}$$

$$J_k \in \Lambda_k \bigcap \mathcal{J}, \qquad k \in [1:N] \tag{4.9c}$$

$$x_1 = z^0, \ |J_N| = 1 \tag{4.9d}$$

Note that (4.6) and (4.9) differ only in (4.6c) and (4.9c): the latter contains an extra intersection with $\mathcal{J}$ to enforce priority.

Lemma 9. *A solution to (4.9) satisfies the monotonicity property in Lemma 8.*

Proof. Let J_k, for $k \in [1:N]$, form a solution to (4.9). Note that all elements of $\mathcal{J}$ are ordered,

i.e., for any $J, J' \in \mathcal{J}$, either $J \subset J'$ or $J' \subset J$ holds. Suppose that $J_{k-1} \subset J_k$, for some $k \in [2\!:\!N]$, and denote $\bar{J} = J_k$. Then, setting $J_{k-1} = \bar{J}$ is feasible with respect to (4.9) (due to Lemma 7) and it increases the objective function value. This contradicts the assumption that $J_{k-1} \subset J_k$. □

Corollary 6. *A solution to* (4.9) *satisfies the prioritization of target and the branch times are ordered, i.e.,*

$$k^{\lambda^n} \leq \ldots \leq k^{\lambda^1}.$$

Proof. Observe that J_k, for $k \in [1\!:\!N]$, which form a solution to (4.9) satisfy the monotonicity property (Lemma 9). Furthermore, $J_k \in \mathcal{J}$, for $k \in [1\!:\!N]$, with $J_1 = [1\!:\!n]$ and $J_N = \{\lambda^1\}$. As a result, the targets are removed from J_k with the correct priority as k increases and the branch times are ordered. □

4.4 Cardinality Minimization

In this section we show that the constrained-reachability-based descriptions for DDTO can be equivalently stated as cardinality minimization problems.

4.4.1 Maximize duration of reachability to a collection of targets

The problem of maximizing the time duration for which targets in $J \subseteq [1\!:\!n]$ are reachable is the same as the problem of generating trajectories to targets in J which stay identical for as long as possible, which can be described as follows

$$\max_{x_k^j} \ \mathfrak{g}^{|J|}(X^J) \tag{4.10a}$$

$$\text{s.t.} \ \ x_{k+1}^j = f(x_k^j, u_k^j), \qquad k \in [1\!:\!N-1],\, j \in J \tag{4.10b}$$

$$x_k^j \in \mathbb{X},\ u_k^j \in \mathbb{U}, \qquad k \in [1\!:\!N-1],\, j \in J \tag{4.10c}$$

$$x_1^j = z^0,\ x_N^j \in \mathcal{Z}^j, \qquad j \in J \tag{4.10d}$$

where X^J concatenates all trajectories and $\mathfrak{g}^m : \mathbb{R}^{mn_x N} \to \mathbb{R}$ evaluates the maximum duration that given m trajectories stay identical, starting from $k = 1$. In particular, we have that

$$\mathfrak{g}^{|J|}(X^J) \triangleq \max \left\{ k' \in [1\!:\!N] \,\middle|\, x_k^j = x_k^i,\ \forall\, k \in [1\!:\!k'],\ \forall\, i,j \in J \right\}. \tag{4.11}$$

Theorem 7. *The optimal value of* (4.10) *is the branch time* k^J *from Definition 8.*

Proof. Let $k^\star$ be the optimal value of (4.10). Suppose that $k^J < k^\star < N$. Then, $z^\star = x_{k^\star}^i \in \mathcal{R}_{k^\star}^j$, for each $i, j \in J$. Hence, $z^\star \in \mathcal{R}_{k^\star}^J$, which contradicts $\mathcal{R}_k^J = \varnothing$, for $k > k^J$.

Next, suppose that $k^\star < k^J < N$. Since $\mathcal{R}_{k^J}^J \neq \varnothing$, let $z^\star \in \mathcal{R}_{k^J}^J$. Then, there exist trajectories $x_1, \ldots, x_{k^J-1}, z^\star, x_{k^J+1}^j, \ldots, x_N^j$, for $j \in J$, with $x_1 = z^0$ and $x_N \in \mathcal{Z}^j$, which are feasible with respect to (4.10). Note that the states of these trajectories stay identical until time k^J, which contradicts the assumption that $k^\star < k^J$. □

The above result also validates Corollary 4.

4.4.2 Maximize reachable targets from trajectory to particular target

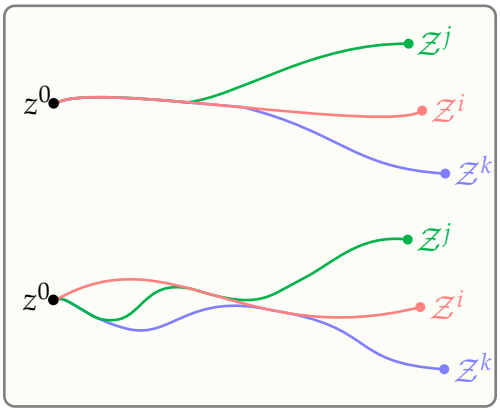


Figure 4.3: Trajectories forming a tree-like structure (shown above) are optimal whereas the trajectories with irregular clumping (shown below) are not.

Given $i \in [1\!:\!n]$, the problem of maximizing the number of reachable targets at each

time instant from a trajectory to target i can be described as follows

$$\min_{x_k^j} \quad \sum_{\substack{j\in[1:n]\\ j\neq i}} \sum_{k=1}^{N-1} \|x_k^i - x_k^j\|_\blacklozenge \tag{4.12a}$$

$$\text{s.t.} \quad x_{k+1}^j = f(x_k^j, u_k^j), \qquad k \in [1:N-1],\, j \in J \tag{4.12b}$$

$$x_k^j \in \mathbb{X},\, u_k^j \in \mathbb{U}, \qquad k \in [1:N-1],\, j \in J \tag{4.12c}$$

$$x_1^j = z^0,\, x_N^j \in \mathcal{Z}^j, \qquad j \in J \tag{4.12d}$$

Lemma 10. *Let trajectories x_k^j, for $k \in [1:N]$ and $j \in [1:n]$, form a solution to (4.12), and let*

$$J_k \triangleq \left\{ j \in [1:n] \,\middle|\, x_k^j = x_k^i \right\}, \quad k \in [1:N]. \tag{4.13}$$

Then the objective function (4.12a) satisfies

$$\sum_{\substack{j\in[1:n]\\ j\neq i}} \sum_{k=1}^{N-1} \|x_k^i - x_k^j\|_\blacklozenge + \sum_{k=1}^{N-1} |J_k| = n(N-1). \tag{4.14}$$

Proof. The result follows from the observation that

$$\sum_{\substack{j\in[1:n]\\ j\neq i}} \|x_k^i - x_k^j\|_\blacklozenge = n - |J_k|$$

for each $k \in [1:N-1]$. □

Lemma 11. *The monotonicity property in Lemma 8 holds for target sets defined in (4.13), i.e., $J_k \subseteq J_{k-1}$, for $k \in [2:N]$.*

Proof. Suppose that $j \in J_k \setminus J_{k-1}$, for some $k \in [2:N-1]$. Then, $x_{k-1}^i \neq x_{k-1}^j$ and $x_k^i = x_k^j$. Replacing the trajectory to target j with $x_1^i, \ldots, x_{k-1}^i, x_k^i, x_{k+1}^j, \ldots, x_N^j$, with $x_1^i = z^0$ and $x_N^i \in \mathcal{Z}^i$, reduces the objective function value. Therefore, we must have $J_k \setminus J_{k-1} = \varnothing$. □

Theorem 8. *If x_k^j, for $k \in [1:N]$ and $j \in [1:n]$, form a solution to (4.12) and sets J_k, for $k \in [1:N]$, are constructed via (4.13), then x_k^i, J_k, for $k \in [1:N]$, form a solution to (4.5). Conversely, if x_k, J_k,*

for $k \in [1:N]$, form a solution to (4.5), then Corollary 5 provides a solution to (4.12).

Proof. Let x_k^j, for $k \in [1:N]$ and $j \in [1:n]$, solve (4.12), and let $\bar{x}_k$, $\bar{J}_k$, for $k \in [1:N]$, solve (4.5). Then, we apply Corollary 5 to construct $\bar{x}_k^j$, for $k \in [1:N]$ and $j \in [1:n]$, which is feasible with respect to (4.12). Suppose that x_k^i, J_k, for $k \in [1:N]$, is feasible but not optimal with respect to (4.5). Therefore,

$$\sum_{k=1}^{N-1} |J_k| < \sum_{k=1}^{N-1} |\bar{J}_k|$$

From the equivalence of the objective functions (4.5a) and (4.12a) shown via (4.14), we conclude that $\bar{x}_k^j$ leads to a strictly lower value of (4.12a) than x_k^j. A contradiction.

Next, let x_k, J_k, for $k \in [1:N]$, solve (4.5), and let $\bar{x}_k^j$, for $k \in [1:N]$ and $j \in [1:n]$, solve (4.12). Then, $\bar{x}_k^i$, $\bar{J}_k$, for $k \in [1:N]$, is feasible with respect to (4.5), where $\bar{J}_k$ is computed using (4.13). Suppose that the trajectories constructed via Corollary 5 are feasible but not optimal with respect to (4.12). Therefore,

$$\sum_{\substack{j \in [1:n] \\ j \neq i}} \sum_{k=1}^{N-1} \|\bar{x}_k^i - \bar{x}_k^j\|_\blacklozenge < \sum_{\substack{j \in [1:n] \\ j \neq i}} \sum_{k=1}^{N-1} \|x_k^i - x_k^j\|_\blacklozenge$$

From the equivalence of the objective functions (4.5a) and (4.12a) shown via (4.14), we conclude that $\bar{x}_k^i$, $\bar{J}_k$ lead to a strictly greater value of (4.5a) than x_k, J_k. A contradiction. □

Remark 20 ***(Convex relaxation).*** *The objective function (4.12a) can be equivalently expressed using 0-norm as follows*

$$\sum_{\substack{j \in [1:n] \\ j \neq i}} \sum_{k=1}^{N-1} \|x_k^i - x_k^j\|_\blacklozenge =$$
$$\sum_{k=1}^{N-1} \|\left(\|x_k^i - x_k^1\|_p, \ldots, \|x_k^i - x_k^n\|_p\right)\|_0.$$

where $\|\square\|_p$ *denotes p-norm for some* $p \geq 1$. *Then a convex relaxation of* (4.12a) *is given by*

$$\sum_{k=1}^{N-1} \|\left(\|x_k^i - x_k^1\|_p, \ldots, \|x_k^i - x_k^n\|_p\right)\|_1 = \sum_{k=1}^{N-1} \sum_{\substack{j\in[1:n]\\ j\neq i}} \|x_k^i - x_k^j\|_p.$$

In practice, choosing $p = 1$ *is desirable. That would encourage sparsity of the vector of concatenated states:* $(x_1^i - x_1^j, \ldots, x_N^i - x_N^j)$, *for each* $j \in [1:n]$.

4.4.3 Maximize reachable targets from trajectory to any target

We can generalize (4.12) by optimizing over $i \in [1:n]$ to pick target $i^\star$. Among all targets, a trajectory to target $i^\star$ can have the maximum possible number of reachable targets at each time instant.

$$\text{min.} \min_{i\in[1:n]} \sum_{\substack{j\in[1:n]\\ j\neq i}} \sum_{k=1}^{N-1} \|x_k^i - x_k^j\|_\blacklozenge \tag{4.15a}$$

$$\text{s.t. } x_{k+1}^j = f(x_k^j, u_k^j), \qquad k \in [1:N-1],\ j \in J \tag{4.15b}$$

$$x_k^j \in \mathbb{X},\ u_k^j \in \mathbb{U}, \qquad k \in [1:N-1],\ j \in J \tag{4.15c}$$

$$x_1^j = z^0,\ x_N^j \in \mathcal{Z}^j, \qquad j \in J \tag{4.15d}$$

Given trajectories denoted by x_k^j, for $k \in [1:N]$ and $j \in [1:n]$, which solve (4.15), let

$$J_k^\star \triangleq \left\{ j \in [1:n] \,\middle|\, x_k^j = x_k^{i^\star} \right\}, \quad k \in [1:N], \tag{4.16}$$

with

$$i^\star \in \operatorname*{argmin}_{i\in[1:n]} \sum_{\substack{j\in[1:n]\\ j\neq i}} \sum_{k=1}^{N} \|x_k^i - x_k^j\|_\blacklozenge. \tag{4.17}$$

Remark 21. *Solving* (4.15) *is equivalent to solving* (4.12) *for each target* $i \in [1\!:\!n]$ *and picking the one which leads to least value for* (4.12a). *Similarly, solving* (4.6) *is equivalent to solving* (4.5) *for each* $i \in [1\!:\!n]$ *in the boundary condition* (4.5d), *and selecting the best solution.*

Theorem 9. *Let* x_k^j, *for* $k \in [1\!:\!N]$ *and* $j \in [1\!:\!n]$, *solve* (4.15). *Sets* $J_k^\star$, *for* $k \in [1\!:\!N]$, *and* $i^\star$ *are given by* (4.16) *and* (4.17), *respectively. Then* $x_k^{i^\star}, J_k^\star$, *for* $k \in [1\!:\!N]$, *solves* (4.6). *Conversely, if* x_k, J_k, *for* $k \in [1\!:\!N]$, *solves* (4.6), *then there exists a solution to* (4.15) *where* x_k, *for* $k \in [1\!:\!N]$, *is the trajectory to target* $i^\star$.

Proof. With the observation in Remark 21 the proof is similar to that of Theorem 8. □

Remark 22. *Given a solution to* (4.6), *we can construct a solution for* (4.15) *by following the procedure in Corollary* 5, *after noting that the single element contained in* J_N *can be denoted by* i, *i.e., the target where the trajectory which solves* (4.6) *terminates.*

Remark 23. *The branch times and branch points (described in Definition* 9*) can be computed for the solutions to* (4.12) *and* (4.15) *using the targets sets in* (4.13) *and* (4.16), *respectively. The branch time* k^j *is the time after which the trajectory to target* j *"detaches" from the trajectory to target* i *(or* $i^\star$*).*

We have established a tree-like structure for the trajectories that solve the optimization problems in Section 4.4 and highlighted their connection to the monotonicity of sets J_k, for $k \in [1\!:\!N]$, which solve the optimization problem in Section 4.3. In other words, it is not optimal for trajectories to clump together arbitrarily as shown in Figure 4.3.

4.5 Solution Method

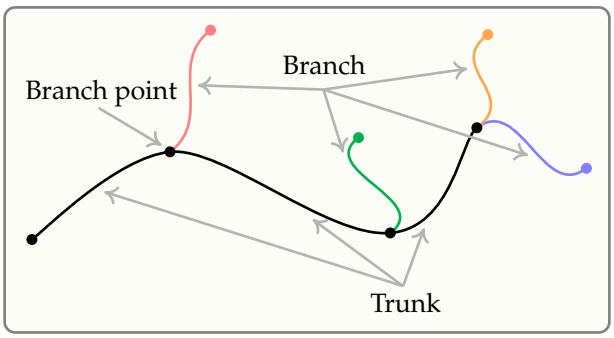


Figure 4.4: Algorithms 4 and 5 recursively compute trunk and branch trajectories connected by branch points while adhering to the given target prioritization.

This section describes a solution method for DDTO that mimics the tree-like structure of the solutions in Section 4.4 (equivalently, the monotonicity of target sets in Section 4.3). The problem of computing the latest branch time is numerically more tractable than the rest of the problems in Section 4.4. This is due to the challenge of obtaining a tight, continuous approximation for the objective functions (4.12a) and (4.15a). Recall that any p-norm is valid for the approximation provided in Remark 20. However, since (4.10) is agnostic to prioritization of targets, we adopt a recursive solution method. In particular, we recursively compute the latest branch time (by solving a problem similar to (4.10)) for a shrinking collection of targets. At each branch time, at least one target is rejected based on a given priority order. Two consecutive branch points are connected by a trajectory segment called *trunk*, where the trajectories to a collection of targets stay identical. At each branch time, the number of trajectories in the trunk reduces by at least one and the trajectory segmented which "detaches" from the trunk to terminate at the "rejected" target, is called a *branch*. The sequence of branches and trunks possess a tree-like structure. See Figure 4.4 for an illustration of branches and trunks. Note that it is possible for two successive branch times to coincide. In which case, more than one target is rejected at the same branch time.

The computation of the latest branch time and the corresponding trajectories can be specialized as: (i) DDTO-QCVX for discrete-time affine systems with convex constraints on the state and control input, and (ii) DDTO-SCP for continuous-time nonlinear systems with nonconvex constraints via a trajectory optimization framework based on sequential convex programming [22]. The complete solution methods for the two specialization are presented in Algorithms 4 and 5. We refer the reader to [26] for an extension to Algorithm 4 with adaptive update of the constraints on cumulative trajectory cost.

4.5.1 DDTO-QCVX

Lemma 12. *For an affine dynamical system subject to convex constraints on the state and input,* (4.10) *is a quasiconvex optimization problem, i.e., when f is an affine function, constraint sets $\mathbb{X}$, $\mathbb{U}$, and targets $\mathcal{Z}^j$, for $j \in [1\!:\!n]$, are closed and convex.*

Proof. When f is an affine function, $\mathbb{X}$ and $\mathbb{U}$ are closed convex sets, the feasible set of

(4.10) is convex. The objective function (4.10a) is quasiconcave because its superlevel sets are convex sets (hyperplane), i.e.,

$$\begin{aligned} & \mathfrak{g}^{|J|}(X^J) \geq k^\star \\ \iff & x_k^j = x_k^i,\ k \in [1{:}k^\star],\ i,j \in J \\ \iff & \sum_{m=1}^{k-1} A^{k-1-m} B(u_m^j - u_m^i) = 0,\ k \in [1{:}k^\star],\ i,j \in J \end{aligned}$$

□

We can efficiently solve (4.10) via bisection method, which solves a sequence of convex feasibility problems [142, Sec. 3], [26, Alg. 1]. The sequence is guaranteed to terminate within a fixed number of iterations.

In the development thus far, we assumed that the trajectories to all targets have the same length, with the understanding that embedded time-dilation (Appendix H) can allow for different final times for each of the trajectories. However, time-dilation introduces nonlinearity in the system dynamics. So, to avoid time-dilation and preserve convexity, we allow the trajectories to different targets to be of different lengths. Similarly, since the cumulative trajectory constraint function is typically nonlinear, we avoid augmenting such a constraint into the system dynamics (with the approach in Remark 14) to preserve convexity. Then, under the DDTO-QCVX specialization, the solution method described in Algorithm 4 recursively solves the following quasiconvex problem.

$$\begin{aligned} \max.\ & \mathfrak{g}^{|J|}(X^J) && && \text{(4.18a)} \\ \text{s.t.}\ & x_{k+1}^j = Ax_k^j + Bu_k^j + c, && k \in [1{:}M^j{-}1],\ j \in J && \text{(4.18b)} \\ & x_k^j \in \mathbb{X},\ u_k^j \in \mathbb{U}, && k \in [1{:}M^j{-}1],\ j \in J && \text{(4.18c)} \\ & \sum_{k=1}^{M^j-1} l(x_k^j, u_k^j) \leq l_{\max}, && j \in J && \text{(4.18d)} \\ & x_1^j = z^0,\ x_N^j \in \mathcal{Z}^j, && j \in J && \text{(4.18e)} \end{aligned}$$

where $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$, and $c \in \mathbb{R}^{n_x}$ define the affine dynamical system, and the

trajectories to the targets in J are concatenated into vector X^J. The target set $J \subset [1\!:\!n]$, the lengths of the trajectories, M^j, for $j \in J$, and the initial state, z^0, are updated after each branch time computation. The optimal value of (4.18) is a branch time and the segment of any of the trajectories until the branch time forms a trunk.

4.5.2 DDTO-SCP

Consider a continuous-time nonlinear dynamical system

$$\dot{x}(t) = F(x(t), u(t)), \quad t \in [0, t_\mathrm{f}]. \tag{4.19}$$

The state and input constraints: $x(t) \in \mathbb{X}$ and $u(t) \in \mathbb{U}$ can be re-expressed as path constraints: $g_i(x(t), u(t)) \leq 0$, for $i \in [1\!:\!n_g]$, and $h_j(x(t), u(t)) = 0$, for $j \in [1\!:\!n_h]$, where g_i and h_j are scalar-valued functions. We consider a free-final-time optimal control problem, i.e., the final time t_f is a decision variable. So, we adopt time-dilation [22, Sec. 2.4] which treats the actual time t as a continuously differentiable, strictly increasing mapping from a known normalized interval $[0, 1]$ to the actual time interval $[0, t_\mathrm{f}]$. The dilation factor, given by

$$s(\tau) \triangleq \overset{\circ}{t}(\tau) = \frac{\mathrm{d}t(\tau)}{\mathrm{d}\tau}, \quad \tau \in [0, 1], \tag{4.20}$$

is treated as an additional control input. Next, the path constraints are subjected to the isoperimetric reformulation [22, Sec. 2.3]. For any $i \in [1\!:\!n_g]$ and $j \in [1\!:\!n_h]$, we have that

$$\begin{aligned} & g_i(x(t), u(t)) \leq 0,\ h_j(x(t), u(t)) = 0, \quad \forall\, t \in [0, t_\mathrm{f}], \\ \iff & \int_0^{t_\mathrm{f}} |g_i(x(t), u(t))|_+^2 + h_j(x(t), u(t))^2 \mathrm{d}t = 0. \end{aligned}$$

For each $\tau \in [0, 1]$, we augment the state and control input as follows

$$\begin{aligned} \tilde{x}(\tau) &\triangleq (x(t(\tau)), y(\tau), t(\tau)), \\ \tilde{u}(\tau) &\triangleq (u(t(\tau)), s(\tau)), \end{aligned}$$

to obtain the following augmented dynamical system defined over interval $[0,1]$

$$\overset{\circ}{\tilde{x}} = s \begin{bmatrix} F(x,u) \\ \sum_{i=1}^{n_g} |g_i(x,u)|_+^2 + \sum_{j=1}^{n_h} h_j(x,u)^2 \\ 1 \end{bmatrix} = \tilde{F}(\tilde{x},\tilde{u}). \tag{4.21}$$

Imposing periodicity boundary conditions on y is equivalent to satisfying the path constraints in continuous-time. The above constraint reformulation approach is especially useful within direct methods for trajectory optimization to avoid inter-sample constraint violation which is commonly encountered after discretization and imposing path constraint at finitely many node points. We refer the reader to [22] for detailed discussions.

Given a target set $J \in [1\!:\!n]$, initial state z^0, and trajectory length M, DDTO-SCP considers the following continuous-time optimal control problem for $|J|+1$ trajectories: trunk trajectory, x^0, defined over interval $[0, t_{\mathrm{f}}^0]$, and branch trajectories, x^j, defined over interval $[0, t_{\mathrm{f}}^j]$, for $j \in J$.

$$\begin{aligned}
\max.\ & {}^{t}E\,\tilde{x}^0(1) && && \text{(4.22a)}\\
\text{s.t.}\ & \overset{\circ}{\tilde{x}}^j(\tau) = \tilde{F}(\tilde{x}^j(\tau), \tilde{u}^j(\tau)), && \tau \in [0,1],\, j \in \check{J} && \text{(4.22b)}\\
& \tilde{u}^j(\tau) \in \tilde{\mathcal{U}}, && \tau \in [0,1],\, j \in \check{J} && \text{(4.22c)}\\
& {}^{y}E\,\tilde{x}^j(0) = {}^{y}E\,\tilde{x}^j(1), && j \in \check{J} && \text{(4.22d)}\\
& {}^{x}E\,\tilde{x}^j(0) = {}^{x}E\,\tilde{x}^0(1), && j \in J && \text{(4.22e)}\\
& {}^{x}E\,\tilde{x}^0(0) = z^0, && && \text{(4.22f)}\\
& P^j({}^{x}E\,\tilde{x}^j(1)) \leq 0,\ Q^j({}^{x}E\,\tilde{x}^j(1)) = 0, && j \in J && \text{(4.22g)}
\end{aligned}$$

where $\check{J} = J \cup \{0\}$. Since the sets $\mathcal{Z}^j$, for $j \in J$, can be nonconvex, we express the boundary condition constraints due to targets using continuously differentiable constraint functions $P^j : \mathbb{R}^{n_x} \to \mathbb{R}^{n_P}$ and $Q^j : \mathbb{R}^{n_x} \to \mathbb{R}^{n_Q}$. Note that a cumulative constraint on the trajectory is transformed to a terminal constraint in (4.22g) through an approach similar to that in Remark 14. By construction, t_{f}^0 is the branch time and the goal is to maximize it. Convex,

compact set $\tilde{\mathcal{U}}$ can encode convex constraints on the control input. We use selector matrices ${}^{x}E$, ${}^{y}E$, and ${}^{t}E$ to select components of $\tilde{x}$ corresponding to x, y, and t, respectively. We assume that all functions appearing in (4.22) are continuously differentiable. Note that in contrast to DDTO-QCVX, there is an explicit distinction between the trunk and branch trajectories in (4.22).

Given a solution to (4.22), the trajectory from z^0 to target $j \in \check{J}$, defined over interval $[0, t_{\mathrm{f}}^0 + t_{\mathrm{f}}^j]$, is obtained as follows

$$x^j(t) \triangleq \begin{cases} {}^{x}E\,\tilde{x}^0(({}^{t}E\,\tilde{x}^0)^{-1}(t)) & \text{if } t \in [0, t_{\mathrm{f}}^0], \\ {}^{x}E\,\tilde{x}^j(({}^{t}E\,\tilde{x}^j)^{-1}(t - t_{\mathrm{f}}^0)) & \text{if } t \in (t_{\mathrm{f}}^0, t_{\mathrm{f}}^0 + t_{\mathrm{f}}^j], \end{cases}$$

where $t_{\mathrm{f}}^j = {}^{t}E\,\tilde{x}^j(1)$. Note that ${}^{t}E\,\tilde{x}^j : [0,1] \to [0, t_{\mathrm{f}}^j]$ is invertible since it is strictly increasing.

Next, to numerically solve (4.22), we discretize $[0,1]$ with a uniformly-spaced grid of length M: $0 = \tau_1 < \ldots < \tau_M = 1$. For each $j \in \check{J}$, the values of the augmented state and control input at the nodes of the grid: $\tilde{x}_k^j$, for $k \in [1\!:\!M]$, and $\tilde{u}_k^j$, for $k \in [1\!:\!M-1]$, respectively, are treated as decision variables. We use a zero-order-hold parameterization for the control input, $\tilde{v}^j : [0,1] \to \mathbb{R}^{n_u+1}$, defined as

$$\tilde{v}^j(\tau) \triangleq \begin{cases} \tilde{u}_k^j & \text{if } \tau \in [\tau_k, \tau_{k+1}) \text{ for some } k \in [1\!:\!M-1], \\ \tilde{u}_{M-1}^j & \text{if } \tau = \tau_M. \end{cases}$$

Then for each $k \in [1\!:\!M-1]$ and $j \in \check{J}$, the exact discretization of (4.22b) is

$$\tilde{x}_{k+1}^j = \tilde{f}(\tilde{x}_k^j, \tilde{u}_k^j) \triangleq \tilde{x}_k^j + \int_{\tau_k}^{\tau_{k+1}} \tilde{F}({}^{k}\tilde{x}^j(\tau), \tilde{v}^j(\tau))\mathrm{d}\tau, \tag{4.23}$$

where ${}^{k}\tilde{x}^j$ is a trajectory for (4.22b) over $[\tau_k, \tau_{k+1}]$ with control input $\tilde{v}^j$ and initial condition $\tilde{x}_k^j$.

We obtain the following nonconvex optimization problem after time-dilation, constraint

reformulation, augmented control input parameterization, and discretization.

$$
\begin{aligned}
\max. \quad & {}^{t}E\,\tilde{x}_M^0 && && \text{(4.24a)}\\
\text{s.t.} \quad & \tilde{x}_{k+1}^j = \tilde{f}(\tilde{x}_k^j, \tilde{u}_k^j), && k \in [1\!:\!M\!-\!1],\ j \in \check{J} && \text{(4.24b)}\\
& \tilde{u}_k^j \in \tilde{\mathcal{U}}, && k \in [1\!:\!M\!-\!1],\ j \in \check{J} && \text{(4.24c)}\\
& {}^{y}E\,(\tilde{x}_{k+1}^j - \tilde{x}_k^j) \leq \epsilon, && k \in [1\!:\!M\!-\!1],\ j \in \check{J} && \text{(4.24d)}\\
& {}^{x}E\,\tilde{x}_1^j = {}^{x}E\,\tilde{x}_M^0, && j \in J && \text{(4.24e)}\\
& {}^{x}E\,\tilde{x}_1^0 = z^0, && && \text{(4.24f)}\\
& P^j({}^{x}E\,\tilde{x}_M^j) \leq 0,\ Q^j({}^{x}E\,\tilde{x}_M^j) = 0, && j \in J && \text{(4.24g)}
\end{aligned}
$$

The augmented trajectory from z^0 to target $j \in J$ is represented with $\tilde{x}_1^0, \ldots, \tilde{x}_M^0, \tilde{x}_2^j, \ldots, \tilde{x}_M^j$. The corresponding discrete-time trajectory and control input sequence are given by

$$
x_k^j \triangleq \begin{cases} {}^{x}E\,\tilde{x}_k^0 & \text{if } k \in [1\!:\!M], \\ {}^{x}E\,\tilde{x}_{k-M+1}^j & \text{if } k \in [M+1\!:\!N], \end{cases} \tag{4.25a}
$$

$$
u_k^j \triangleq \begin{cases} {}^{u}E\,\tilde{u}_k^0 & \text{if } k \in [1\!:\!M-1], \\ {}^{u}E\,\tilde{u}_{k-M+1}^j & \text{if } k \in [M\!:\!N-1], \end{cases} \tag{4.25b}
$$

where $N \triangleq 2M-1$ is the length of the overall trajectory. Note that the equality constraint (4.22d) is relaxed to (4.24d) to avoid automatic violation of constraint qualifications (see [22, Sec. 3.1]). We solve (4.24) using CT-SCVX, an SCP-based nonconvex trajectory optimization framework [22].

4.5.3 Algorithm

The DDTO solution methods, described in Algorithms 4 and 5, recursively solve (4.18) and (4.24), respectively. Both methods are myopic in their computation of branch times, i.e., they are computed sequentially instead of simultaneously, as would be the case if (4.9) is directly solved. Furthermore, after each branch time computation, the cumulative

trajectory constraint must be updated to account for the contribution due to previous trunk segment (i.e., $l_{\max}$ in (4.18d) must be updated). On the other hand, besides handling target prioritization, the recursive nature of the solution methods is beneficial within a closed-loop simulation (as demonstrated in [126]) wherein the computation of the latest branch time can act on any new information acquired about uncertainties by perception.

Algorithms 4 and 5 take initial state, target states, target prioritization, and trajectory length(s), to provide branch and trunk trajectories. The total trajectory length N passed to Algorithm 5 is an odd number.

Algorithm 4 DDTO-QCVX

Input: z^0, z^j, λ^j, N^j, for $j \in [1:n]$

$k^{\lambda^{n+1}} \leftarrow 1$, $J^1 \leftarrow [1:n]$
for $k \in [1:n-1]$ **do**
  $M^j \leftarrow N^j - k^{\lambda^{n-k+1}}$, for $j \in J_k$
  Solve (4.18) via bisection method for target set J^k with initial state z^0 and trajectory lengths M^j
  Store $k^{\lambda^{n-k+1}}$ ▷ *Branch time*
  ▷ *Trunk trajectory and control input sequence*
  Store x_k^0, for $k \in [k^{\lambda^{n-k+2}} : k^{\lambda^{n-k+1}}]$
  Store u_k^0, for $k \in [k^{\lambda^{n-k+2}} : k^{\lambda^{n-k+1}} - 1]$
  ▷ *Branch trajectory and control input sequence*
  Store $x_k^{\lambda^{n-k+1}}$, for $k \in [k^{\lambda^{n-k+1}} : N^{\lambda^{n-k+1}}]$
  Store $u_k^{\lambda^{n-k+1}}$, for $k \in [k^{\lambda^{n-k+1}} : N^{\lambda^{n-k+1}} - 1]$
  if $k = n-1$ **then**
    $k^{\lambda^1} \leftarrow k^{\lambda^2}$
    Store $x_k^{\lambda^1}$, for $k \in [k^{\lambda^2} : N^{\lambda^1}]$
    Store $u_k^{\lambda^1}$, for $k \in [k^{\lambda^2} : N^{\lambda^1} - 1]$
  end if
  $J^{k+1} \leftarrow J^k \setminus \{\lambda^{n-k+1}\}$ ▷ *Reject target*
  $z^0 \leftarrow x^0_{k^{n-k+1}}$ ▷ *Branch point*
  Update cumulative trajectory constraint
end for

Output: x_k^0, u_k^0, for $k \in [1:k^{\lambda^2}]$,
$x_k^{\lambda^j}$, $u_k^{\lambda^j}$, for $k \in [k^{\lambda^j} : N^{\lambda^j}]$, $j \in [1:n]$

Algorithm 5 DDTO-SCP

Input: z^0, z^j, λ^j, for $j \in [1\!:\!n]$, N

$t^{\lambda^{n+1}} \leftarrow 0$, $J^1 \leftarrow [1\!:\!n]$
$M \leftarrow N+1$
for $k \in [1\!:\!n-1]$ **do**
  $M \leftarrow \lceil M/2 \rceil$
  Solve (4.24) via CT-SCVX for target set J^k
  with initial state z^0 and trajectory length M
  $t^{\lambda^{n-k+1}} \leftarrow t^{\lambda^{n-k+2}} + {}^tE\,\tilde{x}^0(1)$ ▷ *Branch time*
  $t_{\mathrm{f}}^{\lambda^{n-k+1}} \leftarrow t^{\lambda^{n-k+1}} + {}^tE\,\tilde{x}^{\lambda^{n-k+1}}(1)$
  ▷ *Trunk trajectory and control input*
  Store $x^0(t)$, for $t \in [t^{\lambda^{n-k+2}}, t^{\lambda^{n-k+1}}]$
  Store $u^0(t)$, for $t \in [t^{\lambda^{n-k+2}}, t^{\lambda^{n-k+1}}]$
  ▷ *Branch trajectory and control input sequence*
  Store $x^{\lambda^{n-k+1}}(t)$, for $t \in [t^{\lambda^{n-k+1}}, t_{\mathrm{f}}^{\lambda^{n-k+1}}]$
  Store $u^{\lambda^{n-k+1}}(t)$, for $t \in [t^{\lambda^{n-k+1}}, t_{\mathrm{f}}^{\lambda^{n-k+1}}]$
  if $k = n-1$ **then**
    $t^{\lambda^1} \leftarrow t^{\lambda^2}$
    Store $x^{\lambda^1}(t)$, for $t \in [t^{\lambda^2}, t_{\mathrm{f}}^{\lambda^1}]$
    Store $u^{\lambda^1}(t)$, for $t \in [t^{\lambda^2}, t_{\mathrm{f}}^{\lambda^1}]$
  end if
  $J^{k+1} \leftarrow J^k \setminus \{\lambda^{n-k+1}\}$ ▷ *Reject target*
  $z^0 \leftarrow x^0(t^{n-k+1})$ ▷ *Branch point*
  Update cumulative trajectory constraint
end for

Output: $x(t)^0, u(t)^0$, for $t \in [0, t^{\lambda^2}]$,
$x^{\lambda^j}(t), u^{\lambda^j}(t)$, for $t \in [t^{\lambda^j}, t_{\mathrm{f}}^{\lambda^j}]$, $j \in [1\!:\!n]$

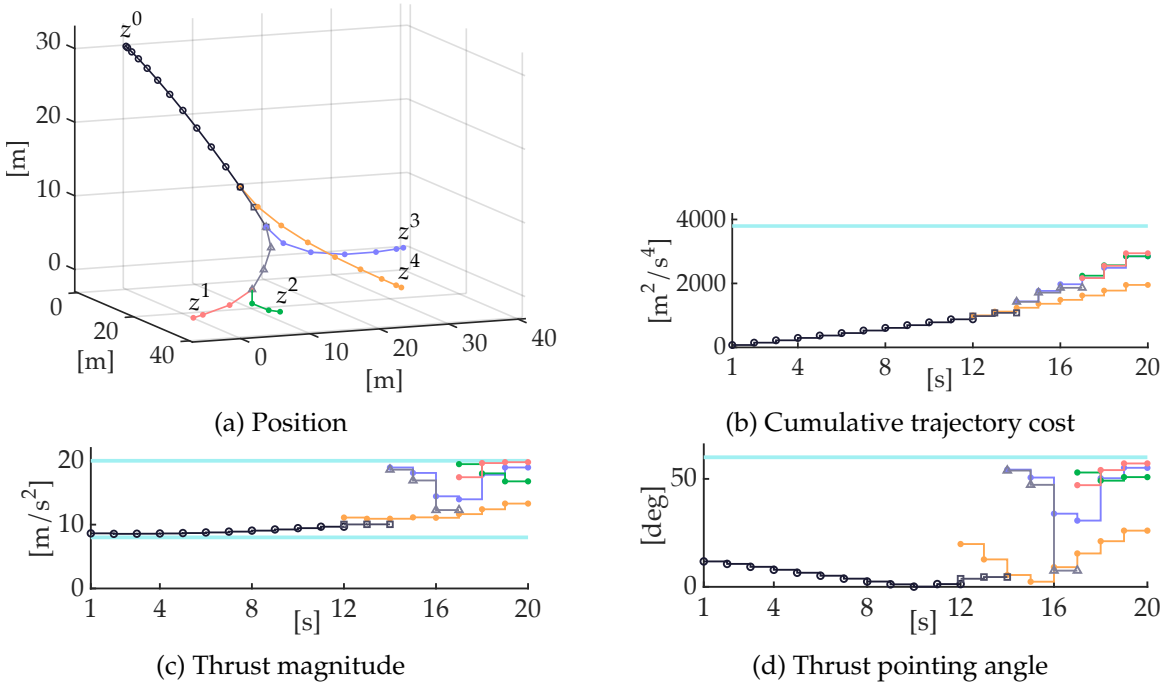
4.6 Numerical Results

This section demonstrates DDTO-QCVX and DDTO-SCP within Algorithms 4 and 5, respectively, using two optimal control applications based on quadrotor motion planning. To ensure reliable numerical performance of the solution methods, we scale the primal variables and path constraints functions so that the values that they take are of similar orders of magnitude. The code used to generate the numerical results is provided at:

https://github.com/purnanandelango/ddto

Figure 4.5 shows the result of Algorithm 4 for a discrete-time convex optimal control

example described in Appendix D.2.1, and Figure 4.6 shows the result of Algorithm 5 for a continuous-time convex optimal control example described in Appendix D.2.2. Both examples consider four target states with a specified priority order. The first trunk is denoted with ⊖, the second trunk with ⊟, and the third trunk with △. The branches are denoted with ●, ●, ●, and ●. The constraint bounds are denoted with ▬. The nodes in Figure 4.5 correspond to the discrete-time state and control input, whereas the nodes in Figure 4.6 denote the SCP solution variables at the discretization nodes.



(a) Position

(b) Cumulative trajectory cost

(c) Thrust magnitude

(d) Thrust pointing angle

Figure 4.5: Algorithm 4 applied to the discrete-time convex optimal control example in Appendix D.2.1.

4.7 Conclusions

We present a trajectory optimization framework called DDTO—deferred decision trajectory optimization—for trajectory optimization in the presence of unmodeled uncertainties and contingencies due to imperfect knowledge of the system model or its environment. The key idea is to ensure that a collection of targets is available for as long as possible, which allows the vehicle to defer the decision to select a target as much as possible. In a closed-loop setting this would provide more time to quantify the uncertainties and contingencies

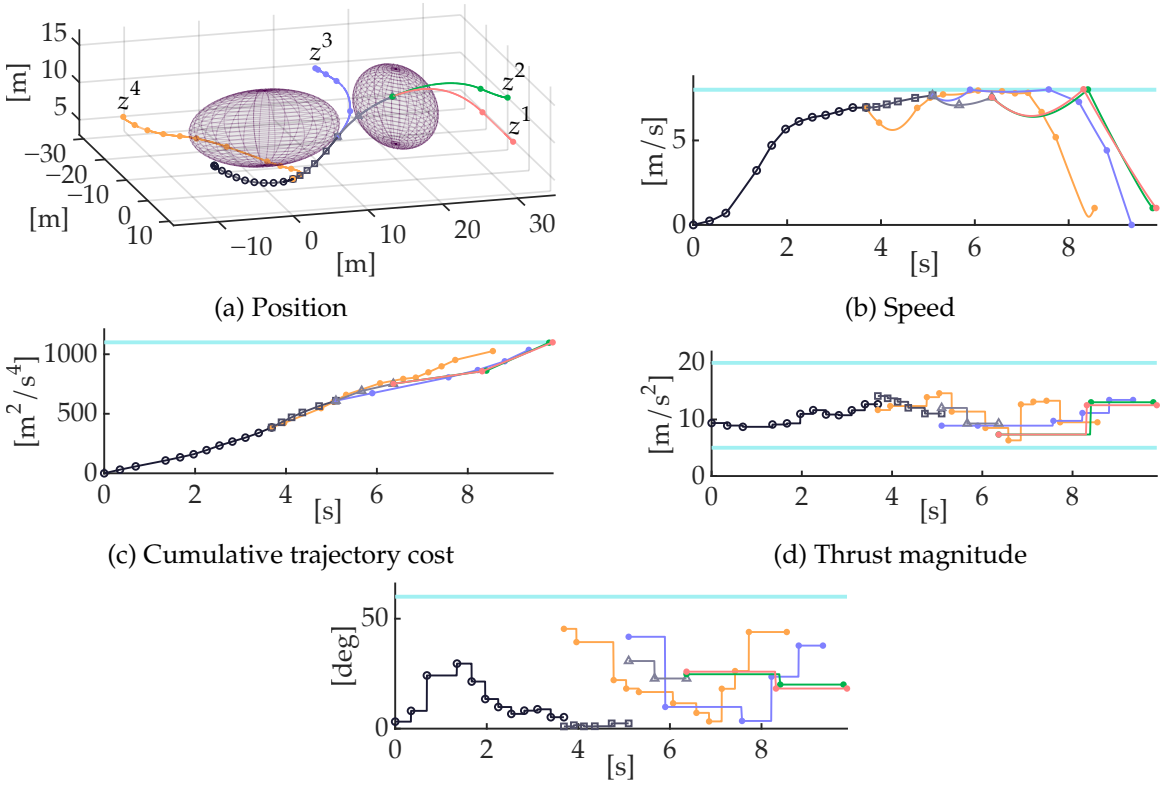


Figure 4.6: Algorithm 5 applied to the continuous-time nonconvex optimal control example in Appendix D.2.2.

so that the most reliable target can be eventually selected. To this end, we proposed a fully-deterministic optimization-based approach for formulating DDTO via constrained reachable sets and provided their equivalent representations as a cardinality minimization problems. Then by inferring the structure of the optimal DDTO solutions, we designed specialized solution methods for convex and nonconvex optimal control problems, which we demonstrated on two optimal control applications based on quadrotor motion planning.

Chapter 5

Closed-Loop Methods for Cislunar Applications

This chapter describes closed-loop solution methods specialized to spacecraft applications in cislunar space, particularly station keeping, formation flight, autonomous optical navigation (OPNAV), and rendezvous on a near rectilinear halo orbit (NRHO).

5.1 Local Eigenmotion Control for Station Keeping

The Lunar Orbital Platform-Gateway (LOP-G), also referred to as the Gateway, will play an important role in facilitating missions in cis-lunar space and beyond [143, 144]. The Gateway will be deployed in proximity to a near rectilinear halo orbit (NRHO), a closed periodic trajectory in the Earth-Moon circular-restricted three-body problem (CR3BP), due to its favorable stability properties and visibility from Earth [145].

The Gateway will be deployed near and not on an NRHO, because the NRHO of the CR3BP does not take into account perturbations such as the gravitational attraction of the Sun, solar radiation pressure (SRP), or lunar J2 effects. Instead of attempting to follow the NRHO of the CR3BP and using fuel to compensate for predictable perturbations, standard practice is to solve via multiple shooting or collocation-based techniques for a high-fidelity trajectory near an NRHO that accounts for all major predictable forces in cis-lunar space [146, 147]. This high-fidelity solution is also referred to as an NRHO, even though it is no longer closed, periodic, nor stable. The benefit of this high-fidelity NRHO solution, however, is that in the absence of any additional perturbing forces, a spacecraft could naturally follow the trajectory without expending any fuel, which is a key performance metric to ensure the long term viability of the Gateway.

However, two factors prevent deploying the Gateway solely based on the high-fidelity

solution. First, navigational uncertainty and unpredictable disturbance forces prevent the spacecraft from being deployed exactly on the computed trajectory. Second, and critically, compared to the ideal CR3BP counterpart the trajectory is highly unstable. As a consequence, small arbitrary deviations from the solution will cause the spacecraft to diverge rapidly from the computed trajectory, and thus require stabilizing control action [148, 149].

In recent years, several station-keeping strategies have been developed for high-fidelity NRHOs, including target point methods [150, 151, 152], crossing control [153, 154, 155], Hamiltonian structure preserving strategy [156, 152] and Floquet mode control [157, 150, 152]. While the Gateway itself could make use of such control strategies, these methods can not be directly applied to visiting spacecraft such as cargo resupply, human transport, or inspection and maintenance missions, which may need to perform long-term bounded, collision-free relative motion about the Gateway. Given this need, we propose a control strategy for reliable, fuel-efficient station keeping and bounded relative motion control for the Gateway and its visiting spacecraft. While formation control for multiple spacecraft on halo orbits has been developed [158, 159, 160], for NRHO in particular, these methods propose control schemes based on the periodic solution in the CR3BP [161] and rely on the computationally expensive process of generating a high-fidelity solution for each spacecraft in the formation [162, 163]. The approach proposed in this work makes use of a single precomputed high-fidelity NRHO solution while ensuring safe separation distance between spacecraft.

The proposed method belongs to family of station-keeping methods which harness the linear approximation of the spacecraft dynamics in the vicinity of the high-fidelity NRHO [148, 164]. State-of-the-art methods in this family include Cauchy-Green tensor (CGT) targeting [164, 165], x-axis crossing control [165], and the STT approach[166]. Although small deviations from a state on the high-fidelity NRHO trajectory will generally lead to rapid divergence, at any given instant there exist states in the vicinity of the NRHO which yield desirable non-diverging natural motion. The set of such states can be estimated using local modal decompositions of receding-horizon state transition matrices (STMs)

associated with the high-fidelity NRHO trajectory. The states in the vicinity of the NRHO that exhibit desirable natural motion arise from eigenvectors of the STMs with eigenvalues lying inside the unit circle. The natural motion resulting from an initial condition along an eigenvector of an STM is termed local eigenmotion. Whenever divergence from the vicinity of the NRHO is detected, the proposed local eigenmotion-based control solves a nonlinear optimization problem to find a fuel-efficient maneuver that transfers the spacecraft to the set of states that yield desirable natural motion.

While a nonlinear optimal control problem could be solved to obtain the optimal station-keeping maneuver for the entire duration of the spacecraft mission, such an approach is computationally prohibitive. The local eigenmotion-based control, on the other hand, exploits natural motion to provide a tractable method for fuel-efficient maneuvering, while also reducing the frequency of control action, which is another key performance metric for the long term viability of the Gateway due to thruster lifecycle. In contrast, LQR-based station keeping requires frequent control action, and re-planning a new trajectory in the high-fidelity model using multiple shooting or collocation whenever the spacecraft diverges away is prohibitively expensive for on-board computation.

Following the notation in this section, the paper proceeds in Section 5.1.1 with a description of NRHOs and the dynamical system model. Section 5.1.2 introduces the notion of local eigenmotion and develops the control algorithm. Section 5.1.3 demonstrates the approach via a case study of two spacecraft in bounded relative motion about a high-fidelity NRHO that take simulated measurements from the Deep Space Network. Finally, concluding remarks are provided in Section 5.1.4.

Notation

The set of natural numbers is $\mathbb{N}$ and the set of real numbers is $\mathbb{R}$. All vectors are column vectors. A real vector of length $n \in \mathbb{N}$ belongs to the set represented by $\mathbb{R}^n$. Vectors are represented as comma separated list of elements enclosed in $[\cdot]$. A vector with two elements $[a, b]$ with real numbers a and b such that $a \leq b$ also represents a closed interval on the real line. Similarly, (a, b) denotes an open interval. A vector constructed by vertically

stacking two vectors c and d can be compactly represented as $[c^\top \ d^\top]^\top$. The vector of length n containing zeros is denoted by 0_n.

The kth element of a vector θ of length $n \geq k$ is denoted by $\theta[k]$. Given a matrix Θ with $n \geq k$ rows and $m \geq k$ columns, row k is denoted by $\Theta[k,:]$, while column k is denoted by $\Theta[:,k]$. If κ is a vector consisting of elements from $\{1,\ldots,n\}$, then $\theta[\kappa]$ is a vector of elements of θ indexed by the elements in the vector κ. Similarly, $\Theta[:,\kappa]$ and $\Theta[\kappa,:]$ are matrices composed of columns and rows of Θ, respectively, indexed by the elements in the vector κ.

The identity matrix is denoted by I, with its dimension inferred by context. An eigenvalue of a matrix is denoted by λ unless otherwise specified.

5.1.1 NRHO Model

Near rectilinear halo orbits (NRHO) are periodic trajectories around the L1 and L2 Lagrange points of the Earth-Moon circular restricted three-body problem (CR3BP). Owing to their favorable stability properties and relatively low station-keeping cost, the NRHO about the L2 point with 9:2 synodic resonance and perilune radius of about 3150 km has been chosen for deploying the Gateway [143, 167]. However, NRHOs don't exist in reality since the CR3BP ignores solar radiation pressure (SRP), gravitational forces due to celestial bodies other than Earth and Moon, and effects like lunar J2 zonal harmonics. Ignoring these higher order effects during mission design would lead to an unacceptably high amount of fuel consumption. Thus, a high-fidelity astrodynamics model with ephemeris data is used in practice to generate a solution which is closest to the NRHO in the CR3BP. This high-fidelity solution is aperiodic and consists of a finite number of revolutions around the Moon. The astrodynamics model considered in this work is based on the one used in [30], which accounts for all major predictable forces acting on a spacecraft in cis-lunar space. Any major predictable force has magnitude larger than that of the largest unpredictable force. For the model considered here, the largest unpredictable force influencing a spacecraft is determined to be the indirect disturbance caused by navigation

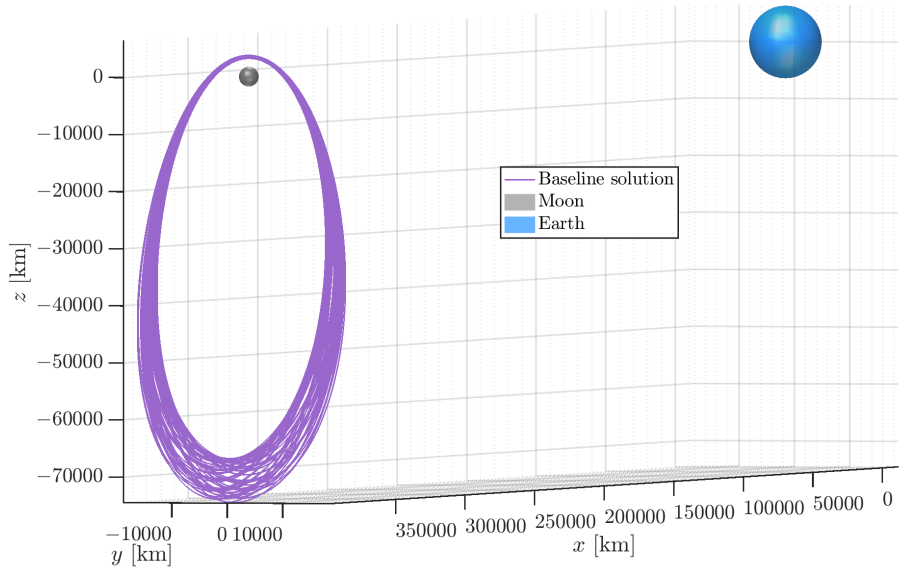


Figure 5.1: The baseline NRHO solution (represented in the Earth-Moon rotating frame) consisting of 60 revolutions around the Moon over 394 days is computed via multiple shooting.

error feeding into the spacecraft's controller [168]. The navigation error is quantified under the assumption that the spacecraft state is estimated using measurements from the Deep Space Network (DSN) [169, 170]. Under these assumptions, the force-magnitude analysis in [30] determined that the major predictable forces acting on a spacecraft in the region of space occupied by the NRHO of the CR3BP are SRP, lunar J2 zonal harmonics, and gravitational forces due to the Earth, Moon and Sun.

Let the high-fidelity dynamical system be represented by

$$\dot{\theta}(t) = g(t, \theta(t), u(t)), \tag{5.1}$$

which describes the equations of motion of a spacecraft in the non-inertial rotating frame (henceforth referred to as the rotating frame) considered in the CR3BP. The origin of this frame is at the Earth-Moon barycenter, with x-axis pointing towards the center of mass of Moon and z-axis along the angular momentum vector of the Earth-Moon system in the CR3BP. This frame is commonly chosen for the analysis of NRHOs since it is relevant for observation and communication from Earth [146, Section 2.1.2]. The right-hand-side of the model in (5.1) accounts for the major predictable external forces acting on the spacecraft

mentioned above. Interested readers can refer to [165, Section 2.3] for further details. The state vector $\theta \triangleq [r^\top \ v^\top]^\top$ consists of the spacecraft position vector r and velocity vector v in the rotating frame, and u represents the control input vector.

5.1.1.1 Control Model

The control approach developed in the following section makes use of impulsive thrusters to execute maneuvers, wherein a thrust impulse is modelled to cause an instantaneous change in velocity of the spacecraft. Consider the motion of the spacecraft in the time interval $[t_1, t_2]$ with a thrust impulse $\zeta \in \mathbb{R}^3$ applied at $t' \in (t_1, t_2)$. The control input $u(t)$ can be represented as

$$u(t) = \begin{bmatrix} 0_3 \\ \zeta \end{bmatrix} \delta(t - t'), \quad \text{for } t \in [t_1, t_2], \tag{5.2}$$

where $\delta(t)$ is the Dirac delta function. The right-hand-side of (5.1) can be rewritten as

$$g(t, \theta(t), u(t)) = f(t, \theta(t)) + u(t), \tag{5.3}$$

for $t \in [t_1, t_2]$, where f includes the previously mentioned external forces acting on the spacecraft. The (uncontrolled) natural motion of the spacecraft is thus given by

$$\dot{\theta}(t) = f(t, \theta(t)). \tag{5.4}$$

The spacecraft state at t_2 obtained after the impulse at t' can be then determined as follows

$$\begin{aligned} \theta(t_2) &= \theta(t_1) + \int_{t_1}^{t_2} g(\tau, \theta(\tau), u(\tau)) \mathrm{d}\tau, \\ &= \theta(t_1) + \int_{t_1}^{t_2} \left(\begin{bmatrix} 0_3 \\ \zeta \end{bmatrix} \delta(\tau - t') + f(\tau, \theta(\tau)) \right) \mathrm{d}\tau, \\ &= \theta(t') + \int_{t'}^{t_2} f(\tau, \theta(\tau)) \mathrm{d}\tau, \end{aligned} \tag{5.5}$$

where

$$\theta(t') = \theta(t_1) + \begin{bmatrix} 0_3 \\ \zeta \end{bmatrix} + \int_{t_1}^{t'} f(\tau, \theta(\tau)) \mathrm{d}\tau. \tag{5.6}$$

While the propagation of spacecraft dynamics with impulsive thrust (5.6) is used in the subsequent development, with minor modifications, the control approach of Section 5.1.2 can be adapted for thrust pulses of finite-duration. Note that finite-duration thrust pulses on the order of minutes can be approximated fairly well by thrust impulses due to the sufficiently slow time scales of the system dynamics (5.4).

5.1.1.2 Baseline Solution and Linearization

Denote the high-fidelity NRHO solution of (5.4) as the baseline solution $\bar{\theta}(t)$. The baseline solution is an uncontrolled, natural motion trajectory which lies near the NRHO of the CR3BP, and is estimated using the multiple shooting approach in [146]. The baseline solution $\bar{\theta}(t)$, shown in Fig. 5.1, consists of $K \triangleq 60$ revolutions around the Moon spanning $t_{\max} \triangleq 394$ days, where $t \in [0, t_{\max}]$ with $t = 0$ corresponding to the Julian date epoch of January 13, 2023. The position in each revolution which is farthest from the moon is referred to as an apolune, and the corresponding time instants are of particular importance to the control strategy of Section 5.1.2. Apolune approach times for K revolutions of $\bar{\theta}(t)$ are given by $t^0_{\text{apo}} < t^1_{\text{apo}} < \ldots < t^K_{\text{apo}}$, where $t^0_{\text{apo}} = 0$ and $t^K_{\text{apo}} = t_{\max}$.

In addition to the baseline, the following section makes use of the linear time-varying system,

$$\dot{\theta}(t) \approx A(t)\theta(t) + b(t), \tag{5.7}$$

where,

$$A(t) = \left.\frac{\partial f(\tau, \omega)}{\partial \omega}\right|_{(t, \bar{\theta}(t))}, \tag{5.8a}$$

$$b(t) = (I - A(t))\bar{\theta}(t), \tag{5.8b}$$

which approximates solutions of (5.4) near $\bar{\theta}(t)$.

The proposed control strategy aims to keep the spacecraft in the region of the state space close to the baseline[1] where the linear approximation (5.7), which, along with $\bar{\theta}(t)$, can be computed a priori, is valid. Hence, the linear model, although imperfect, is useful for predicting future evolution of the nonlinear dynamics (5.4) and for determining appropriate control action.

The finite-time behavior of solutions to (5.4) near the baseline solution is particularly useful and can be analyzed using the discrete-time counterpart of (5.7). More precisely, if the spacecraft is at $\bar{\theta}_1$ in the vicinity of $\bar{\theta}(t_1)$ for some $t_1 \in [0, t_{\max}]$, then its state at $t_2 > t_1$, denoted by $\bar{\theta}_2$, can be approximated as

$$\bar{\theta}_2 \approx \Phi(t_2, t_1)\bar{\theta}_1 + \chi(t_2, t_1), \tag{5.9}$$

where $\Phi(t_2, t_1)$ is the state transition matrix (STM) for the time interval $[t_1, t_2]$ obtained by propagating the linear system

$$\dot{\Phi}(t, t_1) = A(t)\Phi(t, t_1), \quad t \geq t_1, \tag{5.10}$$

over the time interval $[t_1, t_2]$ with initial condition $\Phi(t_1, t_1) = I$, and $\chi(t_2, t_1)$ is the residual

$$\chi(t_2, t_1) = \int_{t_1}^{t_2} \Phi(t_2, \tau)b(\tau)\mathrm{d}\tau. \tag{5.11}$$

For an eigenvector ν of $\Phi(t_2, t_1)$, there exists a complex number λ such that

$$\Phi(t_2, t_1)\nu = \lambda\nu. \tag{5.12}$$

If $|\lambda| \leq 1$ then ν is said to be a *non-expanding* eigenvector, whereas if $|\lambda| > 1$ then ν is said to be an *expanding* eigenvector. The STM provides useful local information about the solutions to (5.4) near $\bar{\theta}(t)$. The natural motion that results from an initial condition along the expanding and non-expanding eigenvectors of an STM is termed as *expanding* and

[1]The region where deviation of the spacecraft from the baseline is much smaller than the scale of the baseline.

non-expanding local eigenmotion, respectively. Ideally, a non-expanding local eigenmotion associated with $\Phi(t_2, t_1)$ does not diverge away from the baseline for the time interval $[t_1, t_2]$, whereas an expanding local eigenmotion diverges away during the same interval. In practice, a non-expanding local eigenmotion can still diverge away for $t < t_2$ owing to the fact that the linear estimate in (5.9) is only an approximation of the propagated nonlinear dynamics (5.4). Since the linear model is only valid close to the baseline, the performance of a non-expanding local eigenmotion can deteriorate when the initial condition is far away from $\bar{\theta}(t)$.

5.1.2 Local Eigenmotion Control

The proposed control approach utilizes STMs computed for a sequence of adjacent time intervals spanning $[0, t_{\max}]$, termed as *receding-horizon* STMs, to determine non-expanding local eigenmotion corresponding to those intervals. Whenever the spacecraft is predicted to diverge beyond a specified threshold from the baseline $\bar{\theta}(t)$, it is maneuvered to an initial condition of one of the non-expanding local eigenmotions in a fuel-efficient manner, see Fig. 5.2.

In principle, we could compute an STM for the entire duration of the baseline, and pick a non-expanding eigenmotion to obtain a bounded relative motion trajectory for the entire duration of the baseline in one shot. However, this approach is computationally infeasible because the condition number of the STM increases as the time duration for which it holds increases. In practice, the longest time duration for which the STM is reliable is typically the time required for 12 revolutions around the Moon, which is about 78 days.

In addition to the length of the time interval for the STM, the initial time of the interval is also critical, particularly for control applications. Local Lyapunov Exponents (LLE) [171] determined for the baseline solution, and shown in Fig. 5.3, indicate that the sensitivity of the solution to perturbations is highest at perilune and lowest at apolune. As a consequence, apolune is the best time to apply control, since when the spacecraft is in a region with high LLE, sources of error such as navigation uncertainty and actuation errors can result in

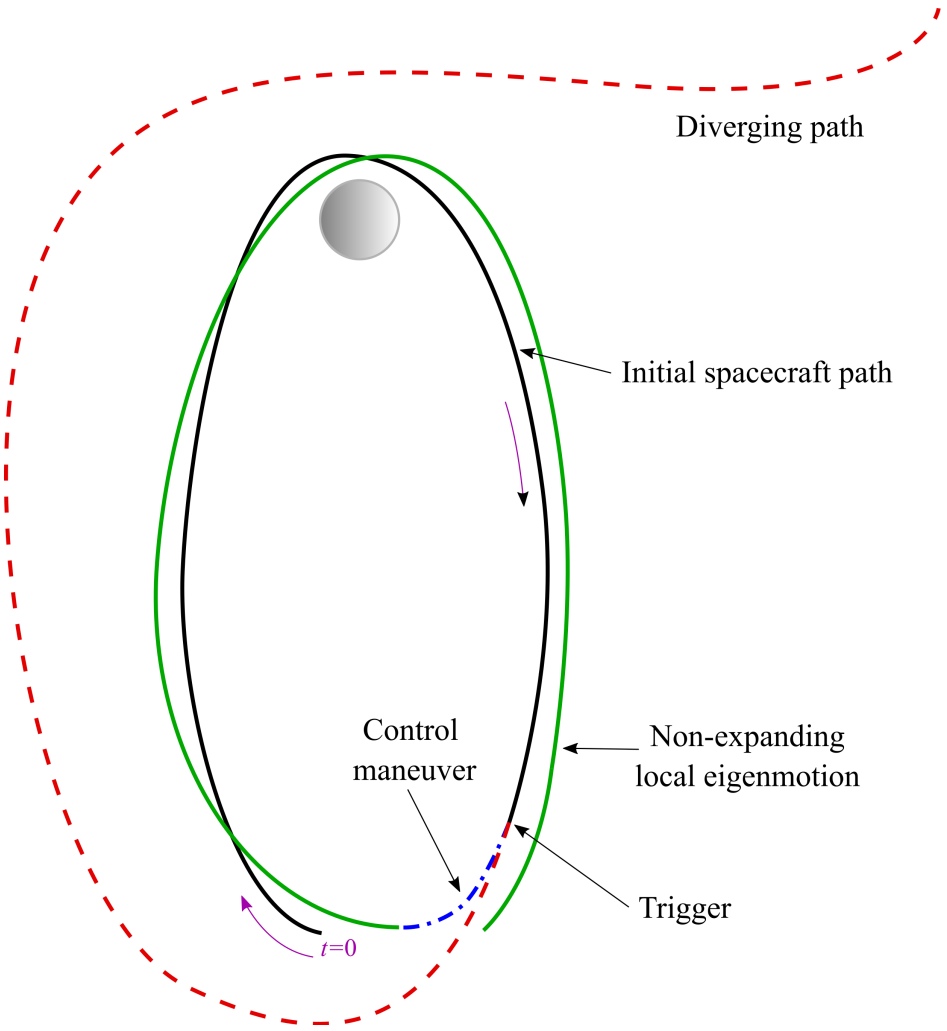


Figure 5.2: Illustration of a maneuver which transfers the spacecraft to a non-expanding local eigenmotion at apolune when the trigger condition is satisfied.

incorrect thruster firing, which can cause the spacecraft to diverge rapidly [30].

The strategy developed in this work considers maneuvers at apolune, but not necessarily every time the spacecraft visits apolune. The reason for this is as follows. For some $G \leq 12$ and $k \leq K - G$, assume that the spacecraft is initialized on a desirable local eigenmotion in proximity to the baseline by precisely placing it on an appropriate eigenvector of $\Phi(t_{\text{apo}}^{k+G}, t_{\text{apo}}^{k})$. Numerical simulations of (5.4) indicate that the spacecraft exhibits satisfactory bounded motion relative to the baseline for at least the following G revolutions around the Moon.

As previously mentioned, the longest time duration for which the STM is numerically

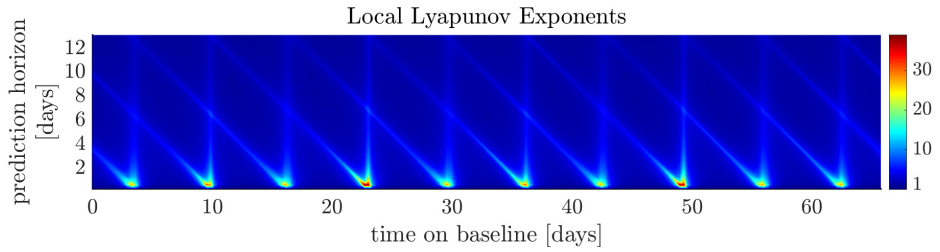


Figure 5.3: A Local Lyapunov Exponent (LLE) is the logarithm of the maximum singular value of an STM normalized by the length of the interval for which it is valid. LLEs for the baseline solution over a duration of 75 days are estimated using STMs computed at each instant with a prediction horizon of 13 days. A larger value of LLE (highlighted by warmer colors) indicates greater sensitivity to perturbations. The LLEs reveal the spacecraft is most vulnerable to perturbations every time it visit a perilune of the baseline.

reliable, which here means a condition number smaller then 10^8, is the time required for $H \triangleq 12$ revolutions around the Moon. The eigenvalues of the STM estimated for $[t^0_{\text{apo}}, t^H_{\text{apo}}]$ are $\text{spec}(\Phi(t^H_{\text{apo}}, t^0_{\text{apo}})) = \{1.14 \times 10^4,\ 4.48,\ -0.88 + \text{i}0.46,\ -0.88 - \text{i}0.46,\ 0.22,\ 8.75 \times 10^{-5}\}$. Observe that this STM has two expanding and four non-expanding eigenvectors. Owing to a significantly large eigenvalue, small perturbations to a spacecraft on the baseline NRHO solution can cause it to diverge away rapidly. This is in stark contrast to the favorable stability properties of the NRHO in CR3BP model. Furthermore, it is observed that the spectrum of the STM estimated for $[t^k_{\text{apo}}, t^{k+H}_{\text{apo}}]$ for any $k \leq K - H$ is qualitatively similar that of the first H revolutions. As a result, typically any vector in the span of the four non-expanding eigenvectors of $\Phi(t^{k+H}_{\text{apo}}, t^k_{\text{apo}})$ give rise to a non-expanding local eigenmotion for any $k \leq K - H$. Let these eigenvectors, indexed by the set $\mathcal{I}^k$, be denoted by v^k_i for $i \in \mathcal{I}^k,\ k \leq K - H$. The distance from the baseline and the position relative to the baseline of a spacecraft moving along some of the local eigenmotions of $\Phi(t^H_{\text{apo}}, t^0_{\text{apo}})$ are shown in Fig. 5.4 and Fig. 5.5, respectively.

The spacecraft maneuvers occur over a short time interval, referred to as the control horizon, just before the spacecraft reaches an apolune. A maneuver to return the spacecraft state to a non-expanding local eigenmotion is initiated when a trigger condition is met. Let the position deviation of the spacecraft at $t = t^k_{\text{apo}}$ be denoted by Δr_k for $k \leq K - H$. The trigger condition adopted in this work considers the change in distance between the spacecraft and the baseline solution since $t = 0$ and since its recent visit to apolune, and is

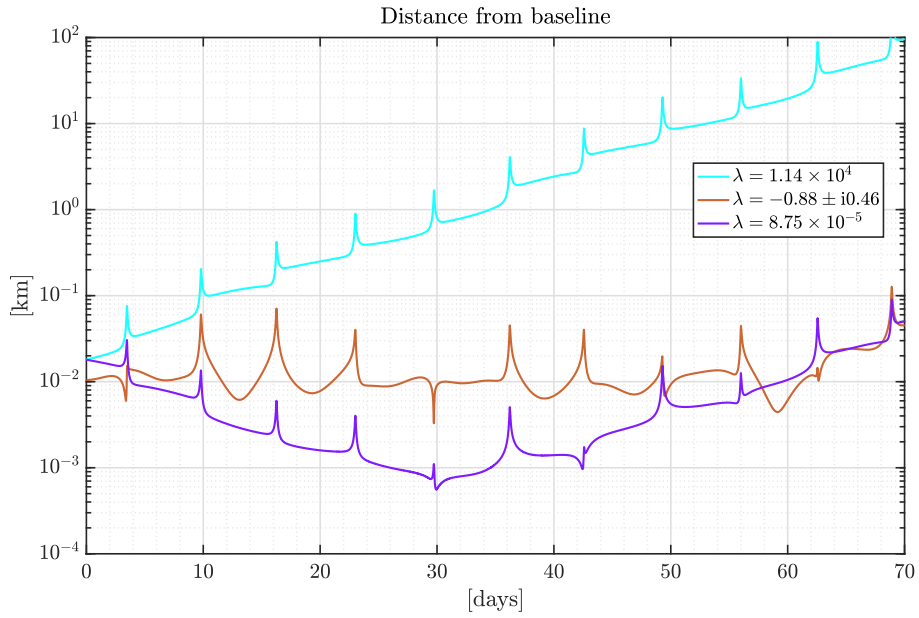


Figure 5.4: The distance to baseline from local eigenmotions resulting from four eigenvectors of $\Phi(t^H_{\text{apo}}, t^0_{\text{apo}})$ are shown for a duration of 70 days. The distance of initial conditions from the baseline are chosen to have similar order of magnitude. Although the expanding and non-expanding nature of these is evident, the non-expanding behavior doesn't in general persist over the entire duration for which the corresponding STM is valid. This is due to the STM being based on the linear system (5.7), which only approximately predicts the behavior of the nonlinear model (5.4). However, in practice, even with the this apparent degradation, the proposed approach requires few maneuvers and low Δv to sustain annual bounded motion.

given by

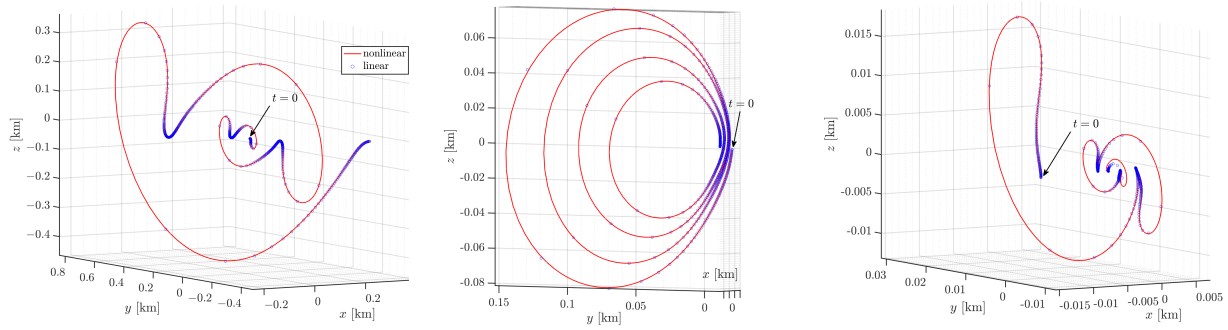
$$\Delta r_k > \min\left\{\rho \Delta r_0, \min\{2\Delta r_{k-1}, \Delta r_0\}\right\}, \tag{5.13}$$

where, ρ is a tuning parameter that contributes to determining the nature of the generated bounded motion trajectory near the baseline.

If the trigger condition (5.13) is satisfied by Δr_k for some $k < K$, a fuel-efficient maneuver for transferring the spacecraft to a non-expanding local eigenmotion starting at t^k_{apo} is computed by solving the discrete-time optimal control problem

$$\underset{\substack{\{\alpha_i\}_{i\in\mathcal{I}},\\ \{u_j\}_{j=0}^{N-2}}}{\text{minimize}} \quad \sum_{j=0}^{N-2} \|u_j\|_2 \tag{5.14a}$$

$$\text{subject to} \quad \theta(\tau^k_{j+1}) = \theta(\tau^k_j) + \begin{bmatrix} 0_3 \\ u_j \end{bmatrix} + \int_{\tau^k_j}^{\tau^k_{j+1}} f(\gamma, \theta(\gamma)) \mathrm{d}\gamma, \;\; 0 \le j \le N-1 \tag{5.14b}$$



(a) $\lambda = 1.14 \times 10^4$ (b) $\lambda = -0.88 \pm \mathrm{i}0.46$ (c) $\lambda = 8.75 \times 10^{-5}$

Figure 5.5: The position relative to the baseline on local eigenmotions resulting from four eigenvectors of $\Phi(t^H_{\text{apo}}, t^0_{\text{apo}})$ are shown for the time interval $[t^0_{\text{apo}}, t^4_{\text{apo}}]$. The nonlinear system (5.4) and the linear approximation (5.7) are propagated using initial conditions aligned with the eigenvectors associated with these local eigenmotions. The position of the spacecraft relative to the baseline at each instant is shown in these plots. Observe that since (b) and (c) correspond to non-expanding local eigenmotion, the relative position converges to origin, whereas in (a) the relative position diverges away owing to the expanding local eigenmotion. (Note the order of magnitude difference in the scale of axes of the plots.)

$$\theta(\tau^k_{N-1}) = \bar{\theta}(\tau^k_{N-1}) + \sum_{i \in \mathcal{I}^k} \alpha_i v^k_i, \tag{5.14c}$$

$$\theta(\tau^k_0) = \hat{\theta}(\tau^k_0), \tag{5.14d}$$

for the time grid $T^k_N \triangleq [\tau^k_0, \ldots, \tau^k_{N-1}]$, where $\tau^k_j \triangleq t^k_{\text{apo}} - (N-1-j)\Delta t$ for $j = 0, \ldots, N-1$. The maneuver consists of $N-1$ thrust impulses $\{u_j\}_{j=0}^{N-2}$ applied at intervals of Δt starting from τ^k_0. Under the assumption of impulsive input, the representation of the propagated equations of motion in (5.6) is utilized for constraint (5.14b). The boundary condition (5.14c) stipulates that the state at end of the control horizon lies in the span of the non-expanding eigenvectors of $\Phi(t^{k+H+1}_{\text{apo}}, t^{k+1}_{\text{apo}})$, which would give rise to a non-expanding local eigenmotion. Furthermore, in a practical implementation, the controller will not have access to the true state of the spacecraft. It receives an estimate, denoted by $\hat{\theta}(\tau^k_0)$ in (5.14d), by the state estimator.

The proposed control approach results in the automatic routine summarized by Algorithm 6. The key components are described below.

- The input to Algorithm 6 consists of the following

Algorithm 6 Local Eigenmotion Control

Input: $\bar{\theta}(t), \{t^j_{\text{apo}}\}_{j=0}^{K}, \Delta r_0, H, \{T^j_N\}_{j=1}^{K}$

$T^0_H \leftarrow [t^0_{\text{apo}}, t^H_{\text{apo}}]$
$t_0 \leftarrow t^0_{\text{apo}}$
$\theta \leftarrow \texttt{EigenMotion}(\Delta r_0, 0, H)$
$\Delta v \leftarrow 0$
$(l, k) \leftarrow (1, 0)$
while $k \leq K - 1 - H$ **do**
    $\bar{\psi} \leftarrow \bar{\theta}(t^{k+1}_{\text{apo}})$
    $T \leftarrow [t^k_{\text{apo}}, T^{k+1}_N[1]]$
    $\widetilde{\theta} \leftarrow \texttt{Propagate}(\theta, T)$
    $T \leftarrow T^{k+1}_N[[1, N]]$
    $\theta \leftarrow \texttt{Propagate}(\widetilde{\theta}, T)$
    $\Delta r_{k+1} \leftarrow \texttt{PosDev}(\theta, \bar{\psi})$
    if $\texttt{Trigger}(\Delta r_{k+1}, \Delta r_k, \Delta r_0)$ **then**
        $T^{k+1}_H \leftarrow [t^{k+1}_{\text{apo}}, t^{k+H+1}_{\text{apo}}]$
        $(U_l, \Delta v_l) \leftarrow \texttt{OptCtrl}(\widetilde{\theta}, \Delta r_{k+1}, T^{k+1}_N, T^{k+1}_H)$
        $\theta \leftarrow \texttt{PropagateCtrl}(\widetilde{\theta}, U_l, T^{k+1}_N)$
        $(\Delta v, \Delta r_{k+1}) \leftarrow (\Delta v + \Delta v_l, \texttt{PosDev}(\theta, \bar{\psi}))$
        $l \leftarrow l + 1$
    end if
    $k \leftarrow k + 1$
end while
$M \leftarrow l - 1$

Output: $\Delta v, \{U_j, \Delta v_j\}_{j=1}^{M}$

- The baseline solution $\bar{\theta}(t)$.
- The time instants t^k_{apo} for $k = 0, \ldots, K$, corresponding to $K + 1$ visits to apolune in the baseline.
- The initial distance Δr_0 of the spacecraft from the baseline.
- Number of downstream revolutions H for which the STM is computed.
- The time intervals before each visit to apolune in the baseline solution (excluding the first), T^k_N for $k = 1, \ldots, K$, each with N grid points uniformly spaced by Δt.

- `Propagate`(ψ, T) returns the state after propagating the high-fidelity prediction model (5.4) with initial condition ψ for the time interval specified by the two-element vector T. When Algorithm 6 operates alongside a state-estimator, as demonstrated in Section 5.1.3, the first argument of `Propagate`, ψ, is an estimate of the spacecraft's current state.

- `PosDev`(θ, ψ) returns the position deviation between states θ and ψ.

- `EigenMotion`$(\Delta r, k, H)$ returns the state with a position deviation of Δr from the state at the kth apolune of the baseline, and lying along the vector $\sum_{i \in \mathcal{I}^k} v_i$, where v_i for $i \in \mathcal{I}^k$ are the non-expanding eigenvectors of $\Phi(t_{\text{apo}}^{k+H}, t_{\text{apo}}^{k})$.

- `Trigger` evaluates to true if the trigger condition (5.13) is satisfied.

- `OptCtrl`$(\psi, \Delta r, T_N^k, T_H^k)$ solves the discrete-time optimal control problem (5.14) for the time interval T_N^k, i.e., for a maneuver to take place right before the spacecraft's kth apolune visit. Constraint (5.14c) is constructed using the non-expanding eigenvectors of the STM for the interval $T_H^k = [t_{\text{apo}}^k, t_{\text{apo}}^{k+H}]$, and the initial condition (5.14d) is specified by ψ. `OptCtrl` returns the control effort required for the maneuver, and the control solution estimated as a matrix with the control input for each τ_j^k for $j = 0, \ldots, N-2$ stacked row-wise.

- `PropagateCtrl`(ψ, U, T_N^k) propagates (5.1) over the control horizon T_N^k using the control solution U returned by `OptCtrl` with initial condition ψ.

- Δv_j for $j = 1, \ldots, M$ is the cost for M maneuvers initiated in Algorithm 6 over the duration $[0, t_{\max}]$. The cumulative cost of the M maneuvers is given by Δv.

5.1.3 Numerical Results

This section demonstrates a practical implementation of the proposed control strategy by augmenting Algorithm 6 with a Kalman Filter which estimates states using simulated range and range-rate measurements from the Deep Space Network (DSN). The measurements

are received at a rate of 6 hr which is reasonable for the DSN, and adequate for accurate state estimation. The errors in range and range-rate measurements come from standard normal distributions with standard deviation of 10 m and 1 mm s^{-1}, respectively. More details on the DSN error statistics can be found in [30].

The simulations were implemented in MATLAB ver. 2019b running on a MacBook Pro with 16 GB RAM and 2.6 GHz Intel 6-Core i7 processor. The proposed approach is demonstrated for the case of two spacecraft executing bounded relative motion in the vicinity of the baseline solution. In particular, the Gateway is subjected to tight station keeping about the baseline while a visiting spacecraft is made to execute collision-free relative motion near the Gateway.

The trajectory computed for the Gateway, referred to as Solution 1, uses $\rho = 10,\ \Delta t = 6$ hr, $\Delta r_0 = 0.5$ km, and $N = 4$, whereas the trajectory for the visiting spacecraft, referred to as Solution 2, is computed with $\rho = 2,\ \Delta t = 12$ hr, $\Delta r_0 = 50$ km, and $N = 6$. Certain parameters in Algorithm 6 influence the nature of the bounded motion solution. In particular, the choice of ρ in (5.13) and Δr_0 for the two Solutions are instrumental in ensuring that they remain collision-free. These parameters are tuned such that the resulting solutions are infrequent and fuel efficient.

It is worthwhile to note that the effect of navigation uncertainty is more prominent for tight station-keeping maneuvers near the baseline. When a maneuver is initiated close to the baseline, the final state (which lies close to the baseline and is aligned with the desired eigenvector) is more easily corrupted by navigation uncertainly owing to its small magnitude measured relative to the baseline. As a consequence, the spacecraft could be maneuvered to a state which is not properly aligned with the desired eigenvector, which will then cause the spacecraft to diverge prematurely. Hence, tight station keeping could potentially necessitate more annual maneuvers. This pitfall is avoided while generating Solution 1 by choosing a small value of Δr_0 and a large value for ρ. This allows the spacecraft to slowly offset from the baseline over a duration of 150 days and settle at a distance of about 9 km from the baseline where maneuvers are not triggered too often.

The annual station-keeping performance of Solutions 1 and 2 are shown in Table 5.1,

Table 5.1: Annual station-keeping performance

	Solution 1	Solution 2
Maneuvers	13	15
Control duty cycle [% time annually]	0.8	2.05
Fuel consumption [m s^{-1}]	0.05	0.7

and the distance of the Solutions from the baseline as a function of time is shown in Fig. 5.6. With the proposed strategy, the annual station-keeping cost for the Gateway is comparable to that of state-of-the art techniques for station keeping on NRHO [30, 164]. The fuel required for the visiting spacecraft is significantly higher since it maintains a larger distance from the baseline than the Gateway. As such, each maneuver for transferring to a non-expanding local eigenmotion of the baseline is more expensive. Although the annual cost for Solution 2 is significantly higher than that for Solution 1, it is still comparable to the cost reported by competing techniques [30].

Another notable benefit of the proposed approach is the relatively few maneuvers required to sustain annual bounded motion. The black segments on Solutions 1 and 2 in Fig. 5.6 highlight the significantly small annual control duty cycle. Furthermore, Solutions 1 and 2 do not pose a risk of collision between the Gateway and the visiting spacecraft. The distance between the solutions is never smaller than 8 km, as shown in Fig. 5.7.

5.1.4 Conclusions

This work proposes a control strategy for bounded motion in the vicinity of a near rectilinear halo orbit (NRHO) where the Gateway will be deployed. The approach exploits the finite-duration characteristic natural motion (termed as local eigenmotion) resulting from eigenvectors of a state transition matrix to generate fuel-efficient bounded motion maneuvers for multiple spacecraft. More specifically, the proposed approach is demonstrated for bounded, collision-free relative motion of the Gateway and a visiting spacecraft. The key idea is to initiate fuel-efficient thruster firing to transfer the spacecraft to a state which yields bounded natural motion for a finite time whenever the spacecraft diverges signifi-

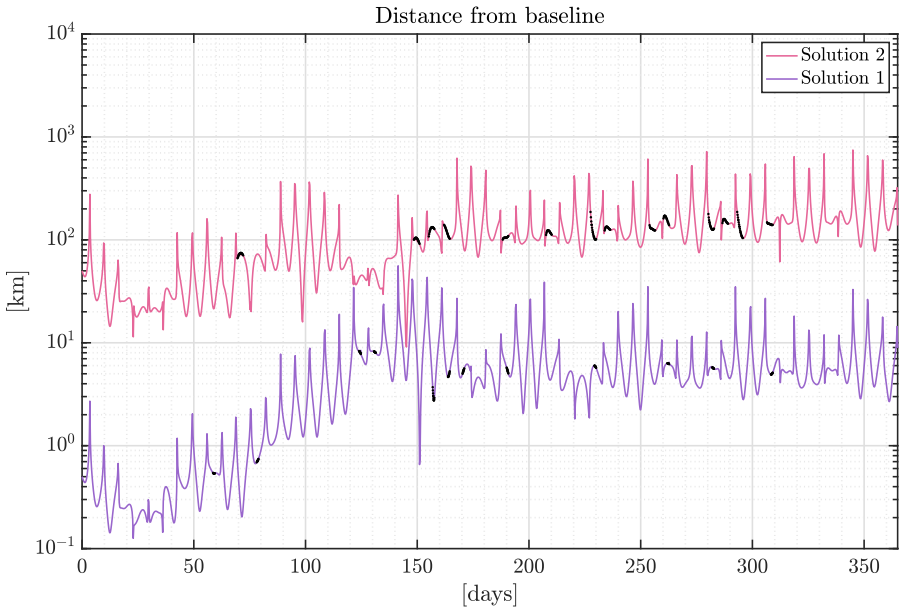


Figure 5.6: Position deviations of Solution 1 (for Gateway) and Solution 2 (for the visiting spacecraft) with respect to the baseline are shown. The 13 maneuver segments of Solution 1 and 15 maneuver segments of Solution 2 are marked in black. The solutions seem to intersect at around 150 days, but that is only due to the log scale of the plot. Fig. 5.7 confirms that the least separation between the two solutions is at least 8 km.

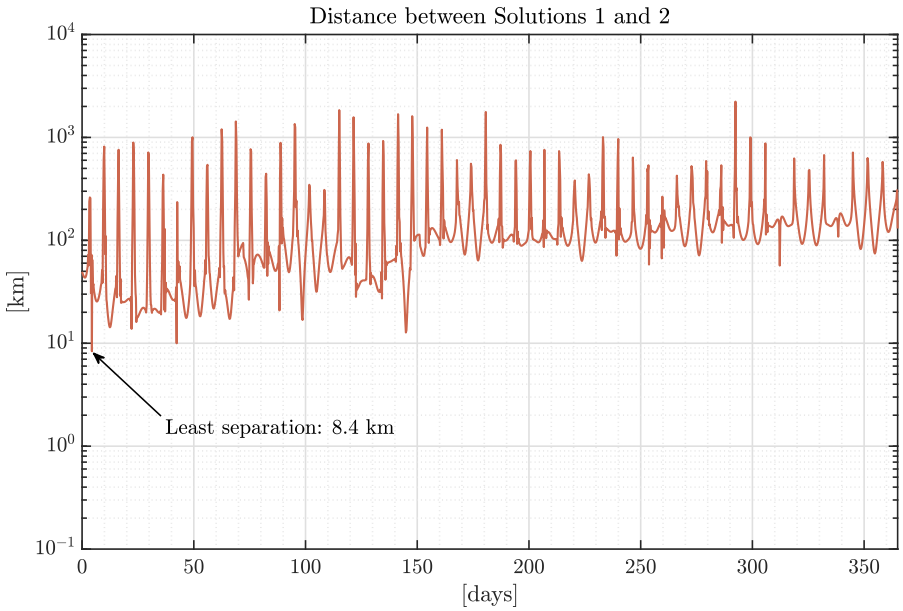


Figure 5.7: Position deviation of Solution 1 (for Gateway) with respect to Solution 2 (for the visiting spacecraft) is shown. The two Solutions maintain a separation distance of at least 8 km at all times.

cantly away from the baseline solution. The effectiveness of the strategy is demonstrated via a realistic simulation which includes a state estimator taking simulated measurements from the Deep Space Network. The annual station-keeping performance in terms of total fuel consumption and the total maneuver duration is shown to be at par or better than competent state-of-the art techniques. Future work will improve the trigger condition which initiates a maneuver. The current trigger condition adopts a myopic approach i.e. the spacecraft is transferred to a non-expanding local eigenmotion when it has deviated significantly from the baseline, without taking into account the quality of the local eigenmotion that is chosen. One relevant metric for assessing the quality of local eigenmotion is the length of time for which the spacecraft state does not diverge away. Another related metric could be the Momentum Integral described in [164].

5.2 Sequential Linearization Station Keeping with Optical Navigation

The recently launched CAPSTONE mission will be the first spacecraft to fly on an NRHO to verify the dynamics of the orbit and validate station-keeping strategies under real conditions [172]. Shortly after trans-lunar injection, CAPSTONE lost communication with NASA's Deep Space Network (DSN), jeopardizing the mission [173]. While the spacecraft eventually regained communication [174], such events highlight the possibility of communication faults and the need for backup autonomous navigation solutions that do not rely on DSN. Furthermore, in the coming years, NRHO will see a host of spacecraft, both crewed and uncrewed, arriving to build LOP-G, supply and service it, and use it as a launching pad for missions to the lunar surface or out to Mars. Given the increased activity, DSN-based navigation services will need to be rationed. As such, there is a need for alternatives. The Lunar GNSS Receiver Experiment (LuGRE) will attempt to use GPS and Galileo signals in the lunar environment and on the lunar surface[175], and CAPSTONE will be testing a new peer-to-peer autonomous navigation system that measures range and range-rate relative to NASA's Lunar Reconnaissance Orbiter (LRO) [172].

Horizon-based optical navigation (OPNAV) is another candidate technology that has recently matured and will be demonstrated on the upcoming Artemis 1 mission, currently slated for launch later this summer [176]. OPNAV is a fully onboard solution that can be deployed in the vicinity of any planet or moon.

In this work, we develop an autonomous station-keeping approach for NRHO that relies on state estimates generated from OPNAV measurements. In recent years, many strategies for station-keeping on high-fidelity NRHOs have been proposed [177, 178, 168, 29, 164, 28, 179, 30, 180, 181, 44]. Such strategies are generally categorized into short-horizon maneuvering, which maintains a spacecraft near an existing baseline, and long-horizon maneuvering, which transitions from one baseline trajectory to another. Using both short- and long-horizon maneuvering enables indefinite station-keeping on NRHOs [30]. Short-horizon maneuvering is further categorized into spectrum-based strategies that align a spacecraft with the stable subspace near the baseline [150, 151, 44], and target point approaches that control the spacecraft state or a portion of the state to attain a desired

value at some future time, see e.g., x-axis crossing control [150, 164]. Target point methods can also be used for long-horizon maneuvering. While long-horizon maneuvers may solve an optimal control problem to transition between two pre-computed baseline trajectories [30], when used for long-horizon maneuvers, target point methods do not require a pre-computed baseline to transition to, but rather generate the baseline trajectory in real-time [177].

We introduce a variant of the long-horizon x-axis crossing control method based on sequential linearization. We use the method in a receding-horizon approach that obviates the need for short-horizon station-keeping [164]. That is, instead of generating a baseline trajectory offline or computing one in real time and then using short-horizon maneuvers to maintain the spacecraft near the baseline, long-horizon maneuvers are computed after each revolution to maintain the spacecraft near an NRHO. Every apolune, the algorithm checks whether the x-component of the predicted spacecraft velocity at perilune relative to the Earth-Moon rotating frame six and half revolutions downstream is beyond a threshold, and if so, triggers the computation of a long-horizon maneuver to ensure the threshold will be satisfied. The long-horizon maneuver is computed based on sequentially linearizing the predicted trajectory. A linear approximation of the high-fidelity nonlinear dynamics is iteratively used to determine a velocity impulse which ensures the desired target velocity is achieved by the nonlinear system. The proposed approach is closely related to the sequential convex programming algorithms specialized to trajectory optimization [3] where the nonlinear dynamics is linearized about an iteratively updated reference trajectory.

We demonstrate the robustness of this baseline-free station-keeping strategy on NRHO to state estimates generated by an extended Kalman filter (EKF) that gets OPNAV position measurements from a high-fidelity ephemeris-based simulation and the synthetic rendering software Basilisk[182] and Vizard [183]. To the authors' knowledge, only a few other past works [168, 184, 185, 186, 187] have demonstrated realistic closed-loop station-keeping simulations with OPNAV, and this work may be the first to do so on NRHO with implementation details.

The rest of the paper is organized as follows. First we introduce the spacecraft and

control model along with the high-fidelity model for NRHO. We then provides details on horizon-based optical navigation and the extended Kalman filter. Next, we describe the sequential linearization-based targeting approach for NRHO station keeping. We then demonstrate the performance of the approach in a closed-loop station-keeping simulation with OPNAV. Finally, concluding remarks are provided.

Notation

We use the following notation throughout this work. The identity matrix in $\mathbb{R}^{n\times n}$ is denoted by I_n, the jth entry of vector x is denoted by $x^{[j]}$, the vector formed by jth through kth entries of vector x is denoted by $x^{[j:k]}$, and the vector formed by entries from columns j through k of the ith row of matrix Y is denoted by $Y^{[i,j:k]}$.

5.2.1 Spacecraft Model

Consider a point-mass spacecraft in cis-lunar space and the synodic frame $\mathrm{F_S}$. The frame $\mathrm{F_S}$ is a non-inertial rotating frame with its x-axis pointing from the Earth-Moon barycenter towards the center of mass of the Moon and its z-axis along the angular momentum vector of the Earth-Moon system in the CR3BP. This frame is commonly chosen for the analysis of NRHOs since it is useful for observation and communication from Earth [146, Section 2.1.2]. The nonlinear equation of motion of the spacecraft is given by

$$\dot{\theta}(t) = g(t, \theta(t), u(t)), \tag{5.15}$$

where $\theta \triangleq [r^\top \; v^\top]^\top$ is the state vector, r is the position of the spacecraft with respect to the Earth-Moon barycenter resolved in $\mathrm{F_S}$, $v \triangleq \dot{r}$ is the velocity of the spacecraft with respect to $\mathrm{F_S}$ resolved in $\mathrm{F_S}$, and u is the control vector of thrust forces acting on the spacecraft, resolved in $\mathrm{F_S}$. The right-hand-side of the model in (5.15) accounts for the major predictable external forces acting on the spacecraft, see [165, Section 2.3] for further details.

We assume that the spacecraft is actuated with impulsive thrusters, wherein a thrust

impulse is modeled to cause an instantaneous change in the velocity of the spacecraft. Consider the motion of the spacecraft in the time interval $[t_1, t_2]$ with a thrust impulse $\zeta \in \mathbb{R}^3$ applied at $t' \in (t_1, t_2)$. The control input $u(t)$ can be represented as

$$u(t) = \begin{bmatrix} 0_3 \\ \zeta \end{bmatrix} \delta(t - t'), \tag{5.16}$$

for $t \in [t_1, t_2]$, where $\delta(t)$ is the Dirac delta function. The right-hand-side of (5.15) can be rewritten as

$$g(t, \theta(t), u(t)) = f(t, \theta(t)) + u(t), \tag{5.17}$$

for $t \in [t_1, t_2]$, where f includes the previously mentioned external forces acting on the spacecraft. The (uncontrolled) natural motion of the spacecraft is thus given by

$$\dot{\theta}(t) = f(t, \theta(t)). \tag{5.18}$$

The spacecraft state at t_2 obtained after the impulse at t' is

$$\begin{aligned} \theta(t_2) &= \theta(t_1) + \int_{t_1}^{t_2} g(\tau, \theta(\tau), u(\tau)) \mathrm{d}\tau, \\ &= \theta(t_1) + \int_{t_1}^{t_2} \left(\begin{bmatrix} 0_3 \\ \zeta \end{bmatrix} \delta(\tau - t') + f(\tau, \theta(\tau)) \right) \mathrm{d}\tau, \\ &= \theta(t') + \int_{t'}^{t_2} f(\tau, \theta(\tau)) \mathrm{d}\tau, \end{aligned} \tag{5.19}$$

where

$$\theta(t') = \theta(t_1) + \begin{bmatrix} 0_3 \\ \zeta \end{bmatrix} + \int_{t_1}^{t'} f(\tau, \theta(\tau)) \mathrm{d}\tau. \tag{5.20}$$

While the propagation of spacecraft dynamics with impulsive thrust (5.20) is used in the subsequent development, with minor modifications it can be adapted to handle thrust pulses of finite-duration. Note that finite-duration thrust pulses on the order of minutes can be approximated fairly well by thrust impulses due to the slow time scales of the system dynamics (5.18) on NRHO, especially near apolune.

5.2.2 Horizon-based Optical Navigation

This work considers horizon-based OPNAV as an autonomous backup navigation strategy for spacecraft station keeping on NRHO, which does not rely on external measurements and communication, e.g. range and range-rate measurements from the Deep Space Network (DSN). We can measure (estimate) the spacecraft position in the Moon-centered inertial frame using the OPNAV algorithm in [188, Figure 5], which is based on detecting the lit limb on the horizon of the Moon using an image taken by an onboard camera. This technique also provides the covariance associated with the position measurement [188, Figure 6]. The position measurement and the covariance can be transformed to the synodic frame F_S.

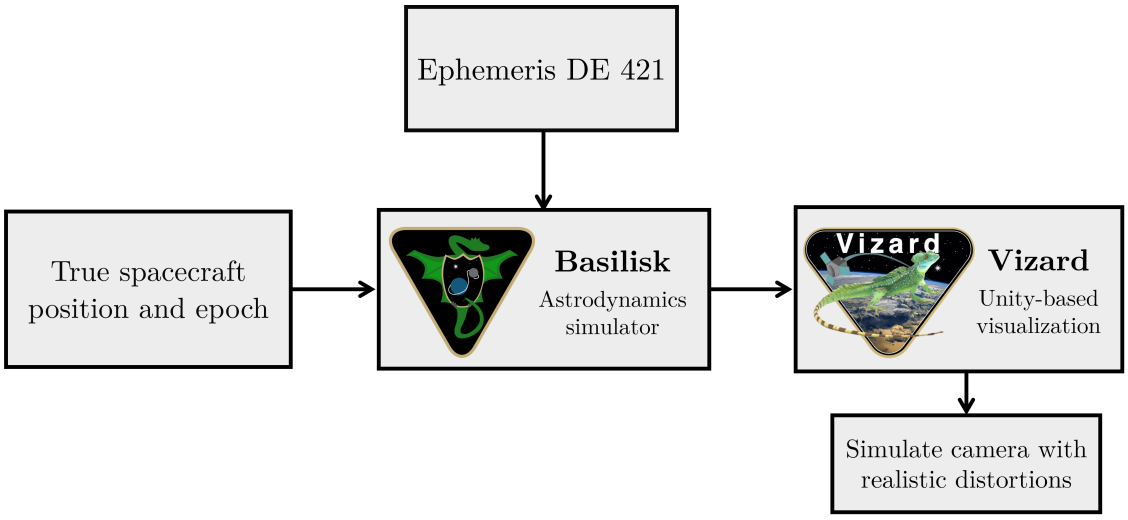


Figure 5.8: Flowchart describing how realistic images of the Moon are acquired from a simulated onboard camera.

To demonstrate closed-loop station keeping with realistic OPNAV we used the astrodynamics simulation and visualization software Basilisk [182] and Vizard [183]. Given a time epoch and the true position of the spacecraft, Basilisk can query the instantaneous positions of Sun, Earth, and Moon from the DE 421 ephemeris [189], and call Vizard to render them along with the spacecraft in an appropriate frame. A pin-hole camera of desired field of view can be attached to the spacecraft, which can be oriented to point to the Moon. Vizard is capable of accurately capturing the illumination of the Moon due

to the Sun in the view from the spacecraft, particularly the lit limb on the Moon horizon, which is a key aspect contributing to the realism of our closed-loop simulator. Moreover, the Vizard camera can simulate commonly observed distortions in images taken from spacecraft. Figure 5.8 illustrates this Basilisk-Vizard pipeline for acquiring Moon images. The limb points on the image of the Moon captured by the spacecraft camera are detected via the Canny transform [190] implemented in OpenCV [191], and then used in the OPNAV algorithm [188, Figure 5] to measure (estimate) the spacecraft position.

Figure 5.9 shows the Vizard windows initialized by Basilisk when the spacecraft is near apolune and perilune. The limb points detected by the Canny transform from the two instances in Figure 5.9 are shown in Figure 5.10.

In our closed-loop simulation we use images of constant resolution (2048×2048 pixels) at all times. Since the computational cost of the OPNAV algorithm is very dependent on the image resolution, we prefer not to vary the resolution of the image captured at different regions of the NRHO trajectory. Since the perilune and apolune of the NRHO are situated at about 3200 km and 70000 km, respectively, to ensure that the Moon occupies roughly the same fraction of area in all camera images, the field of view $\mathcal{F}$ of the camera is varied as follows

$$\mathcal{F} = \min\left\{2.8 \arctan\left(\frac{R_{\text{Moon}}}{d_{\text{Moon}}(t)}\right), \frac{5\pi}{12}\right\} \text{ rad}, \tag{5.21}$$

where R_{Moon} is the mean volumetric radius of the Moon, and $d_{\text{Moon}}(t)$ is the distance between the spacecraft and the Moon. This means that $\mathcal{F}$ ranges between 3° to 78° as the spacecraft moves from apolune to perilune.

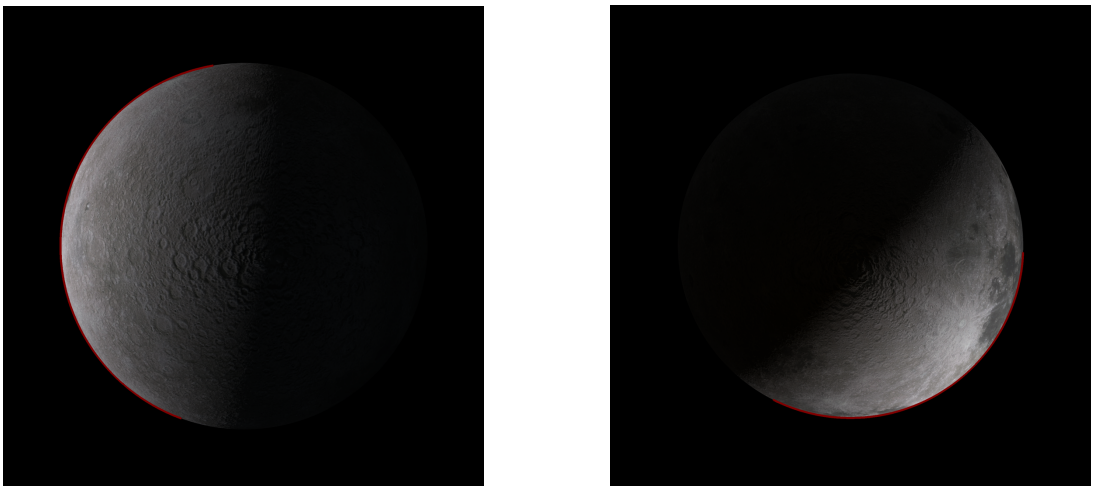
The OPNAV measurement error is empirically observed to range between 2 and 20 km. The large variation in $\mathcal{F}$ causes the OPNAV measurement covariance computed according to [188, Section V] to be inconsistent with the observed measurement error, especially near perilune where $\mathcal{F}$ is large. This inconsistency is a combined effect of the large variation in $\mathcal{F}$ and a potential mismatch in modelling the geometry of the Moon between the OPNAV algorithm and the CAD model of the Moon used in Vizard. Since Moon's oblateness is not very pronounced, we assume that it is a perfect sphere with radius $R_{\text{Moon}} = 1737.4$ km (mean volumetric radius) in [188, Eq. 6] of the OPNAV algorithm. In a

(a) Spacecraft at perilune.

(b) Spacecraft at apolune.

Figure 5.9: Vizard renderings of the Moon and the spacecraft at a perilune and an apolune of a baseline NRHO solution. The windows on the left of each screen-capture show the image of the Moon taken by the spacecraft camera. Vizard renders the realistic lighting conditions of the Moon (as seen by the spacecraft at a given time epoch) based on the positions of Sun, Earth and Moon received from the DE 421 ephemeris via Basilisk.

real mission, we could use a very high resolution camera with constant field of view. This will ensure that limb detection at apolune is reliable even if the Moon occupies a small fraction of the image. Furthermore, if the Moon is imaged at intervals of a few hours, the onboard computers will have sufficient time to operate on the high resolution image of the Moon to extract the OPNAV position measurement. Therefore, in our simulation we scale the measurement covariance from the OPNAV algorithm [188, Figure 6] to ensure that its maximum eigenvalue is at least 16. This ensures that the state estimator will not excessively trust the OPNAV measurements. Future work will examine improvements



(a) Camera image at perilune.

(b) Camera image at apolune.

Figure 5.10: The lit limb (marked in red) detected by the Canny transform performed on the spacecraft camera images from Figure 5.9.

to the simulation framework that allow greater reliance on the OPNAV measurement covariance.

5.2.2.1 State Estimation

We use an EKF [192] to estimate the spacecraft state from the OPNAV position measurements. The true state of the spacecraft at the sampling time t_j, for some $j \in \mathbb{N}$, is

$$\theta_j = \theta_{j-1} + \int_{t_{j-1}}^{t_j} f(\tau, \theta(\tau))\, \mathrm{d}\tau, \tag{5.22}$$

where the true initial state is $\theta(t_0) = \theta_0$. The prediction of the state and covariance at time t_j given the measurements until time t_{j-1} is

$$\widehat{\theta}_{j|j-1} = \widehat{\theta}_{j-1|j-1} + \int_{t_{j-1}}^{t_j} f\big(\tau, \widehat{\theta}_{j-1}(\tau)\big)\, \mathrm{d}\tau, \tag{5.23a}$$

$$\Sigma_{j|j-1} = \Phi(t_j, t_{j-1})\Sigma_{j-1|j-1}\Phi(t_j, t_{j-1})^\top + Q_{j-1}, \tag{5.23b}$$

where $\widehat{\theta}_{j-1}(\tau)$ is the predicted state for $\tau \in [t_{j-1}, t_j]$ with $\widehat{\theta}_{j-1}(t_{j-1}) = \widehat{\theta}_{j-1|j-1}$, and the state transition matrix (STM) $\Phi(t_j, t_{j-1})$ is given by integrating the matrix differential

equation

$$\dot{\Phi}(\tau, t_{j-1}) = \left(\left. \frac{\partial f(\gamma, \theta)}{\partial \theta} \right|_{\left(\tau, \widehat{\theta}_{j-1}(\tau)\right)} \right) \Phi(\tau, t_{j-1}), \tag{5.24}$$

over the time interval $[t_{j-1}, t_j]$ with $\Phi(t_0, t_0) = I_6$. In (5.23b) the STM computed for the predicted trajectory is used to propagate the state covariance. Although there is no mismatch between the actual system dynamics and the system model used in the filter, we introduce process noise to tune the estimator to account for potential model mismatch in the measurement equation[2]. We assume the following form of the process noise covariance (5.23b),

$$Q_j = \begin{bmatrix} q_r^2 I_3 & 0_{3\times 3} \\ 0_{3\times 3} & q_v(t_j)^2 I_3 \end{bmatrix}, \tag{5.25}$$

where

$$q_v(t_j) = q_{v,\mathrm{a}} \left| \frac{\widehat{d}_{\mathrm{Moon}}(t_j) - d_{\mathrm{Moon,p}}}{d_{\mathrm{Moon,a}} - d_{\mathrm{Moon,p}}} \right| + q_{v,\mathrm{p}} \left| \frac{d_{\mathrm{Moon,a}} - \widehat{d}_{\mathrm{Moon}}(t_j)}{d_{\mathrm{Moon,a}} - d_{\mathrm{Moon,p}}} \right|, \tag{5.26}$$

with $q_{v,\mathrm{a}} = 0.05$ cm s^{-1}, $q_{v,\mathrm{p}} = 0.5$ cm s^{-1}, $q_r = 0.5$ km, $d_{\mathrm{Moon,a}} = 71000$ km, $d_{\mathrm{Moon,p}} = 3200$ km, and the predicted distance to Moon denoted by $\widehat{d}_{\mathrm{Moon}}(t_j)$. Since the performance of the station-keeping approach is strongly dependent on the accuracy of the velocity estimates, we prescribe q_v as a function of $\widehat{d}_{\mathrm{Moon}}$ to account for the order of magnitude difference between the apolune speed (~ 0.1 km s^{-1}) and the perilune speed (~ 1.6 km s^{-1}).

The OPNAV position measurement at time t_j entering the filter is denoted by

$$\beta_j = E_r \theta_j + \nu_j, \tag{5.27}$$

where $\nu_j \sim \mathcal{N}(\overline{\nu}_j, R_j)$ with mean $\overline{\nu}_j$ and covariance R_j, and

$$E_r = \begin{bmatrix} I_3 \\ 0_{3\times 3} \end{bmatrix}.$$

[2]Note that we do not account for the highly nonlinear mapping between the Moon image and the OPNAV position measurement described in [188, Figure 5].

Finally, the correction step of the filter for obtaining the optimal state estimate and covariance at time t_j is given by

$$K_j = \Sigma_{j|j-1} E_r^\top \left(E_r \Sigma_{j|j-1} E_r^\top + R_j \right)^{-1}, \tag{5.28a}$$

$$\widehat{\theta}_{j|j} = \widehat{\theta}_{j|j-1} + K_j(\beta_j - E_r \widehat{\theta}_{j|j-1}), \tag{5.28b}$$

$$\Sigma_{j|j} = (I - K_j E_r)\Sigma_{j|j-1}, \tag{5.28c}$$

where K_j is the Kalman gain.

5.2.3 Sequential Linearization-based Targeting

We propose a sequential linearization-based targeting approach for NRHO station keeping in Algorithm 7 based on the well-known x-axis crossing control method [164]. This approach does not require a pre-computed baseline solution. In fact, it can generate one. The key idea behind the algorithm is to test (in Line 11) when the spacecraft arrives near apolune whether the x-component of the velocity at a perilune a few revolutions (typically six [164]) downstream is within a certain threshold ϵ. If it is not, then a sequential linearization approach is used to compute a delta-v correction which ensures that the predicted target perilune state of the nonlinear dynamics (5.18) satisfies the velocity threshold. The proposed approach combines the so called short- and long-horizon maneuvers for station keeping, and reduces vulnerability to accumulation of errors while targeting several revolutions downstream. The typical x-axis crossing control computes delta-v corrections based on the state transition matrix (STM) computed for a single predicted trajectory from current apolune to the target perilune. Since the errors due to linearization accumulate over long duration, the calculated delta-v correction might not result in the intended behavior. The strategy we propose iteratively predicts the target perilune state and updates the delta-v until the desired x-axis velocity at perilune is achieved according to the nonlinear model (5.18). Each iteration of the sequential linearization process solves for the minimum-magnitude delta-v correction which ensures that the magnitude of x-axis velocity at target perilune does not exceed ϵ (Line 12). This problem is equivalent to that of

determining the projection of the origin in $\mathbb{R}^3$ onto the intersection of two half-spaces. The solution of the projection operation is known in closed-form [193, Proposition 29.23]. Any y satisfying the targeting condition in Line 12 belongs to the intersection of two halfspaces

$$\{y \,|\, |a^\top y + b| \leq \epsilon\} \iff \{y \,|\, a^\top y \leq \epsilon - b\} \bigcap \{y \,|\, -a^\top y \leq -\epsilon + b\}, \tag{5.29}$$

where $a^\top = \Phi_{\mathrm{p}}^{[4,4:6]}$ and $b = \dot{x}_{\mathrm{p}}$. Also, observe that when $\epsilon = 0$ Line 12 is equivalent to estimating the projection of the origin onto a subspace. To account for uncertainties with state estimation and system model, the trigger condition is evaluated every time the spacecraft comes close to an apolune. Such a recursive strategy for handling uncertainties obviates the need for separate short-horizon maneuvers to track the solution generated by Algorithm 7.

The input to Algorithm 7 assumes that the true initial state of the spacecraft is close to an apolune state of a known NRHO baseline solution. An estimate for the initial state $\widehat{\theta}_0$ and the associated covariance P_0 are provided as input. It is reasonable to assume that the spacecraft was station-keeping on another baseline solution, and that Algorithm 7 begins execution when the spacecraft reaches the terminal apolune state of that baseline solution. The other inputs to Algorithm 7 are the targeting horizon H, which is the number of revolutions downstream until the target perilune, the planning horizon K, which is the total number of revolutions around the Moon for which station keeping is performed, the sampling time Δt of the OPNAV measurements, and the threshold ϵ for targeting the x-axis velocity at perilune.

The various components of Algorithm 7 are as follows. The $\texttt{Propagate}(\widehat{\theta}, T)$ function returns the final state of the trajectory predicted by propagating the spacecraft dynamics (5.18) with initial condition θ for time interval T. The $\texttt{STM}(\theta, T)$ function returns the STM for time interval T of the linear approximation of the system about the trajectory predicted by $\texttt{Propagate}(\theta, T)$. The $\texttt{PropagateFilter}(\theta, T, \Delta t, P)$ function calls the EKF which accepts OPNAV position measurements at sampling intervals Δt to estimate the true spacecraft state at the end of time interval T, with initial state θ and covariance P. And finally, the $\texttt{SearchPerilune}(\theta, t, T)$ function in Line 5 predicts the target perilune time within the

Algorithm 7 Sequential Linearization-based Targeting

Input: $\widehat{\theta}_0, P_0, K, H, \Delta t, \epsilon$

$E_v \leftarrow \begin{bmatrix} 0_{3\times 3} \\ I_3 \end{bmatrix}$

for $0 \leq k \leq K-1-H$ **do**

  $\widetilde{k} \leftarrow k + H$

  $T \leftarrow \left[\widetilde{k} t_{\text{TBP}}, (\widetilde{k}+1) t_{\text{TBP}}\right]$ ▷ *Time interval containing target perilune after H revolutions*

  $t_{\text{p}} \leftarrow \texttt{SearchPerilune}(\widehat{\theta}_k, T)$ ▷ *Target perilune time instant*

  $T \leftarrow \left[k t_{\text{TBP}}, t_{\text{p}}\right]$

  $\theta_{\text{p}} \leftarrow \texttt{Propagate}(\widehat{\theta}_k, T)$ ▷ *Target perilune state*

  $\Phi_{\text{p}} \leftarrow \texttt{STM}(\widehat{\theta}_k, T)$ ▷ *Linearize about predicted trajectory to target perilune*

  $\dot{x}_{\text{p}} \leftarrow \theta_{\text{p}}^{[4]}$ ▷ *x-axis velocity at target perilune*

  $\Delta v_k \leftarrow [0\ 0\ 0]^\top$

  while $\dot{x}_{\text{p}} \geq \epsilon$ **do**

    ▷ *Compute minimum-norm delta-v as projection on intersection of two half-spaces*

    $\Delta v \leftarrow \underset{y\in\mathbb{R}^3}{\text{argmin}} \|y\|_2 \text{ subject to } |\Phi_{\text{p}}^{[4,4:6]} y + \dot{x}_{\text{p}}| \leq \epsilon$

    $\Delta v_k \leftarrow \Delta v_k + \Delta v$ ▷ *Update delta-v*

    $\theta_{\text{p}} \leftarrow \texttt{Propagate}(\widehat{\theta}_k + E_v \Delta v_k, T)$ ▷ *Predict target perilune state with delta-v correction*

    $\Phi_{\text{p}} \leftarrow \texttt{STM}(\widehat{\theta}_k + E_v \Delta v_k, T)$ ▷ *Linearize about new predicted trajectory*

    $\dot{x}_{\text{p}} \leftarrow \theta_{\text{p}}^{[4]}$

  end while

  $T \leftarrow [k t_{\text{TBP}}, (k+1) t_{\text{TBP}}]$

  ▷ *Estimate state at next apolune with delta-v correction*

  $\left(\widehat{\theta}_{k+1}, P_{k+1}\right) \leftarrow \texttt{PropagateFilter}(\widehat{\theta}_k + E_v \Delta v_k, T, \Delta t, P_k)$

end for

$c \leftarrow \sum_{k=0}^{K-1-H} \|\Delta v_k\|_2$ ▷ *Cumulative station-keeping cost*

Output: $c,\ \{\Delta v_j\}_{j=0}^{K-1-H}$

interval T via a bisection search for determining the perilune state, where the initial state is θ at time t. Note that it is not critical to apply the maneuvers precisely at apolune. Since the time scale of the dynamics is slow near apolune, we can use the time period t_{TBP} of the NRHO in CR3BP to approximate the actual apolune time. However, we cannot use such an approximation for predicting the target perilune states due to the fast time scale of the dynamics near perilune. Even a small error in predicting the target perilune time

may cause the algorithm to target a state significantly different from the actual perilune which will cause the trajectory to diverge. This is because states far from either perilune or apolune are supposed to have significantly large x-axis velocity on a NRHO solution. So targeting them to be close to zero would be incorrect. The `SearchPerilune` function ensures that the target perilune state is accurately predicted.

A block diagram of the simulation stack considered in this work with station keeping, state estimation, and OPNAV is shown in Figure 5.11. The first component of the OPNAV block is described in Figure 5.8. We do not consider spacecraft attitude control in this work. We assume that the spacecraft camera(s) can be oriented to image the Moon in each sampling period.

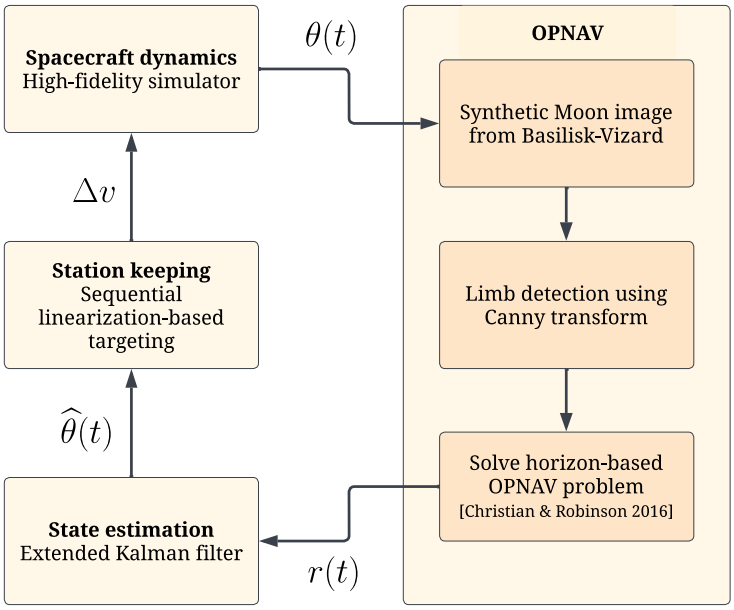


Figure 5.11: Block diagram of the closed-loop station-keeping simulator with OPNAV.

5.2.4 Numerical Results

This section presents the results of station keeping on NRHO from the closed-loop simulation described in Figure 5.11.

Figure 5.12 shows the error in state estimates from the EKF that receives OPNAV measurements at sampling intervals of 3 hours for a spacecraft that followed a known

NRHO baseline solution for 26 days. The 2σ bounds on measurements and state estimates are shown as red and blue patches respectively. The true initial state is an apolune state of the baseline solution. The initial state estimate for the filter is obtained by randomly perturbing the true position by 5 km and true velocity by 5 cm s^{-1}. The initial state covariance is a diagonal matrix chosen to be consistent with the initial state error. The high sensitivity of the spacecraft dynamics near perilune leads to relatively large state estimation error. The spikes of about 40 cm s^{-1} prominently seen in the error in velocity estimates in Figure 5.12 correspond to perilune. Hence, perilune is unsuitable for applying delta-v corrections. However, the velocity estimation error near the apolune region is relatively lower at around 1 cm s^{-1}, which allows maneuvers to be executed effectively.

Furthermore, the measurement error along z-axis in Figure 5.12 is significantly higher than the error along the other two axes. This is partly due to the fact that the boresight direction of the spacecraft camera is closely aligned with the z-axis for a significant duration in each revolution of the NRHO, in particular when the spacecraft is close to apolune and perilune. It is well-known that in horizon-based OPNAV there is larger measurement uncertainty in the camera boresight direction [194].

Given the sensitivity of the spacecraft dynamics, it is possible for the estimated and true states to diverge away after a large delta-v correction ($\sim$ 5 cm s^{-1}) if the filter trusts the predicted state too much without accounting for the discontinuous change in state after the impulse. Therefore, we re-initialize the state covariance matrix to the value at the start of the simulation each time the magnitude of the computed delta-v correction exceeds a specified threshold (1 cm s^{-1}).

Figure 5.14 shows the states estimates and OPNAV measurements for a 1-year station-keeping simulation obtained using Algorithm 7. The OPNAV measurements are obtained at 6 hr intervals. The tolerance ϵ on the target perilune x-axis velocity is 20 m s^{-1}. The estimation error at the apolune region (where the maneuvers are applied) is roughly 1 km in position and 1 cm s^{-1} in velocity. The annual station-keeping cost for this simulation is 114.06 cm s^{-1}. Repeating the simulation with variations in the initial state estimate given to the the filter results in similar station-keeping cost. In comparison to the relevant NRHO

station-keeping result with OPNAV reported for the medium-error quiet spacecraft case in [168, Table 7], the result from the proposed approach requires significantly lower annual delta-v. The cumulative delta-v as a function of time, shown in Figure 5.14a, increases gradually without large jumps, which is desirable for the reliability and safety of the approach.

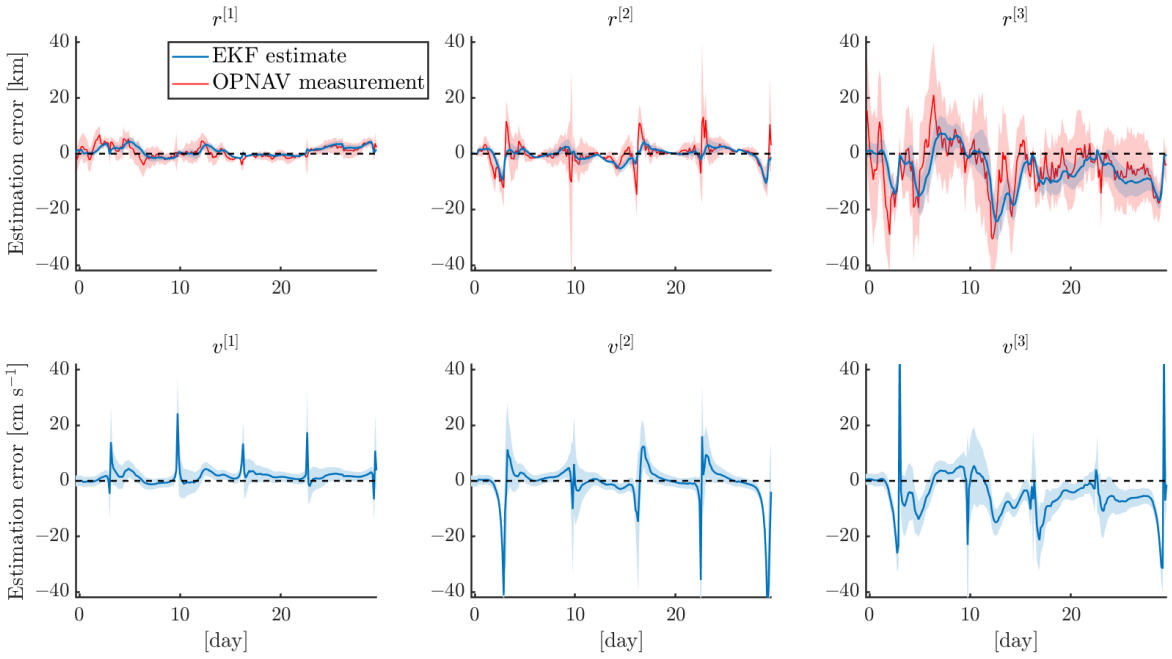


Figure 5.12: EKF state estimation error with OPNAV measurements provided at 3 hour intervals for a spacecraft that follows a NRHO baseline solution for 26 days. The 2σ bands for the OPNAV position measurement (red) and state estimate (blue) are shown.

5.2.5 Conclusions

We proposed a targeting-based approach for station keeping on the near rectilinear halo orbit (NRHO) chosen for the Lunar Gateway and demonstrated it in a closed-loop simulation with a state estimator which receives spacecraft position measurement from horizon-based optical navigation (OPNAV). We use the astrodynamics simulation and visualization software Basilisk and Vizard for generating realistic OPNAV measurements from synthetic images of Moon which accurately show the limb formed on the Moon horizon due to illumination from the Sun. The proposed approach is able to deliver an annual station-

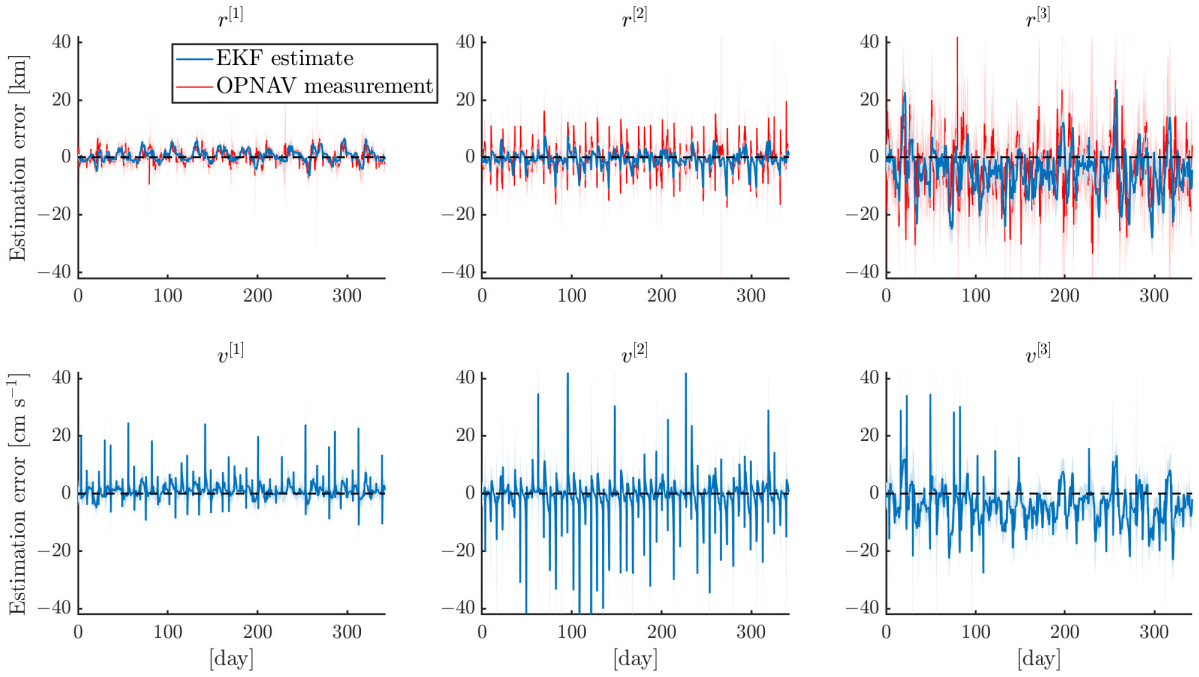
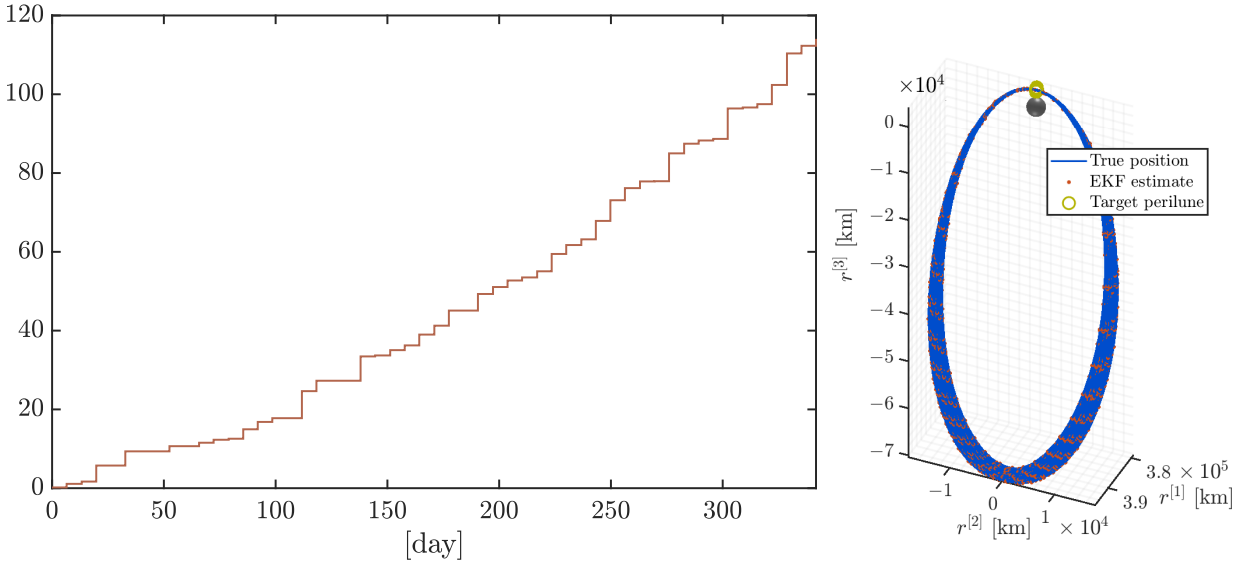


Figure 5.13: The state estimation error for a 1-year closed-loop station-keeping simulation with OPNAV measurements. The 2σ bands for the OPNAV position measurement (red) and state estimate (blue) are shown.



(a) Cumulative delta-v = 114.06 cm s^{-1}.

(b) Position trajectory.

Figure 5.14: Station keeping with OPNAV measurements.

keeping cost close to 1.14 m s^{-1}, and it is robust to measurement uncertainty of about 10 km observed with OPNAV measurements on the NRHO trajectory.

Future work will improve the simulator to make the computation of OPNAV measurement covariance more reliable, and specialize the design of the filter to exploit the dynamical features of a high-fidelity NRHO solution to reduce the state estimation error at perilune, which will potentially contribute to a reduction in the annual station-keeping cost. The robustness and efficiency of the station-keeping approach with DSN and OPNAV measurements will be compared. In addition, the performance of the proposed approach adapted with short-horizon maneuvers will be examined.

5.3 Spacecraft Rendezvous with Continuous-Time Passive Safety

5.3.1 Introduction

Spacecraft rendezvous is a crucial maneuver for operating space stations and for complex space missions (such as in the Apollo and Artemis [195] programs) involving multiple space vehicles. While the International Space Station (ISS) will soon be decommissioned [196], the upcoming deployment of a space station–the Lunar Orbital Platform-Gateway (LOP-G)–on the 9:2 near rectilinear halo orbit (NRHO) [197], has renewed interest in the development of safe and efficient technologies for spacecraft rendezvous on NRHO. In addition, there is an emphasis on enabling autonomous operations to enhance the scale and capacity of space missions [198], since human-assisted operations are slow, expensive, and detrimental for scalability. In realistic simulations and for guidance synthesis, NRHO is computed as a long-duration aperiodic trajectory for a high-fidelity dynamic model with ephemeris data and higher-order effects (e.g. solar radiation pressure and gravitational perturbations) [145]. In the ideal setting, the NRHO is a non-Keplerian periodic motion which solves the circular restricted three-body-problem (CR3BP). However, all instances of rendezvous in space missions thus far, have involved Keplerian orbits (primarily around the Earth and the Moon). Rendezvous of a spacecraft with the LOP-G (also referred to as the Gateway), as a part of the ongoing Artemis program, will be the first attempt of such a maneuver on a non-Keplerian trajectory. Moreover, unlike the literature on rendezvous on elliptic (Keplerian) orbits, which spans more than half a century [20, Sec. 3.2], [199], most of the methods published for rendezvous on the NRHO appeared within the last decade.

A guidance, navigation, and control (GNC) system is a key component for executing a rendezvous maneuver. The focus of this work is on an optimization-based (rendezvous) guidance algorithm, i.e., we design the feedforward block within the control system. More specifically, we develop a guidance algorithm for the rendezvous of a spacecraft to the Gateway while ensuring continuous-time passive safety for a specified duration. Further, to enable autonomy, there is a need for guidance algorithms which can automatically generate the complete rendezvous maneuver (spanning multiple safety zones and hold

points) while guaranteeing properties such as continuous-time constraint satisfaction (including safety) and fuel-optimality. Optimization-based methods are well-suited for this goal, due to the recent advances in their real-time performance and in solving general nonconvex problems [22, 5, 3]. We explore one such approach based on sequential convex programming (SCP) in this work.

We present a solution method for passively-safe rendezvous of a spacecraft to the Gateway (within a high-fidelity dynamic model) based on the continuous-time successive convexification (CT-SCVX) framework [22]. We reformulate and augment the continuous-time passive-safety constraint to the system dynamics to eliminate the commonly-encountered inter-sample constraint violation without the need for computationally expensive mesh-refinement heuristics. In other words, the proposed framework can generate a high-fidelity solution with coarse discretization grids. We solve the constraint-reformulated optimal control problem for rendezvous by combining ℓ_1 exact penalization of nonconvexities and a convergence-guaranteed sequential convex programming (SCP) algorithm, called the prox-linear method [5] for convex-composite problems. Furthermore, we impose chance constraints to incorporate uncertainties encountered in a closed-loop execution, and reduce its conservativeness by jointly synthesizing a feedback controller that inhibits the growth of the state covariance by using the state measurement error statistics.

Related Work

A variety of methods have been explored for incorporating safety constraints in rendezvous guidance. Much of the initial results were geared towards elliptic (Keplerian) orbits [200, 201, 202, 203]. In particular, the approach in [203] imposes the passive-safety constraint by explicitly parameterizing and constraining the discrete-time free-drift trajectories while preserving convexity, which invariably leads to excessive conservativeness and inter-sample constraint violation.

Methods for safe rendezvous on NRHO were proposed in [204, 31, 205, 206, 207, 208, 209]. In particular, [31] provides a case study based on the proposed JAXA HTV-X resupply vehicle which will rendezvous with the Gateway.

Generally, the major limitations of the prior work include: (i) inter-sample violation of path constraints (including passive safety); (ii) use of low-fidelity dynamic model (such as the CR3BP model) to evaluate passive safety [205, 208, 209]; (iii) use of a myopic, receding horizon solution approach which can lead to artificial infeasibility [204].

Notation

We adopt the following notation in this work. $\mathbb{R}, \mathbb{R}_+, \mathbb{R}^n$, and $\mathbb{R}^{n\times m}$ denote the set of reals, nonnegative reals, $n \times 1$ vector of reals, and $n \times m$ matrix of reals, respectively. 0_n and 1_n denote vector of zeros and vector of ones in $\mathbb{R}^n$, respectively, and I_n denotes the identity matrix in $\mathbb{R}^{n\times n}$. $|v|_+$ and v^2 denote $\square \mapsto \max\{0, \square\}$ and $\square \mapsto \square^2$, respectively, applied elementwise to vector $v \in \mathbb{R}^n$, and $\|v\|$ denotes its Euclidean norm. (v, w) denotes the concatenation of vectors $v \in \mathbb{R}^n$, $w \in \mathbb{R}^m$ to form a vector in $\mathbb{R}^{n+m}$, and $[A\ B]$ denotes the concatenation of matrices $A \in \mathbb{R}^{n\times m_1}$ and $B \in \mathbb{R}^{n\times m_2}$ to form a matrix in $\mathbb{R}^{n\times(m_1+m_2)}$. $\nabla_m h(x_1, \ldots, x_n)$ denotes the gradient of h with respect to mth argument ($m \leq n$), evaluated at $x_1, \ldots, x_n$, and $\nabla g(x)$ denotes the gradient of (single-argument function) g evaluated at x. The interior of set $\mathcal{D}$ is denoted by $\operatorname{int}\mathcal{D}$.

5.3.2 Problem Formulation

This section presents the optimal control problem describing the rendezvous of a chaser spacecraft to a target spacecraft while satisfying path constraints on the state and input, and ensuring passive safety of the chaser's trajectory. The uncontrolled translational motion (free-drift) of a spacecraft in the cis-lunar space, which is influenced by the ephemeris states (position and velocity) of the Moon, Earth, and Sun, higher-order gravitational perturbations (Moon J2), and solar radiation pressure (SRP), is represented in the Moon-centered J2000 [210, Sec. 3.7] inertial frame by

$$\dot{z}(t) = \frac{\mathrm{d}z(t)}{\mathrm{d}t} = \check{f}(t, z(t)), \quad t \in [t_\mathrm{i}, t_\mathrm{f}], \tag{5.30}$$

where t_i and t_f are the initial and final time, respectively (see Appendix E for further details). Let $n_x = 6$ and $n_u = 3$ denote the state and control input dimensions, respectively. The state trajectory x of a chaser spacecraft relative to a given free-drift state trajectory $\check{z}$ of a target spacecraft satisfies

$$\dot{x}(t) = f(t, x(t), u(t)) \triangleq \check{f}(t, x(t) + \check{z}(t)) - \check{f}(t, \check{z}(t)) + \begin{bmatrix} 0_{n_u \times n_u} \\ I_{n_u} \end{bmatrix} u(t), \quad t \in [t_\mathrm{i}, t_\mathrm{f}], \tag{5.31}$$

where u is a piecewise continuous control input acting on the chaser, which is either a velocity impulse or an acceleration.

5.3.2.1 Passive Safety

Consider a polytopic avoid set $\mathcal{P} \triangleq \mathrm{int}\,\{\zeta \in \mathbb{R}^{n_x} | H\zeta \leq h\}$, for some $H \triangleq [H_1 \ \ldots \ H_m]^\top \in \mathbb{R}^{m \times n_x}$ and $h \triangleq (h_1, \ldots, h_m) \in \mathbb{R}^m$. We require that the free-drift trajectory of the chaser, starting at time $t \in [t_\mathrm{i}, t_\mathrm{f}]$, does not enter $\mathcal{P}$ for a safety duration of t_s. Given a state trajectory x for the chaser over $[t_\mathrm{i}, t_\mathrm{f}]$, the free-drift trajectory x^t, starting from $x(t)$ at $t \in [t_\mathrm{i}, t_\mathrm{f}]$, satisfies

$$\mathring{x}^t(\tau) = \frac{\mathrm{d}x^t(\tau)}{\mathrm{d}\tau} = f(t + \tau, x^t(\tau), 0_{n_u}), \quad \tau \in [0, t_\mathrm{s}], \tag{5.32}$$

with initial condition $x^t(0) = x(t)$. Then the passive-safety constraint is given by

$$x^t(\tau) \notin \mathcal{P}, \quad \forall\, \tau \in [0, t_\mathrm{s}],\ t \in [t_\mathrm{i}, t_\mathrm{f}]. \tag{5.33}$$

Let $F_\mathrm{fd} : \mathbb{R} \times \mathbb{R} \times \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ return the state on the free-drift trajectory, given the drift duration and initial condition, i.e., for any $\tau \in [0, t_\mathrm{s}]$ and $t \in [t_\mathrm{i}, t_\mathrm{f}]$

$$F_\mathrm{fd}(t + \tau, t, x(t)) \triangleq x(t) + \int_0^\tau f(t + \theta, x^t(\theta), 0_{n_u})\mathrm{d}\theta = x^t(\tau). \tag{5.34}$$

To encode (5.33) within a gradient-based optimization framework, we use an equivalent alternate representation based on the signed-distance function (see Appendix F). For any

$\tau \in [0, t_s]$ and $t \in [t_i, t_f]$, we have that

$$x^t(\tau) \notin \mathcal{P} \iff d_{\mathcal{P}}(x^t(\tau)) \geq 0. \tag{5.35}$$

Next, we use $\Xi : \mathbb{R} \times \mathbb{R}^{n_x} \to \mathbb{R}_+$ to measure the violation in constraint (5.33) via a continuously differentiable exterior penalty function, i.e., for any $t \in [t_i, t_f]$

$$\Xi(t, x(t)) \triangleq \int_0^{t_s} |-d_{\mathcal{P}}(F_{\text{fd}}(t+\tau, t, x(t)))|_+^2 \mathrm{d}\tau. \tag{5.36}$$

Then the following holds for any $t \in [t_i, t_f]$

$$x^t(\tau) \notin \mathcal{P} \quad \forall\, \tau \in [0, t_s] \iff \Xi(t, x(t)) = 0. \tag{5.37}$$

A key step in the proposed solution method is evaluating the gradient of Ξ on another (reference) state trajectory $\bar{x}$ over $[t_i, t_f]$. First, note that for any $\tau \in [0, t_s]$ and $t \in [t_i, t_f]$, we have that $\nabla_3 F_{\text{fd}}(t+\tau, t, \bar{x}(t)) = \Phi_{\text{fd}}(t+\tau, t)$, which is obtained from the following initial value problem

$$\frac{\mathrm{d}}{\mathrm{d}\theta} \begin{bmatrix} \bar{x}^t(\theta) \\ \Phi_{\text{fd}}(t+\theta, t) \end{bmatrix} = \begin{bmatrix} f(t+\theta, \bar{x}^t(\theta), 0_{n_u}) \\ \nabla_2 f(t+\theta, \bar{x}^t(\theta), 0_{n_u}) \Phi_{\text{fd}}(t+\theta, t) \end{bmatrix}, \quad \theta \in [0, \tau]. \tag{5.38}$$

with initial condition $\bar{x}^t(0) = x(t)$ and $\Phi_{\text{fd}}(t, t) = I_{n_x}$. Then for any $t \in [t_i, t_f]$, we have that

$$\nabla_2 \Xi(t, \bar{x}(t)) = -2 \int_0^{t_s} |-d_{\mathcal{P}}(\bar{x}^t(\tau))|_+ \nabla d_{\mathcal{P}}(\bar{x}^t(\tau)) \Phi_{\text{fd}}(t+\tau, t) \mathrm{d}\tau, \tag{5.39}$$

where

$$-2|-d_{\mathcal{P}}(\bar{x}^t(\tau))|_+ \nabla d_{\mathcal{P}}(\bar{x}^t(\tau)) = \begin{cases} 2\big(\bar{x}^t(\tau) - (\bar{x}^t(\tau))^{\partial \mathcal{P}}\big)^\top & \text{if } d_{\mathcal{P}}(\bar{x}^t(\tau)) < 0, \\ 0 & \text{otherwise.} \end{cases} \tag{5.40}$$

5.3.2.2 Backward Reachability

The formulation of the passive-safety constraint in (5.36) is based on *forward reachability*. We can provide an alternate equivalent characterization of passive safety based on *backward reachability*.

The state on the free-drift trajectory can be approximated by linearizing F_{fd} about a reference state trajectory $\bar{x}$, i.e., for $\tau \in [0, t_{\text{s}}]$ and $t \in [t_{\text{i}}, t_{\text{f}}]$, we have that

$$F_{\text{fd}}(t+\tau, t, x(t)) \approx \Phi_{\text{fd}}(t+\tau, t)x(t) + \phi_{\text{fd}}(t+\tau, t), \tag{5.41}$$

where $\phi_{\text{fd}}(t+\tau, t) \triangleq F_{\text{fd}}(t+\tau, t, \bar{x}(t)) - \Phi_{\text{fd}}(t+\tau, t)\bar{x}(t)$. Note that for any $z \in \mathbb{R}^{n_x}$, the following holds

$$\begin{aligned}
& d_{\mathcal{P}}(\Phi_{\text{fd}}(t+\tau, t)z + \phi_{\text{fd}}(t+\tau, t)) \geq 0, && \text{(5.42a)}\\
\iff & \max_{1 \leq i \leq m} H_i^\top \Phi_{\text{fd}}(t+\tau, t)z - h_i - H_i^\top \phi_{\text{fd}}(t+\tau, t) \geq 0, && \text{(5.42b)}\\
\iff & z \notin \mathcal{B}_\tau^t \triangleq \text{int}\{\zeta \in \mathbb{R}^{n_x} \mid H\Phi_{\text{fd}}(t+\tau, t)\zeta \leq h - H\phi_{\text{fd}}(t+\tau, t)\}, && \text{(5.42c)}\\
\iff & d_{\mathcal{B}_\tau^t}(z) \geq 0, && \text{(5.42d)}
\end{aligned}$$

where the parameterized set $\mathcal{B}_\tau^t$ is a backward reachable set (for the system linearized about $\bar{x}$). It defines the set of all states at time t which free-drift into $\mathcal{P}$ after a duration of τ. Next, we define $\bar{\Xi} : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \to \mathbb{R}_+$ as

$$\bar{\Xi}(t, x(t), \bar{x}(t)) \triangleq \int_0^{t_{\text{s}}} |-d_{\mathcal{B}_\tau^t}(x(t))|_+^2 \text{d}\tau, \tag{5.43}$$

so that the following holds for any $t \in [t_{\text{i}}, t_{\text{f}}]$

$$x(t) \notin \bigcup_{\tau \in [0, t_{\text{s}}]} \mathcal{B}_\tau^t \iff \bar{\Xi}(t, x(t), \bar{x}(t)) = 0. \tag{5.44}$$

The convergence of an iterative solution method for computing x and u, where F_{fd} is linearized at each iteration about the previous iterate (denoted by $\bar{x}$), would mean that, for

any $t \in [t_\mathrm{i}, t_\mathrm{f}]$

$$x(t) \to \bar{x}(t) \implies d_{\mathcal{P}}(\Phi_{\mathrm{fd}}(t+\tau, t)x(t) + \phi_{\mathrm{fd}}(t+\tau, t)) \to d_{\mathcal{P}}(F_{\mathrm{fd}}(t+\tau, t, \bar{x}(t))).$$

Then from (5.42a) and (5.42d), we obtain the following for $t \in [t_\mathrm{i}, t_\mathrm{f}]$

$$\Xi(t, \bar{x}(t)) = 0 \iff \bar{\Xi}(t, \bar{x}(t), \bar{x}(t)) = 0. \tag{5.45}$$

We can similarly consider other safety-based constraints via the set-based approach described above (see Appendix G). Note that the parameterized sets $\mathcal{B}_\tau^t$ need to be recomputed at each iteration of the solution method, which, however, need not be a computationally expensive operation since the sets at each iteration are defined for a linearized system.

5.3.2.3 Chance Constraints

Consider a random state trajectory ξ, where $\xi(t) \sim \mathcal{N}(x(t), \Sigma(t))$ for $t \in [t_\mathrm{i}, t_\mathrm{f}]$. Then the passive-safety constraint is imposed with some probability level $\beta \in (0, 1)$, as follows

$$\mathbb{P}(d_{\mathcal{P}}(F_{\mathrm{fd}}(t+\tau, t, \xi(t))) \geq 0) \geq \beta, \quad \tau \in [0, t_\mathrm{s}],\ t \in [t_\mathrm{i}, t_\mathrm{f}], \tag{5.46}$$

Given a reference state trajectory $\bar{x}$, the signed-distance with respect to $\mathcal{P}$ of the state on the free-drift trajectory, starting from $x(t)$ at time $t \in [t_\mathrm{i}, t_\mathrm{f}]$, can be approximated as

$$\begin{aligned}
&d_{\mathcal{P}}(F_{\mathrm{fd}}(t+\tau, t, x(t))) \\
&\approx d_{\mathcal{P}}(F_{\mathrm{fd}}(t+\tau, t, \bar{x}(t))) + \nabla d_{\mathcal{P}}(F_{\mathrm{fd}}(t+\tau, t, \bar{x}(t)))\Phi_{\mathrm{fd}}(t+\tau, t)(x(t) - \bar{x}(t)), && \text{(5.47a)} \\
&= a^t(\tau)^\top x(t) + b^t(\tau), && \text{(5.47b)}
\end{aligned}$$

for $\tau \in [0, t_\mathrm{s}]$, where

$$a^t(\tau) \triangleq \nabla d_{\mathcal{P}}(F_{\mathrm{fd}}(t+\tau, t, \bar{x}(t)))\Phi_{\mathrm{fd}}(t+\tau, t), \tag{5.48a}$$

$$b^t(\tau) \triangleq d_{\mathcal{P}}(F_{\mathrm{fd}}(t+\tau, t, \bar{x}(t))) - a^t(\tau)\bar{x}(t). \tag{5.48b}$$

Since $\xi(t)$ is normally distributed for each $t \in [t_{\mathrm{i}}, t_{\mathrm{f}}]$, we have that

$$a^t(\tau)^\top x(t) + b^t(\tau) + c^t(\tau) \geq 0 \implies \mathbb{P}(a^t(\tau)^\top \xi(t) + b^t(\tau) \geq 0) \geq \beta, \tag{5.49}$$

for $\tau \in [0, t_{\mathrm{s}}]$, where $c^t(\tau) \triangleq -\sqrt{\chi^2_{n_x}(\beta) a^t(\tau)^\top \Sigma(t) a^t(\tau)}$ and $\chi^2_{n_x}$ is the probability density function of the chi-squared distribution with n_x degrees of freedom [211, Cor. 2].

In the proposed iterative solution method the system is linearized at each iteration (about the previous iterate). Then the random state trajectory ξ for the linearized system automatically evolves with a Gaussian distribution if the initial state and all injected uncertainties are Gaussian random variables. Next we define $\bar{\Xi} : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \to \mathbb{R}$ as follows

$$\bar{\Xi}(t, x(t), \bar{x}(t)) \triangleq \int_0^{t_{\mathrm{s}}} |-a^t(\tau)^\top x(t) - b^t(\tau) - c^t(\tau)|_+^2 \mathrm{d}\tau. \tag{5.50}$$

For each $t \in [t_{\mathrm{i}}, t_{\mathrm{f}}]$, we obtain the following from (5.49)

$$\bar{\Xi}(t, x(t), \bar{x}(t)) = 0 \iff \mathbb{P}(a^t(\tau)^\top \xi(t) + b^t(\tau) \geq 0) \geq \beta \quad \forall\, \tau \in [0, t_{\mathrm{s}}]. \tag{5.51}$$

Furthermore, note that the convergence of the solution method leads to the following

$$x(t) \to \bar{x}(t) \implies \bar{\Xi}(t, x(t), \bar{x}(t)) \to \int_0^{t_{\mathrm{s}}} |-d_{\mathcal{P}}(F_{\mathrm{fd}}(t+\tau, t, \bar{x}(t))) - c^t(\tau)|_+^2 \mathrm{d}\tau. \tag{5.52}$$

The limiting value of $\bar{\Xi}$ at convergence is a tightened form of constraint due to Ξ in (5.36) in the deterministic case, which aligns with the expectation that the chance constraint should be more conservative than its deterministic counterpart.

5.3.2.4 Optimal Control Problem

The optimal control problem for fuel-efficient rendezvous of the chaser spacecraft to the target is given by

$$\underset{x,\,u}{\text{minimize}} \int_{t_\text{i}}^{t_\text{f}} \|u(t)\| \mathrm{d}t \tag{5.53a}$$

$$\text{subject to } \dot{x}(t) = f(t, x(t), u(t)), \qquad t \in [t_\text{i}, t_\text{f}] \tag{5.53b}$$

$$g(x(t), u(t)) \leq 0, \qquad t \in [t_\text{i}, t_\text{f}] \tag{5.53c}$$

$$\Xi(t, x(t)) = 0, \qquad t \in [t_\text{i}, t_\text{f}] \tag{5.53d}$$

$$P(x(t_\text{i}), x(t_\text{f})) = 0 \tag{5.53e}$$

The path constraint function $g : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_g}$, with $n_g = 4$, encodes upper-bound u_max and lower-bound u_min on the control input magnitude, and an approach cone constraint with half cone angle θ_ac and cone axis e_ac, as follows

$$g(x(t), u(t)) \triangleq \begin{bmatrix} \|u(t)\|^2 - u_\text{max}^2 \\ u_\text{min}^2 - \|u(t)\|^2 \\ \|\cos\theta_\text{ac}\, {}^r\!E\, x(t)\|^2 - (e_\text{ac}^\top {}^r\!E\, x(t))^2 \\ -e_\text{ac}^\top {}^r\!E\, x(t) \end{bmatrix}, \tag{5.54}$$

where ${}^r\!E$ selects the position coordinates from a vector in $\mathbb{R}^{n_x}$. The scalar-valued components of g are written in a form that ensures continuous differentiability of g, particularly the second-order-cone constraint [112, Sec. 3.2.4].

5.3.2.5 Constraint Reformulation

Given a state trajectory x and control input u over $[t_\text{i}, t_\text{f}]$ the continuous-time satisfaction of path constraints can equivalently expressed through the isoperimetric reformulation [96, Sec. 10].

$$g(x(t), u(t)) \leq 0,\ \Xi(t, x(t)) = 0,\ \forall\, t \in [t_\text{i}, t_\text{f}], \tag{5.55a}$$

$$\iff \int_{t_\mathrm{i}}^{t_\mathrm{f}} (1_{n_g}^\top |g(x(t), u(t))|_+^2, \Xi(t, x(t)))\mathrm{d}t = 0_2. \tag{5.55b}$$

Note that Ξ is not composed with an exterior penalty function since it is nonnegative everywhere. For each $t \in [t_\mathrm{i}, t_\mathrm{f}]$, the differential-algebraic system given by

$$\dot{x}(t) = f(t, x(t), u(t)), \tag{5.56a}$$

$$g(x(t), u(t)) \leq 0,\ \Xi(t, x(t)) = 0, \tag{5.56b}$$

is equivalent to the following two-point boundary value problem

$$\dot{x}(t) = f(t, x(t), u(t)), \tag{5.57a}$$

$$\dot{y}(t) = \begin{bmatrix} 1_{n_g}^\top |g(x(t), u(t))|_+^2 \\ \Xi(t, x(t)) \end{bmatrix}, \tag{5.57b}$$

$$y(t_\mathrm{i}) = y(t_\mathrm{f}), \tag{5.57c}$$

where (5.57b) is a auxiliary dynamical system with state trajectory y for measuring the cumulative continuous-time constraint violation. The periodic boundary conditions (5.57c) ensure that the path constraints are satisfied everywhere within $[t_\mathrm{i}, t_\mathrm{f}]$.

5.3.2.6 Reformulated Optimal Control Problem

Consider the following augmented dynamical system for $t \in [t_\mathrm{i}, t_\mathrm{f}]$

$$\dot{\tilde{x}}(t) = \tilde{f}(t, \tilde{x}(t), u(t)) \triangleq \begin{bmatrix} f(t, x(t), u(t)) \\ 1_{n_g}^\top |g(x(t), u(t))|_+^2 \\ \Xi(t, x(t)) \end{bmatrix}, \tag{5.58}$$

where $\tilde{x} \triangleq (x, y)$ is the augmented state trajectory, with the augmented state dimension denoted by $n_{\tilde{x}} \triangleq n_x + 2$. We use selector matrices ${}^xE$ and ${}^yE$ to select the elements of $\tilde{x}$ corresponding to x and y, respectively.

The optimal control problem (5.53) is reformulated using the augmented system (5.58)

as follows.

$$\underset{\tilde{x},\,u}{\text{minimize}} \int_{t_\text{i}}^{t_\text{f}} \|u(t)\| \mathrm{d}t \tag{5.59a}$$

$$\text{subject to } \dot{\tilde{x}}(t) = \tilde{f}(t, \tilde{x}(t), u(t)), \qquad t \in [t_\text{i}, t_\text{f}] \tag{5.59b}$$

$$^{y}E\left(\tilde{x}(t_\text{f}) - \tilde{x}(t_\text{i})\right) = 0 \tag{5.59c}$$

$$P(^{x}E\,\tilde{x}(t_\text{i}), {}^{x}E\,\tilde{x}(t_\text{f})) = 0 \tag{5.59d}$$

5.3.3 Discretization & Solution Method

Consider a discretization grid of size N within $[t_\text{i}, t_\text{f}]$: $t_\text{i} = t_1 < \ldots < t_N = t_\text{f}$. We treat the augmented states $\tilde{x}_k$ at node points t_k as decision variables. The control input is parameterized via $\nu : [t_\text{i}, t_\text{f}] \to \mathbb{R}^{n_u}$ as follows

$$\nu(t) \triangleq \sum_{k=1}^{N} u_k \Gamma_u^k(t), \tag{5.60}$$

with coefficients $u_k \in \mathbb{R}^{n_u}$ and basis functions $\Gamma_u^k : [t_\text{i}, t_\text{f}] \to \mathbb{R}$, satisfying $\nu(t_k) = u_k$, for $k = 1, \ldots, N$. We assume that the basis functions are chosen such that, within interval $[t_k, t_{k+1}]$ for any $k = 1, \ldots, N-1$, ν is influenced solely by u_k and u_{k+1} (see Appendix C for examples of well-known parameterizations).

Remark 24. *If the basis functions are convex then imposing the convex constraints on the control input only at nodes t_k is sufficient for continuous-time satisfaction; they do not require the isoperimetric reformulation described in (5.55). Further, if the basis functions are piecewise constant (such as in impulse, finite-burn pulse, and zero order hold parameterizations) imposing nonconvex control input constraints only at the nodes of the discretization grid is sufficient for continuous-time satisfaction.*

Next, for each $k = 1, \ldots, N-1$, we define $F_k : \mathbb{R}^{n_{\tilde{x}}} \times \mathbb{R}^{n_{\tilde{x}}} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_{\tilde{x}}}$ as follows

$$F_k(\tilde{x}_{k+1}, \tilde{x}_k, u_{k+1}, u_k) \triangleq \tilde{x}_{k+1} - \tilde{x}_k - \int_{t_k}^{t_{k+1}} \tilde{f}(\tilde{x}^k(t), \nu(t))\mathrm{d}t, \tag{5.61}$$

where the augmented state trajectory $\tilde{x}^k$ satisfies (5.58) on $[t_k, t_{k+1}]$ with control input ν, and initial condition $\tilde{x}_k$. Then the discretization of (5.59b), along with (5.59c), yields

$$F_k(\tilde{x}_{k+1}, \tilde{x}_k, u_{k+1}, u_k) = 0, \tag{5.62a}$$

$${}^{y}E\,(\tilde{x}_{k+1} - \tilde{x}_k) = 0, \tag{5.62b}$$

However, inclusion of (5.62) as constraints in an optimization problem leads to violation in linear independence constraint qualification (LICQ), which is essential for enabling exact penalization, a key step in the proposed framework. We relax (5.62b) to an inequality via a constant $\epsilon > 0$ (see [22, Sec. 3.1] for further details about the implications of this relaxation). Next, due to parameterization (5.60), the cost function (5.59a) can be expressed as

$$\int_{t_\mathrm{i}}^{t_\mathrm{f}} \|u(t)\| = \sum_{k=1}^{N} \alpha_k \|u_k\|, \tag{5.63}$$

where constants $\alpha_k \in \mathbb{R}_+$ depend on the choice of parameterization ν (see Appendix C for examples). Then the reformulated optimal control problem (5.59) transforms under discretization and control input parameterization as follows

$$\underset{\tilde{x}_k,\, u_k j}{\text{minimize}} \ \sum_{k=1}^{N} \alpha_k \|u_k\| \tag{5.64a}$$

$$\text{subject to } F_k(\tilde{x}_{k+1}, \tilde{x}_k, u_{k+1}, u_k) = 0, \qquad k = 1, \ldots, N-1 \tag{5.64b}$$

$${}^{y}E\,(\tilde{x}_{k+1} - \tilde{x}_k) \leq \epsilon, \qquad k = 1, \ldots, N-1 \tag{5.64c}$$

$$u_k \in \mathbb{U}, \qquad k = 1, \ldots, N \tag{5.64d}$$

$$P({}^{x}E\,\tilde{x}_1, {}^{x}E\,\tilde{x}_N) = 0 \tag{5.64e}$$

Compact set $\mathbb{U}$ serves a three-fold purpose: (i) representing control input constraints

depending on the choice of parameterization (see Remark 24); (ii) establishing a pointwise bound on the path constraint violation as a consequence of relaxation (5.64c); and (iii) invoking the convergence guarantees of the prox-linear method which is a key component of the proposed SCP-based solution method. We refer the reader to [22, Sec. 3, 4] for details.

Remark 25. *When the chance constraints described in Section 5.3.2.3 are included, the description of the optimal control problem (5.53) requires a reference state trajectory. This is because function Ξ in (5.58) is replaced with $\bar{\Xi}$ in (5.50) which requires a reference state as additional argument. Consequently, for each $k \in 1, \ldots, N-1$, F_k requires reference quantities $\bar{x}_k$, $\bar{u}_k$, $\bar{u}_{k+1}$ to generate a reference state trajectory $\bar{x}^k$ over $[t_k, t_{k+1}]$ with initial condition $\bar{x}_k$ and control input $\bar{v}$ parameterized with coefficients $\bar{u}_k$ and $\bar{u}_{k+1}$.*

5.3.3.1 Closed-Loop Covariance Correction

The formulation thus far has focused on the feedforward block of the control system without any considerations about the information provided by and uncertainties encountered by sensors. Incorporating some of this information within the feedforward block can improve the performance of the feedback controller and reduce the conservativeness of the chance constraints imposed during the reference signal design. In particular, the use of open-loop state covariance, which grows over time, can lead to prohibitively conservative chance constraints. To reduce their conservativeness we can: (i) include sensor measurement statistics and actuation noise; and (ii) introduce a single-step impulsive deadbeat controller to reduce the growth of the state covariance.

We assume that the time instants at which measurements are received and the impulsive corrections are made, coincide with the nodes of the discretization grid. Suppose that a state trajectory for (5.31) over $[t_\mathrm{i}, t_\mathrm{f}]$ with parameterized control input v are given, which we will qualify as "nominal". Then for each $k = 1, \ldots, N$, let x_k and u_k denote the corresponding state and control input, respectively, at the discretization node t_k. Next, to impose the chance constraint described in Section 5.3.2.3, we require linearizing the system about a reference state trajectory. Suppose that the discretized and parameterized

reference quantities are provided (see Remark 25). The first-order sensitivity of the system (also known as the state transition matrix) is computed about the reference quantities by solving the following initial value problem for each $k = 1, \ldots, N-1$

$$\dot{\Phi}^k(t, t_k) = \nabla_2 f(t, \bar{x}^k(t), \bar{v}(t))\Phi^k(t, t_k), \quad t \in [t_k, t_{k+1}], \tag{5.65}$$

with initial condition $\Phi^k(t_k, t_k) = I_{n_x}$. Then the state transition matrix for the interval $[t_k, t_{k+1}]$ is denoted by $A_k \triangleq \Phi^k(t_{k+1}, t_k)$.

In the real-world closed-loop execution, suppose that the true state of the spacecraft at time t_k is denoted by $x_k^{\mathrm{t}} \triangleq x_k + \delta x_k$, where the true state deviation δx_k is a random variable. At the initial time, we have $\delta x_1 \sim \mathcal{N}(0_{n_x}, \Sigma_1^{\delta x})$ and δx_1 is referred to as the insertion error. The evolution of the true state deviation in proximity to the nominal state trajectory can be approximated as

$$\delta x_{k+1} = A_k \delta x_k + B_k(\delta u_k + \mu_k), \tag{5.66}$$

where δu_k is an impulsive correction corrupted by noise $\mu_k \sim \mathcal{N}(0_{n_u}, \Sigma^{\mu})$, and $B_k \triangleq A_k {}^{v}E^{\top}$. The selector matrix ${}^{v}E$ selects the velocity coordinates from a vector in $\mathbb{R}^{n_x}$. Note that an actuation noise for the nominal control input can also be considered in with minor modifications to the discussion in this section. The spacecraft measures its state using a relative range measurement with respect to the target spacecraft. The measurement of the true state at time t_k is denoted by $x_k^{\mathrm{m}} = x_k^{\mathrm{t}} + \delta y_k$, where $\delta y_k \sim \mathcal{N}(0_{n_x}, \Sigma_k^{\delta y})$ is the measurement noise. From (5.66), the evolution of the measured state deviation $\delta z_k = \delta x_k + \delta y_k = x_k^{\mathrm{m}} - x_k$ in proximity to the nominal state trajectory is given by

$$\begin{aligned} \delta z_{k+1} &= A_k(\delta z_k - \delta y_k) + B_k(\delta u_k + \mu_k) + \delta y_{k+1}, \\ &= A_k \delta z_k + B_k(\delta u_k + \mu_k) + \delta y_{k+1} - A_k \delta y_k, \end{aligned} \tag{5.67}$$

We assume that the impulsive correction is based on feeding back the measured state deviation, i.e., $\delta u_k = K_k \delta z_k$. The gain K_k is computed analytically to ensure that the

feedback action reduces the position components of the covariance of δz_k, i.e.,

$$K_k \triangleq -({}^{r}E\, B_k)^{-1\,r}E\, A_k. \tag{5.68}$$

Then (5.67) can be consolidated as

$$\begin{aligned}\delta z_{k+1} &= \underbrace{(A_k + B_k K_k)}_{A_k^{\mathrm{cl}}} \delta z_k + \delta y_{k+1} - A_k \delta y_k + B_k \mu_k, \\ &= A_k^{\mathrm{cl}} \delta z_k + \underbrace{[I_{n_x} \; -A_k \; B_k]}_{\Upsilon_k} \begin{bmatrix} \delta y_{k+1} \\ \delta y_k \\ \mu_k \end{bmatrix}. \end{aligned} \tag{5.69}$$

Since δx_1 and δy_1 are independent random variables, we have that $\delta z_1 \sim \mathcal{N}(0_{n_x}, \Sigma_1^{\delta z})$, where $\Sigma_1^{\delta z} = \Sigma_1^{\delta x} + \Sigma_1^{\delta y}$. Furthermore, since the measurement noise at different time instances are uncorrelated, we also have

$$\Upsilon_k \begin{bmatrix} \delta y_{k+1} \\ \delta y_k \\ \mu_k \end{bmatrix} \sim \mathcal{N}(0_{n_x}, \Upsilon_k \Omega_k \Upsilon_k^{\top}) \tag{5.70}$$

where $\Omega_k \triangleq \mathrm{blkdiag}(\Sigma_{k+1}^{\delta y}, \Sigma_k^{\delta y}, \Sigma^{\mu})$. Then the covariance of the measured state deviation evolves as follows

$$\Sigma_{k+1}^{\delta z} = A_k^{\mathrm{cl}} \Sigma_k^{\delta z} A_k^{\mathrm{cl}\top} + \Upsilon_k \Omega_k \Upsilon_k^{\top}, \quad k = 1, \ldots, N-1. \tag{5.71}$$

Recall that the continuous-time covariance of the measured state deviation is required by function $\bar{\Xi}$ (which is embedded within $\tilde{f}$) while numerically integrating the augmented dynamical system over $[t_k, t_{k+1}]$. The following result shows that (5.71) can be equivalently expressed in terms of an integration over an arbitrarily fine grid.

Lemma 13. *For any $k = 1, \ldots, N-1$, consider a grid within $[t_k, t_{k+1}]$ of size M_k, denoted by $t_k = t_k^1 < \ldots < t_k^{M_k} = t_{k+1}$. Next, consider the evolution of the covariance of the measured state*

deviation within $[t_k, t_{k+1}]$

$$\Sigma_k^{\delta z,j+1} = \Phi^k(t_k^{j+1}, t_k^j)\Sigma_k^{\delta z,j}\Phi^k(t_k^{j+1}, t_k^j)^\top, \quad j = 2, \dots, M_k - 1, \tag{5.72a}$$

$$\Sigma_k^{\delta z,2} = \bar{A}_k \Sigma_k^{\delta z,1} \bar{A}_k^\top + \bar{Y}_k \Omega_k \bar{Y}_k^\top, \tag{5.72b}$$

$$\Sigma_k^{\delta z,1} = \Sigma_{k-1}^{\delta z,M_{k-1}}, \tag{5.72c}$$

where

$$\bar{A}_k \triangleq \Phi^k(t_k^2, t_k) + \Phi^k(t_{k+1}, t_k^2)^{-1} B_k K_k, \tag{5.73}$$

$$\bar{Y}_k \triangleq \Phi^k(t_{k+1}, t_k^2)^{-1} Y_k. \tag{5.74}$$

Then for $k = 1, \dots, N-1$, *we have* $\Sigma_{k+1}^{\delta z} = \Sigma_k^{\delta z, M_k}$, *where* $\Sigma_0^{\delta z, M_0} \triangleq \Sigma_1^{\delta z}$.

Proof. From (5.72a) and (5.72b), we have

$$\Sigma_k^{\delta z, M_k} = \Phi^k(t_{k+1}, t_k^2)\Sigma_k^{\delta z,2}\Phi^k(t_{k+1}, t_k^2)^\top, \tag{5.75a}$$

$$= \Phi^k(t_{k+1}, t_k^2)(\bar{A}_k \Sigma_k^{\delta z,1} \bar{A}_k^\top + \bar{Y}_k \Omega_k \bar{Y}_k^\top)\Phi^k(t_{k+1}, t_k^2)^\top, \tag{5.75b}$$

$$= \Phi^k(t_{k+1}, t_k^2)\bar{A}_k \Sigma_k^{\delta z,1} \bar{A}_k^\top \Phi^k(t_{k+1}, t_k^2)^\top + \Phi^k(t_{k+1}, t_k^2)\bar{Y}_k \Omega_k \bar{Y}_k^\top \Phi^k(t_{k+1}, t_k^2)^\top, \tag{5.75c}$$

$$= (A_k + B_k K_k)\Sigma_k^{\delta z,1}(A_k + B_k K_k)^\top + Y_k \Omega_k Y_k^\top. \tag{5.75d}$$

Therefore, $\Sigma_1^{\delta z,1}$ and $\Sigma_k^{\delta z, M_k}$, for $k = 1, \dots, N-1$, satisfy (5.71). □

Note that $\Phi^k(t_{k+1}, t_k^2)^{-1} B_k$ consists of the columns of $\Phi^k(t_k^2, t_k)$ corresponding to velocity, i.e.,

$$\Phi^k(t_{k+1}, t_k^2)^{-1} B_k = \Phi^k(t_{k+1}, t_k^2)^{-1} \underbrace{\Phi^k(t_{k+1}, t_k^2)\Phi^k(t_k^2, t_k)}_{A_k} {}^v E^\top = \Phi^k(t_k^2, t_k) {}^v E^\top$$

where matrix ${}^v E$ selects velocity coordinates from a vector in $\mathbb{R}^{n_x}$. Therefore,

$$\bar{A}_k = \Phi^k(t_k^2, t_k)(I_{n_x} + {}^v E^\top K_k) \tag{5.76a}$$

$$\bar{Y}_k = \Phi^k(t_{k+1}, t_k^2)^{-1}[I_{n_x} \ -A_k \ B_k] = \Phi^k(t_k^2, t_k)[A_k^{-1} \ -I_{n_x} \ {}^v E^\top] \tag{5.76b}$$

To summarize, our goal is to solve (5.64) to obtain a nominal state trajectory and control input that satisfy the chance constraint described in Section (5.3.2.3), imposed with a closed-loop-corrected state covariance discussed above. As a consequence of (5.71) and Lemma 13, we can compute the covariance $\Sigma^{\delta z}(t)$ of the measured state deviation for any $t \in [t_\mathrm{i}, t_\mathrm{f}]$. At each iteration of the proposed solution method, after linearizing the system, we have that $x^\mathrm{m}(t) \sim \mathcal{N}(x(t), \Sigma^{\delta z}(t))$, where x and x^m denote the continuous-time nominal and measured state trajectories, respectively. Then we impose the chance constraint on the nominal state trajectory with respect to the measured state trajectory (as shown in Section 5.3.2.3 with $\xi = x^\mathrm{m}$) and embed the resulting path constraint function $\bar{\Xi}$ into the augmented dynamical system (5.58) via the isoperimetric reformulation.

5.3.3.2 ℓ_1-Penalization and Prox-Linear Method

In the solution method proposed for (5.64), we first subject all nonconvex constraints: (5.64b) and (5.64d) (see Remark 24) to ℓ_1 exact penalization, to obtain new optimization problem with a nonconvex convex-composite objective function and a compact, convex feasible set. We then use an SCP algorithm called the prox-linear method [5] to solve the resulting optimization problem via the approach outlined in [22]. The series of steps involved in solving (5.64) using the SCP-based approach is detailed in [43, Sec. II.C].

5.3.4 Numerical Results

We consider the rendezvous of a spacecraft to the Lunar Gateway space station [197] which is deployed on a long-duration station-keeping trajectory computed for the spacecraft dynamic model described in Appendix E. The major considerations while computing such a long-duration station-keeping trajectory are described in [168, 28, 44]. The motion of the spacecraft (relative to the Gateway) described by (5.31) is in the Gateway-centered J2000 frame, since the motion of the Gateway is described in the Moon-centered J2000 frame (see Appendix E). The key parameters for the rendezvous to the Gateway are described in the Sun LVLH frame. We use rotation matrix R_{LVLH} to transform a vector from Sun LVLH to

the Gateway-centered J2000 frame. The rendezvous maneuver is split into three phases to satisfy three layers of safety constraints, i.e., the size of the avoid set is different for each phase. The solution to each phase is obtained sequentially (by solving separate instances of (5.64)), from Phase 1 to Phase 3. Two consecutive phases are stitched using the terminal position, velocity, and the covariance of the measured state.

The polyhedral avoid set $\mathcal{P} \triangleq \{z \in \mathbb{R}^{n_x} \mid Hz \leq h\}$, defined by

$$H \triangleq \begin{bmatrix} \begin{bmatrix} I_3 \\ -I_3 \end{bmatrix} R_{\text{LVLH}} & 0_{3\times 3} \end{bmatrix}, \quad h \triangleq \frac{1}{2} r_{\text{as}} 1_{n_x},$$

forms a box in the position coordinates, where r_{as} is the length of edge of the box. The boundary conditions constraint function P in (5.53e) is defined as

$$P(x(t_{\text{i}}), x(t_{\text{f}})) \triangleq \begin{bmatrix} x(t_{\text{i}}) - (r_{\text{i}}, v_{\text{i}}) \\ x(t_{\text{f}}) - (r_{\text{f}}, v_{\text{f}}) \end{bmatrix},$$

where r_{i} and v_{i} are the initial position and velocity, respectively, of a phase, and r_{f} and v_{f} are the final position and velocity, respectively, of a phase.

In each phase, we assume that the covariance of the relative range measurement of the spacecraft state (with respect to the Gateway) is provided by a linear interpolation of the prescribed initial and final values, denoted by

$$\Sigma^{\delta y}(\square) \triangleq \text{diag}((\sigma_r^{\delta y}(\square)\, 1_3, \sigma_v^{\delta y}(\square)\, 1_3))^2,$$

where $\square = t_{\text{i}}, t_{\text{f}}$. We are provided with the covariance of the insertion error[3] at the beginning of Phase 1, denoted by

$$\Sigma^{\delta x}(t_{\text{i}}) \triangleq \text{diag}((\sigma_r^{\delta x}(t_{\text{i}})\, 1_3, \sigma_v^{\delta x}(t_{\text{i}})\, 1_3))^2.$$

Then the measured state covariance at the beginning of Phase 1 is given by $\Sigma^{\delta z}(t_{\text{i}}) =$

[3]Insertion error refers to the error in maneuvering the spacecraft to the desired state for commencing Phase 1 of rendezvous.

$\Sigma^{\delta x}(t_i) + \Sigma^{\delta y}(t_i)$. After each iteration of CT-SCVX, the covariance of the measured state, used for chance constraints reformulation (in Section 5.3.2.3) and for feedback synthesis (in Section 5.3.3.1), is recomputed by linearizing the system with respect to the most recent CT-SCVX iterate. The actuation noise covariance (considered in Section 5.3.3.1) for each phase is denoted by $\Sigma^{\mu} \triangleq (\sigma^{\mu}\, 1_3)^2$.

We parameterize the control input as impulses applied at each discretization node (see Appendix C.5). A lower bound constraint on the magnitude of the control input is not imposed for the scenario considered here. The initialization for CT-SCVX is a linear interpolation between the desired initial and final states of each phase. Two layers of scaling are required within CT-SCVX to ensure reliable numerical performance: (i) dimensions of hr, km, and km/hr are chosen for time, position, and velocity, respectively, in the spacecraft dynamic model; and (ii) all constraints and decision variables of (5.64) are scaled so that their numerical values are of similar orders of magnitude. Table 5.2 provides parameter values chosen for each of the three phases ($\star$ denotes a free variable).

Figure 5.15 shows the position trajectory of the spacecraft in the Gateway-centered J2000 frame for the three phases. The avoid set is denoted with red cube, the approach cone with blue surface, the CT-SCVX solution at the discretization nodes with black circles, the resulting continuous-time trajectory with black lines, and the 24-hr free drift trajectory with gray lines. The position coordinates are denoted with r^1, r^2, and r^3. The total fuel consumption for the maneuver is 34.56 m/s, with 19.23 m/s in Phase 1, 14.18 m/s in Phase 2, and 1.14 m/s in Phase 3.

5.3.5 Conclusions

We present a solution method based on CT-SCVXfor passively-safe rendezvous of a spacecraft to the Gateway, which will be deployed on the 9:2 NRHO associated with the Earth-Moon L2 point. We reformulate the continuous-time passive-safety constraint as an isoperimetric constraint and augment it to the high-fidelity dynamic model of the spacecraft (with ephemeris and higher-order effects) to eliminate the commonly-encountered

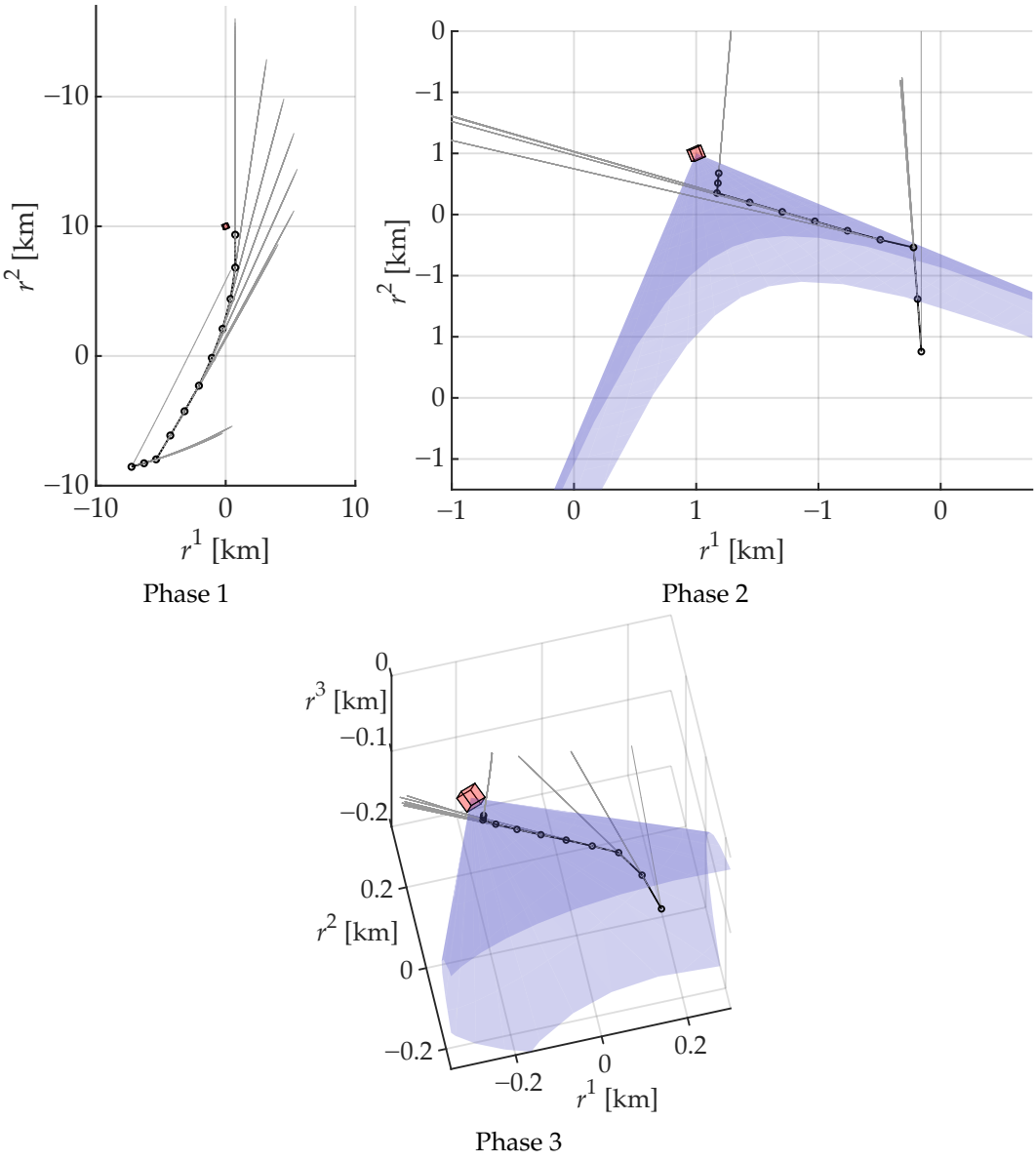


Figure 5.15: Position trajectory for three-phase rendezvous to the Gateway.

Table 5.2: Parameters values chosen for the three phases of rendezvous to Gateway

Parameter	Phase 1	Phase 2	Phase 3
N	11	11	11
$[t_\mathrm{i}, t_\mathrm{f}]$	$[0, 36]$ hr	$[36, 42]$ hr	$[42, 48]$ hr
t_s	24 hr	24 hr	24 hr
v_max	100 km/hr	100 km/hr	10 km/hr
u_max	200 km/hr	200 km/hr	50 km/hr
ϵ	10^{-4}	10^{-4}	10^{-4}
r_as	20 km	2 km	0.4 km
$\sigma_r^{\delta x}(t_\mathrm{i})$, $\sigma_v^{\delta x}(t_\mathrm{i})$	33.333 km, 6 km/hr	NA	NA
$\sigma_r^{\delta y}(t_\mathrm{i})$, $\sigma_v^{\delta y}(t_\mathrm{i})$	6.667 km, 0.252 km/hr	0.085 km, 0.043 km/hr	0.009 km, 0.038 km/hr
$\sigma_r^{\delta y}(t_\mathrm{f})$, $\sigma_v^{\delta y}(t_\mathrm{f})$	0.085 km, 0.043 km/hr	0.009 km, 0.038 km/hr	0.006 km, 0.014 km/hr
β	0.95	0.95	0.95
σ^{μ}	0.276 km/hr	0.237 km/hr	0.013 km/hr
θ_ac	NA	45°	45°
e_ac	NA	$R_{\mathrm{LVLH}}(0, 0, 1)$	$R_{\mathrm{LVLH}}(0, 0, 1)$
r_i	$\frac{1000}{\sqrt{2}} R_{\mathrm{LVLH}}(0, -1, 1)$ km	$\frac{50}{\sqrt{5}} R_{\mathrm{LVLH}}(0, 1, 2)$ km	$\frac{5}{\sqrt{5}} R_{\mathrm{LVLH}}(0, 1, 2)$ km
v_i	$R_{\mathrm{LVLH}}(2.52, 28.44, -21.60)$ km/hr	Phase 1 v_f	Phase 2 v_f
r_f	$\frac{50}{\sqrt{5}} R_{\mathrm{LVLH}}(0, 1, 2)$ km	$\frac{5}{\sqrt{5}} R_{\mathrm{LVLH}}(0, 1, 2)$ km	$\frac{1}{2} R_{\mathrm{LVLH}}(0, 0, 1)$ km
v_f	$\star$	$\star$	0_3 km/hr

inter-sample constraint violation in direct methods for solving optimal control problems. Consequently, computationally expensive mesh-refinement heuristics are not necessary: the proposed approach can generate a high-fidelity solution with coarse discretization grids, which is beneficial for real-time performance. We solve the constraint-reformulated optimal control problem for rendezvous by combining ℓ_1 exact penalization of nonconvexities and a convergence-guaranteed SCP algorithm for convex-composite problems. The solution method is demonstrated using a numerical example based on a three-phase rendezvous to the Gateway.

Planned future work includes the consideration of anisotropy in the statistics of the range and range-rate measurements, and treating the final time of the rendezvous maneuver as a decision variable via time dilation.

Chapter 6

FUTURE DIRECTIONS

This chapter provides a few directions for future research based on the work presented in the preceding chapters.

6.1 Nonconvex Trajectory Optimization

The guarantees within the CT-SCVX framework are dependent on the existence of KKT points, which in turn necessitates certain constraint qualifications. Future work can explore constraint qualifications weaker than LICQ for the exact penalty to hold, with the goal of developing a solution method that directly handles the isoperimetric constraint without relaxation. Furthermore, the reformulated optimal control problem in CT-SCVX exhibits unique structural properties: it is a minimization of a convex function subject to nonlinear equality and linear inequality constraints. This structure, which has gained considerable attention recently [212, 213, 214], opens up opportunities for the development of specialized first-order solvers. Another promising direction is to investigate the link between the use of isoperimetric constraint reformulation in CT-SCVX and the literature on trajectory optimization over manifolds [215, 216, 217].

6.2 Customized Conic Optimization Solver

Several future directions lie ahead for expanding on the capabilities of PIPG. The first involves integrating automatic scaling and preconditioning techniques, like the recently developed QR preconditioner [218], to improve reliability and broaden its applicability as a general-purpose first-order conic optimization tool like SCS [113] and COSMO [219]. Moreover, ensuring reliable numerical performance agnostic to the problem-instance will

necessitate the implementation of commonly used adaptive step-sizes strategies [220], with adjustments based on convergence rates of a desired performance metric.

Second, the development of a tool for automatic customization of PIPG for trajectory optimization, alongside code-generation for embedded systems (such as CVXGEN [17], CASADI [221] and CVXPYGEN [222]), will allow rapid prototyping and extend its use in real-time optimization applications. The in-house-developed code-generation software SCvxGEN (see Section 2.6), which will soon be released, is an effort to this end.

Next, solving nonconvex problems directly with PIPG opens an important new direction, which would eliminate the need for algorithms such as SCP and SQP to solve nonconvex trajectory optimization problems. In the recent years, nonconvex optimization via first order methods has received significant attention [212, 223, 213, 214, 224].

Finally, customization of PIPG for solving SDPs in robust trajectory optimization is a promising avenue [225]. Here, developing a fast solver for funnel synthesis [32, 33, 34] is key, which will further facilitate the simultaneous computation of funnel and trajectory for a variety of constrained nonlinear systems [226].

6.3 Deferred Decision Trajectory Optimization

As it stands, DDTO considers the same path constraints for trajectories to each of the targets. Future work can analyze how different constraints can be imposed on trajectories to different targets without compromising guarantees and computational tractability. Furthermore, the existing framework is designed for open-loop feedforward guidance. A more practical approach, however, is to recursively update the DDTO solution in closed-loop with newly acquired information about the unknowns from perception and sensor measurements. For instance, by embedding a perception-based metric into the optimization problem for DDTO within a recursive setup, the robustness of the trajectories can be improved and the motion of the vehicle can be biased toward regions where the contingencies can be more easily quantified. The recently demonstrated ADAPTIVE-DDTO framework [126] marks a step towards this goal, with potential for further extensions and theoretical analysis.

Appendix A

Gradient of Discretized Dynamics in ct-SCvx

A key step in any gradient-based solution method for (2.22) is to compute the partial derivatives of F_k. We adopt the so-called variational method [98, Sec. 3.2], [97, Sec. 4.2], also referred to as inverse-free exact discretization [42, Sec. 2.3]. To this end, let $\bar{\tilde{X}} = (\bar{\tilde{x}}_1, \ldots, \bar{\tilde{x}}_N)$, $\bar{U}$, and $\bar{S}$ denote a reference solution, and let $\bar{\tilde{v}}$ denote the corresponding parameterization using (2.18). For each $k = 1, \ldots, N-1$, consider the following initial value problem over $[\tau_k, \tau_{k+1}]$

$$\mathring{\Phi}^k_{\tilde{x}}(\tau) = A(\tau)\Phi^k_{\tilde{x}}(\tau), \tag{A.1a}$$
$$\mathring{\Phi}^k_{u}(\tau) = A(\tau)\Phi^k_{u}(\tau) + \Gamma_u(\tau) \otimes B(\tau), \tag{A.1b}$$
$$\mathring{\Phi}^k_{s}(\tau) = A(\tau)\Phi^k_{s}(\tau) + \Gamma_s(\tau) \otimes C(\tau), \tag{A.1c}$$
$$\Phi^k_{\tilde{x}}(\tau_k) = I_{n_{\tilde{x}}}, \tag{A.1d}$$
$$\Phi^k_{u}(\tau_k) = 0_{n_{\tilde{x}} \times n_u N_u}, \tag{A.1e}$$
$$\Phi^k_{s}(\tau_k) = 0_{n_{\tilde{x}} \times N_s}, \tag{A.1f}$$

where $\Phi^k_{\tilde{x}}(\tau) \in \mathbb{R}^{n_{\tilde{x}} \times n_{\tilde{x}}}$, $\Phi^k_{u}(\tau) \in \mathbb{R}^{n_{\tilde{x}} \times n_u N_u}$, and $\Phi^k_{s}(\tau) \in \mathbb{R}^{n_{\tilde{x}} \times N_s}$. The partial derivatives of F evaluated on the augmented state trajectory $\bar{\tilde{x}}^k$ for (2.16b) over $[\tau_k, \tau_{k+1}]$, with augmented control input $\bar{\tilde{v}}$ and initial condition $\bar{\tilde{x}}_k$, are compactly denoted with $A(\tau) \in \mathbb{R}^{n_{\tilde{x}} \times n_{\tilde{x}}}$, $B(\tau) \in \mathbb{R}^{n_{\tilde{x}} \times n_u}$, and $C(\tau) \in \mathbb{R}^{n_{\tilde{x}}}$,

$$A(\tau) = \nabla_{\tilde{x}} F\big(\bar{\tilde{x}}^k(\tau), \bar{\tilde{v}}(\tau)\big), \tag{A.2a}$$
$$B(\tau) = \nabla_{\tilde{u}} F\big(\bar{\tilde{x}}^k(\tau), \bar{\tilde{v}}(\tau)\big)^u E^\top, \tag{A.2b}$$
$$C(\tau) = \nabla_{\tilde{u}} F\big(\bar{\tilde{x}}^k(\tau), \bar{\tilde{v}}(\tau)\big)^s E^\top, \tag{A.2c}$$

for $\tau \in [\tau_k, \tau_{k+1}]$. Then, the partial derivatives of F_k with respect to $\tilde{x}_k$, U, and S (denoted by A_k, B_k, and C_k) evaluated at the reference solution are related to the terminal value of the solution to (A.1) as follows

$$A_k = \Phi_{\tilde{x}}^k(\tau_{k+1}) = -\nabla_{\tilde{x}_k} F_k(\bar{\tilde{x}}_{k+1}, \bar{\tilde{x}}_k, \bar{U}, \bar{S}), \tag{A.3a}$$

$$B_k = \Phi_u^k(\tau_{k+1}) = -\nabla_u F_k(\bar{\tilde{x}}_{k+1}, \bar{\tilde{x}}_k, \bar{U}, \bar{S}), \tag{A.3b}$$

$$C_k = \Phi_s^k(\tau_{k+1}) = -\nabla_s F_k(\bar{\tilde{x}}_{k+1}, \bar{\tilde{x}}_k, \bar{U}, \bar{S}), \tag{A.3c}$$

Finally, the linearization of F_k is given by the mapping

$$(\tilde{x}_{k+1}, \tilde{x}_k, U, S) \mapsto \tilde{x}_{k+1} - \bar{\tilde{x}}^k(\tau_{k+1}) - A_k(\tilde{x}_k - \bar{\tilde{x}}_k) - B_k(U - \bar{U}) - C_k(S - \bar{S}).$$

Note that arbitrarily chosen $\bar{\tilde{X}}$, $\bar{U}$, and $\bar{S}$ may not be continuous-time feasible, i.e., $\bar{\tilde{x}}^k(\tau_{k+1}) \neq \bar{\tilde{x}}_{k+1}$, for $k = 1, \ldots, N-1$. We refer to [3, Fig. 14, 15], [227, Sec. II] and [42, Sec 2.3] for further insights on discretization and linearization of nonlinear dynamics.

Appendix B

PROOF OF THEOREM 1

As a consequence of the relaxation in (2.22c) for a given $\epsilon > 0$, we wish to bound $\hat{g}_i^k(\epsilon)$ and $\hat{h}_j^k(\epsilon)$ on the interval $[t_k, t_{k+1}]$, for any $i = 1, \ldots, n_g$, $j = 1, \ldots, n_h$, and $k = 1, \ldots, N-1$.

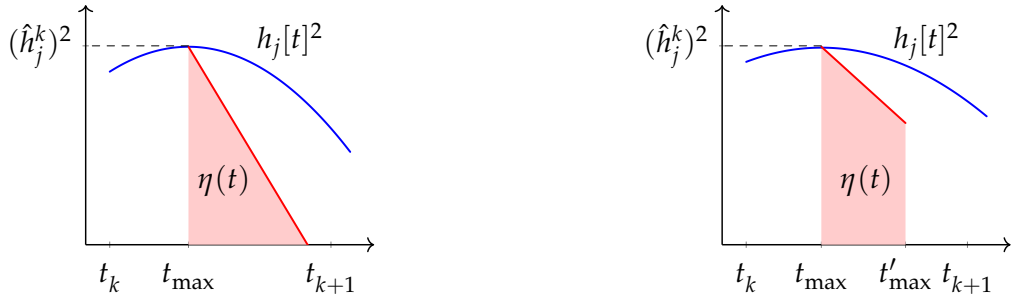
Lemma 14. *Time derivative of path constraint functions g_i and h_j are bounded almost everywhere on the state trajectory x and control input u.*

Proof. Note that $\tilde{v}$ in (2.18) is a piecewise polynomial, which is differentiable almost everywhere in $[0, 1]$. Then due to (2.22d) and Remark 6, $u(t)$ and $s(\tau)$ are bounded for $t \in [t_i, t_f]$, $\tau \in [0, 1]$, and $\dot{u}(t) = \mathring{u}(t(\tau))/s(\tau)$ is well-defined and bounded a.e. $\tau \in [0, 1]$. The state trajectory x is bounded due to Assumption 1. Further, for $k = 1, \ldots, N-1$, Δt_k is bounded. Consequently, f, $\nabla_\square g_i$, and $\nabla_\square h_j$ are bounded, where $\square = t$, x, and u. Then using the chain-rule: $\dot{\square} = \nabla_t \square + \nabla_x \square \dot{x} + \nabla_u \square \dot{u}$, with $\square = g_i$, h_j, $\dot{g}_i$ and $\dot{h}_j$ are bounded almost everywhere. □

We denote the upper bounds on the absolute values of $\dot{g}_i$ and $\dot{h}_j$ (which hold almost everywhere) with ω_{g_i} and ω_{h_j}, respectively. When (2.22c) and the penultimate row of (2.22b) are satisfied, for any $i = 1, \ldots, n_g$, $j = 1, \ldots, n_h$, and $k = 1, \ldots, N-1$, we have that

$$
\begin{aligned}
&\int_{\tau_k}^{\tau_{k+1}} s(\tau)\, 1^\top |g(t(\tau), x(t(\tau)), u(t(\tau)))|_+^2 \\
&\qquad + s(\tau)\, 1^\top h(t(\tau), x(t(\tau)), u(t(\tau)))^2 \,\mathrm{d}\tau \leq \epsilon, \\
\iff &\int_{t_k}^{t_{k+1}} 1^\top |g(t, x(t), u(t))|_+^2 + 1^\top h(t, x(t), u(t))^2 \,\mathrm{d}t \leq \epsilon, \\
\implies &\int_{t_k}^{t_{k+1}} |g_i(t, x(t), u(t))|_+^2 \mathrm{d}t \leq \epsilon, \\
&\int_{t_k}^{t_{k+1}} h_j(t, x(t), u(t))^2 \mathrm{d}t \leq \epsilon,
\end{aligned}
$$

where we use the change of variable: $t(\tau) = t_k + \int_{\tau_k}^{\tau} s(\theta)\mathrm{d}\theta$, for $\tau \in [\tau_k, \tau_{k+1}]$.



(a) $\hat{h}_j^k \leq \Delta t_k \omega_{h_j}$ (b) $\hat{h}_j^k > \Delta t_k \omega_{h_j}$

Figure B.1: Approximation of the area under $h_j[t]^2$.

First, we examine the upper bound for $\hat{h}_j^k(\epsilon)$ (denoted as $\hat{h}_j^k$ for brevity). We denote $h_j(t, x(t), u(t))$ compactly with $h_j[t]$, and assume, without loss of generality, that $\hat{h}_j^k$ is attained at $t_{\max} \in [t_k, t_k + 0.5\Delta t_k]$. Next, we approximate the area under $h_j[t]^2$, for $t \in [t_k, t_{k+1}]$, via an auxiliary function $\eta : [t_{\max}, t_{k+1}] \to \mathbb{R}$, given by

$$\eta(t) = |(\hat{h}_j^k)^2 - 2\omega_{h_j}\hat{h}_j^k(t - t_{\max})|_+.$$

Note that, for $t \in [t_{\max}, t_{k+1}]$, we have

$$\begin{aligned} & h_j[t]^2 - (\hat{h}_j^k)^2 + 2\omega_{h_j}\hat{h}_j^k(t - t_{\max}), \\ \geq\ & h_j[t]^2 - (\hat{h}_j^k)^2 + 2\hat{h}_j^k(\hat{h}_j^k - h_j[t]), \\ =\ & h_j[t]^2 + (\hat{h}_j^k)^2 - 2\hat{h}_j^k h_j[t] = (h_j[t] - \hat{h}_j^k)^2 \geq 0. \end{aligned}$$

Therefore, $h_j[t]^2 - \eta(t)$ is positive for $t > t_{\max}$ and $h_j[t_{\max}]^2 = \eta(t_{\max})$.

If $\hat{h}_j^k \leq \Delta t_k \omega_{h_j}$, we have that $\eta(t'_{\max}) = 0$, where $t'_{\max} = t_{\max} + 0.5\Delta t_k$. The area under $\eta(t)$, for $t \in [t_{\max}, t_{k+1}]$, is triangular (see Figure B.1a). Then

$$\epsilon \geq \int_{t_k}^{t_{k+1}} h_j[t]^2 \mathrm{d}t \geq \int_{t_{\max}}^{t_{k+1}} \eta(t)\mathrm{d}t, \tag{B.1a}$$

$$= \int_{t_{\max}}^{t_{\max} + \frac{\hat{h}_j^k}{2\omega_{h_j}}} (\hat{h}_j^k)^2 - 2\omega_{h_j}\hat{h}_j^k(t - t_{\max})\mathrm{d}t, \tag{B.1b}$$

$$= \int_0^{\frac{\hat{h}_j^k}{2\omega_{h_j}}} (\hat{h}_j^k)^2 - 2\omega_{h_j}\hat{h}_j^k\theta \, \mathrm{d}\theta = \frac{(\hat{h}_j^k)^3}{4\omega_{h_j}}. \tag{B.1c}$$

Hence, from (B.1c), we have that

$$\epsilon \leq \omega_{h_j}^2 \frac{\Delta t_{\min}^3}{4} \implies \hat{h}_j^k(\epsilon) \leq \delta_{h_j}(\epsilon) = (4\epsilon\omega_{h_j})^{\frac{1}{3}}, \tag{B.2}$$

for $j = 1, \ldots, n_h$.

Conversely, if $\hat{h}_j^k > \Delta t_k \omega_{h_j}$, we have that $\eta(t'_{\max}) > 0$. The area under $\eta(t)$ within $[t_{\max}, t'_{\max}]$ is trapezoidal (see Figure B.1b). Then

$$\begin{aligned}
\epsilon &\geq \int_{t_k}^{t_{k+1}} h_j[t]^2 \mathrm{d}t \geq \int_{t_{\max}}^{t_{k+1}} \eta(t) \mathrm{d}t, \\
&> \int_{t_{\max}}^{t_{\max}+\frac{\Delta t_k}{2}} (\hat{h}_j^k)^2 - 2\omega_{h_j}\hat{h}_j^k(t - t_{\max}) \mathrm{d}t, \\
&= \int_0^{\frac{\Delta t_k}{2}} (\hat{h}_j^k)^2 - 2\omega_{h_j}\hat{h}_j^k\theta \, \mathrm{d}\theta, \\
&= (\hat{h}_j^k)^2\frac{\Delta t_k}{2} - \omega_{h_j}\hat{h}_j^k\frac{\Delta t_k^2}{4} > (\hat{h}_j^k)^2\frac{\Delta t_k}{4} > \omega_{h_j}^2\frac{\Delta t_{\min}^3}{4},
\end{aligned}$$

where $\Delta t_{\min} > 0$ is the lower bound for Δt_k; it exists because the dilation factor is positive and $\tau_{k+1} - \tau_k > 0$, for $k = 1, \ldots, N-1$. Note that $\hat{h}_j^k > \Delta t_k \omega_{h_j}$ holds only if $\epsilon > 0.25\omega_{h_j}^2 \Delta t_{\min}^3$ This case can be ignored in practice since we are interested in small, physically insignificant values for ϵ.

We can similarly determine an upper bound for $\hat{g}_i^k(\epsilon)$ by approximating the area under $|g_i(t, x(t), u(t))|_+^2$, for $t \in [t_k, t_{k+1}]$, to obtain

$$\epsilon \leq \omega_{g_i}^2 \frac{\Delta t_{\min}^3}{4} \implies \hat{g}_i^k(\epsilon) \leq \delta_{g_i}(\epsilon) = (4\epsilon\omega_{g_i})^{\frac{1}{3}}, \tag{B.3}$$

for $i = 1, \ldots, n_g$.

Appendix C

Control Input Parameterization

Let $U = (u_1, \ldots, u_{N_u})$ with $u_k \in \mathbb{R}^{n_u}$, for $k = 1, \ldots, N_u$. Then according to (2.18), the control input is given by

$$u(\tau) = \sum_{k=1}^{N_u} \Gamma_u^k(\tau) u_k, \tag{C.1}$$

for $\tau \in [0, 1]$. Note that when time-dilation is not performed, normalized time τ is replaced with actual time t in the following expressions.

C.1 Pseudospectral Methods

In pseudospectral methods, the control input is parameterized with a basis of orthogonal polynomials, i.e.,

$$\Gamma_u^k(\tau) = \prod_{\substack{j=1 \\ j \neq k}}^{N_u} \frac{2\tau - 1 - \eta_j}{\eta_k - \eta_j}, \tag{C.2}$$

is the Lagrange interpolating polynomial [228, Sec. 2.5] of degree $N_u - 1$, where $\eta_k \in [-1, 1]$, for $k = 1, \ldots, N_u$, are related to the roots of orthogonal polynomials (such as members of the Jacobi family [228, Sec. 10.3]). For e.g., choosing η_k to be the roots of the polynomial

$$\eta \mapsto (1 - \eta^2) \frac{\mathrm{d} T_{N_u - 1}(\eta)}{\mathrm{d}\eta},$$

would correspond to Chebyshev-Gauss-Lobatto (CGL) collocation [227, Sec. D.1], where $T_M : [-1, 1] \to \mathbb{R}$ defined by $\eta \mapsto \cos(M \arccos(\eta))$ is the Chebyshev polynomial of

degree M [228, Def. 3.3.1]. The choice of basis functions (C.2) ensures that $\Gamma_u^k\big(\frac{\eta_j+1}{2}\big) = \delta_{jk}$, for each $k, j = 1, \ldots, N_u$. As a result, $u\big(\frac{\eta_k+1}{2}\big) = u_k$, for $k = 1, \ldots, N_u$.

C.2 First-Order-Hold

Choose $N_u = N$ and define the elements of Γ_u as follows

$$\Gamma_u^1(\tau) = \begin{cases} \frac{\tau_2-\tau}{\tau_2-\tau_1} & \text{if } \tau \in [\tau_1, \tau_2], \\ 0 & \text{otherwise,} \end{cases}$$

$$\Gamma_u^k(\tau) = \begin{cases} \frac{\tau-\tau_{k-1}}{\tau_k-\tau_{k-1}} & \text{if } \tau \in [\tau_{k-1}, \tau_k], \\ \frac{\tau_{k+1}-\tau}{\tau_{k+1}-\tau_k} & \text{if } \tau \in [\tau_k, \tau_{k+1}], \quad k = 2, \ldots, N-1, \\ 0 & \text{otherwise,} \end{cases}$$

$$\Gamma_u^N(\tau) = \begin{cases} \frac{\tau-\tau_{N-1}}{\tau_N-\tau_{N-1}} & \text{if } \tau \in [\tau_{N-1}, \tau_N], \\ 0 & \text{otherwise.} \end{cases}$$

Then if $\tau \in [\tau_k, \tau_{k+1}]$, for some $k = 1, \ldots, N-1$, we have

$$u(\tau) = \left(\frac{\tau_{k+1} - \tau}{\tau_{k+1} - \tau_k}\right) u_k + \left(\frac{\tau - \tau_k}{\tau_{k+1} - \tau_k}\right) u_{k+1}.$$

Note that, for $k, j = 1, \ldots, N$, $\Gamma_u^k(\tau_j) = \delta_{jk}$ is the Kronecker delta. Consequently, $u(\tau_k) = u_k$, for $k = 1 \ldots, N$. The coefficients α_k in (5.63) are given by

$$\alpha_1 = \frac{\tau_2 - \tau_1}{2}, \tag{C.3a}$$

$$\alpha_k = \tau_{k+1} - \tau_k, \quad k = 2, \ldots, N-1, \tag{C.3b}$$

$$\alpha_N = \frac{\tau_N - \tau_{N-1}}{2}. \tag{C.3c}$$

C.3 Zero-Order-Hold

Choose $N_u = N - 1$ and define Γ_u as follows

$$\Gamma_u^k(\tau) = \begin{cases} 1 & \text{if } \tau \in [\tau_k, \tau_{k+1}), \\ 0 & \text{otherwise,} \end{cases} \tag{C.4}$$

for $k = 1, \ldots, N-1$. The coefficients α_k in (5.63) are given by $\alpha_k = \tau_{k+1} - \tau_k$, for $k = 1 \ldots, N-1$.

C.4 Finite-Burn Pulse

Choose $N_u = N - 1$ and define Γ_u as follows

$$\Gamma_u^k(\tau) = \begin{cases} 1 & \text{if } \tau \in [\tau_k, t_{\text{burn}}], \\ 0 & \text{otherwise,} \end{cases} \tag{C.5}$$

for $k = 1, \ldots, N-1$, where $t_{\text{burn}} < \tau_{k+1} - \tau_k$. The coefficients α_k in (5.63) are given by $\alpha_k = t_{\text{burn}}$, for $k = 1 \ldots, N-1$.

C.5 Impulse

Choose $N_u = N$ and define Γ_u as follows

$$\Gamma_u^k(\tau) = \delta(\tau - \tau_k), \tag{C.6}$$

for $k = 1 \ldots, N$, where δ is the Dirac delta function. The coefficients α_k in (5.63) are all unity.

Appendix D

Optimal Control Examples

D.1 ct-SCvx

This section describes the optimal control problems considered for the numerical examples in Section 2.6.

D.1.1 Dynamic Obstacle Avoidance

We consider a two-dimensional free-final-time path planning problem. The dynamical system describing the vehicle comprises of position $r \in \mathbb{R}^2$, velocity $v \in \mathbb{R}^2$, cumulative cost $\check{p} \in \mathbb{R}$, and control input $u \in \mathbb{R}^2$. Then $x = (r, v, \check{p})$ is the state, $\tilde{x} = (x, y, t)$ is the augmented state, and $\tilde{u} = (u, s)$ is the augmented control input. We choose a first-order-hold (see Appendix C.2) parameterization for the augmented control input. The augmented system is given by

$$\overset{\circ}{\tilde{x}} = \begin{bmatrix} v \\ u - c_{\mathrm{d}}\|v\|v \\ \|u\|^2 \\ 1^\top |g(t,x,u)|_+^2 \\ 1 \end{bmatrix} s, \tag{D.1}$$

where $c_{\mathrm{d}} \in \mathbb{R}_+$ is the drag coefficient. The vehicle is subject to the following inequality path constraints: avoidance of $n = 10$ moving elliptical obstacles, speed upper bound, and

control input upper- and lower bounds, which are encoded in function g as follows

$$g(t,x,u) = \begin{bmatrix} 1 - \|\check{H}_1(r - \check{q}_1(t))\|^2 \\ \vdots \\ 1 - \|\check{H}_n(r - \check{q}_n(t))\|^2 \\ \|v\|^2 - v_{\max}^2 \\ \|u\|^2 - u_{\max}^2 \\ u_{\min}^2 - \|u\|^2 \end{bmatrix}. \tag{D.2}$$

The shape matrix and center of the ith obstacle at time t are $\check{H}_i$ and $\check{q}_i(t)$, respectively, for $i = 1, \ldots, n$. A sinusoidal motion is prescribed for the centers of the obstacles: $\check{q}_i(t) = (\psi_i + \delta\psi_i \sin(\theta_i t + \angle\psi_i), 0)$, with amplitude $\delta\psi_i$, phase angle $\angle\psi_i$, and frequency θ_i, about the nominal center ψ_i. The speed upper bound and the control input upper- and lower bounds are $v_{\max}$, $u_{\max}$, and $u_{\min}$, respectively. The boundary conditions are specified through function Q as follows

$$Q(t_{\mathrm{i}}, x(t_{\mathrm{i}}), t_{\mathrm{f}}, x(t_{\mathrm{f}})) = \begin{bmatrix} r(t_{\mathrm{i}}) - r_{\mathrm{i}} \\ v(t_{\mathrm{i}}) - v_{\mathrm{i}} \\ \check{p}(t_{\mathrm{i}}) \\ r(t_{\mathrm{f}}) - r_{\mathrm{f}} \\ v(t_{\mathrm{f}}) - v_{\mathrm{f}} \end{bmatrix}, \tag{D.3}$$

where r_{i} and v_{i} are the initial position and velocity, respectively, and r_{f} and v_{f} are the final position and velocity, respectively. We set the terminal state cost function to $L(t_{\mathrm{f}}, x(t_{\mathrm{f}})) = \check{p}(t_{\mathrm{f}})$. Constraint functions h and P are not utilized in this example.

Table D.1 shows the parameter values chosen for the system and the algorithm to generate the results in Section 2.6.1. In addition, We choose the following parameter values for prescribing the motion of the obstacles

$$\psi = \begin{bmatrix} 34 & -32 & 42 & -24 & 34 & -32 & 42 & -24 & 34 & -32 \\ 20 & 20 & 10 & 10 & 0 & 0 & -10 & -10 & -20 & -20 \end{bmatrix},$$

$$\delta\psi = (10)\,1_{1\times n},\ \theta = \left(\frac{\pi}{20}\right)1_{1\times n},$$
$$\angle\psi = \frac{\pi}{2}\begin{bmatrix}1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1\end{bmatrix},$$

where $\square_i$ is the ith column of $\square$, for $\square = \psi, \delta\psi, \theta, \angle\psi$. The scenario with static obstacles in Section 2.6.1 is generated by setting $\delta\psi = 0_{1\times n}$.

Table D.1: Parameter values for dynamic obstacle avoidance example.

Parameter	Value
c_d	$0.01\ \mathrm{m}^{-1}$
$r_\mathrm{i}, r_\mathrm{f}$	$(0,-28), (0,28)$ m
$v_\mathrm{i}, v_\mathrm{f}$	$(0.1,0), (0.1,0)\ \mathrm{m\ s}^{-1}$
v_max	$6\ \mathrm{m\ s}^{-1}$
$u_\mathrm{min}, u_\mathrm{max}$	$0.5, 6\ \mathrm{m\ s}^{-2}$
$H_i, i = 1,\ldots,n$	$\begin{bmatrix} 0 & 0.45 \\ 0.03 & 0 \end{bmatrix}$
t_i	0 s
ϵ	10^{-5}
$\mathcal{U}, \mathcal{S}$	$\{u \in \mathbb{R}^2 \mid \|u\|_\infty \leq u_\mathrm{max}\}$, $[1,60]$
N	10
γ	6.67×10^3
ρ	1.5×10^{-3}

D.1.2 6-DoF Rocket Landing

We consider a free-final-time 6-DoF lunar landing scenario based on the formulation given in [79], and adopt the notation therein. Then $x = (m, \boldsymbol{r}_\mathcal{I}, \boldsymbol{v}_\mathcal{I}, \boldsymbol{q}_{\mathcal{B}\leftarrow\mathcal{I}}, \boldsymbol{\omega}_\mathcal{B}) \in \mathbb{R}^{14}$ is the state, $u = \boldsymbol{T}_\mathcal{B} \in \mathbb{R}^3$ is the control input, $\tilde{x} = (x, y)$ is the augmented state, and $\tilde{u} = (\boldsymbol{T}_\mathcal{B}, s)$ is the augmented control input. We choose a first-order-hold (see Appendix C.2) parameterization for the augmented control input. The augmented dynamical system

is given by

$$\overset{\circ}{\tilde{x}} = \begin{bmatrix} -\breve{\alpha}\|\boldsymbol{T}_{\mathcal{B}}\| \\ \boldsymbol{v}_{\mathcal{I}} \\ \frac{1}{m}C_{\mathcal{I}\leftarrow\mathcal{B}}(\boldsymbol{q}_{\mathcal{B}\leftarrow\mathcal{I}})\boldsymbol{T}_{\mathcal{B}} + \boldsymbol{g}_{\mathcal{I}} \\ \frac{1}{2}\Omega(\boldsymbol{\omega}_{\mathcal{B}})\boldsymbol{q}_{\mathcal{B}\leftarrow\mathcal{I}} \\ J_{\mathcal{B}}^{-1}\left(\boldsymbol{r}_{T,\mathcal{B}} \times \boldsymbol{T}_{\mathcal{B}} - \boldsymbol{\omega}_{\mathcal{B}} \times J_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}}\right) \\ 1^{\top}|g(t,x,u)|_{+} \end{bmatrix} s. \tag{D.4}$$

The path constraint function g is given by

$$g(t,x,u) = \begin{bmatrix} m_{\text{dry}} - m \\ \|\cot\gamma_{\text{gs}} H_{\gamma}\boldsymbol{r}_{\mathcal{I}}\|^2 - (\boldsymbol{e}_1^{\top}\boldsymbol{r}_{\mathcal{I}})^2 \\ -\boldsymbol{e}_1^{\top}\boldsymbol{r}_{\mathcal{I}} \\ \|\boldsymbol{v}_{\mathcal{I}}\|^2 - v_{\max}^2 \\ \|2H_{\theta}\boldsymbol{q}_{\mathcal{B}\leftarrow\mathcal{I}}\|^2 - (\cos\theta_{\max} - 1)^2 \\ \|\boldsymbol{\omega}_{\mathcal{B}}\|^2 - \omega_{\max}^2 \\ \|\cos\delta_{\max}\boldsymbol{T}_{\mathcal{B}}\|^2 - (\boldsymbol{e}_1^{\top}\boldsymbol{T}_{\mathcal{B}})^2 \\ -\boldsymbol{e}_1^{\top}\boldsymbol{T}_{\mathcal{B}} \\ \|\boldsymbol{T}_{\mathcal{B}}\|^2 - T_{\max}^2 \\ T_{\min}^2 - \|\boldsymbol{T}_{\mathcal{B}}\|^2 \end{bmatrix}, \tag{D.5}$$

where the second-order-cone constraints on $\boldsymbol{r}_{\mathcal{I}}$ and $\boldsymbol{T}_{\mathcal{B}}$ are equivalently reformulated to quadratic forms to ensure continuous differentiability [112, Sec. 3.2.4]. The boundary

conditions are specified through function Q as follows

$$Q(t_\mathrm{i}, x(t_\mathrm{i}), t_\mathrm{f}, x(t_\mathrm{f})) = \begin{bmatrix} m(t_\mathrm{i}) - m_\mathrm{wet} \\ \boldsymbol{r}_\mathcal{I}(t_\mathrm{i}) - \boldsymbol{r}_\mathrm{i} \\ \boldsymbol{v}_\mathcal{I}(t_\mathrm{i}) - \boldsymbol{v}_\mathrm{i} \\ \boldsymbol{\omega}_\mathcal{B}(t_\mathrm{i}) \\ \boldsymbol{r}_\mathcal{I}(t_\mathrm{f}) - \boldsymbol{r}_\mathrm{f} \\ \boldsymbol{v}_\mathcal{I}(t_\mathrm{f}) - \boldsymbol{v}_\mathrm{f} \\ \boldsymbol{q}_{\mathcal{B}\leftarrow\mathcal{I}}(t_\mathrm{f}) - \boldsymbol{q}_\mathrm{id} \\ \boldsymbol{\omega}_\mathcal{B}(t_\mathrm{f}) \end{bmatrix}, \tag{D.6}$$

where $\boldsymbol{r}_\mathrm{i}$ and $\boldsymbol{v}_\mathrm{i}$ are the initial position and velocity, respectively, $\boldsymbol{r}_\mathrm{f}$ and $\boldsymbol{v}_\mathrm{f}$ are the final position and velocity, respectively, and $\boldsymbol{q}_\mathrm{id}$ is the unit quaternion. The terminal state cost function is set to $L(t_\mathrm{f}, x(t_\mathrm{f})) = -m(t_\mathrm{f})$. Constraint functions h and P are not utilized in this example. Table D.2 shows the parameter values chosen for the system and the algorithm to generate the results in Section 2.6.2. The definitions of H_γ and H_θ are taken from [79].

D.1.3 3-DoF Rocket Landing

We consider a convex, fixed-final-time Mars landing scenario based on the formulation in [51]. Then $x = (r, v, z) \in \mathbb{R}^7$ is the state and $u = (\check{T}, \check{\sigma}) \in \mathbb{R}^4$ is the control input. The vehicle mass is given by $m = \exp z$, the thrust vector by $T = m\check{T}$, and $\check{\sigma}$ is an auxiliary control input (slack variable) introduced as a part of the lossless convexification procedure. The system dynamics is given by

$$\dot{x} = \begin{bmatrix} v \\ \check{T} + (0, 0, -g_\mathrm{m}) \\ -\check{\alpha}\check{\sigma} \end{bmatrix} \tag{D.7}$$

Table D.2: Parameter values for 6-DoF rocket landing example.

Parameter	Value
$\breve{\alpha}$	4.53×10^{-4} s m^{-1}
$\boldsymbol{g}_{\mathcal{I}}$	$(-1.61, 0, 0)$ m s^{-2}
$\boldsymbol{r}_{T,\mathcal{B}}$	$(-0.25, 0, 0)$ m
$J_{\mathcal{B}}$	$\mathrm{diag}(19150, 13600, 13600)$ kg m^2
m_{dry}, m_{wet}	2100, 3250 kg
$\boldsymbol{r}_{\mathrm{i}}$, $\boldsymbol{r}_{\mathrm{f}}$	$(433, 0, 250)$, $(10, 0, -30)$ m
$\boldsymbol{v}_{\mathrm{i}}$, $\boldsymbol{v}_{\mathrm{f}}$	$(10, 0, -30)$, $(-1, 0, 0)$ m s^{-1}
γ_{gs}	85°
$v_{\max}$	50 m s^{-1}
$\theta_{\max}$	60°
$\omega_{\max}$	10° s^{-1}
$\delta_{\max}$	45°
t_{i}	0 s
$T_{\min}$, $T_{\max}$	5000, 22000 N
ϵ	10^{-4}
$\mathcal{U}$, $\mathcal{S}$	$\{u \in \mathbb{R}^3 \mid \|u\|_\infty \leq T_{\max}\}$, $[1, 60]$
N	5
γ	2×10^3
ρ	2.5×10^{-3}

$$= \underbrace{\begin{bmatrix} 0_{3\times3} & I_3 & 0_{3\times1} \\ & 0_{3\times7} & \\ & 0_{1\times7} & \end{bmatrix}}_{A} x + \underbrace{\begin{bmatrix} & 0_{3\times4} & \\ I_3 & & 0_{3\times1} \\ 0_{1\times3} & & -\breve{\alpha} \end{bmatrix}}_{B} u + \underbrace{\begin{bmatrix} 0_{5\times1} \\ -g_{\mathrm{m}} \\ 0 \end{bmatrix}}_{w},$$

where g_{m} is the acceleration due to gravity on Mars, and $\breve{\alpha}$ determines the mass depletion rate. We choose a zero-order-hold parameterization for the control input.

The path constraint function g is given by

$$g(t, x, u) = \tag{D.8}$$

$$
\begin{bmatrix}
\| \cot \gamma_{\mathrm{gs}} E_{12} r \|^2 - (e_3^\top r)^2 \\
-(e_3^\top r) \\
\|v\|^2 - v_{\max}^2 \\
z - z_1(t) \\
-z + \max\{\log m_{\mathrm{dry}}, z_0(t)\} \\
-e_3^\top \check{T} + \check{\sigma} \cos\theta_{\max} \\
\|\check{T}\|^2 - \check{\sigma}^2 \\
-\check{\sigma} \\
\mu_{\min}(t) \left(1 - (z - z_0(t)) + \frac{1}{2}(z - z_0(t))^2\right) - \check{\sigma} \\
-\mu_{\max}(t)(1 - (z - z_0(t))) + \check{\sigma}
\end{bmatrix},
$$

where $E_{12} = [I_2 \; 0_{2\times 1}]$, $e_3 = (0, 0, 1)$,

$$
\begin{aligned}
z_0(t) &= \log(m_{\mathrm{wet}} - \check{\alpha} T_{\max} t), \\
z_1(t) &= \log(m_{\mathrm{wet}} - \check{\alpha} T_{\min} t), \\
\mu_{\min}(t) &= T_{\min} \exp(-z_0(t)), \\
\mu_{\max}(t) &= T_{\max} \exp(-z_0(t)),
\end{aligned}
$$

Lossless convexification holds if $\|T(t)\| = \sigma(t) = m(t)\check{\sigma}(t)$ a.e. $t \in [t_{\mathrm{i}}, t_{\mathrm{f}}]$. The boundary conditions are specified through function Q as follows

$$
Q(t_{\mathrm{i}}, x(t_{\mathrm{i}}), t_{\mathrm{f}}, x(t_{\mathrm{f}})) = \begin{bmatrix}
r(t_{\mathrm{i}}) - r_{\mathrm{i}} \\
v(t_{\mathrm{i}}) - v_{\mathrm{i}} \\
z(t_{\mathrm{i}}) - \log m_{\mathrm{wet}} \\
r(t_{\mathrm{f}}) - r_{\mathrm{f}} \\
v(t_{\mathrm{f}}) - v_{\mathrm{f}}
\end{bmatrix}, \tag{D.9}
$$

where r_{i} and v_{i} are the initial position and velocity, respectively, and r_{f} and v_{f} are the final position and velocity, respectively. The terminal state cost function is set to $L(t_{\mathrm{f}}, x(t_{\mathrm{f}})) = -z(t_{\mathrm{f}})$. The running cost function Y, and constraint functions h and P, are not utilized in this example.

Table D.3 shows the parameter values chosen for the system and the algorithm to generate the results in Section 2.6.3.

Table D.3: Parameter values for 3-DoF rocket landing example.

Parameter	Value
$\check{\alpha}$	4.53×10^{-4} s m^{-1}
g_{m}	3.71 m s^{-2}
r_{i}, r_{f}	$(2000, 0, 1500)$, $(0, 0, 0)$ m
v_{i}, v_{f}	$(80, 30, -75)$, $(0, 0, 0)$ m s^{-1}
m_{dry}, m_{wet}	1505, 1905 kg
γ_{gs}	84°
$v_{\max}$	139 m s^{-1}
$\theta_{\max}$	40°
$T_{\min}$, $T_{\max}$	4971.6, 13258 N
$[t_{\mathrm{i}}, t_{\mathrm{f}}]$	$[0, 84]$ s
ϵ	10^{-5}
$\mathcal{U}$	$\{u \in \mathbb{R}^4 \mid \|u\|_\infty \leq T_{\max}\}$
N	8
γ	6.67×10^3
ρ	1.5×10^{-3}

D.2 DDTO

The following sections describe a discrete-time convex and a continuous-time nonconvex optimal control problem based on quadrotor motion planning, which are used for the numerical results in Section 4.6.

D.2.1 Discrete-Time Convex Problem

We consider a discrete-time model of a point-mass aerial vehicle where the state $x_k = (r_k, v_k)$ consists of three-dimensional position $r_k \in \mathbb{R}^3$, velocity $v_k \in \mathbb{R}^3$, and the control input $u_k \in \mathbb{R}^3$ is an acceleration. The dynamical system is given by

$$x_{k+1} = \underbrace{\begin{bmatrix} I_3 & \Delta t I_3 \\ 0_{3\times 3} & I_3 \end{bmatrix}}_{A} x_k + \underbrace{\begin{bmatrix} \frac{\Delta t^2}{2} I_3 \\ \Delta t I_3 \end{bmatrix}}_{B} u_k + \underbrace{\begin{bmatrix} \frac{\Delta t^2}{2} a \\ \Delta t a \end{bmatrix}}_{c}, \tag{D.10}$$

where the vehicle has unit mass, $a \in \mathbb{R}^3$ is the acceleration due to gravity, Δt is the sampling time. The control input is subject to the following path constraints

$$\|u_k\| \leq u_{\max}, \tag{D.11a}$$

$$\|u_k\| \leq \sec \delta_{\max} \hat{e}^\top u_k, \tag{D.11b}$$

$$\hat{e}^\top u_k \geq u_{\min}, \tag{D.11c}$$

where $u_{\max}$ and $u_{\min}$ are the upper and lower bounds, respectively, on the control input magnitude, $\delta_{\max}$ is the maximum angle between the control input and the pointing vector $\hat{e}$. Note that in the presence of the pointing constraint (D.11b), a conservative convex approximation for the (nonconvex) lower bound constraint on the control input magnitude is provided by (D.11c). For each $j \in J$, the targets are singleton sets denoted by $\mathcal{Z}^j = \{z^j\}$. The stage cost function l in cumulative constraint (4.18d) is given by

$$l(x_k, u_k) \triangleq \|u_k\|^2. \tag{D.12}$$

Table D.4 shows the parameter values chosen for the system and Algorithm 4 to generate the results in Figure 4.5.

Table D.4: Parameter values for discrete-time convex optimal control problem.

Parameter	Value
Δt	0.5 s
a	$(0, 0, -9.806)$ m/s^2
n	4
N^j for $j \in [1\!:\!n]$	40
$u_{\max}, u_{\min}$	$20, 8$ m/s^2
$\hat{e}, \delta_{\max}$	$(0, 0, 1), 60^\circ$
$l_{\max}$	3794 m^2/s^4
z^0	$(0, 0, 30, 0, 0, 0)$ m, m/s
z^1	$(39.5, -6.25, 0, 0, 0, 0)$ m, m/s
z^2	$(39.5, 6.25, 0, 0, 0, 0)$ m, m/s
z^3	$(28.3, 28.3, 0, 0, 0, 0)$ m, m/s
z^4	$(40, 0, 0, 0, 0, 0)$ m, m/s
$\lambda^1, \lambda^2, \lambda^3, \lambda^4$	$1, 2, 3, 4$

D.2.2 Continuous-Time Nonconvex Problem

We consider a continuous-time model of a point-mass aerial vehicle where the state $x(t) = (r(t), v(t), \theta(t))$ consists of three-dimensional position $r(t) \in \mathbb{R}^3$, velocity $v(t) \in \mathbb{R}^3$, and cumulative trajectory cost $\theta(t) \in \mathbb{R}$, and the control input $u(t) \in \mathbb{R}^3$ is an acceleration. The dynamical system is given by

$$\dot{x}(t) = \begin{bmatrix} v(t) \\ u(t) - c_{\mathrm{d}}\|v(t)\|v(t) + a \\ \|u(t)\|^2 \end{bmatrix} = F(x(t), u(t)), \tag{D.13}$$

where the vehicle has unit mass, $a \in \mathbb{R}^3$ is the acceleration due to gravity, and c_{d} is the drag coefficient. The path constraints on the vehicle are defined by $g : \mathbb{R}^7 \times \mathbb{R}^3 \to \mathbb{R}^7$ as follows

$$g(x(t), u(t)) = \begin{bmatrix} -\|H^1_{\mathrm{obs}}(r(t) - q^1_{\mathrm{obs}})\|^2 + 1 \\ -\|H^2_{\mathrm{obs}}(r(t) - q^2_{\mathrm{obs}})\|^2 + 1 \\ \|v(t)\|^2 - v^2_{\max} \\ \|u(t)\|^2 - u^2_{\max} \\ -\|u(t)\|^2 + u^2_{\min} \\ \|u(t)\|^2 - (\sec \delta_{\max} \hat{e}^\top u(t))^2 \\ -\hat{e}^\top u(t) \end{bmatrix}. \tag{D.14}$$

where H^i_{obs} and q^i_{obs}, for $i = 1, 2$, are the shape matrices and centers, respectively, of ellipsoidal obstacles, $v_{\max}$ is the upper bound on speed, $u_{\max}$ and $u_{\min}$ are the upper and lower bounds, respectively, on the control input magnitude, and $\delta_{\max}$ is the maximum angle between the control input and the pointing vector $\hat{e}$. Note that g_i, for $i \in [1\!:\!n_g]$ with $n_g = 7$, in (4.21) denote the scalar-valued components of g. The second-order-cone constraint is equivalently reformulated to a quadratic form to ensure continuous differentiability [112, Sec. 3.2.4]. The boundary condition constraint functions P^j and Q^j

are defined as

$$P^j(x(t_\mathrm{f})) \triangleq \theta(t_\mathrm{f}) - l_\mathrm{max}, \tag{D.15a}$$

$$Q^j(x(t_\mathrm{f})) \triangleq (r(t_\mathrm{f}), v(t_\mathrm{f})) - z^j, \tag{D.15b}$$

where $z^j \in \mathbb{R}^6$ specify the target position and velocity, for $j \in J$, and $x(t_\mathrm{f})$ is the terminal state of a trajectory.

Table D.5 shows the parameter values chosen for the system and Algorithm 5 to generate the results in Figure 4.6.

Table D.5: Parameter values for continuous-time nonconvex optimal control problem.

Parameter	Value
a	$(0, 0, -9.806)$ m/s^2
c_d	0.01 1/m
n, N	$4, 23$
v_max	8 m/s
$u_\mathrm{max}, u_\mathrm{min}$	$20, 5$ m/s^2
$\hat{e}, \delta_\mathrm{max}$	$(0, 0, 1), 60^\circ$
l_max	1100 m^2/s^4
$H^1_\mathrm{obs}, H^2_\mathrm{obs}$	$\mathrm{diag}(0.2, 0.1, 0.2), \mathrm{diag}(0.1, 0.2, 0.2)$
$q^1_\mathrm{obs}, q^2_\mathrm{obs}$	$(-5, 1, 10), (-10, 20, 10)$ m
z^0	$(10, -10, 10, 0, 0, 0, 0)$ m, m/s, m^2/s^4
z^1	$(10, 30, 10, 1, 0, 0)$ m, m/s
z^2	$(-10, 35, 10, 0, 1, 0)$ m, m/s
z^3	$(-30, 15, 10, 0, 0, 0)$ m, m/s
z^4	$(-15, -15, 10, 0, 1, 0)$ m, m/s
$\lambda_1, \lambda_2, \lambda_3, \lambda_4$	$1, 2, 3, 4$
$\tilde{\mathcal{U}}^j$	$\{u \in \mathbb{R}^3 \mid \|u\|_\infty \leq u_\mathrm{max}\} \times [1, 15]$
ϵ	10^{-5}

Appendix E

Equations of Motion in Cislunar Space

The spacecraft equations of motion (5.30) in the cislunar space are represented in the Earth-centered inertial frame (ECI) defined by Earth's Mean Equator and Mean Equinox (MEME) at 12:00 Terrestrial Time on January 1, 2000, with the origin at the instantaneous center of the Moon (see [210, Sec. 3.7] for further details). This frame is labelled as `J2000` in the NAIF SPICE toolkit [229], which is a widely used for ephemeris calculations. The spacecraft state evolves according to the following nonlinear ordinary differential equation

$$
\begin{aligned}
\dot{r}_{\mathrm{sc}}(t) &= v_{\mathrm{sc}}(t), \qquad\qquad (E.1)\\
\dot{v}_{\mathrm{sc}}(t) &= -GM_{\mathrm{M}}\frac{r_{\mathrm{sc}}(t)}{\|r_{\mathrm{sc}}(t)\|_2^3} + GM_{\mathrm{E}}\left(\frac{r_{\mathrm{E}}(t)-r_{\mathrm{sc}}(t)}{\|r_{\mathrm{E}}(t)-r_{\mathrm{sc}}(t)\|_2^3} - \frac{r_{\mathrm{E}}(t)}{\|r_{\mathrm{E}}(t)\|_2^3}\right)\\
&\quad + GM_{\mathrm{S}}\left(\frac{r_{\mathrm{S}}(t)-r_{\mathrm{sc}}(t)}{\|r_{\mathrm{S}}(t)-r_{\mathrm{sc}}(t)\|_2^3} - \frac{r_{\mathrm{S}}(t)}{\|r_{\mathrm{S}}(t)\|_2^3}\right) - \frac{k_{\mathrm{sc}}A_{\mathrm{sc}}S_0r_0^2}{M_{\mathrm{sc}}c}\left(\frac{r_{\mathrm{S}}(t)-r_{\mathrm{sc}}(t)}{\|r_{\mathrm{S}}(t)-r_{\mathrm{sc}}(t)\|_2^3}\right)\\
&\quad + \frac{3}{2}GM_{\mathrm{M}}M_{\mathrm{J2}}R_{\mathrm{M}}^2\frac{r_{\mathrm{sc}}(t)}{\|r_{\mathrm{sc}}(t)\|_2^5}\left(3\sin^2\left(\arccos\left(\frac{r_{\mathrm{E}}(t)^\top \bar{r}_{\mathrm{sc}}(t)}{\|r_{\mathrm{E}}(t)\|_2\|\bar{r}_{\mathrm{sc}}(t)\|_2}\right)+\theta_{\mathrm{eq}}\right)-1\right),
\end{aligned}
$$

where

$$
\begin{aligned}
\bar{r}_{\mathrm{sc}} &= r_{\mathrm{sc}} - \frac{r_{\mathrm{sc}}^\top \bar{v}_{\mathrm{E}}}{\|\bar{v}_{\mathrm{E}}\|_2^2}\bar{v}_{\mathrm{E}},\\
\bar{v}_{\mathrm{E}} &= -r_{\mathrm{E}}\times(r_{\mathrm{E}}\times v_{\mathrm{E}}) = -r_{\mathrm{E}}^\top(v_{\mathrm{E}}\times r_{\mathrm{E}}).
\end{aligned}
$$

Note that $z \triangleq (r_{\mathrm{sc}}, v_{\mathrm{sc}})$ and the right-hand-side of (E.1) defines function $\check{f}$ in (5.30). We use the DE 421 ephemeris [189] within SPICE to query the position and velocity of Earth and Sun with respect to Moon. Vectors $r_{\mathrm{E}}, \bar{v}_{\mathrm{E}}$ and $r_{\mathrm{E}}\times v_{\mathrm{E}}$ form a right-handed orthogonal set. The projection of spacecraft position vector onto the plane formed by r_{E} and $r_{\mathrm{E}}\times v_{\mathrm{E}}$ is

denoted by $\bar{r}_{sc}$. The angle between $\bar{r}_{sc}$ and r_E, denoted by λ_{sc}, quantifies the Moon latitude closest to the spacecraft. The cannonball model of solar radiation pressure assumed in (E.1) represents the spacecraft as a sphere. As a result, the cross-sectional area A_{sc} experiencing solar radiation is independent of spacecraft orientation. Figure E.1 and Table E.1 describe the quantities appearing in (E.1).

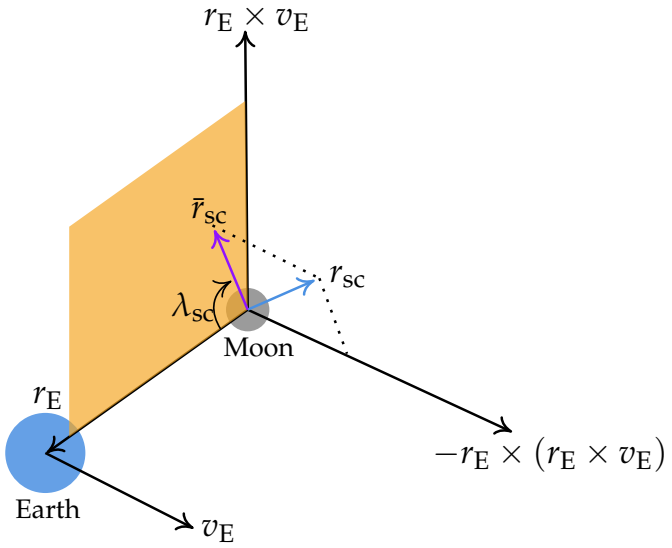


Figure E.1: Illustration of the vectors defining the spacecraft equations of motion in the cislunar space.

Table E.1: Parameters in the spacecraft equations of motion

r_{sc}	Position of spacecraft with respect to Moon
v_{sc}	Velocity of spacecraft with respect to Moon
r_E	Position of Earth with respect to Moon
v_E	Velocity of Earth with respect to Moon
r_S	Position of Sun with respect to Moon
k_{sc}	Reflectivity of spacecraft body
r_0	1 AU
A_{sc}	Cross-sectional area of spacecraft
S_0	Solar flux at distance r_0 from Sun
c	Speed of light in vacuum
G	Universal gravitational constant
M_{sc}	Mass of spacecraft
M_E	Mass of Earth
M_M	Mass of Moon
M_S	Mass of Sun
M_{J2}	J2 zonal harmonic coefficient for Moon, 2.024×10^{-4}
R_M	Radius of Moon, 1737.1 km
θ_{eq}	Equatorial inclination of Moon, 6.68°

Appendix F

SIGNED DISTANCE

The signed-distance [92, Chap. IV Sec. 1.3] of $z \in \mathbb{R}^n$ with respect to a nonempty convex set $\mathcal{D} \subset \mathbb{R}^n$, denoted by $d_{\mathcal{D}}(z)$, is given by

$$d_{\mathcal{D}}(z) = \inf_{y \in \mathcal{D}} \|z - y\| - \inf_{x \notin \mathcal{D}^c} \|z - x\|. \tag{F.1}$$

The gradient of $d_{\mathcal{D}}$ evaluated at z, denoted by $\nabla d_{\mathcal{D}}(z)$, is given by

$$\nabla d_{\mathcal{D}}(z) = \frac{(z - z^{\partial\mathcal{D}})^\top}{d_{\mathcal{D}}(z)}, \tag{F.2}$$

where $z^{\partial\mathcal{D}} = \operatorname*{argmin}_{y \in \partial\mathcal{D}} \|z - y\|$ is the projection of z onto the boundary of $\mathcal{D}$, denoted by $\partial\mathcal{D}$.

Consider a polytope of the form: $\mathcal{D} = \{z \in \mathbb{R}^n \mid Hz \leq h\}$, where $H = [H_1 \ \dots \ H_m]^\top$ and $h = (h_1, \dots, h_m)$, with $H_i \in \mathbb{R}^n$, $h_i \in \mathbb{R}$, for $i = 1, \dots, m$. Then

$$d_{\mathcal{D}}(z) = \begin{cases} \min\limits_{y \in \mathcal{D}} \|z - y\| & \text{if } z \notin \mathcal{D} \\ \min\limits_{1 \leq i \leq m} \frac{|H_i^\top z - h_i|}{\|H_i\|} & \text{otherwise} \end{cases} \tag{F.3}$$

where the first case amounts to solving an inequality-constrained QP and the second case is a simple algebraic enumeration spanning at most all the faces of the polytope. Further, the projection of z onto the boundary of $\mathcal{D}$ is given by

$$z^{\partial\mathcal{D}} = \begin{cases} \operatorname*{argmin}\limits_{y \in \mathcal{D}} \|z - y\| & \text{if } z \notin \mathcal{D} \\ z - \frac{(H_{i^\star}^\top z - h_{i^\star})}{\|H_{i^\star}\|^2} H_{i^\star} & \text{otherwise} \end{cases} \tag{F.4}$$

where $i^\star = \underset{1 \le i \le m}{\operatorname{argmin}} |H_i^\top z - h_i| / \|H_i\|$.

Appendix G

PARAMETERIZED SETS FOR SAFETY

Suppose that the system model in (5.31) also includes process noise, i.e., function f takes a process noise vector from $\mathbb{R}^{n_w}$ as an additional argument. Let $C([a,b],\mathbb{R}^n)$ denote the space of piecewise continuous functions that map $[a,b]$ to $\mathbb{R}^n$.

Given a state trajectory x over $[t_\mathrm{i}, t_\mathrm{f}]$ for the dynamical system, along with a safety horizon $[0, t_\mathrm{s}]$, we can consider more general safety constraints

$$x(t) \notin \mathcal{B}_\tau^t, \quad \forall\, \tau \in [0, t_\mathrm{s}],\ t \in [t_\mathrm{i}, t_\mathrm{f}], \tag{G.1}$$

where $\mathcal{B}_\tau^t$ are parameterized compact sets. Given an avoid set $\mathcal{P} \subset \mathbb{R}^{n_x}$, and compact sets $\mathcal{U} \subset \mathbb{R}^{n_u}$ and $\mathcal{W} \subset \mathbb{R}^{n_w}$, the following are some examples of parameterized set-based constraints.

- Backward reachable set (BRS):

$$\mathcal{B}_\tau^t = \left\{ z \in \mathbb{R}^{n_x} \;\middle|\; \begin{array}{l} \dfrac{\mathrm{d}x(\gamma)}{\mathrm{d}\gamma} = f(\gamma, x(\gamma), 0_{n_u}, 0_{n_w}),\ \gamma \in [t, t+\tau] \\ x(t) = z,\ x(t+\tau) \in \mathcal{P} \end{array} \right\} \tag{G.2}$$

- Robust BRS (with off-nominal actuation from $\mathcal{U}$):

$$\mathcal{B}_\tau^t = \left\{ z \in \mathbb{R}^{n_x} \;\middle|\; \begin{array}{l} \forall u \in C([t, t+\tau], \mathcal{U}) \\ \dfrac{\mathrm{d}x(\gamma)}{\mathrm{d}\gamma} = f(\gamma, x(\gamma), u(\gamma), 0_{n_w}),\ \gamma \in [t, t+\tau] \\ x(t) = z,\ x(t+\tau) \in \mathcal{P} \end{array} \right\} \tag{G.3}$$

- Robust BRS (with process noise from $\mathcal{W}$):

$$\mathcal{B}_\tau^t = \left\{ z \in \mathbb{R}^{n_x} \;\middle|\; \begin{array}{l} \exists w \in C([t, t+\tau], \mathcal{W}) \\ \dfrac{\mathrm{d}x(\gamma)}{\mathrm{d}\gamma} = f(\gamma, x(\gamma), 0_{n_u}, w(\gamma)),\ \gamma \in [t, t+\tau] \\ x(t) = z,\ x(t+\tau) \in \mathcal{P} \end{array} \right\} \tag{G.4}$$

- Robust BRS (with process noise from $\mathcal{W}$ and off-nominal actuation from $\mathcal{U}$):

$$\mathcal{B}_\tau^t = \left\{ z \in \mathbb{R}^{n_x} \;\middle|\; \begin{array}{l} \exists w \in C([t, t+\tau], \mathcal{W}),\ \forall u \in C([t, t+\tau], \mathcal{U}) \\ \dfrac{\mathrm{d}x(\gamma)}{\mathrm{d}\gamma} = f(\gamma, x(\gamma), 0_{n_u}, w(\gamma)),\ \gamma \in [t, t+\tau] \\ x(t) = z,\ x(t+\tau) \in \mathcal{P} \end{array} \right\} \tag{G.5}$$

The set (G.2) is the same as the one described in Section 5.3.2.2, the set (G.3) considers the failure scenario where all off-nominal actuation from $\mathcal{U}$ causes the spacecraft to enter $\mathcal{P}$, the set (G.4) considers the failure scenario where at least one process noise realization from $\mathcal{W}$ causes the spacecraft to enter $\mathcal{P}$, and set (G.5) considers the failure scenario where there exists at least one process noise realization from $\mathcal{W}$ such that all off-nominal actuation from $\mathcal{U}$ cause the spacecraft to enter $\mathcal{P}$. Here, off-nominal actuation refers to a damaged actuation system which does not provide the expected (nominal) performance.

When the dynamics function f is affine, control input u and process noise w are subject to zero-order-hold or impulse parameterization, and the avoid set $\mathcal{P}$ is a polytope, we can efficiently compute $\mathcal{B}_\tau^t$. Otherwise, these sets are intractable to compute.

Appendix H

UNIFORM-GRID DISCRETIZATION FOR FREE-FINAL-TIME PROBLEMS

Consider a discrete-time dynamical system given by

$$x_{k+1} = f_k(x_k, u_k), \quad k \geq 1. \tag{H.1}$$

We consider a trajectory and a control input sequence defined over the grid: $0 = t_1 < \ldots < t_N = t_\mathrm{f}$. Suppose that the underlying continuous-time dynamical system is

$$\dot{x}(t) = F(x(t), u(t)), \quad t \in [0, t_\mathrm{f}] \tag{H.2}$$

and assume that the continuous-time control input u is subject to a zero-order-hold parameterization. Then for each $k \in [1\!:\!N-1]$, we have the following

$$f_k(x_k, u_k) \triangleq x_k + \int_{t_k}^{t_{k+1}} F({}^k x(t), u_k) \mathrm{d}t, \tag{H.3}$$

where ${}^k x$ is a trajectory for (H.2) over $[t_k, t_{k+1}]$ with constant control input u_k and initial condition x_k. Note that subscript "k" in f_k accounts for the possibility that grid t_k, for $k \in [1\!:\!N]$, is nonuniformly-spaced.

Next we apply time-dilation, which considers a new independent variable $\tau \in [0, 1]$ by mapping $[0, 1]$ to $[0, t_\mathrm{f}]$ using a strictly increasing, piecewise linear function t. For each $k \in [1\!:\!N-1]$, define dilation factor as follows

$$s_k \triangleq \frac{\mathrm{d}t(\tau)}{\mathrm{d}\tau}, \quad \tau \in [\tau_k, \tau_{k+1}], \tag{H.4}$$

where $0 = \tau_1 < \ldots < \tau_N = 1$ is a uniformly-spaced grid. We treat the dilation factor as an additional control input, i.e., $\tilde{u}_k \triangleq (u_k, s_k)$ for each $k \in [1\!:\!N-1]$, to obtain a new dynamical with an augmented control input

$$\begin{aligned}\tilde{f}(x_k, \tilde{u}_k) &\triangleq x_k + \int_{\tau_k}^{\tau_{k+1}} s_k F({}^k x(t(\tau)), u_k) \mathrm{d}\tau, \\ &= f_k(x_k, u_k) = x_{k+1}.\end{aligned}$$

We constrain s to be positive and bounded (for strict monotonicity and physically meaningful values of t). While all trajectories of length N for the new dynamical system are defined over $[0, 1]$, they can each correspond to a different final time. Suppose that a collection of trajectories with same initial state but different target states are required to be identical for as long as possible, then each of those trajectories can have different final time in spite of being the same length. The dilation factors of all trajectories will be the same in the identical portion of trajectory; beyond that, they may differ (since the dilation factor serves a role similar to the original control inputs).

BIBLIOGRAPHY

[1] NASA. Image of the Week - Week 149. https://mars.nasa.gov/mars2020/multimedia/raw-images/image-of-the-week/week-149, December 2023. Accessed: 2024-06-25.

[2] David Crandall, Michael Hamilton, Mihir Rimjha, and Brandon Robinette. Noise Assessment for Amazon Prime Air MK27-2 Flight Test and Durability & Reliability Operations at the Pendleton Unmanned Aircraft Systems Range, Oregon. [Online] Accessed: 07-14-2024, September 2022. HMMH Report No. 313090.002 002-1.

[3] Danylo Malyuta, Taylor P. Reynolds, Michael Szmuk, Thomas Lew, Riccardo Bonalli, Marco Pavone, and Behçet Açıkmeşe. Convex Optimization for Trajectory Generation: A Tutorial on Generating Dynamically Feasible Trajectories Reliably and Efficiently. *IEEE Control Systems Magazine*, 42(5):40–113, 2022. doi:10.1109/MCS.2022.3187542.

[4] Ajay Sathya, Pantelis Sopasakis, Ruben Van Parys, Andreas Themelis, Goele Pipeleers, and Panagiotis Patrinos. Embedded nonlinear model predictive control for obstacle avoidance using PANOC. In *2018 European Control Conference (ECC)*, pages 1523–1528, 2018. doi:10.23919/ECC.2018.8550253.

[5] Dmitriy Drusvyatskiy and Adrian Lewis. Error bounds, quadratic growth, and linear convergence of proximal methods. *Mathematics of Operations Research*, 43(3):919–948, 2018. doi:10.1287/moor.2017.0889.

[6] Juan L. Jerez, Paul J. Goulart, Stefan Richter, George A. Constantinides, Eric C. Kerrigan, and Manfred Morari. Embedded online optimization for model predictive control at megahertz rates. *IEEE Transactions on Automatic Control*, 59(12):3238–3251, 2014. doi:10.1109/TAC.2014.2351991.

[7] Antonin Chambolle and Thomas Pock. On the ergodic convergence rates of a first-order primal-dual algorithm. *Mathematical Programming*, 159(1-2):253–287, 2016. doi:10.1007/s10107-015-0957-3.

[8] Jonathan Eckstein. Parallel alternating direction multiplier decomposition of convex programs. *Journal of Optimization Theory and Applications*, 80(1):39–62, 1994. doi:10.1007/BF02196592.

[9] Yue Yu, Purnanand Elango, Ufuk Topcu, and Behçet Açıkmeşe. Proportional–integral projected gradient method for conic optimization. *Automatica*, 142:110359, August 2022. doi:10.1016/j.automatica.2022.110359.

[10] Alexander Domahidi, Eric Chu, and Stephen Boyd. Ecos: An socp solver for embedded systems. In *2013 European Control Conference (ECC)*. IEEE, July 2013. doi:10.23919/ecc.2013.6669541.

[11] MOSEK ApS. *The MOSEK optimization toolbox for MATLAB manual. Version 10.1.*, 2024. URL: http://docs.mosek.com/latest/toolbox/index.html.

[12] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, March 2006. doi:10.1007/s10107-004-0559-y.

[13] Richard H. Byrd, Jorge Nocedal, and Richard A. Waltz. Knitro: An integrated package for nonlinear optimization. In *Nonconvex Optimization and Its Applications*, pages 35–59. Springer US, 2006. doi:10.1007/0-387-30065-1_4.

[14] Johan Lofberg. YALMIP: a toolbox for modeling and optimization in MATLAB. In *2004 IEEE International Conference on Robotics and Automation*, CACSD-04. IEEE, 2004. doi:10.1109/cacsd.2004.1393890.

[15] Michael Grant and Stephen P. Boyd. CVX: MATLAB software for disciplined convex programming, version 2.1, 2014. URL: https://cvxr.com/cvx/doc/.

[16] Steven Diamond and Stephen P. Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[17] Jacob Mattingley and Stephen P. Boyd. CVXGEN: a code generator for embedded convex optimization. *Optimization and engineering*, 13(1):1–27, March 2012. doi:10.1007/s11081-011-9176-9.

[18] Nikolai Matni, Aaron D. Ames, and John C. Doyle. A quantitative framework for layered multirate control: Toward a theory of control architecture. *IEEE Control Systems Magazine*, 44(3):52–94, June 2024. doi:10.1109/mcs.2024.3382388.

[19] Rachid Alami, Raja Chatila, Sara Fleury, Malik Ghallab, and Félix Ingrand. An architecture for autonomy. *The International Journal of Robotics Research*, 17(4):315–337, 1998. doi:10.1177/027836499801700402.

[20] Danylo Malyuta, Yue Yu, Purnanand Elango, and Behçet Açıkmeşe. Advances in trajectory optimization for space vehicle control. *Annual Reviews in Control*, 52:282–315, 2021. doi:10.1016/j.arcontrol.2021.04.013.

[21] Philip E. Gill, Walter Murray, and Michael A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Review*, August 2006. doi:10.1137/S0036144504446096.

[22] Purnanand Elango, Dayou Luo, Abhinav G. Kamath, Samet Uzun, Taewan Kim, and Behçet Açıkmeşe. Successive convexification for trajectory optimization with continuous-time constraint satisfaction, 2024. (submitted to Automatica). arXiv:2404.16826.

[23] Purnanand Elango, Abhinav G. Kamath, Yue Yu, John M. Carson III, Mehran Mesbahi, and Behçet Açıkmeşe. A customized first-order solver for real-time powered-descent guidance. In *AIAA SciTech 2022 Forum*. AIAA, January 2022. doi:10.2514/6.2022-0951.

[24] Yue Yu, Purnanand Elango, Behçet Açıkmeşe, and Ufuk Topcu. Extrapolated proportional-integral projected gradient method for conic optimization. *IEEE Control Systems Letters*, 7:73–78, 2023. doi:10.1109/lcsys.2022.3186647.

[25] Abhinav G. Kamath, Purnanand Elango, Taewan Kim, Skye Mceowen, Yue Yu, John M. Carson III, Mehran Mesbahi, and Behçet Açıkmeşe. Customized real-time first-order methods for onboard dual quaternion-based 6-DoF powered-descent guidance. In *AIAA SciTech 2023 Forum*. AIAA, January 2023. doi:10.2514/6.2023-2003.

[26] Purnanand Elango, Selahattin Burak Sarsılmaz, and Behçet Açıkmeşe. Deferring Decision in Multi-target Trajectory Optimization. In *AIAA SciTech 2022 Forum*, San Diego, CA, 2022. American Institute of Aeronautics and Astronautics. doi:10.2514/6.2022-1583.

[27] Purnanand Elango, Selahattin Burak Sarsılmaz, and Behçet Açıkmeşe. Deferred decision trajectory optimization. *Automatica*, 2024. (to be submitted).

[28] Clark P. Newman, Diane C. Davis, Ryan J. Whitley, Joseph R. Guinn, and Mark S. Ryne. Stationkeeping, orbit determination, and attitude control for spacecraft in near rectilinear halo orbits. In *AAS Astrodynamics Specialist Conference*, pages 1–20, August 2018.

[29] Diane C. Davis, Sean M. Phillips, Kathleen C. Howell, Srianish Vutukuri, and Brian P. McCarthy. Stationkeeping and transfer trajectory design for spacecraft in cislunar space. In *AAS/AIAA Astrodynamics Specialist Conference*, pages 1–20, 2017.

[30] Vivek Muralidharan, Avishai Weiss, and Uros V. Kalabic. Control strategy for long-term station-keeping on near-rectilinear halo orbits. In *AIAA Scitech 2020 Forum*, 2020. arXiv: https://arc.aiaa.org/doi/pdf/10.2514/6.2020-1459, doi:10.2514/6.2020-1459.

[31] Ryo Nakamura, Junji Kikuchi, Takahiro Sasaki, Yuki Matsumoto, Moeko Hidaka, Naomi Murakami, Satoshi Ueda, and Naoki Satoh. Rendezvous trajectory design of logistics resupply missions to the lunar gateway in near-rectilinear halo orbit. *Journal of Space Safety Engineering*, 10(2):144–154, June 2023. doi:10.1016/j.jsse.2023.03.001.

[32] Taylor P. Reynolds, Danylo Malyuta, Mehran Mesbahi, and Behçet Açıkmeşe. Temporally-interpolated funnel synthesis for nonlinear systems. In *2nd RSS Workshop on Robust Autonomy: Tools for Safety in Real-World Uncertain Environments (RSS 2020)*, 2020.

[33] Taylor Reynolds, Danylo Malyuta, Mehran Mesbahi, Behçet Açıkmeşe, and John M. Carson III. Funnel synthesis for the 6-DOF powered descent guidance problem. In *AIAA Scitech 2021 Forum*, Reston, Virginia, January 2021. American Institute of Aeronautics and Astronautics. doi:10.2514/6.2021-0504.

[34] Taewan Kim, Purnanand Elango, Taylor P. Reynolds, Behçet Açıkmeşe, and Mehran Mesbahi. Optimization-based constrained funnel synthesis for systems with lipschitz nonlinearities via numerical optimal control. *IEEE Control Systems Letters*, 7:2875–2880, 2023. doi:10.1109/lcsys.2023.3290229.

[35] Anirudha Majumdar and Russ Tedrake. Funnel libraries for real-time robust feedback motion planning. *The International Journal of Robotics Research*, 36(8):947–982, 2017. doi:10.1177/0278364917712421.

[36] James Anderson, John C Doyle, Steven H Low, and Nikolai Matni. System level synthesis. *Annual Reviews in Control*, 47:364–393, 2019. doi:10.1016/j.arcontrol.2019.03.006.

[37] Michael Neunert, Cédric De Crousaz, Fadri Furrer, Mina Kamel, Farbod Farshidian, Roland Siegwart, and Jonas Buchli. Fast nonlinear model predictive control for unified trajectory optimization and tracking. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1398–1404. IEEE, 2016. doi:10.1109/ICRA.2016.7487274.

[38] Chao Shen, Yang Shi, and Brad Buckham. Integrated path planning and tracking control of an auv: A unified receding horizon optimization approach. *IEEE/ASME Transactions on Mechatronics*, 22(3):1163–1173, 2016.

[39] Sylvia L. Herbert, Mo Chen, SooJean Han, Somil Bansal, Jaime F. Fisac, and Claire J Tomlin. FaSTrack: a modular framework for fast and guaranteed safe motion planning. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 1517–1522. IEEE, 2017. doi:10.1109/CDC.2017.8263867.

[40] Venkata Ramana Makkapati, Mehregan Dor, and Panagiotis Tsiotras. Trajectory desensitization in optimal control problems. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 2478–2483. IEEE, 2018.

[41] Hans Seywald and Kevin L Seywald. Desensitized optimal control. *Journal of Guidance, Control, and Dynamics*, pages 1–14, 2024. doi:10.2514/1.G008284.

[42] Abhinav G. Kamath, Purnanand Elango, Yue Yu, Skye Mceowen, Govind M. Chari, John M. Carson, III, and Behçet Açıkmeşe. Real-time sequential conic optimization for multi-phase rocket landing guidance. *IFAC-PapersOnLine*, 56(2):3118–3125, 2023. doi:10.1016/j.ifacol.2023.10.1444.

[43] Govind M. Chari, Abhinav G. Kamath, Purnanand Elango, and Behçet Açıkmeşe. Fast monte carlo analysis for 6-DoF powered-descent guidance via GPU-accelerated sequential convex programming. In *AIAA SciTech 2024 Forum*. AIAA, January 2024. doi:10.2514/6.2024-1762.

[44] Purnanand Elango, Stefano Di Cairano, Uroš Kalabić, and Avishai Weiss. Local eigenmotion control for near rectilinear halo orbits. In *2022 American Control Conference (ACC)*, pages 1822–1827, 2022. doi:10.23919/ACC53348.2022.9867672.

[45] Purnanand Elango, Stefano Di Cairano, Karl Berntorp, and Avishai Weiss. Sequential linearization-based station keeping with optical navigation for NRHO. In *AAS/AIAA Astrodynamics Specialist Conference 2022*, North Carolina, 2022.

[46] Purnanand Elango, Behçet Açıkmeşe, Abraham P. Vinod, Stefano Di Cairano, and Avishai Weiss. Successive convexification for spacecraft rendezvous on near rectilinear halo orbit with continuous-time safety. *Journal of Guidance, Control, and Dynamics*, 2024. (to be submitted).

[47] Panagiotis Tsiotras and Mehran Mesbahi. Toward an algorithmic control theory. *Journal of Guidance, Control, and Dynamics*, 40(2):194–196, 2017. doi:10.2514/1.G002754.

[48] John T. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. Advances in Design and Control. SIAM, 2nd edition, January 2010. doi:10.1137/1.9780898718577.

[49] Daniel Liberzon. *Calculus of Variations and Optimal Control Theory: A Concise Introduction*. Princeton University Press, December 2011.

[50] Donggun Lee, Shankar A. Deka, and Claire J. Tomlin. Convexifying state-constrained optimal control problems. *IEEE Transactions on Automatic Control*, pages 1–8, 2022. doi:10.1109/TAC.2022.3221704.

[51] Behçet Açıkmeşe, John M. Carson III, and Lars Blackmore. Lossless convexification of nonconvex control bound and pointing constraints of the soft landing optimal control problem. *IEEE Transactions on Control Systems Technology*, 21(6):2104–2113, November 2013. doi:10.1109/tcst.2012.2237346.

[52] Daniel Dueri, Yuanqi Mao, Zohaib Mian, Jerry Ding, and Behçet Açıkmeşe. Trajectory optimization with inter-sample obstacle avoidance via successive convexification. In *2017 IEEE 56th Conference on Decision and Control (CDC)*, pages 1150–1156. IEEE, December 2017. doi:10.1109/cdc.2017.8263811.

[53] Anil V. Rao. A survey of numerical methods for optimal control. *Advances in the Astronautical Sciences*, 135(1):497–528, 2009.

[54] John Schulman, Yan Duan, Jonathan Ho, Alex Lee, Ibrahim Awwal, Henry Bradlow, Jia Pan, Sachin Patil, Ken Goldberg, and Pieter Abbeel. Motion planning with sequential

convex optimization and convex collision checking. *International Journal of Robotics Research*, 33(9):1251–1270, August 2014. doi:10.1177/0278364914528132.

[55] Taylor A. Howell, Kevin Tracy, Simon Le Cleac'h, and Zachary Manchester. CALIPSO: A differentiable solver for trajectory optimization with conic and complementarity constraints. In *Springer Proceedings in Advanced Robotics*, pages 504–521. Springer Nature Switzerland, Cham, 2023. doi:10.1007/978-3-031-25555-7_34.

[56] Behçet Açıkmeşe, Daniel Scharf, Fred Hadaegh, and Emmanuell Murray. A convex guidance algorithm for formation reconfiguration. In *AIAA Guidance Navigation and Control Conference*, Reston, Virigina, August 2006. AIAA. doi:10.2514/6.2006-6070.

[57] Arthur Richards and Oliver Turnbull. Inter-sample avoidance in trajectory optimizers using mixed-integer linear programming. *International Journal of Robust and Nonlinear Control*, 25(4):521–526, 2015. doi:10.1002/rnc.3101.

[58] Changliu Liu, Chung-Yen Lin, and Masayoshi Tomizuka. The convex feasible set algorithm for real time optimization in motion planning. *SIAM Journal of Control and Optimization*, 56(4):2712–2733, January 2018. doi:10.1137/16M1091460.

[59] Vincent Roulet, Siddhartha Srinivasa, Dmitriy Drusvyatskiy, and Zaid Harchaoui. Iterative linearized control: Stable algorithms and complexity guarantees. In *Proceeding of the 36th International Conference on Machine Learning*, volume 97, pages 5518–5527. PMLR, 2019. URL: https://proceedings.mlr.press/v97/roulet19a.html.

[60] Xinfu Liu and Ping Lu. Solving nonconvex optimal control problems by convex optimization. *Journal of Guidance, Control, and Dynamics*, 37(3):750–765, May 2014. doi:10.2514/1.62110.

[61] Tobia Marcucci, Mark Petersen, David von Wrangel, and Russ Tedrake. Motion planning around obstacles with convex optimization. *Science Robotics*, 8(84), November 2023. doi:10.1126/scirobotics.adf7843.

[62] L. S. Jennings, M. E. Fisher, Kok Lay Teo, and C. J. Goh. MISER3: Solving optimal control problems–an update. *Advances in Engineering Software and Workstation*, 13(4):190–196, July 1991. doi:10.1016/0961-3552(91)90016-w.

[63] Oskar von Stryk. Numerical solution of optimal control problems by direct collocation. In *Optimal Control*, pages 129–143. Birkhäuser Basel, 1993. doi:10.1007/978-3-0348-7539-4_10.

[64] Per E. Rutquist and Marcus M. Edvall. *PROPT-MATLAB optimal control software*. TOMLAB Optimization, April 2010.

[65] Michael A. Patterson and Anil V. Rao. 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*, 41(1):1–37, October 2014. doi:10.1145/2558904.

[66] Robin Verschueren, Gianluca Frison, Dimitris Kouzoupis, Jonathan Frey, Niels van Duijkeren, Andrea Zanelli, Branimir Novoselnik, Thivaharan Albin, Rien Quirynen, and Moritz Diehl. Acados-a modular open-source framework for fast embedded optimal control. *Mathematical programming computation*, 14(1):147–183, March 2022. doi:10.1007/s12532-021-00208-8.

[67] Boris Houska, Hans Joachim Ferreau, and Moritz Diehl. ACADO toolkit–an open-source framework for automatic control and dynamic optimization. *Optimal Control Applications and Methods*, 32(3):298–312, May 2011. doi:10.1002/oca.939.

[68] Florian Messerer, Katrin Baumgärtner, and Moritz Diehl. Survey of sequential convex programming and generalized gauss-newton methods. *ESAIM: Proceedings and Surveys*, 71:64–88, August 2021. doi:10.1051/proc/202171107.

[69] Moritz Diehl and Sebastien Gros. Numerical optimal control. https://www.syscop.de/files/2020ss/NOC/book-NOCSE.pdf, 2024. Accessed: 04-24-2024.

[70] Yuanqi Mao, Michael Szmuk, Xiangru Xu, and Behçet Açıkmeşe. Successive convexification: A superlinearly convergent algorithm for non-convex optimal control problems, 2019. arXiv:1804.06539.

[71] Riccardo Bonalli, Thomas Lew, and Marco Pavone. Analysis of theoretical and numerical properties of sequential convex programming for continuous-time optimal control. *IEEE Transactions on Automatic Control*, pages 1–16, 2022. doi:10.1109/TAC.2022.3207865.

[72] Lei Xie, Xiang Zhou, Hong-Bo Zhang, and Guo-Jian Tang. Descent property in sequential second-order cone programming for nonlinear trajectory optimization. *Journal of Guidance, Control, and Dynamics*, pages 1–16, October 2023. doi:10.2514/1.g007494.

[73] Dayou Luo, Purnanand Elango, and Behçet Açıkmeşe. Remarks on "Successive Convexification: A Superlinearly Convergent Algorithm for Non-convex Optimal Control Problems", 2024. arXiv:2403.00733.

[74] Kenshiro Oguri. Successive convexification with feasibility guarantee via augmented lagrangian for non-convex optimal control problems, 2024. arXiv:2304.14564.

[75] Florian Messerer and Moritz Diehl. Determining the exact local convergence rate of sequential convex programming. In *2020 European Control Conference (ECC)*, pages 1280–1285. IEEE, May 2020. doi:10.23919/ecc51009.2020.9143749.

[76] Adrian Lewis and Stephen Wright. A proximal method for composite minimization. *Mathematical Programming*, 158(1-2):501–546, July 2016. doi:10.1007/s10107-015-0943-9.

[77] Coralia Cartis, Nicholas I. M. Gould, and Philippe L. Toint. On the evaluation complexity of composite function minimization with applications to nonconvex nonlinear programming. *SIAM Journal on Optimization*, 21(4):1721–1739, October 2011. doi:10.1137/11082381X.

[78] Zhenbo Wang and Michael J. Grant. Constrained trajectory optimization for planetary entry via sequential convex programming. *Journal of Guidance, Control, and Dynamics*, 40(10):2603–2615, October 2017. URL: http://doi.org/10.2514/1.G002150, doi:10.2514/1.g002150.

[79] Michael Szmuk, Taylor P. Reynolds, and Behçet Açıkmeşe. Successive convexification for real-time six-degree-of-freedom powered descent guidance with state-triggered constraints. *Journal of Guidance, Control, and Dynamics*, 43(8):1399–1413, August 2020. doi:10.2514/1.G004549.

[80] Marco Sagliano. Pseudospectral convex optimization for powered descent and landing. *Journal of Guidance, Control, and Dynamics*, 41(2):320–334, February 2018. URL: http://doi.org/10.2514/1.G002818, doi:10.2514/1.g002818.

[81] Hans Georg Bock and Karl-Josef Plitt. A multiple shooting algorithm for direct solution of optimal control problems. *IFAC Proceedings Volumes*, 17(2):1603–1608, 1984. doi:10.1016/S1474-6670(17)61205-9.

[82] Robert McGill. Optimum control, inequality state constraints, and the generalized newton-raphson algorithm. *Journal of the Society for Industrial and Applied Mathematics*, 3(2):291–298, January 1965. doi:10.1137/0303021.

[83] R. W. H. Sargent and G. R. Sullivan. The development of an efficient optimal control package. In *Proceedings of the 8th IFIP Conference on Optimization Techniques, Würzburg, September 5–9, 1977*, volume 7, pages 158–168, Berlin/Heidelberg, 1977. Springer-Verlag. doi:10.1007/bfb0006520.

[84] Kok Lay Teo and C. Goh. A simple computational procedure for optimization problems with functional inequality constraints. *IEEE Transactions on Automatic Control*, 32(10):940–941, October 1987. doi:10.1109/tac.1987.1104471.

[85] R. C. Loxton, K. L. Teo, V. Rehbock, and K. F. C. Yiu. Optimal control problems with a continuous inequality constraint on the state and the control. *Automatica*, 45(10):2250–2257, October 2009. doi:10.1016/j.automatica.2009.05.029.

[86] Qun Lin, Ryan Loxton, Kok Lay Teo, Yong Hong Wu, and Changjun Yu. A new exact penalty method for semi-infinite programming problems. *Journal of Computational and Applied Mathematics*, 261:271–286, May 2014. doi:10.1016/j.cam.2013.11.010.

[87] Yuanbo Nie and Eric C. Kerrigan. Solving dynamic optimization problems to a specified accuracy: An alternating approach using integrated residuals. *IEEE Transactions on Automatic Control*, 68(1):548–555, 2023. doi:10.1109/TAC.2022.3144131.

[88] Dmitriy Drusvyatskiy and Courtney Paquette. Efficiency of minimizing compositions of convex functions and smooth maps. *Mathematical Programming*, 178:503–558, 2019. doi:10.1007/s10107-018-1311-3.

[89] Samet Uzun, Purnanand Elango, Abhinav G. Kamath, Taewan Kim, and Behçet Açıkmeşe. Successive convexification for nonlinear model predictive control with continuous-time constraint satisfaction. In *8th IFAC Conference on Nonlinear Model Predictive Control*, Kyoto, Japan, August 2024. arXiv:2405.00061.

[90] Taewan Kim, Abhinav G. Kamath, Niyousha Rahimi, Jasper Corleis, Behçet Açıkmeşe, and Mehran Mesbahi. Six-degree-of-freedom aircraft landing trajectory planning with runway alignment, 2024. arXiv:2405.16680.

[91] Stephen P. Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, March 2004. doi:10.1017/CBO9780511804441.

[92] Jean-Baptiste Hiriart-Urruty and Claude Lemarechal. *Convex Analysis and Minimization Algorithms I: Fundametals*. Grundlehren der mathematischen Wissenschaften. Springer, Berlin, Germany, 1993. doi:10.1007/978-3-662-02796-7.

[93] Leonard D. Berkovitz. *Optimal Control Theory*. Applied Mathematical Sciences. Springer, New York, NY, February 1974. doi:10.1007/978-1-4757-6097-2.

[94] Francis H. Clarke, Yuri S. Ledyaev, Ronald J. Stern, and Peter R. Wolenski. *Nonsmooth Analysis and Control Theory*, volume 178 of *Graduate Texts in Mathematics*. Springer Verlag, New York, 1998. doi:10.1007/b97650.

[95] Gerald B. Folland. *Real Analysis: Modern Techniques and their Applications*, volume 40. John Wiley & Sons, 1999.

[96] Richard F Hartl, Suresh P Sethi, and Raymond G Vickson. A survey of the maximum principles for optimal control problems with state constraints. *SIAM Review*, 37(2):181–218, June 1995. doi:10.1137/1037043.

[97] R. Quirynen, M. Vukov, M. Zanon, and M. Diehl. Autogenerating microsecond solvers for nonlinear MPC: A tutorial using ACADO integrators. *Optimal Control Applications and Methods*, 36(5):685–704, September 2015. doi:10.1002/oca.2152.

[98] Qun Lin, Ryan Loxton, and Kok Lay Teo. The control parameterization method for nonlinear optimal control: A survey. *Journal of Industrial and Management Optimization*, 10(1):275–309, 2014. doi:10.3934/jimo.2014.10.275.

[99] John T. Betts. Survey of numerical methods for trajectory optimization. *Journal of Guidance, Control, and Dynamics*, 21(2):193–207, 1998. doi:10.2514/2.4231.

[100] Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer, 2nd edition, July 2006. doi:10.1007/978-0-387-40065-5.

[101] S. P. Han and Olvi L. Mangasarian. Exact penalty functions in nonlinear programming. *Mathematical Programming*, 17:251–269, 1979. doi:10.1007/BF01588250.

[102] Fernando A. C. C. Fontes and Luis Tiago Paiva. Guaranteed constraint satisfaction in continuous-time control problems. *IEEE Control Systems Letters*, 3(1):13–18, 2019. doi:10.1109/LCSYS.2018.2849853.

[103] Jun Fu and Fangyin Tian. Dynamic optimization of nonlinear systems with guaranteed feasibility of inequality-path-constraints. *Automatica*, 127(109516):109516, May 2021. doi:10.1016/j.automatica.2021.109516.

[104] G. Di Pillo and L. Grippo. On the exactness of a class of nondifferentiable penalty functions. *Journal of Optimization Theory and Applications*, 57(3):399–410, 1988. doi:10.1007/BF02346160.

[105] Francis Clarke. *Functional Analysis, Calculus of Variations and Optimal Control*, volume 264. Springer, 2013. doi:10.1007/978-1-4471-4820-3.

[106] Amir Beck. *First-Order Methods in Optimization*. SIAM, 2017. doi:10.1137/1.9781611974997.

[107] G. Di Pillo and L. Grippo. Exact penalty functions in constrained optimization. *SIAM Journal of Control and Optimization*, 27(6):1333–1360, 1989. doi:10.1137/0327068.

[108] Taylor P. Reynolds and Mehran Mesbahi. The crawling phenomenon in sequential convex programming. In *2020 American Control Conference (ACC)*, pages 3613–3618, July 2020. doi:10.23919/ACC45564.2020.9147550.

[109] Taylor P. Reynolds, Michael Szmuk, Danylo Malyuta, Mehran Mesbahi, Behçet Açıkmeşe, and John M. Carson III. Dual quaternion-based powered descent guidance with state-triggered constraints. *Journal of Guidance, Control, and Dynamics*, 43(9):1584–1599, September 2020. doi:10.2514/1.G004536.

[110] Taylor Reynolds, Danylo Malyuta, Mehran Mesbahi, Behçet Açıkmeşe, and John M. Carson III. A real-time algorithm for non-convex powered descent guidance. In *AIAA SciTech 2020 Forum*. AIAA, January 2020. doi:10.2514/6.2020-0844.

[111] Michael Szmuk, Carlo Alberto Pascucci, and Behçet Açıkmeşe. Real-time quad-rotor path planning for mobile obstacle avoidance using convex optimization. In *2018 IEEE/RSJ In-*

ternational Conference on Intelligent Robots and Systems (IROS). IEEE, October 2018. doi:10.1109/iros.2018.8594351.

[112] MOSEK ApS. *MOSEK Modeling Cookbook 3.3.0*, 2024. Accessed: 2024-06-28. URL: https://docs.mosek.com/modeling-cookbook/index.html.

[113] Brendan O'Donoghue, Eric Chu, Neal Parikh, and Stephen P. Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, 169(3):1042–1068, June 2016. doi:10.1007/s10957-016-0892-3.

[114] Yue Yu, Purnanand Elango, and Behçet Açıkmeşe. Proportional-integral projected gradient method for model predictive control. *IEEE Control Systems Letters*, 5(6):2174–2179, December 2021. doi:10.1109/LCSYS.2020.3044977.

[115] Heinz H. Bauschke and Patrick L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, volume 408. Springer, 2017.

[116] Behçet Açıkmeşe, John M. Carson III, and Lars Blackmore. Lossless convexification of non-convex control bound and pointing constraints of the soft landing optimal control problem. *IEEE Transactions on Control Systems Technology*, 21(6):2104–2113, 2013.

[117] Weijie Su, Stephen P. Boyd, and Emmanuel J. Candes. A differential equation for modeling nesterov's accelerated gradient method: Theory and insights. *J. Mach. Learn. Res.*, 17(1):5312–5354, 2016.

[118] Yangyang Xu. Accelerated first-order primal-dual proximal methods for linearly constrained composite convex programming. *SIAM Journal on Optimization*, 27(3):1459–1484, 2017. doi:10.1137/16M1082305.

[119] Yang Wang and Stephen Boyd. Fast model predictive control using online optimization. *IEEE Trans. Control Syst. Technol.*, 18(2):267–278, 2009.

[120] Bartholomew Stellato, Goran Banjac, Paul Goulart, Alberto Bemporad, and Stephen Boyd. OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, 2020. doi:10.1007/s12532-020-00179-2.

[121] Brendan O'Donoghue. Operator splitting for a homogeneous embedding of the linear complementarity problem. *SIAM Journal on Optimization*, 31(3):1999–2023, 2021. doi:10.1137/20M136630.

[122] Quanyan Zhu and Tamer Başar. Disentangling resilience from robustness: Contextual dualism, interactionism, and game-theoretic paradigms. *IEEE Control Systems*, 44(3):95–103, June 2024. doi:10.1109/mcs.2024.3382415.

[123] Mo Chen and Claire J. Tomlin. Hamilton–jacobi reachability: Some recent theoretical advances and applications in unmanned airspace management. *Annual Review of Control, Robotics, and Autonomous Systems*, 1(1):333–358, May 2018. doi:10.1146/annurev-control-060117-104941.

[124] David L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006. doi:10.1109/TIT.2006.871582.

[125] Elaine Crespo Marques, Nilson Maciel, Lirida Naviner, Hao Cai, and Jun Yang. A review of sparse recovery algorithms. *IEEE Access*, 7:1300–1322, 2019. doi:10.1109/access.2018.2886471.

[126] Christopher R. Hayner, Samuel C. Buckner, Daniel Broyles, Evelyn Madewell, Karen Leung, and Behçet Açıkmeşe. HALO: Hazard-Aware Landing Optimization for Autonomous Systems. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2023. doi:10.1109/icra48891.2023.10160655.

[127] Andreas Orthey, Constantinos Chamzas, and Lydia E. Kavraki. Sampling-based motion planning: A comparative review. *Annual Review of Control, Robotics, and Autonomous Systems*, 7(1), November 2023. doi:10.1146/annurev-control-061623-094742.

[128] Stephen Zobell and Craig Wanke. Deferability: A concept for incremental air traffic management decision making. In *9th AIAA Aviation Technology, Integration, and Operations Conference (ATIO)*. American Institute of Aeronautics and Astronautics, 2009. doi:10.2514/6.2009-6964.

[129] S. B. H. Bruder and K. Wedeward. Terrain aided INS robot navigation: a deferred decision making approach. In *42nd Midwest Symposium on Circuits and Systems (Cat. No.99CH36356)*, volume 1, pages 135–139, 1999. doi:10.1109/MWSCAS.1999.867227.

[130] P. O. M. Scokaert and D. Q. Mayne. Min-max feedback model predictive control for constrained linear systems. *IEEE Transactions on Automatic Control*, 43(8):1136–1142, August 1998. doi:10.1109/9.704989.

[131] Avik Jain, Lawrence Chan, Daniel S. Brown, and Anca D. Dragan. Optimal cost design for model predictive control. In *Proceedings of the 3rd Conference on Learning for Dynamics and Control*, volume 144, pages 1205–1217. PMLR, 2021.

[132] Yuxiao Chen, Ugo Rosolia, Wyatt Ubellacker, Noel Csomay-Shanklin, and Aaron D Ames. Interactive multi-modal motion planning with branch model predictive control. *IEEE Robotics and Automation Letters*, 7(2):5365–5372, April 2022. doi:10.1109/LRA.2022.3156648.

[133] Renzi Wang, Mathijs Schuurmans, and Panagiotis Patrinos. Interaction-aware model predictive control for autonomous driving. In *2023 European Control Conference (ECC)*, pages 1–6. IEEE, June 2023. doi:10.23919/ECC57647.2023.10178332.

[134] Ping Lu and Sergio A. Sandoval. Abort guidance during powered descent for crewed lunar missions. In *AIAA Scitech 2021 Forum*. American Institute of Aeronautics and Astronautics, 2021. doi:10.2514/6.2021-0505.

[135] Jason Hardy and Mark Campbell. Contingency planning over probabilistic obstacle predictions for autonomous road vehicles. *IEEE Transactions on Robotics*, 29(4):913–929, 2013. doi:10.1109/TRO.2013.2254033.

[136] John P. Alsterda, Matthew Brown, and J. Christian Gerdes. Contingency model predictive control for automated vehicles. In *2019 American Control Conference (ACC)*, pages 717–722, 2019. doi:10.23919/ACC.2019.8815260.

[137] Jackey Z. Yan and Chris Chu. DeFer: Deferred decision making enabled fixed-outline floorplanner. In *2008 45th ACM/IEEE Design Automation Conference*, pages 161–166, 2008. doi:10.1145/1391469.1391512.

[138] Y. Baram. Partial classification: the benefit of deferred decision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):769–776, 1998. doi:10.1109/34.709564.

[139] Ivo Punčochář and Ondřej Straka. Multiple-model active fault diagnosis with deferred decisions. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 6340–6345, December 2018. doi:10.1109/CDC.2018.8619089.

[140] Amos Tversky and Eldar Shafir. Choice under conflict - the dynamics of deferred decision. *Psychological Science*, 1992.

[141] Adam Grant. The surprising habits of original thinkers. [Online]. Available from: https://youtu.be/fxbCHn6gE3U, February 2016.

[142] Akshay Agrawal and Stephen Boyd. Disciplined quasiconvex programming. *Optimization Letters*, 14(7):1643–1657, October 2020. doi:10.1007/s11590-020-01561-8.

[143] Kathleen C. Laurini, Bernhard Hufenbach, Juergen Hill, and Alain Ouellet. The global exploration roadmap and expanding human/robotic exploration mission collaboration opportunities. In *IAF 66th International Astronautical Congress*, pages 1–9, October 2015.

[144] B. Hufenbach, K. Laurini, N. Satoh, C. Lange, R. Martinez, J. Hill, M. Landgraf, and A. Bergamasco. International missions to lunar vicinity and surface - near-term mission scenario of the global space exploration roadmap. In *IAF 66th International Astronautical Congress*, pages 1–11, October 2015.

[145] D. Lee. Gateway destination orbit model: A continuous 15 year NRHO reference trajectory. Technical report, NASA Johnson Space Center, 2019.

[146] Thomas A. Pavlak. *Trajectory design and orbit maintenance strategies in multi-body dynamical regimes*. PhD thesis, Purdue University, 2013.

[147] Zubin P. Olikara and Daniel J. Scheeres. Numerical method for computing quasi-periodic orbits and their stability in the restricted three-body problem. *Advances in the Astronautical Sciences*, 145(911-930), 2012.

[148] David Folta and Frank Vaughn. A survey of Earth-Moon libration orbits: stationkeeping strategies and intra-orbit transfers. In *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, page 4741, 2004.

[149] Emily M. Zimovan, Kathleen C. Howell, and Diane C. Davis. Near rectilinear halo orbits and their application in cis-lunar space. In *3rd IAA Conference on Dynamics and Control of Space Systems, Moscow, Russia*, volume 20, 2017.

[150] Kathleen C. Howell and Timothy M. Keeter. Station-keeping strategies for libration point orbits-target point and Floquet mode approaches. *Spaceflight mechanics 1995*, pages 1377–1396, 1995.

[151] Gerard Gómez, Kathleen C. Howell, J. Masdemont, and Carles Simó. Station-keeping strategies for translunar libration point orbits. *Advances in Astronautical Sciences*, 99(2):949–967, 1998.

[152] Stefania Soldini, Camilla Colombo, and Scott Walker. Comparison of Hamiltonian structure-preserving and Floquét mode station-keeping for libration-point orbits. In *AIAA/AAS astrodynamics specialist conference*, page 4118, 2014.

[153] D. C. Folta, Thomas A. Pavlak, Kathleen C. Howell, M. A. Woodard, and D. W. Woodfork. Stationkeeping of lissajous trajectories in the Earth-Moon system with applications to ARTEMIS. In *AAS/AIAA Space Flight Mechanics Meeting*, 2010.

[154] Dave Rohrbaugh and Conrad Schiff. Station-keeping approach for the microwave anisotropy probe (map). In *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, page 4429, 2002.

[155] Donald J. Dichmann, Cassandra M. Alberding, and Wayne H. Yu. Stationkeeping monte carlo simulation for the James Webb Space Telescope. In *Proceedings 24th International Symposium on Space Flight Dynamics–24 th ISSFD, Laurel, MD, USA*, 2014.

[156] Kiarash Tajdaran. *Incorporation of mission design constraints in Floquet mode and Hamiltonian structure-preserving orbital maintenance strategies for libration point orbits*. PhD thesis, Purdue University, 2015.

[157] Gerard Gómez, Jaume Llibre, R. Martınez, and Carles Simó. Station keeping of a quasiperiodic halo orbit using invariant manifolds. In *Proceed. 2nd Internat. Symp. on spacecraft flight dynamics, Darmstadt*, pages 65–70, 1986.

[158] Belinda G. Marchand and Kathleen C. Howell. Control strategies for formation flight in the vicinity of the libration points. *Journal of guidance, control, and dynamics*, 28(6):1210–1219, 2005.

[159] Mai Bando and Akira Ichikawa. Formation flying along halo orbit of circular-restricted three-body problem. *Journal of Guidance, Control, and Dynamics*, 38(1):123–129, 2015.

[160] Wei Wang, Giovanni Mengali, Alessandro A. Quarta, and Jianping Yuan. Distributed adaptive synchronization for multiple spacecraft formation flying around Lagrange point orbits. *Aerospace Science and Technology*, 74:93–103, 2018.

[161] Seungyun Jung and Youdan Kim. Formation flying along unstable libration point orbits using switching Hamiltonian structure-preserving control. *Acta Astronautica*, 158:1–11, 2019.

[162] Brian T. Barden and Kathleen C. Howell. Fundamental motions near collinear libration points and their transitions. *The Journal of the Astronautical Sciences*, 46(4):361–378, 1998.

[163] Kathleen C. Howell and Belinda G. Marchand. Natural and non-natural spacecraft formations near the L1 and L2 libration points in the Sun–Earth/Moon ephemeris system. *Dynamical Systems*, 20(1):149–173, 2005.

[164] Davide Guzzetti, Emily M. Zimovan, Kathleen C. Howell, and Diane C. Davis. Stationkeeping analysis for spacecraft in lunar near rectilinear halo orbits. In *27th AAS/AIAA Space Flight Mechanics Meeting*, pages 1–20, 2017.

[165] Vivek Muralidharan. *Stretching Directions in Cislunar Space: Stationkeeping and an Application to Transfer Trajectory Design*. PhD thesis, Purdue University, 2021.

[166] Spencer Boone and Jay Mcmahon. Directional state transition tensors for capturing dominant nonlinear dynamical effects. In *AAS/AIAA Astrodynamics Specialist Conference*, pages 1–21, August 2021.

[167] Matthew Duggan, Xavier Simon, and Travis Moseman. Lander and cislunar gateway architecture concepts for lunar exploration. In *2019 IEEE Aerospace Conference*, pages 1–9. IEEE, 2019.

[168] Diane Davis, Sagar Bhatt, Kathleen Howell, Jiann-Woei Jang, Ryan Whitley, Fred Clark, Davide Guzzetti, Emily Zimovan, and Gregg Barton. Orbit maintenance and navigation of human spacecraft at cislunar near rectilinear halo orbits. In *AAS/AIAA Space Flight Mechanics Meeting*, pages 1–20, 2017.

[169] Catherine L. Thornton and James S. Border. *Radiometric tracking techniques for deep-space navigation*. John Wiley & Sons, 2003.

[170] Kathleen C. Howell and Steven C. Gordon. Orbit determination error analysis and a station-keeping strategy for Sun-Earth L1 libration point orbits. *Journal of the Astronautical Sciences*, 42(2):207–228, 1994.

[171] Rodney L. Anderson, Martin W. Lo, and George H. Born. Application of local Lyapunov exponents to maneuver design and navigation in the three-body problem. In *AAS/AIAA Astrodynamics Specialist Conference*, pages 1–20, August 2003.

[172] Bradley Cheetham. Cislunar Autonomous Positioning System Technology Operations and Navigation Experiment (CAPSTONE). In *ASCEND 2021*. American Institute of Aeronautics and Astronautics, November 2021. doi:10.2514/6.2021-4128.

[173] Further details on communications issues with NASA's CAPSTONE. https://blogs.nasa.gov/artemis/2022/07/05/further-details-on-communications-issues-with-nasas-capstone/, 2022. Accessed: July 5.

[174] Mission Team Determines Cause of Communications Issues for NASA's CAPSTONE. https://blogs.nasa.gov/artemis/2022/07/07/mission-team-determines-cause-of-communications-issues-for-nasas-capstone/, 2022. Accessed: July 7.

[175] Joel J. K. Parker, Fabio Dovis, Benjamin Anderson, Luigi Ansalone, Benjamin Ashman, Frank H. Bauer, Giuseppe D'Amore, Claudia Facchinetti, Samuele Fantinato, Gabriele Impresario, et al. The Lunar GNSS Receiver Experiment (LuGRE). In *Proceedings of the 2022 International Technical Meeting of The Institute of Navigation*, pages 420–437, 2022. doi:10.33012/2022.18199.

[176] Teams Press Ahead Toward Artemis I Launch in Late August. https://blogs.nasa.gov/artemis/2022/07/22/teams-press-ahead-toward-artemis-i-launch-in-late-august/, 2022. Accessed: July 22.

[177] David C. Folta, Thomas A. Pavlak, Amanda F. Haapala, Kathleen C. Howell, and Mark A. Woodard. Earth–moon libration point orbit stationkeeping: Theory, modeling, and operations. *Acta Astronautica*, 94(1):421–433, Jan 2014. doi:10.1016/j.actaastro.2013.01.022.

[178] Lorenzo Bucci, Michele Lavagna, and Rüdiger Jehn. Station keeping techniques for near rectilinear orbits in the earth-moon system. In *Proceedings of 10th international ESA conference on GNC systems, Salzburg, Austria*, volume 29, 2017.

[179] Xiaoyu Fu, Nicola Baresi, and Roberto Armellin. A high-order target point approach to the stationkeeping of near rectilinear halo orbits. In *71th International Astronautical Congress, CyberSpace Edition*. University of Surrey, 2020.

[180] Vivek Muralidharan and Kathleen C. Howell. Stationkeeping in earth-moon near rectilinear halo orbits. In *AAS/AIAA Astrodynamics Specialist Conference, South Lake Tahoe, California, USA*, 2020.

[181] Xiaoyu Fu, Nicola Baresi, and Roberto Armellin. Stochastic optimization for stationkeeping of near rectilinear halo orbits using a high-order target point approach. In *AAS/AIAA Astrodynamics Specialist Conference, Big Sky*, 2021.

[182] Patrick W. Kenneally, Scott Piggott, and Hanspeter Schaub. Basilisk: A flexible, scalable and modular astrodynamics simulation framework. *Journal of aerospace information systems*, 17(9):496–507, 2020.

[183] Jennifer Wood, Mar Cols Margenet, Patrick Kenneally, Hanspeter Schaub, and Scott Piggott. Flexible basilisk astrodynamics visualization software using the unity rendering engine. In *AAS Guidance and Control Conference, Breckenridge, CO*, 2018.

[184] Vittorio Franzese, Simone Ceccherini, Karthik V. Mani, Pierluigi Di Lizia, and Francesco Topputo. Feasibility assessment of autonomous optical navigation in lumio mission. In *69th International Astronautical Congress (IAC 2018)*, pages 1–6. International Astronautical Federation, IAF, 2018.

[185] Greg N. Holt, Christopher N. D'Souza, and David W. Saley. Orion Optical Navigation Progress Toward Exploration Mission 1. In *2018 Space Flight Mechanics Meeting*, page 1978, 2018.

[186] Thibaud Teil, Samuel Bateman, and Hanspeter Schaub. Closed-loop software architecture for spacecraft optical navigation and control development. *The Journal of the Astronautical Sciences*, 67(4):1575–1599, 2020. doi:10.1007/s40295-020-00216-1.

[187] Thibaud Teil, Hanspeter Schaub, and Daniel Kubitschek. Centroid and apparent diameter optical navigation on mars orbit. *Journal of Spacecraft and Rockets*, 58(4):1107–1119, 2021. doi:10.2514/1.A34815.

[188] John A. Christian and Shane B. Robinson. Noniterative horizon-based Optical Navigation by Cholesky Factorization. *Journal of Guidance, Control, and Dynamics*, 39(12):2757–2765, 2016. doi:10.2514/1.G000539.

[189] William M. Folkner, James G. Williams, and Dale H. Boggs. The Planetary and Lunar Ephemeris de 421. *IPN Progress Report*, 42(178):1–34, August 2009. URL: https://ipnpr.jpl.nasa.gov/progress_report/42-178/178C.pdf.

[190] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986. doi:10.1109/TPAMI.1986.4767851.

[191] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[192] Arthur Gelb and The Analytic Sciences Corporation. *Applied Optimal Estimation*. MIT Press, 1974.

[193] Heinz H. Bauschke and Patrick L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, volume 408. Springer, 2011.

[194] John A. Christian. A tutorial on horizon-based optical navigation and attitude determination with space imaging systems. *IEEE Access*, 9:19819–19853, 2021. doi:10.1109/ACCESS.2021.3051914.

[195] Marshall Smith, Douglas Craig, Nicole Herrmann, Erin Mahoney, Jonathan Krezel, Nate McIntyre, and Kandyce Goodliff. The artemis program: An overview of nasa's activities to return humans to the moon. In *2020 IEEE Aerospace Conference*, pages 1–10. IEEE, March 2020. doi:10.1109/aero47225.2020.9172323.

[196] Matthew J. Hart, Robert J. Kinsey, Austin S. Lee, and Justin S. Yoshida. International space station life extension. In *2010 IEEE Aerospace Conference*, pages 1–15. IEEE, March 2010. doi:10.1109/aero.2010.5446890.

[197] Sean Fuller, Emma Lehnhardt, Christina Zaid, and Kate Halloran. Gateway program status and overview. *Journal of Space Safety Engineering*, 9(4):625–628, 2022. doi:10.1016/j.jsse.2022.07.008.

[198] Joseph A. Starek, Behçet Açıkmeşe, Issa A. Nesnas, and Marco Pavone. *Spacecraft Autonomy Challenges for Next-Generation Space Missions*, chapter 1, pages 1–48. Springer Berlin Heidelberg, September 2015. doi:10.1007/978-3-662-47694-9_1.

[199] W. H. Clohessy and R. S. Wiltshire. Terminal guidance system for satellite rendezvous. *Journal of the Aerospace Sciences*, 27(9):653–658, September 1960. doi:10.2514/8.8704.

[200] Daniel Aguilar-Marsillach, Stefano Di Cairano, and Avishai Weiss. Abort-safe spacecraft rendezvous in case of partial thrust failure. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 1490–1495. IEEE, December 2020. doi:10.1109/cdc42340.2020.9303782.

[201] Abraham P. Vinod, Avishai Weiss, and Stefano Di Cairano. Abort-safe spacecraft rendezvous under stochastic actuation and navigation uncertainty. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 6620–6625. IEEE, December 2021. doi:10.1109/cdc45484.2021.9683322.

[202] Daniel Aguilar-Marsillach, Stefano Di Cairano, and Avishai Weiss. Abort-safe spacecraft rendezvous on elliptic orbits. *IEEE Transactions on Control Systems Technology*, 31(3):1133–1148, May 2023. doi:10.1109/tcst.2022.3216077.

[203] Louis Breger and Jonathan P. How. Safe trajectories for autonomous rendezvous of spacecraft. *Journal of Guidance, Control, and Dynamics*, 31(5):1478–1489, September 2008. doi:10.2514/1.29590.

[204] Daniel Aguilar-Marsillach, Stefano Di Cairano, Uros Kalabic, and Avishai Weiss. Fail-safe spacecraft rendezvous on near-rectilinear halo orbits. In *2021 American Control Conference (ACC)*, pages 2980–2985. IEEE, May 2021. doi:10.23919/acc50511.2021.9483328.

[205] E. Blazquez, L. Beauregard, S. Lizy-Destrez, F. Ankersen, and F. Capolupo. Rendezvous design in a cislunar near rectilinear halo orbit. *The Aeronautical Journal*, 124(1276):821–837, October 2019. doi:10.1017/aer.2019.126.

[206] Timothy Goulet, David Woffinden, Nathan Collins, and Blythe Andrews. Robust trajectory design for rendezvous in a near rectilinear halo orbit. In *45th Annual AAS Guidance, Navigation and Control (GN&C) Conference*, pages 1–25, February 2023.

[207] Lorenzo Bucci, Andrea Colagrossi, and Michèle Lavagna. Rendezvous in lunar near rectilinear halo orbits. *Advances in Astronautics Science and Technology*, 1(1):39–43, August 2018. doi:10.1007/s42423-018-0012-6.

[208] Giordana Bucchioni and Mario Innocenti. Ephemeris validation of rendezvous guidance in lunar NRHO. In *AIAA Scitech 2021 Forum*, pages 1–17, Reston, Virginia, January 2021. American Institute of Aeronautics and Astronautics. doi:10.2514/6.2021-0975.

[209] Giordana Bucchioni and Mario Innocenti. Open loop safe trajectory design for cislunar NRHO rendezvous. In *2020 American Control Conference (ACC)*, pages 4337–4342. IEEE, July 2020. doi:10.23919/acc45564.2020.9147502.

[210] David A. Vallado. *Fundamentals of Astrodynamics and Applications*. Microcosm Press, 4th edition, 2013.

[211] Lukas Hewing, Andrea Carron, Kim P. Wabersich, and Melanie N. Zeilinger. On a correspondence between probabilistic and robust invariant sets for linear systems. In *2018 European Control Conference (ECC)*, pages 1642–1647. IEEE, June 2018. doi:10.23919/ecc.2018.8550160.

[212] Mingyi Hong. The proximal primal-dual approach for nonconvex linearly constrained problems. DIMACS Workshop on Distributed Opt., Information Process., and Learning, August 2017. URL: http://dimacs.rutgers.edu/archive/Workshops/Learning/Slides/Mingyi.pdf.

[213] Jiawei Zhang and Zhi-Quan Luo. A proximal alternating direction method of multiplier for linearly constrained nonconvex minimization. *SIAM Journal on Optimization*, 30(3):2272–2302, January 2020. doi:10.1137/19M1242276.

[214] Daoli Zhu, Lei Zhao, and Shuzhong Zhang. A first-order primal-dual method for nonconvex constrained optimization based on the augmented lagrangian. *Mathematics of Operations Research*, March 2023. doi:10.1287/moor.2022.1350.

[215] Ricard Bordalba, Tobias Schoels, Lluis Ros, Josep M. Porta, and Moritz Diehl. Direct collocation methods for trajectory optimization in constrained robotic systems. *IEEE Transactions on Robotics*, pages 1–20, 2022. doi:10.1109/TRO.2022.3193776.

[216] Riccardo Bonalli, Andrew Bylard, Abhishek Cauligi, Thomas Lew, and Marco Pavone. Trajectory optimization on manifolds: A theoretically-guaranteed embedded sequential convex programming approach. In *Robotics: Science and Systems 2019*, 2019. arXiv:1905.07654.

[217] Michael Watterson, Sikang Liu, Ke Sun, Trey Smith, and Vijay Kumar. Trajectory optimization on manifolds with applications to quadrotor systems. *The International Journal of Robotics Research*, 39(2-3):303–320, 2020. doi:10.1177/0278364919891775.

[218] Govind M. Chari, Yue Yu, and Behçet Açıkmeşe. Constraint preconditioning and parameter selection for a first-order primal-dual method applied to model predictive control, 2024. arXiv:2403.15656.

[219] Michael Garstka, Mark Cannon, and Paul Goulart. COSMO: A conic operator splitting method for convex conic problems. *Journal of Optimization Theory and Applications*, 190(3):779–810, September 2021. doi:10.1007/s10957-021-01896-x.

[220] Antonin Chambolle, Claire Delplancke, Matthias J. Ehrhardt, Carola-Bibiane Schönlieb, and Junqi Tang. Stochastic primal dual hybrid gradient algorithm with adaptive step-sizes, 2023. arXiv:2301.02511.

[221] Joel A. E. Andersson, Joris Gillis, Greg Horn, James B. Rawlings, and Moritz Diehl. CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 2018. doi:10.1007/s12532-018-0139-4.

[222] Maximilian Schaller, Goran Banjac, Steven Diamond, Akshay Agrawal, Bartolomeo Stellato, and Stephen P. Boyd. Embedded Code Generation With CVXPY. *IEEE Control Systems Letters*, 6:2653–2658, 2022. doi:10.1109/lcsys.2022.3173209.

[223] Tuomo Valkonen. First-order primal-dual methods for nonsmooth non-convex optimisation. In *Handbook of Mathematical Models and Algorithms in Computer Vision and Imaging: Mathematical Imaging and Vision*, pages 1–42. Springer International Publishing, Cham, 2021. doi:10.1007/978-3-030-03009-4_93-1.

[224] Richard H. Byrd, Frank E. Curtis, and Jorge Nocedal. An inexact newton method for nonconvex equality constrained optimization. *Mathematical Programming*, 122(2):273–299, April 2010. doi:10.1007/s10107-008-0248-3.

[225] Xin Jiang and Lieven Vandenberghe. Bregman primal-dual first-order method and application to sparse semidefinite programming. *Computational Optimization and Applications*, 81(1):127–159, 2022. doi:10.1007/s10589-021-00339-7.

[226] Taewan Kim, Purnanand Elango, and Behçet Açıkmeşe. Joint synthesis of trajectory and controlled invariant funnel for discrete-time systems with locally lipschitz nonlinearities. *International Journal of Robust and Nonlinear Control*, 34(6):4157–4176, January 2024. doi:10.1002/rnc.7186.

[227] Danylo Malyuta, Taylor P. Reynolds, Michael Szmuk, Mehran Mesbahi, Behçet Açıkmeşe, and John M. Carson III. Discretization performance and accuracy analysis for the rocket powered descent guidance problem. In *AIAA SciTech 2019 Forum*, Reston, Virginia, January 2019. doi:10.2514/6.2019-0925.

[228] Philip J. Davis. *Interpolation and Approximation*. Dover Publications, 1975.

[229] Charles H. Acton. Ancillary data services of NASA's Navigation and Ancillary Information Facility. *Planetary and Space Science*, 44(1):65–70, January 1996. doi:10.1016/0032-0633(95)00107-7.