

©Copyright 2026

Samuel C. Buckner

Interaction-Aware Trajectory Optimization: Perception and Contact in Autonomous Systems

Samuel C. Buckner

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2026

Reading Committee:

Behçet Açıkmese, Chair

Mehran Mesbahi

Karen Leung

Abhishek Gupta

Aleksandr Aravkin

Purnanand Elango

Program Authorized to Offer Degree:

William E. Boeing Department of Aeronautics & Astronautics

University of Washington

Abstract

Interaction-Aware Trajectory Optimization: Perception and Contact in Autonomous Systems

Samuel C. Buckner

Chair of the Supervisory Committee:

Professor Behçet Açıkmeşe

William E. Boeing Department of Aeronautics & Astronautics

The capabilities of autonomous systems have been challenged in recent years by the complexity of environmental interaction combined with agile motion demands. Such tasks require reasoning about high-dimensional parameterizations of the physical world, and are critical to accomplish meaningful interaction-driven autonomy. This dissertation develops modeling and optimization techniques for three fundamental challenges in interaction-aware trajectory optimization: active spatial perception of unknown environments, decision-making under unmodeled environmental uncertainty, and multibody contact interaction. In each case, the interaction requirement is embedded directly in the continuous-time optimal control problem and solved by sequential convex programming, in most cases at real-time or near-real-time rates.

First, we address the set-based viewplanning problem, in which a sensor field of view must contain an extended region of interest, and prove conditions under which the resulting infinite-dimensional constraint admits an exact finite-dimensional representation. Next, we lift the overconstrained structure of multi-landmark viewplanning by embedding a differentiable continuous-time filtering model directly into the optimal control problem, yielding an active simultaneous localization and mapping formulation that jointly optimizes vehicle mo-

tion and estimation uncertainty. Then, we introduce a graph-based framework which poses multi-segment trajectory optimization over a directed acyclic graph of trajectory segments, and use it to solve general nonconvex deferred-decision trajectory optimization problems (maintaining reachability to multiple candidate targets for as long as possible) more than an order of magnitude faster than the prior state of the art. An adaptive algorithm is introduced to operationalize this capability for perception-driven safe landing. Finally, we extend continuous-time constraint satisfaction to multibody contact by introducing integral cross-complementarity constraints, enabling contact-implicit trajectory optimization that proposes contact sequences without a prescribed mode schedule. The resulting methods are validated on rocket-powered descent, quadrotor landing and legged locomotion tasks, in both high-fidelity simulators and hardware flight tests, along with consistent benchmarking against relevant baselines at each step.

TABLE OF CONTENTS

	Page
List of Figures	v
List of Tables	x
Glossary	xi
Chapter 1: Introduction	1
1.1 Thesis Contributions	2
1.2 Literature Review	4
1.3 Preliminaries	5
1.3.1 Nonconvex Optimal Control Problems	5
1.3.2 Sequential Convex Programming	7
Chapter 2: Set-Based Viewplanning	14
2.1 Introduction	14
2.1.1 Related Literature	14
2.1.2 Contributions	15
2.2 The Set-Based Viewplanning Problem	16
2.2.1 6-DoF State Space Model	16
2.2.2 Modeling FOV Coverage of a Set-Based Region	18
2.3 Constrained Conic Intersection Theory	21
2.3.1 Connection to the FOV Set Inclusion Problem	29
2.4 The Constrained Visibility Guidance Problem Statement	29
2.4.1 Pointing Events as State-Triggered Constraints	29
2.4.2 Other State & Control Path Constraints	31
2.4.3 Boundary Conditions	32
2.4.4 Reference Trajectory Initialization	33
2.5 Numerical Results	36
2.5.1 Scenario Parameterization	36

2.5.2	Analysis & Discussion	39
Chapter 3:	Active Simultaneous Localization & Mapping	41
3.1	Introduction	41
3.1.1	Related Literature	41
3.1.2	Contributions	46
3.2	Continuous-Time Differentiable Filtering	47
3.2.1	Differentiable SLAM Modeling	50
3.3	The ACT-SLAM Optimization Problem	57
3.3.1	Co-design Configurations	58
3.3.2	Embedded Numerical Continuation	59
3.4	Numerical Results	60
3.4.1	Scenario Parameters	60
3.4.2	Baseline Scenario	61
3.4.3	Model Validation	62
3.4.4	Literature Comparison	63
Chapter 4:	Graph Successive Convexification	72
4.1	Introduction	72
4.1.1	Related Literature	73
4.1.2	Contributions	76
4.2	Problem Formulation	77
4.2.1	Notation	77
4.2.2	Optimal Control Formulation	77
4.2.3	Continuous-Time Constraint Satisfaction	80
4.2.4	Generalized Time Dilation	81
4.3	Transcription	82
4.3.1	Parameterization & Discretization	82
4.3.2	Vertex Segments	85
4.3.3	Edge Connections	86
4.3.4	Discrete-time Problem Formulation	86
4.4	Sequential Convex Programming	87
4.4.1	Exact Penalization	87
4.4.2	The Prox-linear Method	90
4.5	Deferred-Decision Trajectory Optimization	91
4.5.1	Graph Structure	92

4.5.2	Objective & Constraints	92
4.5.3	Initialization	94
4.6	Adaptive-DDTO	96
4.7	Experimental Results	101
4.7.1	Implementation Details	101
4.7.2	DDTO Method Comparison	101
4.7.3	Trade Study: Hazardous Soft Landing	106
4.7.4	Hardware Test: Dynamic Obstacle Avoidance	117
4.7.5	Numerical Parameters	121
Chapter 5:	Contact-Implicit Successive Convexification	123
5.1	Introduction	123
5.1.1	Related Literature	124
5.1.2	Contributions	125
5.2	Contact-Implicit Multibody Dynamics	126
5.2.1	Notation	128
5.2.2	Whole-Body Model	128
5.2.3	Single-Rigid-Body Model	132
5.3	Integral Cross-Complementarity	135
5.4	Contact-Implicit Trajectory Optimization Problem	138
5.4.1	Continuous-Time Dynamics	139
5.4.2	Hybrid Optimal Control	139
5.4.3	Complementarity Relaxation	140
5.4.4	Nonconvex Optimization Problem	142
5.4.5	Control Parameterization	144
5.5	Sequential Convex Programming for CITO Problems	145
5.5.1	Backtracking δ -Homotopy	145
5.5.2	Nested δ -Homotopy	146
5.5.3	SRB to WB Warmstarting	147
5.5.4	Real-Time Implementation	149
5.6	Experimental Results	151
5.6.1	Monoped Study	151
5.6.2	Biped Study	153
5.6.3	Monte Carlo Analysis	156
5.6.4	Half-Cheetah Study	159

Chapter 6: Conclusions	163
Appendix A: Phase-Based Stitching	165
A.1 Modeling Phase-Based Stitching	165
A.2 Discretizing Phase-Based Stitching	167
A.3 Implementing Phase-Based Stitching in SCP	170
Bibliography	172

LIST OF FIGURES

Figure Number		Page
1.1	Illustration of the relation between generalized autonomy modules and the environment (colored modules represent the focus of this dissertation).	1
2.1	An illustration of the problem geometry defined by the FOV set inclusion problem.	19
2.2	A visual alternative to the proof provided in Theorem 2.1.	22
2.3	This diagram illustrates the necessary steps to derive a proximal 6-DOF attitude profile from a 3-DOF solution, with the tilt restriction on the left, and roll restriction on the right.	34
2.4	This figure presents a visualization of the vehicle trajectory and terrain scan events in regards to the candidate powered descent guidance scenario. The vehicle is depicted with a gray cylindrical body at each solver knot point, along with an orange arrow corresponding to the (normalized) thrust vector. Terrain scanning events 1-3 are colored red , blue , and green , respectively, with the boresight pointing vector, FOV cone and target region color-coded. Additionally, the simulated trajectory is displayed in black, with colored segments during the activation of each scanning event.	38
2.5	This figure presents various time signals of interest in regards to the candidate powered descent guidance scenario introduced, with color-coded scan events and constraint bounds.	39
3.1	The autonomy module architecture framed in the context of the active SLAM problem (otherwise referred to as active perception and guidance-navigation codesign in the aerospace context).	42
3.2	Comparison of the estimation process between different SLAM problems.	43
3.3	Illustration demonstrating the modeling approach behind the ACT-SLAM technique applied to the active sensing problem.	45
3.4	A visual interpretation of the set $\Psi_{\mathcal{C}(c,\theta)}$	53

3.5	Left: Visual interpretation of the geometry underlying the FOV set $\mathcal{F}_j$. Right: Visual interpretation of the sigmoid smoothing approximation Φ_κ and what it provides for gradient-based trajectory optimization. The gradient of this function, $\nabla\Phi_\kappa$, is non-zero at all points in its domain, and when performing numerical gradient descent to minimize uncertainty associated with the measurement noise σ_j (or some proximal objective that implicitly seeks to minimize this uncertainty), the optimizer "pushes" in the direction towards perceiving landmarks in the FOV of the vehicle's range-based sensor.	55
3.6	Illustrative diagram of the embedded numerical continuation process to provide an update law for the iterate-varying smoothing parameter κ_k .	59
3.7	Left: Trajectory timelapse for the baseline PDG scenario using ACT-SLAM, where perceptive scans to each landmark are displayed with a 3 second delay between successive scans, and color-coded according to the first , second and third landmarks. Right: Snapshots of the baseline scenario in a high-fidelity visualization environment at different points in time.	61
3.8	Example covariance evolution for the baseline scenario across the timeframe $t \in [0, t_f]$, where relative magnitudes between elements are represented by grayscale color intensity (<i>darker = higher magnitude</i>) under an element-wise square root mapping for visual clarity. Matrix partitions are demarcated by lines.	61
3.9	Vehicle trajectory signals and corresponding constraint sets associated with simulation of the baseline scenario.	62
3.10	Monte Carlo analysis of ACT-SLAM model validation simulation over $n = 1000$ trials. Plots displaying navigation-truth dispersions are shown for each element of the vehicle position, where trajectories are color-coded according to the magnitude of the dispersion at $t = 0$. The 3σ level sets from the ACT-SLAM-output covariance are displayed by symmetric black curves.	64
3.11	Differential entropy history over time between the passive perception, Info-PDG and ACT-SLAM methods.	68
3.12	Trajectory timelapses for the passive perception, Info-PDG and ACT-SLAM methods, using the same visualization conventions as figure 3.7.	68
3.13	Visualization of the terminal covariance matrix, $\Sigma(t_f)$, for the passive perception, Info-PDG and ACT-SLAM methods, using the same visualization conventions as figure 3.8.	69
4.1	Conceptual illustration of GRAPH-SCvx problem modeling approach with application to DDTO problems.	73

4.2	Graphical illustration of the GRAPH-SCvx modeling process, starting with the original OCP stated by problem (4.1), and converted to the transformed OCP stated by problem (4.7), where all trajectory segments are defined on the same normalized time grid.	78
4.3	Illustration of the GRAPH-DDTO warmstart generation process described by algorithm 4.2. Left: four single-target solutions to terminal condition sets $\mathcal{Z}^1$ - $\mathcal{Z}^4$. Right: the converted warmstart for GRAPH-DDTO, where the black segment denotes the trunk and the colored segments denote the branches, with deferral indices $k_1^d, \dots, k_4^d$ and $\lambda = (1, 2, 3, 4)$	94
4.4	Comparison between DDTO and ADAPTIVE-DDTO in the quadrotor soft landing scenario.	97
4.5	Comparison across three different trajectory bundles generated by 1. single-target planning; 2. LEX-DDTO; and 3. GRAPH-DDTO. Deferral times, $t_d^j = t^j(\tau_d^j)$, and suboptimality to each target with respect to the corresponding single-target solution, ϵ^{j*} , are displayed for the latter two cases. We use a global suboptimality upper bound of $\epsilon^j = 0.2$ and uniform GRAPH-DDTO deferral time weighting of $\alpha^j = 1$, $\forall j \in [n_{\text{targ}}]$	102
4.6	Monte Carlo comparison between LEX-DDTO and GRAPH-DDTO across multiple cumulative deferral times, algorithm solve time and percent convergence. For deferral and solve times, the mean and 1- σ uncertainty are displayed by a solid line and a transparent band about the line, respectively.	105
4.7	Left: visual clarification of the different demonstration maps tested on in AirSim. Right: Demonstration of an Adaptive-DDTO landing maneuver on map #3 with snapshots indicating: (A) initial descent with a full ($n_{\text{targ}} = 3$) GRAPH-DDTO solution; (B) midway through descent wherein a target has been lost due to exceeding its deferral time (branched off); (C) final descent locked to a single target.	107
4.8	Comparison of descent profiles between GRAPH-DDTO and comparative methods for a selected problem scenario on map #3. Divert trajectories are color-coded according to temporal progress at the time they were computed relative to the overall maneuver duration.	108
4.9	Monte Carlo statistics of induced energy between GRAPH-DDTO, GR-1 and GR- ∞ across all three demonstration maps. Data is visualized using violin plots characterizing the sampled probability distribution, along with quantile information displayed by box plot overlays and outliers (defined according to table 4.3) plotted with dots outside the box plot ranges.	110
4.10	GUI-based illustration of the hardware test environment for autonomous quadrotor landing with dynamic obstacles.	118

4.11	Timelapses of dynamic obstacle avoidance droptests for quadrotor landing under various operating conditions. For flight tests, camera footage is shown in the top row, and the GUI perspective is shown in the bottom row; for simulated tests, only the GUI perspective is shown. On the far right of each timelapse, the full descent profile and trajectory bundles are displayed under the same conventions as in figure 4.8.	119
4.12	Flight test results in terms of both vertical descent velocity and planner tracking error across all three test scenarios for dynamic obstacle avoidance droptests.	120
5.1	Illustration of the types of multibody models considered in this chapter. . . .	127
5.2	Illustration of how the forward-kinematic (FK) box constraint set is defined for the SRB model.	134
5.3	Illustration of a signal pair a, b satisfying integral cross-complementarity over a time interval $t \in [t_k, t_{k+1}]$ with $a(t) = 0, \forall t \in [t_k, t_{k+1}]$	137
5.4	Illustration of the Fischer-Burmeister relaxation process	141
5.5	Illustration of the warmstart process from an SRB model to a WB model. . .	148
5.6	Illustration of the real-time implementation pipeline for ci-SCvx.	150
5.7	Monoped locomotion solution on a flat surface generated by ci-SCvx with a minimal-coordinate WB model. The solution converges in 873 SCP iterations, taking 37.53 seconds to solve.	152
5.8	Monoped locomotion solution on a flat surface generated by ci-SCvx with an SRB-warmstarted minimal-coordinate WB model. The solution converges in 340 SCP iterations (SRB) + 167 SCP iterations (WB), taking 11.48 seconds elapsed time to solve.	153
5.9	Time histories of the normal and tangential contact forces and impulses for the warmstarted monoped solution. Grid intervals with sustained contact are denoted by gray regions.	154
5.10	Bipedal locomotion solutions on different surfaces. The path of the center of mass (CoM) of the torso is shown by the orange trace, and the path of each contact point (foot connected to each leg) is shown by the cyan traces. . . .	155
5.11	Time histories of tangential velocity, Lagrange multiplier and friction cone term for the hind foot of the biped on the sinusoidal surface (figure 5.10c). . .	156
5.12	Monte Carlo analysis of the elapsed solve time across the monoped and biped models between all five candidate multibody models. The sample mean of each model's elapsed solve time is displayed by a color-coded bar, with the error bars depicting one sample standard deviation.	157

5.13	Monte Carlo analysis of the objective value across the monoped and biped models between all five candidate multibody models. The sample mean of each model's objective value is displayed by a color-coded bar, with the error bars depicting one sample standard deviation.	158
5.14	Ablation study of the δ -homotopy scheme on the biped model. The sample mean of each method's solve time and objective value is displayed by a color-coded bar, with the error bars depicting one sample standard deviation. . . .	159
5.15	HalfCheetah locomotion solutions on a flat surface across multiple tested configurations. The path of the center of mass (CoM) of torso is shown by the orange trace, and the path of the each contact point (foot connected to each leg) is shown by the cyan traces.	160
5.16	Comparison of open-loop (blue) and closed-loop (red) gait solutions for the HalfCheetah model on a flat surface when deployed in the MuJoCo simulator, against the optimized CI-SCvx solution directly propagated on its own dynamics model (gray).	160
A.1	Illustration of how variable elimination works with the non-redundant stitching formulation. In the redundant case, post-discretization, $\tilde{x}_1^j$ is a redundant variable. In the non-redundant case, $\tilde{x}_1^j$ is able to serve the same utility as the second discretization point, $\tilde{x}_2^j$, in the redundant case.	169

LIST OF TABLES

Table Number		Page
2.1	Terrain scanning scenario parameters.	37
3.1	Supplementary scenario parameters.	66
3.2	Literature Comparison Summary.	71
4.1	DDTO algorithm comparison.	105
4.2	Ablation study on GRAPH-DDTO warmstart.	106
4.3	Monte Carlo comparison across various metrics.	112
4.4	Ablation study on $E_{\max}$	117
5.1	Half-Cheetah comparisons between CI-SCvx and MJPC.	158

GLOSSARY

CT-SCVX: (Continuous-Time) Successive Convexification

GRAPH-SCVX: Graph Successive Convexification

CI-SCVX: Contact-Implicit Successive Convexification

ACT-SLAM: Active Continuous-Time Simultaneous Localization and Mapping

DDTO: Deferred-Decision Trajectory Optimization

HALO: Hazard-Aware Landing Optimization

CVG: Constrained Visibility Guidance

CT-OCP: Continuous-Time Optimal Control Problem

CTCS: Continuous-Time Constraint Satisfaction

SCP: Sequential Convex Programming

MPC: Model Predictive Control

STC: State-Triggered Constraint

SOC(P): Second-Order Cone (Programming)

TPBVP: Two-Point Boundary Value Problem

CITO: Contact-Implicit Trajectory Optimization

PDG: Powered Descent Guidance

SLAM: Simultaneous Localization and Mapping

EKF: Extended Kalman Filter

EKBF: Extended Kalman-Bucy Filter

FOV: Field of View

SDF: Signed Distance Function

DEM: Digital Elevation Map

DOF: Degree of Freedom

EDL: Entry, Descent and Landing

DCM: Direction Cosine Matrix

ENU: East-North-Up Reference Frame

ROS: Robot Operating System

WB: Whole-Body

SRB: Single-Rigid-Body

COM: Center of Mass

FB: Fischer-Burmeister

ZOH: Zero-Order Hold

FOH: First-Order Hold

DFOH: Discontinuous First-Order Hold

ATE: Average Trajectory Error

ACKNOWLEDGMENTS

At the center of any doctoral student’s research journey is collaboration with their advisor. To my advisor, Prof. Behçet Açıkmese, I am immensely grateful for the opportunity to join your incredible research group. Without your guidance, I would not have been able to make the progress I have made in the past five years. The many back-and-forth discussions we have had have been invaluable in shaping my research and developing my collaborative skillset. I am endlessly impressed by the pace with which you can pick up and understand a new technical problem from scratch, and I hope to one day acquire such a skillset as well. I would also like to thank you for giving me the flexibility to pursue my own research directions, which has led to me solving problems I had never envisioned prior to starting my Ph.D.

I am grateful to have had the ability to collaborate extensively with Purnanand Elango, both on DDTO while at UW and CI-SCvx while interning at MERL. Your attention to detail and formalization of problem solving have played a large part in shaping my research style. Over the years, I have come to see how rare and valuable that mindset is in a world where clarity and rigor are too often overlooked. Thank you for inspiring me to be a better and more thorough researcher. Thank you for serving on my dissertation committee as well, it was a pleasure having you as a part of my journey. Thank you to the rest of my committee members, namely Prof. Mehran Mesbahi, Prof. Karen Leung, Prof. Abhishek Gupta and Prof. Aleksandr Aravkin, for your time and effort in reviewing my dissertation and providing valuable feedback about my research and future directions at each step.

I would also like to thank the Autonomous Controls Lab (ACL) flight test crew, namely Chris Hayner, Jason Zhou, Skye Mceowen, Ben Chung, Natalia Pavlasek and Aman Tiwary. The many hours and late nights we spent troubleshooting all sorts of software and hardware

issues for the lab’s quadrotor stack to prepare flight demonstrations have been a welcome mixture of frustration and joy, and have been an irreplaceable experience. Thank you for bearing with me on preparing and conducting DDTO flight tests as well. I am sincerely grateful for your patience and perseverance.

To my NASA colleagues, namely John Carson, Jeff Gutkowski, Daniel Matz, Breanna Johnson, Davis Adams, Gavin Mendeck, Julia Briden and the rest of the team I have collaborated with, I would like to thank you for all of the incredible opportunities to operationalize our research on practical powered descent guidance problems. You all represent the pinnacle of guidance, navigation and control engineering, and I am grateful for the chance to have worked with such a talented and dedicated group of people. I would also like to thank Joshua Schaffer from Astrobotic for collaborating with me on various guidance problems related to the Griffin lunar lander. I could not envision a more exciting mission to demonstrate real-time convex optimization in action.

To Abhi Kamath, it was a pleasure collaborating on trajectory optimization problems while interning with you at NASA. To Samet Uzun, thank you for all of your insight and feedback towards new ways to solve optimization problems, your versatility and creativity in problem solving have been a constant source of inspiration. And thank you for being a great gym buddy as well, I will miss our occasional workouts together. To Fabio Spada and Avi Mittal, I have always deeply enjoyed our random whiteboard conversations on miscellaneous topics, which have always been a welcome break from day-to-day research. I also would like to thank the rest of the ACL community I have had the opportunity to chat and collaborate with over the years, namely Taewan Kim, Govind Chari, Oliver Sheridan, Dylan Janak, Sarah Li, Jake Gonzales, Justin Ganiban and Carlos Morales.

Finally, I would like to thank my parents, Brandon and Patricia, as well as my siblings, Nate and Maddie, for your encouragement throughout both my academic and personal journey. I would not have been able to make it this far without your love and support.

This material is based upon work supported by the National Science Foundation Gradu-

ate Research Fellowship Program under Grant No. DGE-2140004. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation. I also gratefully acknowledge the support of the Achievement Rewards for College Scientists (ARCS) Foundation, Seattle Chapter. The Graph-SCvx material in Chapter 4 was sponsored by the Department of the Navy, Office of Naval Research, under ONR award number N00014-20-1-2288. Any opinions, findings, and conclusions or recommendations expressed in that material are those of the author and do not necessarily reflect the views of the Office of Naval Research.

In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of the University of Washington's products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink. If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

DEDICATION

to my friends and family

Chapter 1

INTRODUCTION

Autonomous systems have developed substantial interest at the turn of the 21st century, in a diverse array of applications such as aerospace, automotive and robotics. Increasingly, these systems are required to interface with complex and dynamic environments while maintaining safety, reliability and efficiency. The prevailing architecture for solutions to these autonomy problems typically comprises: (i) task planning, (ii) trajectory planning, (iii) state estimation, and (iv) feedback control (their interconnected relationships are displayed by Figure 1.1). Task planning determines the high-level objectives/requirements that the system should adhere to. Trajectory planning synthesizes a trajectory the system should take to satisfy these objectives while maintaining physical realism. State estimation localizes the autonomous system within its surrounding environment (which may also include estimating the state of the environment itself). Finally, feedback control allows the system to correct for errors between the desired state and estimated state, thus producing a closed-loop feedback response to unmodeled uncertainties.

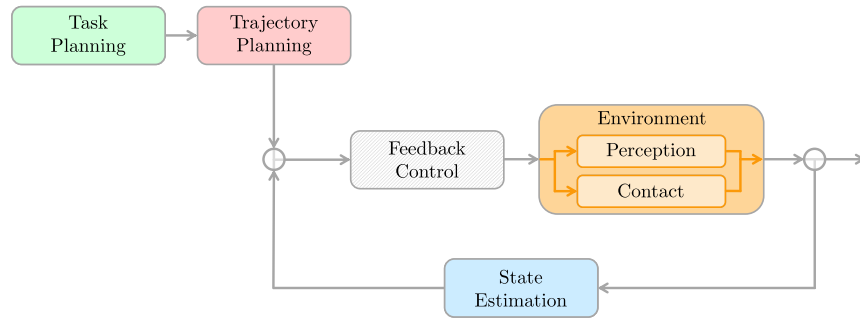


Figure 1.1: Illustration of the relation between generalized autonomy modules and the environment (colored modules represent the focus of this dissertation).

In this dissertation, we will focus on the problem of trajectory planning through numer-

ical optimization, and in particular, the problem of planning for autonomous systems to interact with their surrounding environment. This class of methods will hence be referred to as *interaction-aware trajectory optimization*. In doing this, we will observe couplings emerge between the trajectory planning module, and both task planning and state estimation modules. Such couplings have been used to improve autonomous system design; for example, trajectory planning may provide the ability to proactively maneuver to reduce state estimation uncertainty [115], and may also be used to inform mission-level or vehicular requirements at the task planning stage [51]. In the interaction-aware context, we may, for example, be interested in perceiving an environment from a distance to reduce its uncertainty before physical (contact-based) interaction [104].

Notably, all such problems will be posed as *nonconvex continuous-time trajectory optimization* problems. These problems come with several key considerations. First, *nonconvexity* requires either the direct conversion of a nonconvex problem to a convex one [5, 6], or an iterative (and often computationally-challenging) process to solve the problem [25, 37]. Second, *continuous-time* modeling requires conversion of an infinite-dimensional problem to an equivalent or approximate finite-dimensional proxy (this process is often referred to as *discretization*, or transcription), introducing further computational challenges [82]. In the literature, many techniques have been proposed to tackle this class of problems, as will be discussed in the following literature review section. However, there is a third complexity that is often overlooked in the discussion of fundamental techniques: environmental interaction. Autonomous systems do not operate in a vacuum, and planning for such interactions can often introduce substantial modeling and computational complexity to the problem. Examples of such environmental interactions include physical (contact [23]) and non-physical (spatial perception [104]). In this dissertation, we will investigate a variety of novel techniques aimed at handling environmental interactions in the context of both physical contact and spatial perception, collectively referred to as *interaction-aware trajectory optimization*.

1.1 Thesis Contributions

In the context of interaction-aware trajectory optimization, it has been observed that systematically accounting for environmental interactions requires significant augmentation of

problem complexity. We present an assortment of novel techniques which describe candidate solutions to these problems aimed at reducing model complexity and being amenable to modern computing architectures. In addition, we suggest numerous research extensions along each avenue. Specifically, the following chapters are introduced:

1. **Chapter 2.** [19] introduces a constraint modeling approach aimed at satisfying task planning objectives that require perceptive observations to geometric regions in the surrounding environment, labeled the *set-based viewplanning* problem. This approach, extending on methods which constrain observations to singular points, converts the resulting infinite-dimensional set containment problem towards a finite dimensional equivalency suitable for numerical optimization. Then, the constrained visibility guidance (CVG) framework is proposed to impose set containment constraints to an enumeration of regions using state-triggered constraints in a successive convexification-based trajectory optimization problem. The resulting framework is demonstrated on a practical powered descent guidance scenario for spacecraft landing with terrain scanning objectives.
2. **Chapter 3.** [16] explores a different approach to perception-aware trajectory optimization through an information-theoretic lens, where predicted uncertainty evolution in both the autonomous agent and its surrounding environment are jointly characterized and minimized for (referred to as the *active SLAM* problem). This effort includes defining a simultaneous localization and mapping (SLAM) model with the continuous-time extended Kalman-Bucy filter, proposing steps to obtain a $\mathcal{C}^1$ -differentiable approximation to the filter dynamics, and employing successive convexification with embedded numerical continuation to recover the original dynamics model at convergence. The resulting method is compared against related techniques in the context of powered descent guidance, showing up to 44.3% performance improvement in terms of perceptive information gain.
3. **Chapter 4.** [15, 52] proposes a numerically-efficient and robust approach to planning under unmodeled uncertainty (including those driven by perception) through the concept of decision deferral, enabling the autonomous agent to continuously gather

information before committing to a terminal objective (formalized as the *deferred decision trajectory optimization* (DDTO) problem). The graph successive convexification (GRAPH-SCVX) framework is introduced to solve DDTO problems in real-time under flexible objective and constraint specification. Additionally, a receding-horizon algorithm (ADAPTIVE-DDTO) is introduced to operationalize DDTO within the context of perception-aware maneuvers. This approach is demonstrated for quadrotor soft landing in hazardous environments using both high-fidelity flight simulators and onboard hardware demonstrations.

4. **Chapter 5.** [17, 18] takes a departure from designing around perceptive interactions, and instead considers the case of modeling for direct contact interactions for large-scale multibody systems. A generalized approach referred to as *contact-implicit successive convexification* (CI-SCVX) is introduced, including customized contact-implicit dynamics models for both whole-body and reduced-order systems, a reformulation of contact interaction constraints using integral cross-complementarity, a complementarity homotopy algorithm with backtracking to systematically tighten the smooth approximations to these constraints, and a parallelizable solver architecture to obtain solutions to these problems in real-time.

1.2 Literature Review

A large focus of this dissertation surrounds a specific class of algorithms to solve nonconvex optimization problems known as *sequential convex programming* (SCP) [116]. SCP defines a class of structure-exploiting algorithms which apply *convexification* to non-convex components of an optimization problem, and otherwise impose convex constraints directly, such that the resulting problem is a convex approximation of the original problem. This process is repeated iteratively and recursively about the previous iterate’s solution until a convergence condition is met. Related techniques, such as sequential linear programming (SLP) [20, 100] and sequential quadratic programming (SQP) [49, 108], have been explored extensively as well. In 2016, Mao et al. [84] introduced the first formalization of SCP applied to the continuous-time trajectory optimization problem template. In 2018, Mao et al. [85] showed superlinear convergence properties of this algorithm class. In the same year, Szmuk

et al. [124] developed an application of SCP-based trajectory optimization towards the 6 degree-of-freedom (DoF) powered descent guidance (PDG) problem.

In the following several years, SCP-based trajectory optimization has been explored for a variety of applications, including the fuel-optimal powered descent guidance (rocket landing) problem [112, 124, 125, 129], spacecraft rendezvous problem with integer constraints [80, 81] and quadrotor flight problem [127, 128]. These applications share strict real-time computation requirements for autonomous operation, motivating the additional development of first-order convex solvers suited for SCP-based trajectory optimization problems [58, 59, 148]. Recently, the continuous-time successive convexification framework [37] was proposed to formalize these techniques under flexible problem parameterization, including adaptive time grids and continuous-time satisfaction of path constraints. Subsequent relevant developments include auto-tuned primal-dual methods for successive convexification [87, 88] and prox-linear-based trajectory optimization with signal temporal logic specifications [139].

In this dissertation, we will cover several variations of continuous-time successive convexification, including a preliminary introduction in section 1.3.2 and a collection of modeling techniques catered to the requirements of successive convexification. We will also discuss relevant literature with respect to perception and contact-based interaction-aware trajectory optimization in the body of each following chapter.

1.3 Preliminaries

Before covering the main technical contributions of this dissertation, we first introduce some important preliminary concepts which unify all key contributions. This includes the general-form definition of a nonconvex optimal control problem, and a class of SCP methods to solve these problems.

1.3.1 Nonconvex Optimal Control Problems

We first introduce a general class of nonconvex continuous-time optimal control problems. These problems are described as an infinite-dimensional optimization of the form:

Continuous-Time Optimal Control Problem (1.1)

$$\underset{x(\cdot), u(\cdot)}{\text{minimize}} \quad J(x(t_f)) \quad (1.1a)$$

$$\text{subject to} \quad \dot{x}(t) = f(t, x(t), u(t)), \quad (1.1b)$$

$$g(t, x(t), u(t)) \leq 0, \quad (1.1c)$$

$$h(t, x(t), u(t)) = 0, \quad (1.1d)$$

$$p(x(t_0), x(t_f)) \leq 0, \quad (1.1e)$$

$$q(x(t_0), x(t_f)) = 0, \quad (1.1f)$$

$$\forall t \in [t_0, t_f].$$

Problem (1.1) describes the minimization of an objective function $J : \mathbb{R}^{n_x} \rightarrow \mathbb{R}$ over a state space $x \in \mathbb{R}^{n_x}$ and control space $u \in \mathbb{R}^{n_u}$ across a finite time interval $t \in [t_0, t_f] \subset \mathbb{R}$. The problem is subject to a differential constraint describing the state-space *dynamics* with a function $f : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_x}$, as well as *path constraints* described by functions $g : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_g}$ and $h : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_h}$, and *boundary conditions* described by functions $p : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_p}$ and $q : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_q}$. The canonical Mayer-form objective (1.1a) is chosen in this problem representation, however any other objective functional (such as a time-varying integration term) can be converted to this form without loss of generality.

Definition 1.1 (*Convex Optimal Control Problem*). *In the case where functions $\{f, h, q\}$ are linear and functions $\{J, g, p\}$ are convex [12], problem (1.1) may be referred to as a convex optimal control problem.*

As previously stated, convex optimal control problems are desirable due to their computational tractability and global optimality guarantees. The following subsection describes a solution strategy to problem (1.1) in the general case where this property does not hold.

1.3.2 Sequential Convex Programming

There exist many candidate techniques to solve problem (1.1) in the general nonconvex case. Sequential convex programming (SCP) [83] describes one such class of techniques, where a discretized version of the problem is iteratively approximated as a convex problem by recursively using the previous iterate as a reference point (e.g., for linearization), and solved until a convergence criterion is met. Continuous-Time Successive Convexification (CT-SCvx) [37] presents one specific instance of SCP tailored to the canonical-form continuous-time optimal control problem, and will be the focus of the remainder of this subsection. We make use of the compressed description in [16] with modifications that unify all techniques presented in this dissertation. For a more formalized description of CT-SCvx, we refer the interested reader to [37].

Model Augmentation

We first consider a series of augmentations to the state-space model (across $x(t), u(t)$) to capture continuous-time modeling complexities. The first augmentation enables us to solve problem (1.1) such that t_f may implicitly become an optimization variable (as well as all other times $t \in [t_0, t_f]$), making the problem *free-final-time*. We consider the time dilation formulation introduced in [19, 37], where wall-clock time is defined as a smooth, strictly-monotonic mapping $t : [0, 1] \rightarrow \mathbb{R}_+$ such that $t(0) = t_0$ and $t(1) = t_f$. We will consequently define a normalized time $\tau \in [0, 1]$ and time derivative $s(\tau) = \partial t(\tau) / \partial \tau > 0$. Through the chain rule, we can now define a dynamical relationship on the normalized time grid as

$$\dot{x}(t(\tau)) = \frac{\partial x(t(\tau))}{\partial \tau} = s(\tau) f(x(t(\tau)), u(t(\tau))), \quad (1.2)$$

where $s(\tau)$ becomes an explicit optimization variable. The second augmentation concerns guaranteeable satisfaction of path constraints at all points in time. It has been shown in [37, Lemma 2] that equations (1.1c)-(1.1d) are equivalently solved by satisfaction of the integral constraint

$$\int_{t_0}^{t_f} h(x(t), u(t))^2 + \max(0, g(x(t), u(t)))^2 dt = 0. \quad (1.3)$$

Moreover, we can define an additional explicit optimization variable $\theta \in \mathbb{R}$ that satisfies

$$\dot{\theta}(t) = \Lambda(x(t), u(t)) \triangleq h(x(t), u(t))^2 + \max(0, g(x(t), u(t)))^2, \quad (1.4)$$

$$\theta(t_0) = \theta(t_f), \quad (1.5)$$

which equivalently satisfies equation (1.3) and, consequently, the path constraints (1.1c)-(1.1d). Finally, we can combine these two model augmentations together, by considering the following modifications to the state-space model. We start by redefining the optimization state and input variables through concatenation

$$\tilde{x}(\tau) = (x(\tau), \theta(\tau), t(\tau)), \quad (1.6)$$

$$\tilde{u}(\tau) = (u(\tau), s(\tau)). \quad (1.7)$$

Then, the dynamics function can be redefined as

$$F(\tilde{x}, \tilde{u}) \triangleq s \begin{bmatrix} f(x, u) \\ \Lambda(x, u) \\ 1 \end{bmatrix}. \quad (1.8)$$

Linearization & Discretization of the Dynamics

One of the key details behind the CT-SCvx algorithm is the direct linearization of the non-convex dynamics to obtain a convex constraint representation in the CT-SCvx subproblem formulation. We consider the following jacobian terms associated with the dynamics

$$A(\tau) = \nabla_{x=\bar{x}(\tau)} F(x, \tilde{u}(\tau)), \quad (1.9a)$$

$$B(\tau) = \nabla_{u=\bar{u}(\tau)} F(\tilde{x}(\tau), u), \quad (1.9b)$$

where $(\bar{x}, \bar{u})$ represent a reference trajectory. Then, the continuous-time linear time varying (CT-LTV) dynamics are described through a first-order Taylor series expansion

$$\dot{\tilde{x}}(\tau) = A(\tau)\tilde{x}(\tau) + B(\tau)\tilde{u}(\tau) + w(\tau), \quad (1.10)$$

where $w(\tau)$ corresponds to the Taylor series residual term

$$w(\tau) = F(\bar{x}(\tau), \bar{u}(\tau)) - A(\tau)\bar{x}(\tau) - B(\tau)\bar{u}(\tau), \quad (1.11)$$

and $(\bar{x}(\tau), \bar{u}(\tau))$, $\tau \in [0, 1]$ defines a reference trajectory. Next, to perform discretization, the time interval is discretized into a discrete set of knot points $\tau_k \in [0, 1]$, $k = 1, \dots, N$, and a finite-dimensional control parameterization is chosen, with parameters for each knot point interval denoted as $U_k \in \mathbb{R}^{n_U}$, $k = 1, \dots, N-1$. We can consequently solve the initial value problem over each knot point interval to arrive at the discrete-time linear time varying (DT-LTV) dynamical relationship

$$\tilde{x}_{k+1} = A_k \tilde{x}_k + B_k U_k + w_k. \quad (1.12)$$

Details on derivation of the terms A_k, B_k, w_k can be found in [37] for the interested reader. Additionally, under this finite-dimensional control parameterization, we may require continuity constraints between knot interval parameters (for example, continuity of first-order hold). To do this, we must impose the additional convex constraint

$$(U_1, \dots, U_{N-1}) \in \mathcal{U}. \quad (1.13)$$

Linearization of Pointwise Constraints

For numerical & methodology purposes, we may also wish to impose certain path constraints as pointwise constraints, such that they are only applied at knot points in the transformed (discretized & parameterized) problem. To do this, we will specify pointwise

path constraints

$$g_p(\tilde{x}_k, U_k) \leq 0, \quad (1.14a)$$

$$h_p(\tilde{x}_k, U_k) = 0. \quad (1.14b)$$

Then, these constraints must be linearized directly about a discrete-time reference trajectory $(\bar{x}_k, \bar{U}_k)_{k=1}^{N-1}$

$$A_k^g \tilde{x}_k + B_k^g U_k + w_k^g \leq 0, \quad (1.15a)$$

$$A_k^h \tilde{x}_k + B_k^h U_k + w_k^h = 0, \quad (1.15b)$$

where

$$A_k^g = \nabla_{x=\bar{x}_k} g_p(x, \bar{U}_k), \quad (1.16a)$$

$$B_k^g = \nabla_{U=\bar{U}_k} g_p(\bar{x}_k, U), \quad (1.16b)$$

$$w_k^g = g_p(\bar{x}_k, \bar{U}_k) - A_k^g \bar{x}_k - B_k^g \bar{U}_k, \quad (1.16c)$$

$$A_k^h = \nabla_{x=\bar{x}_k} h_p(x, \bar{U}_k), \quad (1.16d)$$

$$B_k^h = \nabla_{U=\bar{U}_k} h_p(\bar{x}_k, U), \quad (1.16e)$$

$$w_k^h = h_p(\bar{x}_k, \bar{U}_k) - A_k^h \bar{x}_k - B_k^h \bar{U}_k. \quad (1.16f)$$

Forming the Convex Subproblem

With the aforementioned details in place, we are now ready to form the convex subproblem, as part of the CT-SCvx framework, through implementation of the *prox-linear method*. This technique involves the embedding of a trust region term on the optimization variables directly, as well as an ℓ_1 -norm penalization term on slack variables applied to convexified constraints (which, for this problem formulation, refer only to the nonlinear dynamics), to the main objective function. These terms help alleviate concerns of *artificial infeasibility* and *unboundedness* [83], and are important for numerical implementation. We recall that the original objective is denoted as $J(\cdot)$, and define the following additional components of

the objective function

$$J_\eta(\tilde{x}_K, U_K) \triangleq \sum_{k=1}^{N-1} \delta \tilde{x}_k^\top \delta \tilde{x}_k + \delta U_k^\top \delta U_k + \delta \tilde{x}_N^\top \delta \tilde{x}_N, \quad (1.17a)$$

$$J_\nu(\nu_K) \triangleq \|\nu_K\|_1, \quad (1.17b)$$

where

$$\delta \tilde{x}_k \triangleq \tilde{x}_k - \bar{x}_k, \quad (1.18a)$$

$$\delta U_k \triangleq U_k - \bar{U}_k, \quad (1.18b)$$

$$\tilde{x}_K \triangleq (\tilde{x}_1, \dots, \tilde{x}_N), \quad (1.18c)$$

$$U_K \triangleq (U_1, \dots, U_{N-1}), \quad (1.18d)$$

$$\nu_K \triangleq (\nu_1, \dots, \nu_N). \quad (1.18e)$$

Here, we define $\nu_k \in \mathbb{R}^{n_z}$ as slack variables applied to linearized constraints which enable constraint violation during non-converged iterates to prevent artificial infeasibility. As such, we include slack terms for the dynamics constraint, ν_k^d , the pointwise inequality constraint, ν_k^g , and the pointwise equality constraint, ν_k^h , and define $\nu_k = (\nu_k^d, \nu_k^g, \nu_k^h)$. The aforementioned reference trajectory either corresponds to the solution to the previous subproblem iteration, or to an initial guess on the first iterate. We note that, in the case that J is nonconvex, a convex surrogate (e.g. linearization) of this cost must be used in its place to ensure a convex problem definition.

ℓ_1 -Penalized Prox-Linear Subproblem (1.19)

$$\underset{\tilde{x}_K, U_K, \nu_K}{\text{minimize}} \quad w_o J(\tilde{x}_K) + w_\eta J_\eta(\tilde{x}_K, U_K) + w_\nu J_\nu(\nu_K) \quad (1.19a)$$

$$\text{subject to} \quad \tilde{x}_{k+1} - A_k \tilde{x}_k - B_k U_k - w_k = \nu_k^d, \quad (1.19b)$$

$$A_k^g \tilde{x}_k + B_k^g U_k + w_k^g \leq \nu_k^g, \quad (1.19c)$$

$$A_k^h \tilde{x}_k + B_k^h U_k + w_k^h = \nu_k^h, \quad (1.19d)$$

$$p(x_1, x_N) \leq 0, \quad (1.19e)$$

$$q(x_1, x_N) = 0, \quad (1.19f)$$

$$\theta_{k+1} - \theta_k \leq \epsilon_{\text{CTCS}}, \quad (1.19g)$$

$$\tilde{x}_k \in \mathcal{X}_k, \quad (1.19h)$$

$$U_K \in \mathcal{U}, \quad (1.19i)$$

$$\forall k \in [N - 1].$$

Problem (1.19) contains the expanded objective weighted with trust region and dynamic violation penalization terms in (1.19a), where $w_o, w_\eta, w_\nu \in \mathbb{R}_+$ correspond to relative weighting terms for J , J_η and J_ν , respectively. In terms of constraints, this problem contains the DT-LTV dynamics with a slack violation term in constraint (1.19b), the linearized pointwise constraints with additional slack violation terms in constraints (1.19c)-(1.19d), the boundary conditions in constraints (1.19e)-(1.19f), the pointwise convex state constraint (1.19h), and the control parameterization constraint (1.19i). Notably, an additional (convex) constraint (1.19g) is applied to ensure satisfaction of equation (1.5), with a small slack term $\epsilon_{\text{CTCS}} \sim 1e-4$ to ensure satisfaction of the linear independence constraint qualification (LICQ) condition [37]. This subproblem is solved in an iterative manner until the convergence criteria

$$J_\eta(\tilde{x}_K, U_K) \leq \epsilon_\eta, \quad (1.20a)$$

$$J_\nu(\nu_K) \leq \epsilon_\nu, \quad (1.20b)$$

are satisfied.

Chapter 2

SET-BASED VIEWPLANNING

2.1 Introduction

This chapter initiates an examination of perceptive trajectory planning through a deterministic trajectory optimization lens. We assume control over a single rigid-body vehicle that has a perceptive sensor (e.g. camera, LiDAR) attached, with the goal of imposing constraints which enforce strict field-of-view (FOV) visibility over a defined region of interest under specific set parameterizations. We refer to this problem as the *set-based viewplanning* problem. This chapter is based on [19]; the presentation here follows that paper’s structure. A related approach including direct convexification of the visibility constraint set has also been jointly proposed with the same underlying theory [118], however is not discussed in this chapter.

2.1.1 Related Literature

In this chapter, we focus our efforts on studying set-based viewplanning in the context of the *powered descent guidance* (PDG) problem. The PDG problem statement concerns the theoretical and algorithmic design of pinpoint landing strategies on celestial bodies under critical hardware-driven constraints (such as computational requirements). SCP-based trajectory optimization algorithms have been deployed extensively for both translation-only [125] (otherwise known as 3-DoF) and translation-rotation [112, 124] (otherwise known as 6-DoF) lander parameterizations. In the literature, extensions towards FOV visibility constraints have been made which constrain sensor line-of-sight (LOS)–within the FOV envelope–towards a singular point of interest, such as the landing site [111]. These ap-

This chapter is based on work previously published as [19]. Reprinted by permission of the American Institute of Aeronautics and Astronautics, Inc.

proaches are not limited to landing sites, and can be generalized towards keep-in and keep-out regions (in attitude space); in such cases, the problems may even be simplified under rotation-only maneuvers [65, 70]. Recent approaches aim to impose LoS constraints only when the vehicle is within a subset of the state-space; this approach is typically referred to as a state-triggered constraint (STC) [59, 111, 112]. For example, sensor LOS may only be required when between a minimum and maximum range from the point of interest.

There has also been a growing interest in the multiple point LoS problem, where the goal is to impose constraints such that the sensor LoS is maintained towards a set of points. Using signal temporal logic (STL), tractable solutions have been proposed [137, 138] which can implicitly select the optimal ordering of points to constrain LoS towards. When a collection of points is assumed to correspond to a region of interest, other methods will jointly impose visibility to the entire collection of points, either as hard constraints [53] or as a soft objective [107]. However, these methods do not formally handle visibility to continuous regions, motivating an approach which can tractably handle joint visibility towards such sets.

2.1.2 Contributions

In this chapter, we investigate an extension towards viewing a non-singular *region of interest*, rather than a point of interest. This problem is notably difficult due to the inclusion of an infinite-dimensional constraint set; using constrained conic intersection theory, we show how this constraint set can be equivalently posed as a finite-dimensional constraint and tractably solved for modern PDG scenarios.

The resulting formulation, **Constrained Visibility Guidance** (CVG), contains the following novel extensions relative to the aforementioned literature:

1. A new CVG-specific constraint set is formulated which enables guaranteed sensor line-of-sight to an *entire* circular region of interest (rather than a single point) with arbitrary centroid location and radius. Furthermore, an approximation is introduced which enables extraction of a closed-form solution and facilitates generalization to guaranteed visibility of any set that can be reasonably approximated with a 2-norm

ball.

2. A new theory of constrained conic intersections is introduced which validates the closed-form solution of the 2-norm ball approximated CVG constraint set mentioned previously.
3. An integrated formulation of various state-of-the-art techniques in SCP modeling and optimal control is developed and catered to the mission-practical specification of terrain scanning events.
4. A methodology for initialization of SCP algorithms under this problem class is presented which facilitates conversion of any 3-DOF solution to a proximal 6-DOF solution.

The remainder of the chapter is outlined as follows. Section 2.2 introduces the infinite-dimensional set-based viewplanning constraint. Section 2.3 introduces constrained conic intersection theory to reformulate the prior constraint set. Section 2.4 presents the CVG framework as a solution strategy for set-based viewplanning problems. Section 2.5 provides numerical results on a practical PDG scenario.

2.2 The Set-Based Viewplanning Problem

In this section, we introduce the set-based viewplanning problem as an infinite-dimensional constraint set imposed jointly on a vehicle's translation and rotation space. To do this, we must first introduce the concept of a cartesian 6-DoF state space model.

2.2.1 6-DoF State Space Model

We begin our definition of a 6-DoF state space model with definitions of reference frames. Specifically, we consider inertial and body-fixed reference frames defined as follows:

- $\mathcal{I}$: An inertially-fixed East-North-Up (ENU) reference frame with the origin located at the landing site.
- $\mathcal{B}$: A body-fixed reference frame with $+x$ pointing along the longitudinal axis of the vehicle, $+y$ pointing out the side of the vehicle, and $+z$ completing the right-handed system.

The state space model follows a standard quaternion-based (cartesian) 6-DOF formulation for the powered descent guidance problem as in [19, 124]. The vehicle state $x(t) \in \mathbb{R}^{n_x}$ consists of a vertical concatenation of vehicle mass $m(t) \in \mathbb{R}$, inertial position $r_{\mathcal{I}}(t) \in \mathbb{R}^3$, inertial velocity $v_{\mathcal{I}}(t) \in \mathbb{R}^3$, body-relative unit quaternion attitude $q_{\mathcal{I} \rightarrow \mathcal{B}}(t) \in \mathbb{R}^4$ and body-fixed angular velocity $\omega_{\mathcal{B}}(t) \in \mathbb{R}^3$, with $n_x = 14$. Furthermore, the vehicle control $u(t) \in \mathbb{R}^{n_u}$ consists of a vertical concatenation of body-fixed thrust $T_{\mathcal{B}}(t) \in \mathbb{R}^3$ (under a single-engine model) and body-fixed moment $M_{\mathcal{B}}(t) \in \mathbb{R}^3$ (applied through a reaction control system rather than through gimbaling), with $n_u = 6$. Time $t \in \mathbb{R}_+$ is bounded by $t_0 = 0$ and t_f , such that $t \in [0, t_f]$. The full system dynamics can be expressed as

$$\textbf{Mass:} \quad \dot{m}(t) = -\alpha \|T_{\mathcal{B}}(t)\|_2, \quad (2.1a)$$

$$\textbf{Position:} \quad \dot{r}_{\mathcal{I}}(t) = v_{\mathcal{I}}(t), \quad (2.1b)$$

$$\textbf{Velocity:} \quad \dot{v}_{\mathcal{I}}(t) = \frac{1}{m(t)} \mathcal{R}_{\mathcal{I} \rightarrow \mathcal{B}}^\top(t) T_{\mathcal{B}}(t) + g_{\mathcal{I}}, \quad (2.1c)$$

$$\textbf{Attitude:} \quad \dot{q}_{\mathcal{I} \rightarrow \mathcal{B}}(t) = \frac{1}{2} \Omega(\omega_{\mathcal{B}}(t)) q_{\mathcal{I} \rightarrow \mathcal{B}}(t), \quad (2.1d)$$

$$\textbf{Angular Velocity:} \quad \dot{\omega}_{\mathcal{B}}(t) = J_{\mathcal{B}}^{-1} (M_{\mathcal{B}}(t) - [\omega_{\mathcal{B}}(t) \times] J_{\mathcal{B}} \omega_{\mathcal{B}}(t)), \quad (2.1e)$$

where $\alpha \in \mathbb{R}$ denotes the vehicle's engine proportionality constant in terms of vacuum-specific impulse $I_{\text{sp}} \in \mathbb{R}$ and Earth's standard gravity constant $g_0 \in \mathbb{R}$, $g_{\mathcal{I}} \in \mathbb{R}^3$ denotes the local gravitational acceleration expressed in the inertial frame, and $J_{\mathcal{B}} \in \mathbb{R}^{3 \times 3}$ denotes the vehicle's moment of inertia matrix in the body frame. Additionally, the skew-symmetric matrix operations on an input vector $\xi \in \mathbb{R}^3$ are defined as

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

Finally, an expression of the vehicle's attitude in terms of a direction cosine matrix, $\mathcal{R}_{\mathcal{I} \rightarrow \mathcal{B}}(t) \in \text{SO}(3)$, is given by considering a transformation from unit quaternion $q_{\mathcal{I} \rightarrow \mathcal{B}}(t) = [q_0, q_1, q_2, q_3]^\top$

to a direction cosine matrix defined as

$$\mathcal{R}(q) = \begin{bmatrix} 1 - 2(q_2^2 + q_3^2) & 2(q_1q_2 + q_0q_3) & 2(q_1q_3 - q_0q_2) \\ 2(q_1q_2 - q_0q_3) & 1 - 2(q_1^2 + q_3^2) & 2(q_2q_3 + q_0q_1) \\ 2(q_1q_3 + q_0q_2) & 2(q_2q_3 - q_0q_1) & 1 - 2(q_1^2 + q_2^2) \end{bmatrix}.$$

2.2.2 Modeling FOV Coverage of a Set-Based Region

With the 6-DoF state space model defined, we can now formally define the set-based view-planning constraint we wish to impose. We start by considering a body-fixed boresight vector $b_{\mathcal{B}} \in \mathbb{R}^3$ (with unit magnitude) associated with a vision-based sensor onboard the vehicle (such as the central pointing direction of a camera or LiDAR-based system), coupled with a field-of-view (FOV) half-angle $\beta \in \mathbb{R}$. The prototypical boresight pointing constraint seeks to align the vehicle's boresight vector towards the relative direction of a point $z \in \mathbb{R}^3$ (which may represent a landing site or some other point-of-interest) such that the angle between the boresight vector and relative direction $z - r(t)$ is less than β (moreover, such that the point z is in the field of view of the vehicle's vision sensor). This constraint can be formally constructed as a *biconic* constraint in $r_{\mathcal{I}}(t)$ and $b_{\mathcal{I}}(t)$ (that is, the constraint defines a convex cone in each variable if the other is held constant; in this case, a second-order cone, specifically). We will denote this constraint set by $\mathcal{F}(z)$ as

$$\mathcal{F}(z) = \left\{ (r_{\mathcal{I}}(t), b_{\mathcal{I}}(t)) \mid \|z_{\mathcal{I}} - r_{\mathcal{I}}(t)\| \leq \frac{(z_{\mathcal{I}} - r_{\mathcal{I}}(t))^{\top} b_{\mathcal{I}}(t)}{\cos \beta} \right\}, \quad (2.2)$$

where $b_{\mathcal{I}}(t) = \mathcal{R}_{\mathcal{I} \rightarrow \mathcal{B}}^{\top}(t) b_{\mathcal{B}}$. If we instead choose to consider a *region* of interest $\mathcal{Z} \subseteq \mathbb{R}^3$, we can extend this definition as

$$\mathcal{F}(\mathcal{Z}) = \left\{ (r_{\mathcal{I}}(t), b_{\mathcal{I}}(t)) \mid \|\tilde{z}_{\mathcal{I}} - r_{\mathcal{I}}(t)\| \leq \frac{(\tilde{z}_{\mathcal{I}} - r_{\mathcal{I}}(t))^{\top} b_{\mathcal{I}}(t)}{\cos \beta}, \forall \tilde{z} \in \mathcal{Z} \right\} \quad (2.3a)$$

$$= \bigcap_{\tilde{z} \in \mathcal{Z}} \left\{ (r_{\mathcal{I}}(t), b_{\mathcal{I}}(t)) \mid \|\tilde{z}_{\mathcal{I}} - r_{\mathcal{I}}(t)\| \leq \frac{(\tilde{z}_{\mathcal{I}} - r_{\mathcal{I}}(t))^{\top} b_{\mathcal{I}}(t)}{\cos \beta} \right\}. \quad (2.3b)$$

For most non-trivial forms of $\mathcal{Z}$, the above constraint set would be infinite-dimensional (if $\mathcal{Z}$ itself does not have finite cardinality), and cannot be expressed as a tractable set of

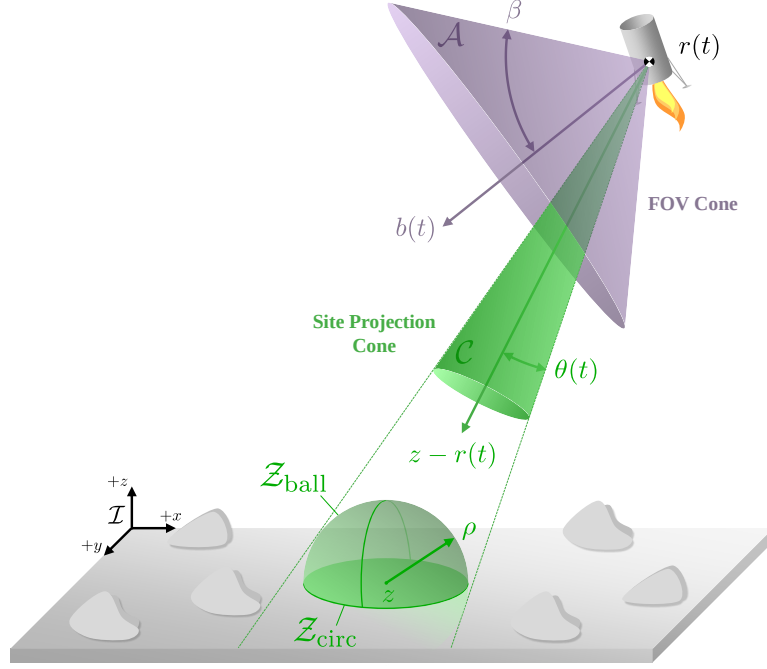


Figure 2.1: An illustration of the problem geometry defined by the FOV set inclusion problem.

constraints in a numerical optimization problem. In the context of the PDG problem, we may seek to consider a circular region of interest on the ground, with radius ρ , constrained to the inertial x - y plane (under a flat-terrain model), denoted as the set $\mathcal{Z}_{\text{circ}} \subseteq \mathbb{R}^3$ with

$$\mathcal{Z}_{\text{circ}} \triangleq \{\tilde{z} \in \mathbb{R}^3 \mid \|H(z - \tilde{z})\| \leq \rho, e_z^\top \tilde{z} = 0\}, \quad H = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad (2.4)$$

where H denotes the projection matrix onto the terrain plane. The constraint set we ultimately wish to enforce is $\mathcal{F}(\mathcal{Z}_{\text{circ}})$, however, as earlier noted, this set does not lend itself to a finite-dimensional parameterization. An approximation can be formed, however, by instead considering the ball $\mathcal{Z}_{\text{ball}}$ which circumscribes $\mathcal{Z}_{\text{circ}}$ as

$$\mathcal{Z}_{\text{ball}} \triangleq \{\tilde{z} \in \mathbb{R}^3 \mid \|z - \tilde{z}\| \leq \rho\}. \quad (2.5)$$

This approximation will henceforth be termed the *ball approximation* on the region $\mathcal{Z}_{\text{circ}}$. We note that $\mathcal{Z}_{\text{ball}}$ acts as a conservative overapproximation of $\mathcal{Z}_{\text{circ}}$ (since $\mathcal{Z}_{\text{circ}} \subset \mathcal{Z}_{\text{ball}}$), and consequently introduces further restrictions on the feasible set from the definition of an intersection

$$\mathcal{Z}_{\text{circ}} \subset \mathcal{Z}_{\text{ball}} \implies \mathcal{F}(\mathcal{Z}_{\text{circ}}) \supseteq \mathcal{F}(\mathcal{Z}_{\text{ball}}).$$

Next, we can reframe this problem by considering the rays which extend from some $r_{\mathcal{I}}(t)$ towards all points in $\mathcal{Z}_{\text{ball}}$, denoted as $\xi \in \mathbb{R}^3$. Geometrically, this set represents a second-order cone (SOC) pointing in the direction $z_{\mathcal{I}} - r_{\mathcal{I}}(t)$ with a half-angle $\theta(t)$ such that it represents the tangent cone to the set $\mathcal{Z}_{\text{ball}}$ (that is, this set defines the cone that *projects* to $\mathcal{Z}_{\text{ball}}$). This set, denoted $\mathcal{C}$, can be stated as

$$\mathcal{C} = \left\{ (r_{\mathcal{I}}(t), \xi) \mid \|z_{\mathcal{I}} - r_{\mathcal{I}}(t)\| \|\xi\| \leq \frac{(z_{\mathcal{I}} - r_{\mathcal{I}}(t))^{\top} \xi}{\cos \theta(t)} \right\}, \quad (2.6)$$

where

$$\theta(t) = \arcsin \left(\frac{\rho}{\|z_{\mathcal{I}} - r_{\mathcal{I}}(t)\|} \right). \quad (2.7)$$

In the above, $\theta(t)$, implicitly considered a function of $r_{\mathcal{I}}(t)$, can be derived from simple trigonometric relations. We can also consider the set defining the sensor FOV cone, which can be simply stated by the set $\mathcal{A}$ as

$$\mathcal{A} = \left\{ (b_{\mathcal{I}}(t), \xi) \mid \|\xi\| \leq \frac{b_{\mathcal{I}}(t)^{\top} \xi}{\cos \beta} \right\}. \quad (2.8)$$

A visualization of both sets $\mathcal{A}$ and $\mathcal{C}$ is provided in Fig. 2.1 for a fixed $r_{\mathcal{I}}(t)$ and $b_{\mathcal{I}}(t)$. The problem we would like to formally pose is to find all $r_{\mathcal{I}}(t), b_{\mathcal{I}}(t) \in \mathbb{R}^3$ such that $\mathcal{C} \subseteq \mathcal{A}$. Informally, this means that the FOV cone $\mathcal{A}$ obtains full coverage of the approximate landing site region $\mathcal{Z}_{\text{ball}}$, since $\mathcal{C}$ projects to $\mathcal{Z}_{\text{ball}}$. This is equivalent to finding the set $\mathcal{B}$ in $(r_{\mathcal{I}}(t), b_{\mathcal{I}}(t))$ that satisfies the relationship

$$\bigcap_{\xi \in \mathcal{C}} \mathcal{A} = \mathcal{B}. \quad (2.9)$$

It can be shown through theorem 2.1 that the desired constraint set $\mathcal{B}$ has the form

$$\mathcal{B} = \left\{ (r_{\mathcal{I}}(t), b_{\mathcal{I}}(t)) \mid \|z_{\mathcal{I}} - r_{\mathcal{I}}(t)\| \leq \frac{(z_{\mathcal{I}} - r_{\mathcal{I}}(t))^{\top} b_{\mathcal{I}}(t)}{\cos(\beta - \theta(t))} \right\}. \quad (2.10a)$$

We note that this constraint set is almost identical to the original pointing constraint to a single point (2.2), with the $\cos \beta$ term replaced by a $\cos(\beta - \theta(t))$ term (which provides direct quantification of the further reduction in feasible space to impose this constraint). Finally, this constraint set is only valid to impose if the vehicle's relative position is large enough to capture the entire region of interest within the sensor field-of-view (such that $\beta > \theta(t)$). Formally, this consideration can be imposed as an additional constraint

$$\|z_{\mathcal{I}} - r_{\mathcal{I}}(t)\| > \frac{\rho}{\sin \beta}, \quad (2.11)$$

which can be approximated as

$$\frac{\rho}{\sin \beta} - \|z_{\mathcal{I}} - r_{\mathcal{I}}(t)\| \leq -\epsilon_{\rho}, \quad (2.12)$$

where $\epsilon_{\rho} \approx 1e - 6$ to encode a tractable inequality constraint (since strict inequalities cannot be traditionally imposed in numerical solvers). We will additionally impose a design restriction that $\beta < \pi/2$, which is generally reasonable for practical purposes.

2.3 Constrained Conic Intersection Theory

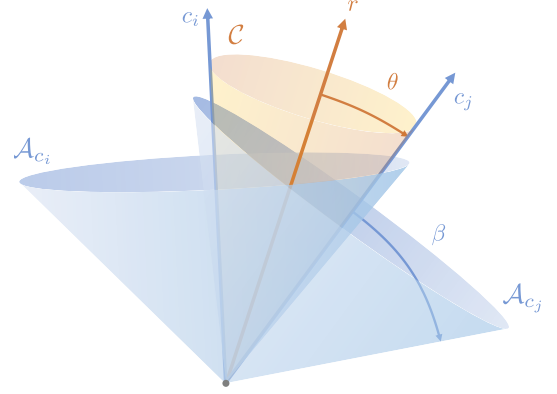
In this section, we develop the theory necessary to prove the relationship established in (2.9) holds. Moreover, we concern ourselves with determining the equivalence of two closed-form sets, wherein one set represents the infinite intersection of second-order cones. In theory, even the intersection of two second-order cones may not have an analytical closed-form solution under non-trivial conditions, however, it can be shown that this set equivalence can be made if the conic pointing vector of the intersected second-order cones is chosen as the varied parameter for the intersection operation, and this vector itself belongs to another second-order cone. A visual interpretation of Theorem 2.1 is also provided in Fig. 2.2.

Theorem 2.1 (*Intersection of Conics Constrained to a Conic*). *We seek to show*

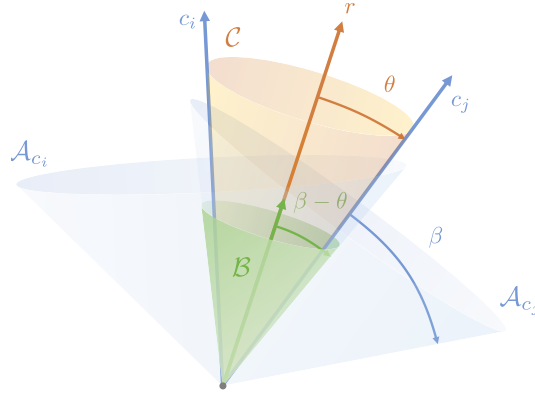
that a closed-form set equivalence for embedded intersection of second-order cones (SOCs)



(a) We start by constructing the second-order cone that defines the intersection operation, $\mathcal{C}$. This cone contains any $c \in \mathcal{C}$ that can be used to construct $\mathcal{A}_c$.



(b) We can then select two elements $c_i, c_j \in \mathcal{C}$ that are coplanar to r on the boundary of $\mathcal{C}$, and visualize their corresponding second-order cones $\mathcal{A}_{c_i}, \mathcal{A}_{c_j}$.



(c) The intersecting second-order cone, denoted as $\mathcal{B}$, is geometrically shown to have a half-angle of $\beta - \theta$ as stated by equation (2.13). We note that to truly visualize the intersecting set, we would need to consider all c_j in the boundary of $\mathcal{C}$, since $\text{int}(\mathcal{C})$ is not activated.

Figure 2.2: A visual alternative to the proof provided in Theorem 2.1.

holds, such that

$$\bigcap_{c \in \mathcal{C}} \left\{ x \in \mathbb{R}^3 \mid \|c\| \|x\| \leq \frac{c^\top x}{\cos \beta} \right\} = \left\{ x \in \mathbb{R}^3 \mid \|r\| \|x\| \leq \frac{r^\top x}{\cos(\beta - \theta)} \right\}, \quad (2.13)$$

where the set $\mathcal{C} \subseteq \mathbb{R}^3$ is defined by another SOC as

$$\mathcal{C} \triangleq \left\{ c \in \mathbb{R}^3 \mid \|r\| \|c\| \leq \frac{r^\top c}{\cos \theta} \right\}, \quad (2.14)$$

such that $r \in \mathbb{R}^3$ and $\theta, \beta \in \mathbb{R}$ are constant, where we additionally impose the angular constraints

$$0 \leq \theta < \beta < \frac{\pi}{2}. \quad (2.15)$$

Proof. We can apply the shorthand set notation to the problem defined in equation (2.13) as

$$\bigcap_{c \in \mathcal{C}} \mathcal{A}_c = \mathcal{B},$$

where

$$\begin{aligned} \mathcal{A}_c &\triangleq \left\{ x \in \mathbb{R}^3 \mid \|c\| \|x\| \leq \frac{c^\top x}{\cos \beta} \right\}, \\ \mathcal{B} &\triangleq \left\{ x \in \mathbb{R}^3 \mid \|r\| \|x\| \leq \frac{r^\top x}{\cos(\beta - \theta)} \right\}. \end{aligned}$$

To show set equivalence, we must show the following two properties hold:

1. **Set inclusivity condition** ($\cap \mathcal{A}_c \supseteq \mathcal{B}$): show that if $x \in \mathcal{B}$, then $x \in \mathcal{A}_c$ for every $c \in \mathcal{C}$.
2. **Set exclusivity condition** ($\cap \mathcal{A}_c \subseteq \mathcal{B}$): show that if $x \notin \mathcal{B}$, then there exists some $c \in \mathcal{C}$ such that $x \notin \mathcal{A}_c$.

Part 1: set inclusivity condition. Consider a linear decomposition of vector c to a basis $\{r, v, w\}$ such that

$$c = \alpha_r r + \alpha_v v + \alpha_w w, \quad (2.16)$$

with $r \perp v$, $r \perp w$ and $v \perp w$. For some vector $c \in \mathcal{C}$, we can choose a proper $v \in \mathbb{R}^3$ such that $\alpha_w = 0$ and $\alpha_v \geq 0$ (polar coordinate representation), reducing the linear decomposition to

$$c = \alpha_r r + \alpha_v v. \quad (2.17)$$

For $c \in \mathcal{C}$, the inequality relationship

$$\cos \theta \leq \frac{r^\top c}{\|r\| \|c\|} \leq 1 \quad (2.18)$$

holds. Then, we ultimately seek to show that

$$\frac{r^\top c}{\|r\| \|c\|} \geq \cos \theta \implies \left| \frac{v^\top c}{\|v\| \|c\|} \right| \leq \sin \theta. \quad (2.19)$$

To do this, consider the right triangle formed by $\alpha_r r$, $\alpha_v v$ and c . We can define the angle between r and c as ϕ , and the angle between v and c as ψ , noting that under the right triangle, $\phi + \psi = \pi/2$. By definition, we also know that $\phi \leq \theta \implies \cos \phi \geq \cos \theta$, which forms the relationship

$$\frac{r^\top c}{\|r\| \|c\|} = \cos \phi \geq \cos \theta. \quad (2.20)$$

This consequently implies that $\psi \geq \pi/2 - \theta \implies \cos \psi \leq \cos(\pi/2 - \theta) = \sin \theta$. We also note that the angle ψ remains the same regardless of the signage of v . Thus, we have the relationships

$$\frac{v^\top c}{\|v\| \|c\|} = \cos \psi \leq \sin \theta, \quad (2.21a)$$

$$-\frac{v^\top c}{\|v\| \|c\|} = \cos \psi \leq \sin \theta, \quad (2.21b)$$

which, when combined, allow us to obtain the relationship described in (2.19). Then, we arrive at the bounds

$$-\sin \theta \leq \frac{v^\top c}{\|v\| \|c\|} \leq \sin \theta. \quad (2.22)$$

By plugging in the decomposition of c from equation (2.17) into equations (2.18) and (2.22),

we arrive at a set of relationships in terms of the basis weights

$$\frac{\|c\|}{\|r\|} \cos \theta \leq \alpha_r \leq \frac{\|c\|}{\|r\|}, \quad (2.23a)$$

$$0 \leq \alpha_v \leq \frac{\|c\|}{\|v\|} \sin \theta, \quad (2.23b)$$

where we note that equation (2.23a) implies $\alpha_r \geq 0$ (since $\cos \theta > 0$ when $\theta < \pi/2$), and that α_v is lower-bounded to zero due to the aforementioned domain restriction. We can then apply a similar linear decomposition of vector x with the same basis $\{r, v, w\}$ as

$$x = \gamma_r r + \gamma_v v + \gamma_w w. \quad (2.24)$$

Then, we can use the definition of a vector $x \in \mathcal{B}$ to apply (2.13) such that

$$\cos(\beta - \theta) \leq \frac{r^\top x}{\|r\| \|x\|} \leq 1, \quad (2.25)$$

and apply the same argument used for (2.22) to arrive at the relationships

$$-\sin(\beta - \theta) \leq \frac{v^\top x}{\|v\| \|x\|} \leq \sin(\beta - \theta), \quad (2.26a)$$

$$-\sin(\beta - \theta) \leq \frac{w^\top x}{\|w\| \|x\|} \leq \sin(\beta - \theta). \quad (2.26b)$$

Consequently, we can plug in the decomposition of x from equation (2.24) into equations (2.25)-(2.26b) to arrive at a set of relationships in terms of basis weights

$$\frac{\|x\|}{\|r\|} \cos(\beta - \theta) \leq \gamma_r \leq \frac{\|x\|}{\|r\|}, \quad (2.27a)$$

$$-\frac{\|x\|}{\|v\|} \sin(\beta - \theta) \leq \gamma_v \leq \frac{\|x\|}{\|v\|} \sin(\beta - \theta), \quad (2.27b)$$

$$-\frac{\|x\|}{\|w\|} \sin(\beta - \theta) \leq \gamma_w \leq \frac{\|x\|}{\|w\|} \sin(\beta - \theta), \quad (2.27c)$$

where we note, once again, that equation (2.27a) implies $\gamma_r \geq 0$ (since $\cos(\beta - \theta) > 0$ when $\beta - \theta < \pi/2$). Ultimately, we need to show that for any $c \in \mathcal{C}$ and $x \in \mathcal{B}$, that $x \in \mathcal{A}_c$, such

that the relationship

$$\cos \beta \leq \frac{c^\top x}{\|c\| \|x\|} \quad (2.28)$$

is satisfied. We can plug in the decompositions of c and x to arrive at the relationship

$$\frac{c^\top x}{\|c\| \|x\|} = \frac{\alpha_r \gamma_r \|r\|^2 + \alpha_v \gamma_v \|v\|^2}{\|c\| \|x\|}. \quad (2.29)$$

Then, we can minimize the right-hand side to obtain an inequality

$$\min \left\{ \frac{\alpha_r \gamma_r \|r\|^2 + \alpha_v \gamma_v \|v\|^2}{\|c\| \|x\|} \right\} \leq \frac{c^\top x}{\|c\| \|x\|}. \quad (2.30)$$

We have, by definition, that $\alpha_r, \alpha_v, \gamma_r \geq 0$ and $\gamma_v \in \mathbb{R}$. Then, to perform the minimization in equation (2.30), we need to minimize $\alpha_r, \gamma_r, \gamma_v$ and maximize α_v . We can use the previously-defined bounds on each term to do this; plugging in yields the relationship

$$\min \left\{ \frac{\alpha_r \gamma_r \|r\|^2 + \alpha_v \gamma_v \|v\|^2}{\|c\| \|x\|} \right\} \quad (2.31a)$$

$$= \cos \theta \cos(\beta - \theta) - \sin \theta \sin(\beta - \theta) \quad (2.31b)$$

$$= \cos \beta, \quad (2.31c)$$

where the last relationship can be shown through trigonometric expansion. Therefore, we have that equation (2.28) is satisfied, and so the set inclusivity condition is shown.

Part 2: set exclusivity condition. We now wish to look at the case in which $x \notin \mathcal{B}$, such that

$$\frac{r^\top x}{\|r\| \|x\|} < \cos(\beta - \theta). \quad (2.32)$$

Moreover, we wish to find some $c \in \mathcal{C}$ such that $x \notin \mathcal{A}_c$, such that

$$\frac{c^\top x}{\|c\| \|x\|} < \cos \beta. \quad (2.33)$$

We can start by using the definition of the projection x onto r , stated as

$$\text{proj}_r x = \frac{r^\top x}{r^\top r} r, \quad (2.34)$$

We would like to consider the vector $c \in \mathcal{C}$ that is as *far away* from the vector x (in terms of angular displacement) as possible. In geometric consideration of this, we construct c on the boundary of $\mathcal{C}$ with a component perpendicular to r pointing in the direction of $-\perp \text{proj}_r x$ (defined under $\text{proj}_r x + \perp \text{proj}_r x = x$). We can once again consider a decomposition of c as

$$c = \alpha r - \perp \text{proj}_r x, \quad (2.35a)$$

$$= \alpha r + \frac{r^\top x}{r^\top r} r - x, \quad (2.35b)$$

where α can be found using the relationship

$$\frac{r^\top c}{\|r\| \|c\|} = \cos \theta \quad (2.36a)$$

$$\implies \frac{\|r\|}{\|c\|} \alpha + \frac{r^\top x}{\|r\| \|c\|} - \frac{r^\top x}{\|r\| \|c\|} = \cos \theta \quad (2.36b)$$

$$\implies \alpha = \frac{\|c\|}{\|r\|} \cos \theta. \quad (2.36c)$$

We can also find $\|c\|^2 = c^\top c$ from equation (2.35) simplified as

$$\|c\|^2 = \alpha^2 \|r\|^2 + \|x\|^2 \sin^2(\psi) \quad (2.37a)$$

$$\implies \frac{\|x\|}{\|c\|} = \frac{\sin \theta}{\sin \psi}, \quad (2.37b)$$

where ψ is the angle between r and x such that $\cos \psi = r^\top x / (\|r\| \|x\|)$, and the last relationship comes from plugging in the definition of α from equation (2.36). Then, we can

use these substitutions to transform equation (2.33) into the relationship

$$\frac{c^\top x}{\|c\|\|x\|} = \alpha \frac{r^\top x}{\|c\|\|x\|} + \frac{(r^\top x)^2}{\|c\|\|x\|\|r\|^2} - \frac{\|x\|}{\|c\|} \quad (2.38a)$$

$$= \frac{r^\top x}{\|r\|\|x\|} \cos \theta + \frac{\|x\|}{\|c\|} \left(\frac{r^\top x}{\|r\|\|x\|} \right)^2 - \frac{\|x\|}{\|c\|} \quad (2.38b)$$

$$= \cos \theta \cos \psi - \frac{\|x\|}{\|c\|} (1 - \cos^2 \psi) \quad (2.38c)$$

$$= \cos \theta \cos \psi - \frac{\|x\|}{\|c\|} \sin^2 \psi \quad (2.38d)$$

$$= \cos \theta \cos \psi - \sin \theta \sin \psi \quad (2.38e)$$

$$= \cos(\theta + \psi), \quad (2.38f)$$

where relation (2.38a) comes from equation (2.35), relation (2.38b) comes from equation (2.36), relation (2.38c) comes from the definition of ψ , relation (2.38e) comes from equation (2.37), and the remaining relations come from trigonometric identities. We can also observe that, when $x \notin \mathcal{B}$, the relationship

$$\cos \psi < \cos(\beta - \theta) \quad (2.39a)$$

$$\implies \theta + \psi > \beta \quad (2.39b)$$

$$\implies \cos(\theta + \psi) < \cos \beta \quad (2.39c)$$

must hold. Thus, we have shown that $x \notin \mathcal{A}_c$ when $x \notin \mathcal{B}$ as

$$\frac{c^\top x}{\|c\|\|x\|} = \cos(\theta + \psi) < \cos \beta, \quad (2.40)$$

and, consequently, the set exclusivity condition is shown. $\square$

2.3.1 Connection to the FOV Set Inclusion Problem

We can revisit the relationship in equation (2.9), which is stated in expanded form as follows:

$$\bigcap_{(r_{\mathcal{I}}(t), \xi) \in \mathcal{C}} \left\{ (b_{\mathcal{I}}(t), \xi) \mid \|\xi\| \leq \frac{\xi^\top b_{\mathcal{I}}(t)}{\cos \beta} \right\} = \left\{ (r_{\mathcal{I}}(t), b_{\mathcal{I}}(t)) \mid \|z_{\mathcal{I}} - r_{\mathcal{I}}(t)\| \leq \frac{(z_{\mathcal{I}} - r_{\mathcal{I}}(t))^\top b_{\mathcal{I}}(t)}{\cos(\beta - \theta(t))} \right\}, \quad (2.41a)$$

$$\mathcal{C} = \left\{ (r_{\mathcal{I}}(t), \xi) \mid \|z_{\mathcal{I}} - r_{\mathcal{I}}(t)\| \|\xi\| \leq \frac{(z_{\mathcal{I}} - r_{\mathcal{I}}(t))^\top \xi}{\cos \theta(t)} \right\}. \quad (2.41b)$$

Using Theorem 2.1, we can see that the above equivalence holds for *any* $r_{\mathcal{I}}(t) \in \mathbb{R}^3$ such that $\theta(t) < \beta$, which we explicitly constrained by imposing equation (2.12). Then, it follows that we can impose the right-hand side of the equivalence as an additional constraint even when $r_{\mathcal{I}}(t)$ is a decision variable, provided $r_{\mathcal{I}}(t)$ belongs to a valid domain enforced by equation (2.12).

2.4 The Constrained Visibility Guidance Problem Statement

In this section, we define the full constrained visibility guidance (CVG) framework by imposing the set-based viewplanning constraint set defined by equations (2.10a) and (2.12). Moreover, these constraints are labeled *pointing events* and imposed as state-triggered constraints, along with other practical path constraints for practical PDG maneuvers. We show how the resulting formulation can be stated as a nonconvex trajectory optimization problem as in section 1.3.1, and how the problem can be solved using the CT-SCvx algorithm as in section 1.3.2.

2.4.1 Pointing Events as State-Triggered Constraints

The constraint set highlighted in equations (2.10a) and (2.12) enables guaranteed viewplanning of a set-based circular region of interest $\mathcal{Z}_{\text{circ}}$, however in traditional mission operations, these visibility constraints do not need to be applied at all times $t \in [0, t_f]$. Moreover, attempting to do so may lead to infeasible problems, motivating the importance of state-triggered constraints (STCs), which are constraints triggered conditionally based off an evaluated criterion as a function of state $x(t)$. We consider the compound STC formulation introduced in [129], wherein $g(\cdot)$ is a trigger function and $c^j(\cdot), j \in [n_c]$ are a series

of constraint functions (with n_c being the number of constraints to conditionally apply), by enforcing the condition

$$g(x(t)) \leq 0 \implies \bigwedge_{j=1}^{n_c} c^j(x(t)) \leq 0. \quad (2.42)$$

In this section, we propose an extension to the single-crossing trigger constraint set formulated in [58]. We consider a number of events n_{event} where, for each event $j \in [n_{\text{event}}]$, we have an event activation trigger g_j^+ and deactivation trigger g_j^- , with associated trigger times t_j^+ and t_j^- . We further require that these trigger conditions be strictly monotonic¹ and feasible, such that

$$0 \leq t_j^+ < t_j^- \leq t_f. \quad (2.43)$$

Then, we can impose the STC constraint set $\forall j \in [n_{\text{event}}]$ as

$$g_j^+(x(t)) \geq 0, \quad \forall t \in [0, t_j^+], \quad (2.44a)$$

$$g_j^+(x(t)) = 0, \quad t = t_j^+, \quad (2.44b)$$

$$g_j^+(x(t)) \leq 0, \quad \forall t \in [t_j^+, t_j^-], \quad (2.44c)$$

$$c^1(x(t)) \leq 0, \quad \forall t \in [t_j^+, t_j^-], \quad (2.44d)$$

$$\vdots$$

$$c^{n_c}(x(t)) \leq 0, \quad \forall t \in [t_j^+, t_j^-], \quad (2.44e)$$

$$g_j^-(x(t)) \geq 0, \quad \forall t \in [t_j^-, t_f], \quad (2.44f)$$

$$g_j^-(x(t)) = 0, \quad t = t_j^-, \quad (2.44g)$$

$$g_j^-(x(t)) \leq 0, \quad \forall t \in [t_j^-, t_f]. \quad (2.44h)$$

For our problem, we will consider n_{scan} separate terrain scanning events, with the two aforementioned constraints enforced for each event corresponding to a region of interest

¹We note that this condition is typically acceptable in the powered descent context when considering a trigger function such as altitude, which is typically expected to be monotonically decreasing with time. This approach is also exact in time, removing intersample constraint violations observed with previous methods. See [58] for a more in-depth explanation.

with centroid and radius $(z_{\mathcal{I}}^j, \rho^j)$, denoted as

$$c^1(x(t)) = \|z_{\mathcal{I}}^j - r_{\mathcal{I}}(t)\| - \frac{(z_{\mathcal{I}}^j - r_{\mathcal{I}}(t))^{\top} \mathcal{R}_{\mathcal{I} \rightarrow \mathcal{B}}^{\top}(t) b_{\mathcal{B}}}{\cos(\beta - \theta^j(t))}, \quad (2.45a)$$

$$c^2(x(t)) = \frac{\rho^j}{\sin \beta} - \|z_{\mathcal{I}}^j - r_{\mathcal{I}}(t)\| - \epsilon_{\rho}, \quad (2.45b)$$

where

$$\theta^j(t) = \arcsin \left(\frac{\rho^j}{\|z_{\mathcal{I}}^j - r_{\mathcal{I}}(t)\|} \right). \quad (2.46)$$

As a shorthand, we will denote the application of triggers in equations (2.44a)-(2.44c) and (2.44f)-(2.44h) with the compound constraint $\mathcal{T}(g_j^+, g_j^-, t_j^+, t_j^-) \leq 0$. To maintain consistency with the CT-SCvx algorithm, we must impose all of these constraints pointwise by selecting a subset of knot points to impose them on a normalized discrete-time grid $\tau_K \subset [0, 1]$. For each j , we choose τ_j^+, τ_j^- and impose the constraint $\mathcal{T}(g_j^+, g_j^-, \tau_j^+, \tau_j^-) \leq 0$ as pointwise equality and inequality constraints. Notably, equations (2.45a)-(2.45b) represent nonconvex functions which must be linearized.

2.4.2 Other State & Control Path Constraints

The remainder of constraints considered in this specific problem definition are reasonably standard for PDG applications. In terms of state constraints, we first consider bounds on velocity terms. Moreover, translational velocity, $v_{\mathcal{I}}(t)$, and angular velocity, $\omega_{\mathcal{B}}(t)$, are upper bounded by terms $v_{\max}$ and $\omega_{\max}$, respectively, as

$$\|v_{\mathcal{I}}(t)\| \leq v_{\max}, \quad (2.47a)$$

$$\|\omega_{\mathcal{B}}(t)\| \leq \omega_{\max}. \quad (2.47b)$$

Additionally, a minimum altitude constraint is specified to ensure the vehicle does not fly below the terrain (alternatively, a glide-slope constraint, as used in [5], could also be considered), stated as

$$e_z^{\top} r_{\mathcal{I}}(t) \geq 0. \quad (2.48)$$

In terms of control constraints, we consider convex SOC constraints on both thrust, $T_{\mathcal{B}}(t)$, and torque, $M_{\mathcal{B}}(t)$, with respect to quantities $T_{\max}$ and $M_{\max}$, respectively, as

$$\|T_{\mathcal{B}}(t)\| \leq T_{\max}, \quad (2.49a)$$

$$\|M_{\mathcal{B}}(t)\| \leq M_{\max}. \quad (2.49b)$$

We additionally consider a lower bound on thrust to account for standard engine minimum throttling limitations with quantity $T_{\min}$ as

$$T_{\min} \leq \|T_{\mathcal{B}}(t)\|. \quad (2.50)$$

Notably, this constraint is nonconvex. Finally, if we opt to study a non-gimbaled engine configuration, we must ensure the vehicle's thrust is constrained to the body frame $+x$ axis. To do this, we must further constrain the thrust as

$$e_x^\top T_{\mathcal{B}}(t) \geq 0, \quad (2.51a)$$

$$e_y^\top T_{\mathcal{B}}(t) = 0, \quad (2.51b)$$

$$e_z^\top T_{\mathcal{B}}(t) = 0, \quad (2.51c)$$

where $\{e_x, e_y, e_z\}$ represent the standard basis in $\mathbb{E}^3$ space. Going forwards, we note that these constraints can be imposed both pointwise and continuously using the CTCS technique mentioned in section 1.3.2; we will opt for pointwise constraint specification in the numerical results.

2.4.3 Boundary Conditions

In terms of boundary conditions, each state term must be handled separately. The initial mass of the vehicle is set to be the mass at the beginning of the maneuver, $m_0 \in \mathbb{R}$, whereas the terminal mass is left as a free variable, lower-bounded by the minimum allowable

remaining mass at the end of the maneuver, $m_{\min}$, as

$$m(0) = m_0, \quad (2.52a)$$

$$m(t_f) \geq m_{\min}. \quad (2.52b)$$

Position, velocity and angular velocity are equality-constrained at both initial and terminal times as

$$r_{\mathcal{I}}(0) = r_0, \quad r_{\mathcal{I}}(t_f) = r_f, \quad (2.53a)$$

$$v_{\mathcal{I}}(0) = v_0, \quad v_{\mathcal{I}}(t_f) = v_f, \quad (2.53b)$$

$$\omega_{\mathcal{B}}(0) = \omega_0, \quad \omega_{\mathcal{B}}(t_f) = \omega_f. \quad (2.53c)$$

The main distinction, with respect to previous literature, is in the specification of attitude boundary conditions. Due to the inclusion of the FOV constraints, we opted to leave attitude free both at initial and terminal times to enable the optimizer to discover the best attitude conditions. However, while quaternion dynamics (given by equation (2.1d)) are *norm-preserving*, they require restriction of quaternion magnitude to unity at *at least one* point in time. Thus, the nonconvex constraint must also be applied as

$$\|q_{\mathcal{I} \rightarrow \mathcal{B}}(\hat{t})\| = 1, \quad (2.54)$$

where we choose $\hat{t} = 0$ in implementation.

2.4.4 Reference Trajectory Initialization

As was mentioned in section 1.3.2, the CT-SCvx algorithm requires an input reference trajectory to serve as the initial guess (linearization point for the first iteration of the algorithm). Traditionally, linear interpolation has been used for this class of problems, e.g. as in [124]. However, we present a new approach to producing an accurate initial guess for this specific problem structure. In practice, this new initialization has shown to produce stronger convergence properties, and is – in many cases – necessary to converge to a solution at all. We start by using the *lossless convexification* approach introduced in [4–6, 11] to produce

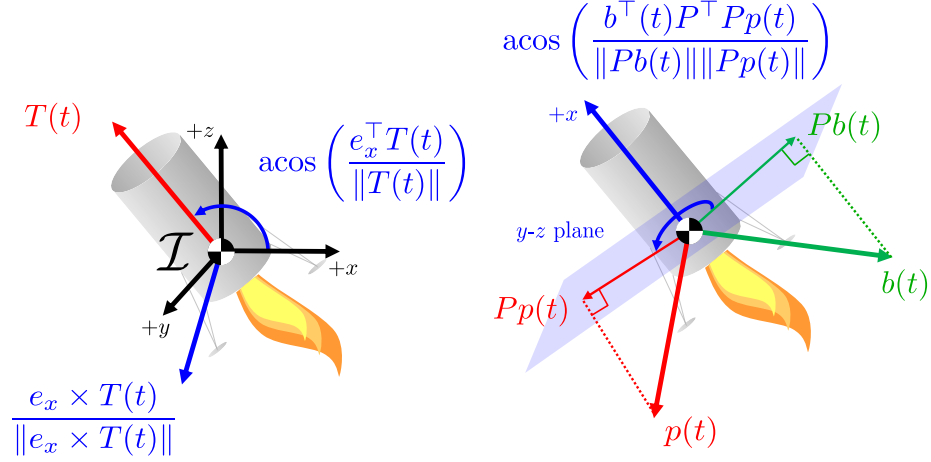


Figure 2.3: This diagram illustrates the necessary steps to derive a proximal 6-DOF attitude profile from a 3-DOF solution, with the **tilt restriction** on the left, and **roll restriction** on the right.

a 3-DOF powered descent guidance solution. That is to say, we encode all aforementioned constraints on states $m(t)$, $r_{\mathcal{I}}(t)$, $v_{\mathcal{I}}(t)$ and control $T_{\mathcal{I}}(t)$ (defined in the inertial frame rather than body-fixed frame), with the introduction of a lossless convexification constraint to replace (2.50). Notably, this does not include the scanning STC constraints, as these must be imposed on the attitude $q_{\mathcal{I} \rightarrow \mathcal{B}}(t)$. Additionally, while the lossless convexification formulation is *fixed-final-time*, a higher-level line-search can be implemented to find a solution paired with an optimal time-of-flight.

Next, we would like to convert this 3-DOF problem into a *proximal* 6-DOF problem, starting with the attitude of the vehicle parameterized by $q_{\mathcal{I} \rightarrow \mathcal{B}}(t)$. A two-step process can be used to define the body frame $\mathcal{B}$ relative to the inertial frame $\mathcal{I}$ in the 3-DOF context. We will consider the construction of a unit quaternion q by a rotation angle-axis pair $(\theta, \hat{n})$, where $q \triangleq [\cos(\theta/2), \sin(\theta/2)\hat{n}^\top]^\top$.

1. Tilt restriction. We start by considering an intermediate frame $\mathcal{B}'$, where we apply a *shortest-arc rotation* from inertial $+x$ to $T_{\mathcal{I}}(t)$, hence constraining the vehicle's $+x$ (tilt)

axis. Then, we can define $q_{\mathcal{I} \rightarrow \mathcal{B}'}(t)$ with angle-axis pair

$$\theta_{\mathcal{I} \rightarrow \mathcal{B}'} = \arccos \left(\frac{e_x^\top T_{\mathcal{I}}(t)}{\|T_{\mathcal{I}}(t)\|} \right), \quad (2.55a)$$

$$\hat{n}_{\mathcal{I} \rightarrow \mathcal{B}'} = \frac{[e_x \times] T_{\mathcal{I}}(t)}{\|[e_x \times] T_{\mathcal{I}}(t)\|}. \quad (2.55b)$$

2. Roll restriction. Next, in the frame $\mathcal{B}'$, we would like to roll the vehicle about the $+x$ axis such that the boresight vector $b_{\mathcal{B}'}(t)$ is as closely-aligned with a desired pointing direction $p_{\mathcal{B}'}(t)$ as possible. To do this, we can project $b_{\mathcal{B}'}(t)$ and $p_{\mathcal{B}'}(t)$ to the $\mathcal{B}'$ y - z plane with a matrix P , defined as

$$P \triangleq \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (2.56)$$

Consequently, the closest alignment is achieved when the boresight projection $Pb_{\mathcal{B}'}(t)$ is *aligned* with desired projection $Pp_{\mathcal{B}'}(t)$. This secondary transformation will result in the final proximal body-frame definition $\mathcal{B}$, defined by $q_{\mathcal{B}' \rightarrow \mathcal{B}}(t)$ with angle-axis pair

$$\theta_{\mathcal{B}' \rightarrow \mathcal{B}} = \arccos \left(\frac{b_{\mathcal{B}'}^\top(t) P^T P p_{\mathcal{B}'}(t)}{\|Pb_{\mathcal{B}'}(t)\| \|Pp_{\mathcal{B}'}(t)\|} \right), \quad (2.57a)$$

$$\hat{n}_{\mathcal{B}' \rightarrow \mathcal{B}} = e_x. \quad (2.57b)$$

Then, we have that the proximal attitude realization $q_{\mathcal{I} \rightarrow \mathcal{B}}(t) = q_{\mathcal{I} \rightarrow \mathcal{B}'}(t) \otimes q_{\mathcal{B}' \rightarrow \mathcal{B}}(t)$. An illustration of these transformations is displayed in Fig. 2.3. An important followup question can be asked: *how do we obtain $p_{\mathcal{B}'}(t)$?* With consideration of aforementioned pointing events, we can apply the following logic:

- **During an event:** if $t \in [t_j^-, t_j^+]$ for any $j \in [n_{\text{scan}}]$, simply point towards the centroid of the event scanning region, such that $p_{\mathcal{B}'}(t) = z_{\mathcal{B}'}^j - r_{\mathcal{B}'}(t)$.
- **Between events:** If $t \in (t_j^-, t_{j+1}^+)$ for any $j \in [n_{\text{scan}} - 1]$, perform a spherical interpolation on $p_{\mathcal{B}'}(t)$ between $p_{\mathcal{B}'}(t_j^-)$ and $p_{\mathcal{B}'}(t_{j+1}^+)$ with interpolation fraction $f = (t - t_j^-)/(t_{j+1}^+ - t_j^-)$.
- **Before the first event:** If $t \in [0, t_1^+)$, hold first event pointing direction $p_{\mathcal{B}'}(t) =$

$$p_{\mathcal{B}'}(t_1^+).$$

- **After the last event:** If $t \in (t_{n_{\text{scan}}}^-, t_f]$, hold last event pointing direction $p_{\mathcal{B}'}(t) = p_{\mathcal{B}'}(t_{n_{\text{scan}}}^-)$.

We note that this approach assumes events do not overlap, which can be enforced by appropriate construction of the trigger functions and times. From this, we can use equations (2.1d) and (2.1e) (or their discrete-time counterparts in the case of initial guess generation for SCP) to compute the angular velocity profile $\omega_{\mathcal{B}}(t)$. Consequently, equation (2.1e) can be used to obtain the moment profile $M_{\mathcal{B}}(t)$ as well. However, these can also be interpolated between boundary conditions or set to zero; the quaternion attitude profile is most critical to obtaining a reliable initial guess for this problem class.

2.5 Numerical Results

In this section, we introduce a practical scenario in regards to perception-based exploration of a planetary terrain environment during a powered descent guidance maneuver. Specifically, we consider a mission objective wherein a reference vehicle is commanded to scan different terrain regions while descending, with each scanning event triggered in a different altitude envelope above ground. In the following subsections, we will discuss parameterization of this scenario and assess simulation results.

2.5.1 Scenario Parameterization

The aforementioned scenario will consider three separate (non-overlapping) terrain scanning events $n_{\text{scan}} = 3$, with each trigger based strictly on the altitude of the vehicle. Moreover, for an event $j \in [n_{\text{scan}}]$, we consider the constraint functions

$$\begin{aligned} g_j^+(r_{\mathcal{I}}(t)) &= e_z^\top r_{\mathcal{I}}(t) - h_j^+, \\ g_j^-(r_{\mathcal{I}}(t)) &= e_z^\top r_{\mathcal{I}}(t) - h_j^-. \end{aligned}$$

Table 2.1: Terrain scanning scenario parameters.

Parameter	Value	Parameter	Value
m_0	1500 kg	$m_{\min}$	750 kg
r_0	$[4000, 600, 3000]^\top$ m	r_f	$[0, 0, 0]^\top$ m
v_0	$[-60, 30, -30]^\top$ m/s	v_f	$[0, 0, 0]^\top$ m/s
ω_0	$[0, 0, 0]^\top$ °/s	ω_f	$[0, 0, 0]^\top$ °/s
$T_{\min}$	600 N	$v_{\max}$	90 m/s
$T_{\max}$	3000 N	$\omega_{\max}$	5°/s
$M_{\max}$	50 N · m		
I_{sp}	300 s	$J_{\mathcal{B}}$	$\text{diag}([0.6, 4.2, 4.2])$ kg · m ²
g_0	9.81 m/s ²	β	20°
$g_{\mathcal{I}}$	$-[0, 0, 1.625]^\top$ m/s ²	$b_{\mathcal{B}}$	$[-\sqrt{3}/2, 0, 0.5]^\top$
N	30	w_o	1e-1
ϵ_η	1e-6	w_η	1e2
ϵ_ν	1e-6	w_ν	1e3

Under this definition, the following terrain scanning STC parameters are considered:

$$\begin{aligned}
\textbf{Scan 1:} \quad & h_1^+ = 2500 \text{ m}, \quad h_1^- = 2100 \text{ m}, \quad \rho^1 = 500 \text{ m}, \quad z_{\mathcal{I}}^1 = [3000, 1000, 0]^\top \text{ m}, \\
\textbf{Scan 2:} \quad & h_2^+ = 1700 \text{ m}, \quad h_2^- = 1300 \text{ m}, \quad \rho^2 = 300 \text{ m}, \quad z_{\mathcal{I}}^2 = [2000, -1000, 0]^\top \text{ m}, \\
\textbf{Scan 3:} \quad & h_3^+ = 900 \text{ m}, \quad h_3^- = 500 \text{ m}, \quad \rho^3 = 100 \text{ m}, \quad z_{\mathcal{I}}^3 = [1000, 0, 0]^\top \text{ m}.
\end{aligned}$$

Additionally, the trigger indices k_j^+ and k_j^- , $j \in [n_{\text{scan}}]$, associated with chosen normalized trigger times $\tau_j^- = \tau_{k_j^-}$ and $\tau_j^+ = \tau_{k_j^+}$, are chosen to be uniformly distributed across the interval $k \in [1, N]$, but not such that the outer events start at the endpoints of the interval. Moreover, we can construct a vector of indices $K \in \mathbb{R}^{N+2}$ which stores uniform linear interpolations between 1 and N over $N+2$ elements (with rounding to the nearest integer for each element), from which we can assign the values:

$$\{K_2, \dots, K_{N+1}\} \rightarrow \{k_1^+, k_1^-, \dots, k_{n_{\text{scan}}}^+, k_{n_{\text{scan}}}^-\}.$$

This trigger node allocation heuristic has been (empirically) proven to work well for many

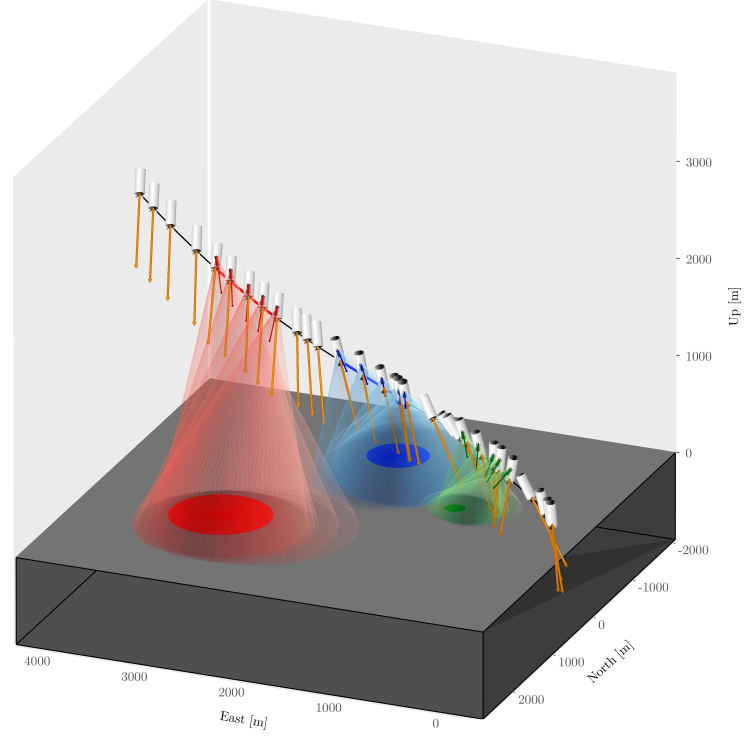


Figure 2.4: This figure presents a visualization of the vehicle trajectory and terrain scan events in regards to the candidate powered descent guidance scenario. The vehicle is depicted with a gray cylindrical body at each solver knot point, along with an **orange** arrow corresponding to the (normalized) thrust vector. Terrain scanning events 1-3 are colored **red**, **blue**, and **green**, respectively, with the boresight pointing vector, FOV cone and target region color-coded. Additionally, the simulated trajectory is displayed in black, with colored segments during the activation of each scanning event.

scenario variations in this problem class. We also consider a minimum-cumulative-thrust objective for this problem, such that

$$J(\tilde{x}, \tilde{u}) = \int_{t_0}^{t_f} \|T_{\mathcal{B}}(t)\| dt.$$

All other scenario parameters are presented in Table 2.1.

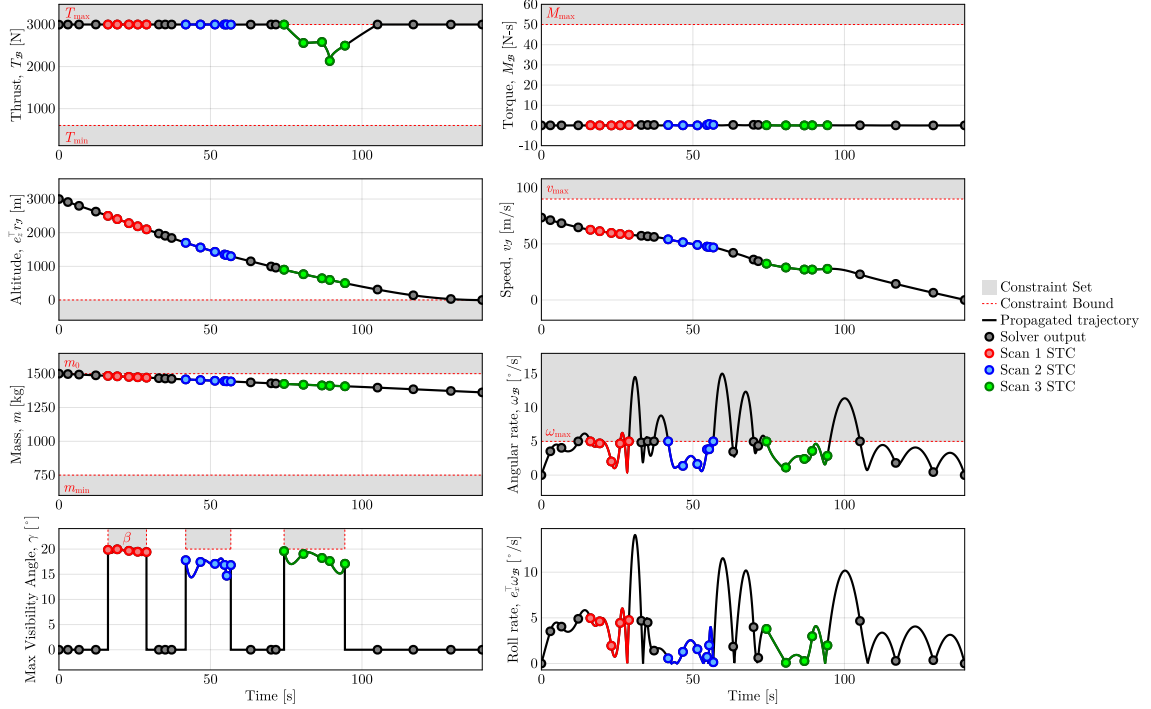


Figure 2.5: This figure presents various time signals of interest in regards to the candidate powered descent guidance scenario introduced, with color-coded scan events and constraint bounds.

2.5.2 Analysis & Discussion

From Fig. 2.4, a visualization of the vehicle maneuver can be seen, including the solver output at each knot point, as well as the simulated continuous-time solution (using fourth order Runge-Kutta integration). The solution was obtained on a 2023 Razer Blade 15” laptop² using the JuMP (Julia programming language) parser [35] and open-source ECOS second-order cone program solver [33], achieving a net solve time of 4.35 seconds over 19 prox-linear iterations with $N = 30$ nodes. From the aforementioned figure, it can be visually observed that all scanning targets are kept within view (as shown by the overlaid conic projections onto the surface). To validate that this is indeed the case, we can define the *maximum visibility angle*, γ , which records the maximum angle between the boresight vector

²This laptop has an AMD Ryzen 9 6900HX CPU with 3301 Mhz clock speed and 8 cores.

$b_{\mathcal{I}}(t)$ and all feasible³ relative directions to points within the target set $\mathcal{Z}_{\text{circ}}$. More formally, this term is defined as

$$\gamma = \max \left\{ \text{acos} \left(\frac{(\tilde{z} - r_{\mathcal{I}}(t))^{\top} b_{\mathcal{I}}}{\|\tilde{z} - r_{\mathcal{I}}(t)\|} \right), \forall \tilde{z} \in \mathcal{Z}_{\text{circ}} \right\}.$$

From Fig. 2.5, we can observe this signal $\gamma(t)$ for all three scanning events, and see that it falls below the maximum threshold, which is equivalent to the FOV half-angle, β . It would appear that the constraint is activated in the first scan, however, upon closer inspection, there is indeed a small gap explained by the conservatism of the $\mathcal{Z}_{\text{ball}}$ set approximation, as discussed in section 2.2. All other signals are validated to satisfy their required constraints at the discretized knot points, however in some cases, we can observe non-trivial intersample constraint violation between knot points. Particularly in regards to the angular rate $\omega_{\mathcal{B}}(t)$, intersample constraint violations between pointing events are of large violation magnitude. Upon further inspection, it can be seen that these violations are concentrated on the roll rate $e_x^{\top} \omega_{\mathcal{B}}(t)$, suggesting that the violations are a result of large roll transition requirements between scanning events. As an extension of these numerical results, we propose that testing with the CTCS specification in the CT-SCvx algorithm may allow us to remedy this concern in future developments.

³In practice, we uniformly sample points about the circumference of $\mathcal{Z}_{\text{circ}}$ to approximately compute the maximum.

Chapter 3

ACTIVE SIMULTANEOUS LOCALIZATION & MAPPING

3.1 Introduction

In the previous chapter, we introduced a collection of techniques to model direct line-of-sight to structured sets in cartesian space. However, a series of important questions still linger:

1. How do we determine *how long* to look at a point or region, and in *what order*?
Moreover, how should we parameterize the aforementioned state-triggered constraints?
2. How can we relate these decisions to the core objectives we want to satisfy during task planning, such as meeting a 3σ uncertainty requirement at the terminal condition?

In terms of state estimation, the problem of jointly estimating both the intrinsic pose of the perceiving vehicle, and the locations of map elements, has been historically addressed as the *simultaneous localization and mapping (SLAM)* problem [21]. More recently, the related *active SLAM* problem has been introduced to address the combination of SLAM state estimation techniques with forward-horizon (predictive) trajectory planning [104]. In this chapter, the state-of-the-art in active SLAM techniques is analyzed, and a novel modeling technique is presented (ACT-SLAM) based on the work of [16].

3.1.1 Related Literature

We begin by detailing the various forms of the SLAM problem and their relation to each other from a discrete-time perspective. We consider a vehicle state at a time t_k denoted $x_k \in \mathbb{R}^{n_x}$, with a corresponding finite set of map elements denoted as $\{m_k^j\}_{j=1}^{n_\ell}$ with n_ℓ denoting the number of map elements. These map elements are often referred to as *landmarks* and

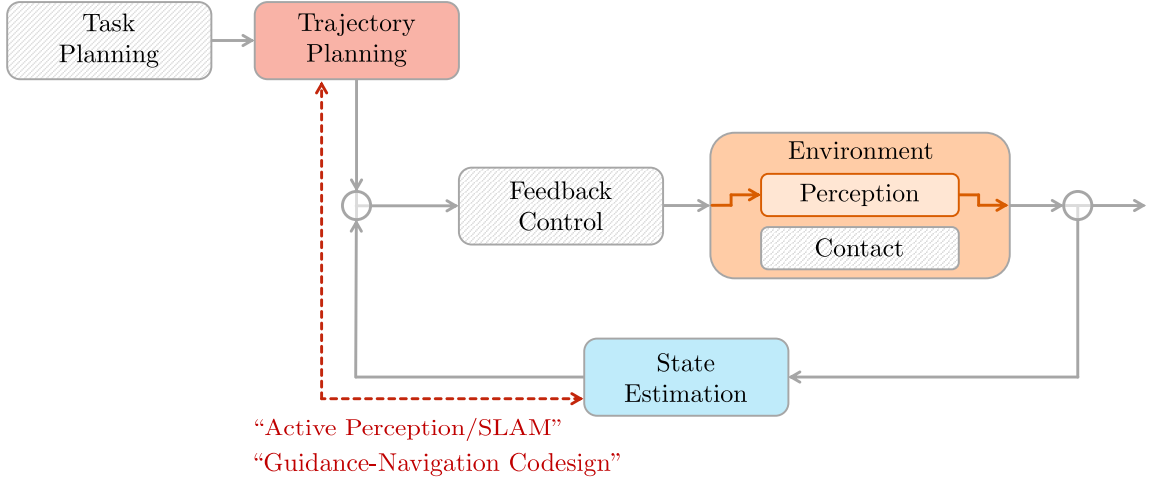


Figure 3.1: The autonomy module architecture framed in the context of the active SLAM problem (otherwise referred to as active perception and guidance-navigation codesign in the aerospace context).

expressed with position information in the context of spatial perception (such that m_k^j denotes the position of landmark j at time t_k). We can then denote a joint system state $z_k \triangleq (x_k, m_k)$ where $m_k \triangleq (m_k^1, \dots, m_k^{n_\ell})$ to concisely express both the vehicle and landmark states. In addition, the system is also capable of taking perceptive measurements, denoted as $y_k \in \mathbb{R}^{n_y}$. Then, under the Markov assumption [132], we can estimate the posterior distribution on z_k conditioned on the most recent measurement y_k and previous state z_{k-1} as

$$p(z_k \mid z_{k-1}, y_k). \quad (3.1)$$

Estimating this posterior in isolation is often referred to as the *online SLAM* problem. A common extension of this modeling technique concerns jointly estimating a collection of posteriors over some backwards-horizon history of measurements over sample count N , where the posterior at each time-step acts as a prior for the following time-step. This class of techniques is often referred to as *offline* or *batch SLAM*, where the batch posterior is

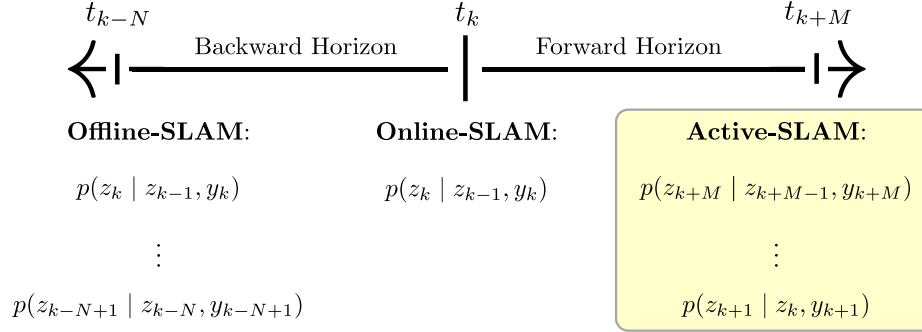


Figure 3.2: Comparison of the estimation process between different SLAM problems.

expressed as

$$p(z_k | z_{k-N:k-1}, y_{k-N+1:k}) = \left(\prod_{i=k-N+1}^k p(z_i | z_{i-1}, y_i) \right) p(z_{k-N}), \quad (3.2)$$

where $p(z_{k-N})$ corresponds to the batch prior (oldest measurement in a sliding window, for example). This approach is commonly chosen over online SLAM to produce a more accurate posterior estimation and apply outlier robustness measures. Similarly, the *active SLAM* problem can be expressed over a forward horizon over sample count M with a similar batch approach to estimating the posterior

$$p(z_{k+M} | z_{k:k+M-1}, y_{k+1:k+M}) = \left(\prod_{i=k+1}^{k+M} p(z_i | z_{i-1}, y_i) \right) p(z_k), \quad (3.3)$$

where $p(z_k)$ still corresponds to a batch prior, in this case the known state distribution at the beginning of the planning maneuver. A visual comparison between these problems is provided in figure 3.2. The remainder of this section will focus on expressing solutions to the active SLAM problem while borrowing from modeling approaches expressed with online and offline SLAM.

Remark 3.1. While the prior SLAM problem statements are presented in discrete-time form, they can be readily extended towards continuous-time modeling by taking $\Delta t \rightarrow 0$ where $\Delta t = (t_k - t_{k-1})$. In the case where measurements must be treated as quantized (online

and offline SLAM), this modeling approach may present unwarranted additional complexity, however in the active SLAM context, measurements may be reasonably approximated with a continuous-time model, as will be seen in the following subsection. Moving forward, all active SLAM problems will be posed in continuous-time.

In the aerospace parlance, we may often-times refer to the problem of trajectory planning as *guidance*, and to the problem of state estimation as *navigation*. As illustrated by figure 3.1, we can then consider the problem of active SLAM to be akin to the problem of guidance-navigation co-design, where we seek to jointly satisfy both guidance and navigation objectives simultaneously. We will now revisit the powered descent guidance (PDG) maneuver mentioned in the previous chapter. Beyond the pointing event/state-triggered constraint approach mentioned previously, the problem of guidance-navigation co-design has been addressed more directly through a number of approaches. Robust Trajectory Optimization (RTO) [22] has been proposed to optimize entry, descent and landing (EDL) maneuvers through the use of a genetic algorithm as the (sampling-based) optimizer over covariance terms from a forward propagation (simulation) of linear covariance analysis [43,44]. Although this method provides a capable co-design approach for upstream phases of EDL maneuvers, the sampling-based nature of the algorithm limits it to optimization over only a few variables (in the case of Calkins et al. [22], two variables). For the downstream PDG maneuvering phase, substantially more resolution in terms of optimization variables is required to find a solution which can satisfy all path constraints when using numerical optimal control techniques such as those aforementioned¹. This would be prohibitively expensive or even intractable to optimize with sampling-based methods, motivating a new approach for co-design under highly-constrained mission scenarios. Ticozzi et al. [133] present a factor graph-based technique for guidance-navigation co-design in the context of perceptive spacecraft maneuvers by directly solving the active SLAM problem, however are still restricted to the same limitations of zeroth-order (sampling-based) optimization.

¹For example, a trajectory optimization with 10 discrete knot points and 6 control elements (thrust and torque terms), parameterized with a first-order hold, corresponds to $10 \times 6 = 60$ variables that must be optimized for at a minimum.

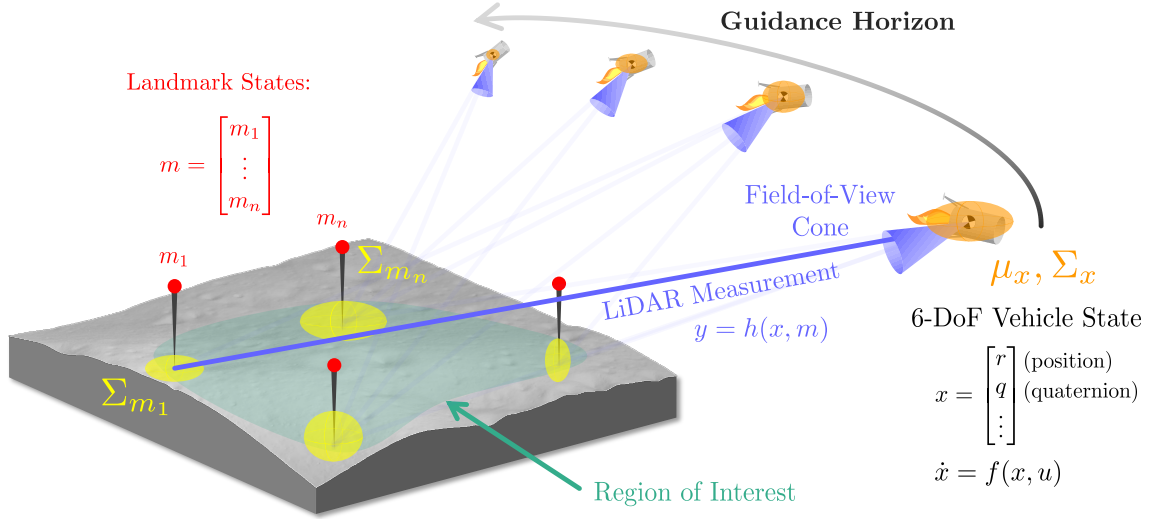


Figure 3.3: Illustration demonstrating the modeling approach behind the ACT-SLAM technique applied to the active sensing problem.

Other techniques have been developed that attempt to embed guidance-navigation co-design directly within gradient-based trajectory optimization templates. This includes a map covariance minimization technique for PDG scenarios [54] and discrete-time active mapping without path constraints applied to robotic locomotion in closed environments [66, 67]. In this work, we attempt to extend these existing developments to a point of further generality that makes the approach amenable to the canonical continuous-time optimal control problem (CT-OCP) template given by problem (1.1). We focus our efforts specifically on application to the PDG problem with a perceptive range-based sensor (e.g. LiDAR) to form the measurement model for estimation, where the system relies on these measurements to produce an accurate navigational solution in terms of its own state and the terrain environment. Guidance-navigation co-design for perceptive maneuvers is notably difficult due to the tight positional and attitude restrictions necessary to perform pointing maneuvers while landing, making them a suitable stress test for the proposed approach. We refer to this problem as the *active sensing* problem going forwards.

3.1.2 Contributions

In this chapter, we introduce a novel, computationally-tractable technique towards solving the general-form active sensing problem as an instantiation of active SLAM. This technique provides a synthesis of optimal estimation and optimal control properties while being embedded directly into a CT-OCF problem template, making it an appropriate candidate for modern PDG maneuver design relative to pre-existing approaches in the literature. This modeling approach considers a vehicle state $x(t) \in \mathbb{R}^{n_x}$ and corresponding control input $u(t) \in \mathbb{R}^{n_u}$, where the vehicle state must necessarily contain *at least* a six degree-of-freedom (6-DoF) rigid body representation [126] to model LiDAR measurements, typically parameterized as an inertial position $r(t) \in \mathbb{R}^3$ and inertial-to-body quaternion $q(t) \in \mathcal{S}^3$, where $\mathcal{S}^3$ represents the unit 3-sphere manifold in $\mathbb{R}^4$. Additionally, a *region of interest* is identified with respect to the ground terrain, and a set of positional landmarks are characterized within this region as $m_j(t) \in \mathbb{R}^3$, $j = 1, \dots, n_\ell$, with the full map state represented as $m(t) \triangleq (m_1(t), \dots, m_{n_\ell}(t))$. The vehicle and map states are consequently modeled as multivariate Gaussian random variables such that $x(t) \sim \mathcal{N}(\mu_x(t), \Sigma_x(t))$ and $m(t) \sim \mathcal{N}(\mu_m(t), \Sigma_m(t))$, respectively, with cross-correlation terms $\Sigma_{xm}(t)$. LiDAR measurements are used to directly correlate and reduce both vehicle uncertainty $\Sigma_x(t)$ and mapping uncertainty $\Sigma_m(t)$. A free-final-time trajectory optimization problem is modeled under decision variables $d(t) \triangleq (\mu_x(t), \mu_m(t), \Sigma_x(t), \Sigma_m(t), \Sigma_{xm}(t), u(t))$ over time span $t \in [0, t_f]$, where $\mu_x(t), u(t)$ must satisfy traditional PDG path constraints as part of the guidance requirements², and $\Sigma_x(t), \Sigma_m(t)$ must satisfy 3σ error budget constraints as part of the navigation requirements. The optimization output $\mu_x(t), u(t)$ can then serve as the nominal trajectory, which can be computed either offline (in a trajectory design/shaping context), or online (recursively revising the guidance solution based on new priors or a shifting region of interest). This explicit characterization of covariance propagation in trajectory optimization relates our technique to the well known class of methods known as *covariance steering* [36, 95, 113], with the additional characterization of measurement covariance contributions (as opposed to purely process covariance) as will be explored in the following

²Examples of path constraints typical to PDG problems include bounds on thrust and torque control terms and glideslope constraints.

sections.

The proposed approach is labeled **Active Continuous-Time Simultaneous Localization and Mapping** (ACT-SLAM), which consists of the following key components that form its core contribution to the literature:

1. **Actively** optimizes over a Gaussian trajectory distribution over the forward time horizon through explicit mean & covariance representation in a continuous-time optimal control problem (CT-OCP). Specifically, the continuous-time successive convexification (CT-SCvx) [37] technique is chosen for real-time nonconvex trajectory optimization.
2. Uses a **continuous-time** optimal estimation model—the *extended Kalman-Bucy filter* [114]—to embed the aforementioned Gaussian trajectory distribution as a set of differential equations within a CT-OCP template.
3. Models the joint relationships between uncertainty in the vehicle’s position and uncertainty in the mapped terrain environment through the use of **simultaneous localization and mapping (SLAM)** [122] modeling approaches.

A graphical illustration of the ACT-SLAM approach is provided in figure 3.3. The remainder of this chapter is structured as follows. Section 3.2 presents a continuously-differentiable Gaussian filtering model posed in continuous-time. Section 3.3 uses this filtering model to propose the ACT-SLAM optimization problem solved with the CT-SCvx algorithm. Section 3.4 provides high-fidelity simulation, model validation and literature comparison results.

3.2 Continuous-Time Differentiable Filtering

We first concern ourselves with deriving a continuously-differentiable representation of an optimal measurement filtering model representing a solution to equation (3.3), enabling us to embed a Gaussian trajectory distribution in the optimization dynamics. This effort can be considered the state estimation (*navigation*) component of the aforementioned co-design process. We start by defining a continuous-time stochastic state-space system, parameter-

ized with a system state vector $z(t) \in \mathbb{R}^{n_z}$ and observation vector $y(t) \in \mathbb{R}^{n_y}$, such that

$$\dot{z}(t) = f(z(t), u(t)) + w(t), \quad (3.4)$$

$$y(t) = h(z(t)) + v(t), \quad (3.5)$$

where $f : \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_z}$ corresponds to the process dynamics, $h : \mathbb{R}^{n_z} \rightarrow \mathbb{R}^{n_y}$ corresponds to the observation model, and $w(t) \in \mathbb{R}^{n_z}, v(t) \in \mathbb{R}^{n_y}$ correspond to zero-mean temporally-uncorrelated Gaussian processes for the process and observation models, with covariances $W(t) \in \mathbb{R}^{n_z \times n_z}$ and $V(t) \in \mathbb{R}^{n_y \times n_y}$ defined, for a fixed $t \in \mathbb{R}$ and for any $\hat{t} \in \mathbb{R}$, as

$$\mathbb{E}[w(t)w(\hat{t})^\top] = W(t)\delta(t - \hat{t}), \quad (3.6)$$

$$\mathbb{E}[v(t)v(\hat{t})^\top] = V(t)\delta(t - \hat{t}), \quad (3.7)$$

where δ corresponds to the Dirac delta function. We note that, under this definition, the system state will also define a Gaussian process such that, for each fixed time $t \in \mathbb{R}$, its marginal distribution is characterized with a mean $\mu(t) \in \mathbb{R}^{n_z}$ and covariance $\Sigma(t) \in \mathbb{R}^{n_z \times n_z}$ such that

$$\mu(t) = \mathbb{E}[z(t)], \quad (3.8)$$

$$\Sigma(t) = \mathbb{E}[(z(t) - \mu(t))(z(t) - \mu(t))^\top]. \quad (3.9)$$

We remark that this notation to describe stochastic systems avoids the formalized use of stochastic differential equations, however is a common and accepted shorthand [43, 56, 86], and all following results apply without loss of generality to the formalized expression. Traditional spaceflight-grade navigation filters make use of the hybrid extended Kalman filter (hybrid-EKF), which splits the mean and covariance propagation into prediction and update steps over a time interval $t \in [t_{k-1}, t_k]$. Note that we use the shorthand $\phi_k = \phi(t_k)$ for a variable $\phi(t)$ to denote evaluation of a continuous-time variable at discrete time t_k .

The hybrid-EKF is summarized with the following system of equations:

$$(\textbf{predict}) \quad \dot{\mu}(t) = f(\mu(t), u(t)), \quad (3.10)$$

$$\dot{\Sigma}(t) = F(t)\Sigma(t) + \Sigma(t)F(t)^\top + W(t), \quad (3.11)$$

$$(\textbf{update}) \quad \mu_k = \mu_k^- + K_k(y_k - H_k\mu_k^-), \quad (3.12)$$

$$\Sigma_k = (I - K_k H_k)\Sigma_k^-, \quad (3.13)$$

$$K_k = \Sigma_k^- H_k^\top (H_k \Sigma_k^- H_k^\top + V_k)^{-1}, \quad (3.14)$$

where $F(t), H(t)$ define measurement partials about a given reference $(\bar{z}(t), \bar{u}(t))$

$$F(t) = \left. \frac{\partial f(z(t), u(t))}{\partial z(t)} \right|_{(\bar{z}(t), \bar{u}(t))}, \quad (3.15)$$

$$H(t) = \left. \frac{\partial h(z(t))}{\partial z(t)} \right|_{\bar{z}(t)}. \quad (3.16)$$

This technique involves integrating the prediction-side system of differential equations, then performing discrete updates at a frequency that is notionally aligned with a measurement acquisition rate. Specifically, equations (3.10)-(3.11) are integrated across $t \in [t_{k-1}, t_k]$ from μ_{k-1}, Σ_{k-1} to produce μ_k^-, Σ_k^- , and a discrete measurement update at time t_k is applied to μ_k^-, Σ_k^- to produce μ_k, Σ_k . This approach has been explored for both online, embedded implementations [42] and offline linear covariance analysis [144, 146]. Consider a concatenated filtering state $\zeta(t) \triangleq (\mu(t), \text{vec}(\Sigma(t))) \in \mathbb{R}^{n_\zeta}$, where $n_\zeta = n_z + n_z^2$ and define mapping $\tilde{g} : \mathbb{R}^{n_\zeta \times n_y \times n_u} \rightarrow \mathbb{R}^{n_\zeta}$ such that $\dot{\zeta} = \tilde{g}(\zeta, y, u)$ satisfies equations (3.10)-(3.14). Due to the *discrete* update equations (3.12)-(3.14), this mapping is defined as a *hybrid dynamical system* [45]. While this system is considered sound within the context of state estimation, hybrid dynamical systems present substantial numerical complexity when embedded within a CT-OCP problem formulation [102], motivating an alternative formulation which removes the discrete-time update step.

To address this concern, we can instead consider an alternative formulation for the Kalman filter which is expressed entirely as a set of continuous-time differential equations, known as the *extended Kalman-Bucy filter* (EKBF) [114]. Moreover, this filter contains the following

system of differential equations to couple the aforementioned predict and update steps as

$$\dot{\mu}(t) = f(\mu(t), u(t)) + K(t)(y(t) - h(\mu(t))), \quad (3.17a)$$

$$\dot{\Sigma}(t) = F(t)\Sigma(t) + \Sigma(t)F(t)^\top + W(t) - K(t)H(t)\Sigma(t), \quad (3.17b)$$

$$K(t) = \Sigma(t)H(t)^\top V(t)^{-1}. \quad (3.17c)$$

Considering the same filtering state $\zeta(t)$, we can define mapping $g : \mathbb{R}^{n_\zeta \times n_y \times n_u} \rightarrow \mathbb{R}^{n_\zeta}$ such that $\dot{\zeta} = g(\zeta, y, u)$ satisfies equations (3.17a)-(3.17c), and note that this new dynamical equation is fully continuous, thus is readily amenable to CT-OCP formulations.

3.2.1 Differentiable SLAM Modeling

Now that we have formed the continuous-time Kalman filtering model, we can relate this model to the SLAM problem, which pertains to modeling the joint correlations between the vehicle state and landmark states to concurrently reduce both. The SLAM problem [122] provides an assortment of well-established approaches towards directly parameterizing these vehicle-map relationships, and has been explored for powered descent guidance problems in a strictly estimation sense [42, 102]. One such approach that is particularly well-suited towards trajectory optimization is the EKF-SLAM approach, wherein the vehicle and landmark states are directly concatenated to form the system state, and filtered through an EKF. In this work, we extend EKF-SLAM towards a continuous-time variant, EKBF-SLAM, and assert certain formulation nuances to recover a model consistent with perceptual estimation yet tractable with numerical trajectory optimization methods.

We start by parameterizing a vehicle state $x(t) \in \mathbb{R}^{n_x}$ with a corresponding control $u(t) \in \mathbb{R}^{n_u}$, and additionally parameterize the environment with a discrete set of position landmarks $m_j(t) \in \mathbb{R}^3$, $j = 1, \dots, n_\ell$ such that the full map state is represented as $m(t) = (m_j(t))_{j=1}^{n_\ell} \in \mathbb{R}^{3n_\ell}$. Then, the system state is defined as the concatenation of the vehicle and map states, such that $z(t) = (x(t), m(t))$, which can be directly fed into the system of EKBF equations defined in (3.17a)-(3.17c). Notably, this design leads to vehicle-map partitions on the stochastic state-space model terms, where subscripts x, m index out

the vehicle and mapping components of each term, stated as

$$\mu = \begin{bmatrix} \mu_x \\ \mu_m \end{bmatrix}, \quad \Sigma = \begin{bmatrix} \Sigma_x & \Sigma_{xm} \\ \Sigma_{xm}^\top & \Sigma_m \end{bmatrix}, \quad f = \begin{bmatrix} f_x \\ f_m \end{bmatrix}, \quad W = \begin{bmatrix} W_x & 0 \\ 0 & W_m \end{bmatrix}.$$

We show all partitions for generality, however note that for a static terrain-based environment, the map dynamics will be taken to be $f_m(\cdot) = 0$ with $W_m(\cdot) = 0$. At this point, we must assert some system assumptions and modifications to the EKBF model to relate our formulation to the SLAM problem. Notably, we must relate the vehicle and map state uncertainties through the concept of perceptive measurements in a restricted field of view. To do this, we must assert several assumptions about our vehicle model and configuration. We start by defining an inertial (world) frame $\mathcal{I}$ and six degree-of-freedom (6-DOF) rigid body frame representation $\mathcal{B}$ affixed to the vehicle. Then, the following assumptions are made:

Assumption 3.1. *The vehicle state $x(t)$ contains a 6-DOF pose representation parameterized by an inertial position $r_{\mathcal{I}}$ and inertial-to-body transformation matrix $\mathcal{R}_{\mathcal{I} \rightarrow \mathcal{B}}(t)$. We could also consider a reduced-order attitude representation, such as a quaternion, however we must be able to recover a transformation matrix via some closed-form relation. In the remainder of this subsection, we will drop the subscripts for these terms for notational convenience.*

Assumption 3.2. *The vehicle contains a restricted field-of-view (FOV) perceptive sensor that acquires range-based measurements (such as a LiDAR sensor), with a body-fixed bore-sight pointing vector b and FOV half-angle $\beta_{\max}$.*

We must further modify the definition of our measurement noise term to be state-dependent, such that $v(t) = v(z(t)) \implies V(t) = V(z(t))$. Let us first define a range-based measurement model to each landmark as

$$h_j(z(t)) = \|m_j(t) - r(t)\|_2. \quad (3.18)$$

We consequently have $n_y = n_\ell$ measurements corresponding to each landmark, with the full measurement function defined as $h(z(t)) = (h_j(z(t)))_{j=1}^{n_\ell}$. We can also define a range-based

standard deviation of the measurement noise to each landmark as

$$\tilde{\sigma}_j(z(t)) = \gamma_1 \exp(\gamma_2 h_j(z(t))), \quad (3.19)$$

where γ_1, γ_2 represent hardware constants. This type of exponential noise model for range-based measurements is common practice in the modeling of LiDAR sensors, and has been validated for accuracy in the context of both spacecraft [130] and autonomous driving [141]. Because our range-based measurements require line-of-sight to the landmark, we must further define the line-of-sight set $\mathcal{F}_j$ such that $z(t) \in \mathcal{F}_j$ implies that landmark m_j is in line-of-sight of the sensor. More rigorously, we define angle $\beta(t)$ as the angle between relative distance $\Delta r_j(t) = m_j(t) - r(t)$ and $\mathcal{R}(t)^\top b$ (where application of $\mathcal{R}(t)$ is necessary to express the boresight vector in the inertial frame). Then, the objective is to find $\mathcal{F}_j$ as the set such that $\beta(t) \leq \beta_{\max}$. This leads to the definition

$$\mathcal{F}_j = \left\{ r, \mathcal{R}, m^j \in z \mid \frac{(\mathcal{R}^\top b)^\top (m^j - r)}{\|m^j - r\|_2} \geq \cos \beta_{\max} \right\}. \quad (3.20)$$

Because of the line-of-sight requirement, we must revise our noise model to be *state-dependent*, such that the noise standard deviation is (effectively) infinite when the landmark is outside the FOV of the vehicle, since no measurement can be made under this condition. This can be done with a model building on the noise definition in equation (3.19) stated as

$$\sigma_j(z(t)) = \begin{cases} \tilde{\sigma}_j(z(t)), & z(t) \in \mathcal{F}_j, \\ \infty, & \text{else.} \end{cases} \quad (3.21)$$

Moreover, we are interested in the reciprocal of the noise

$$\sigma_j^{-1}(z(t)) = \begin{cases} \tilde{\sigma}_j^{-1}(z(t)), & z(t) \in \mathcal{F}_j, \\ 0, & \text{else.} \end{cases} \quad (3.22)$$

This definition notably contains a *discontinuity* when crossing the boundary of set $\mathcal{F}_j$. As will be seen in the next section, gradient-based trajectory optimization techniques (typically)

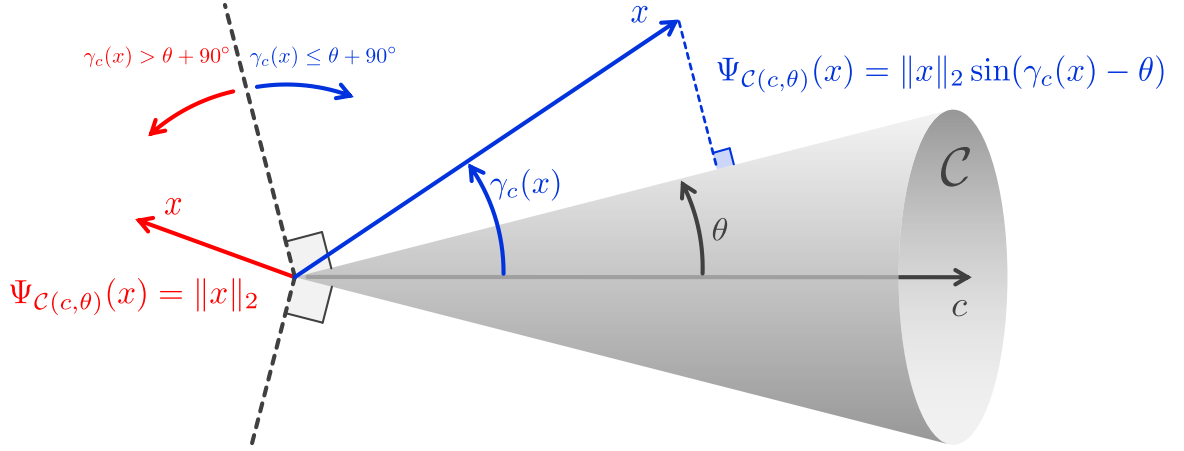


Figure 3.4: A visual interpretation of the set $\Psi_{\mathcal{C}(c,\theta)}$.

have the requirement that the dynamics model be expressed as $\mathcal{C}^1$ -differentiable. To address the discontinuity in equation (3.22), we must first establish several important definitions.

Definition 3.1 (Signed distance function). The signed distance function (SDF) of a set $\mathcal{S}$ is defined as

$$\Psi_{\mathcal{S}}(x) \triangleq \begin{cases} -\min_{x^* \in \partial \mathcal{S}} \|x - x^*\|_2, & x \in \mathcal{S}, \\ \min_{x^* \in \partial \mathcal{S}} \|x - x^*\|_2, & x \notin \mathcal{S}. \end{cases} \quad (3.23)$$

Definition 3.2 (Dot-product second order cone). The class of dot-product second order cones $\mathcal{C}$ can be parameterized with a pointing vector $c \in \mathbb{R}^n$ and half-angle $\theta \in \mathbb{R}$ using the trigonometric dot product relation such that

$$\mathcal{C}(c, \theta) \triangleq \left\{ x \in \mathbb{R}^n \mid \frac{c^\top x}{\|c\|_2 \|x\|_2} \geq \cos \theta \right\}. \quad (3.24)$$

By applying definitions 3.1 and 3.2, we can derive the SDF for a second-order cone $\mathcal{C}$ as

$$\Psi_{\mathcal{C}(c,\theta)}(x) = \begin{cases} \|x\|_2 \sin(\gamma_c(x) - \theta), & \gamma_c(x) \leq \theta + \pi/2, \\ \|x\|_2, & \gamma_c(x) > \theta + \pi/2, \end{cases} \quad (3.25)$$

where γ_c represents the angle between x and c

$$\gamma_c(x) = \cos^{-1} \left(\frac{x^\top c}{\|x\|_2 \|c\|_2} \right). \quad (3.26)$$

Using this definition, the SDF for a set $\mathcal{F}_j$ can be defined as

$$\Psi_{\mathcal{F}_j}(z(t)) = \Psi_{\mathcal{C}(b, \beta_{\max})}(\mathcal{R}(t)(m_j(t) - r(t))). \quad (3.27)$$

A visualization of this set is provided in figure 3.4. The geometric intuition behind the SDF function (3.23) is that it defines the magnitude of the orthogonal projection of any vector input to the boundary of a set, with a sign to indicate whether this point belongs inside or outside the set, giving it favorable numerical properties when applying smoothing operations about the set boundary. With this definition in place, we still need to deal with the discontinuity existing in the aforementioned noise model.

Definition 3.3 (Sigmoid Function). *The sigmoid function $\Phi_\kappa : \mathbb{R} \rightarrow \mathbb{R}$ [69] provides a $\mathcal{C}^1$ -differentiable expression which tends towards 0 as $x \rightarrow -\infty$ and towards 1 as $x \rightarrow +\infty$, with a sharpening parameter $\kappa \in \mathbb{R}$, where $\kappa \rightarrow \infty$ recovers the Heaviside (step) function, and is defined as*

$$\Phi_\kappa(x) \triangleq 1 - (1 + \exp(\kappa x))^{-1}. \quad (3.28)$$

Using definition 3.3, we can develop a smooth approximation to the Heaviside (step) function, and compose this approximation with the numerically well-conditioned SDF in equation (3.27) to obtain an approximation to the range-based noise standard deviation, which we denote $\sigma_{\kappa,j}^{-1}(z)$ as

$$\sigma_{\kappa,j}^{-1}(z) = \tilde{\sigma}_j^{-1}(z) (1 - \Phi_\kappa(\Psi_{\mathcal{F}_j}(z))). \quad (3.29)$$

Then, the measurement noise covariance matrix is defined as a diagonal matrix, where the j th diagonal element of the matrix inverse is defined as

$$(V_\kappa(z)^{-1})_{j,j} = \left(\sigma_{\kappa,j}^{-1}(z) \right)^2. \quad (3.30)$$

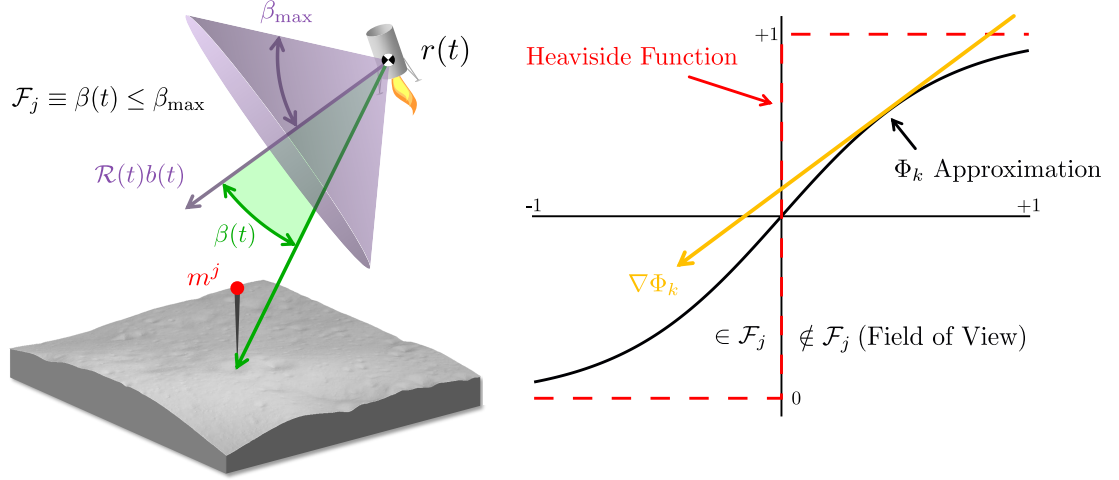


Figure 3.5: **Left:** Visual interpretation of the geometry underlying the FOV set $\mathcal{F}_j$. **Right:** Visual interpretation of the sigmoid smoothing approximation Φ_κ and what it provides for gradient-based trajectory optimization. The gradient of this function, $\nabla \Phi_\kappa$, is non-zero at all points in its domain, and when performing numerical gradient descent to minimize uncertainty associated with the measurement noise σ_j (or some proximal objective that implicitly seeks to minimize this uncertainty), the optimizer "pushes" in the direction towards perceiving landmarks in the FOV of the vehicle's range-based sensor.

We now revisit the EKBF system of equations (3.17a)-(3.17c), and modify the covariance update component with the smoothed approximation to the Kalman gain as

$$\dot{\Sigma}(t) = F(t)\Sigma(t) + \Sigma(t)F(t)^\top + W(t) - K_\kappa(\mu(t))H(t)\Sigma(t), \quad (3.31)$$

$$K_\kappa(\mu(t)) = \Sigma(t)H(t)^\top V_\kappa(\mu(t))^{-1}. \quad (3.32)$$

While the covariance dynamics above are well-defined in a forward-horizon (predictive control) sense, we must now confront a separate issue, in that the observation vector $y(t)$ is not well-defined before any actual measurements can be made. To address this issue, we must leverage one further assumption:

Assumption 3.3. *In the absence of online measurements, we assume that the observation*

vector matches the predictive measurement model, such that

$$y(t) = h(\mu(t)), \quad \forall \mu(t) \in \mathbb{R}^{n_z}. \quad (3.33)$$

Using assumption 3.3, we see that the EKBF mean update equation (3.17a) has the Kalman update term canceled out, such that it simplifies to

$$\dot{\mu}(t) = f(\mu(t), u(t)). \quad (3.34)$$

Using the new mean and covariance system of differential equations (3.31),(3.32),(3.34), we can then define the smoothed approximation to the dynamical mapping $g_\kappa : \mathbb{R}^{n_\zeta \times n_u} \rightarrow \mathbb{R}^{n_\zeta}$ with no explicit measurements such that the following relationship satisfies the aforementioned system of equations

$$\dot{\zeta}(t) = g_\kappa(\zeta(t), u(t)). \quad (3.35)$$

In the following section, equation (3.35) will be the primary dynamical relationship used to express a Gaussian trajectory distribution as a system of smoothed differential equations. Figure 3.5 provides visual intuition for the FOV set and why the sigmoid function approximation based around this set is valuable for gradient-based trajectory optimization. In section 3.3, we will discuss a technique which allows us to systematically update the sharpening term κ to recover the true noise model (3.21) while maintaining numerical stability.

Remark 3.2 (State-dependent noise). *While the focus of this work is on the construction of perceptive noise models, we note that many other common aerospace sensors may exhibit these discontinuous state-dependent limitations, such as altimeters (with a corresponding maximum altitude limit) or star trackers (with both FOV restrictions and sun incidence angle avoidance). To this end, both the discontinuous noise modeling approach and the consequent SDF-composed sigmoid smoothing methodology can be readily applied to all such sensor types.*

Remark 3.3 (Noise parameter terms). *This approach can also readily support inclusion*

of noise parameter terms, such as biases, scaling factors and misalignment errors, at the cost of additional states to model.

3.3 The ACT-SLAM Optimization Problem

In this section, we make use of the well-posed, $\mathcal{C}^1$ -differentiable filtering dynamics model derived in the previous section in equation 3.35, and propose a general-purpose continuous-time optimal control problem (CT-OCP) formulation (ACT-SLAM) to solve the active sensing guidance problem. This process connects the state estimation (*navigation*) component with solution to a trajectory optimization problem (*guidance*), and serves as a solidified instance of guidance-navigation co-design. Consider the following free-final-time CT-OCP defined over a time interval $t \in [t_0, t_f]$:

ACT-SLAM Problem (3.36)

$$\min_{\zeta(\cdot), u(\cdot)} J(\zeta, u) \quad (3.36a)$$

$$\text{subject to } \dot{\zeta}(t) = g_{\kappa}(\zeta(t), u(t)), \quad (3.36b)$$

$$g(\mu(t), u(t)) \leq 0, \quad (3.36c)$$

$$h(\mu(t), u(t)) = 0, \quad (3.36d)$$

$$p(\zeta(t_0), \zeta(t_f)) \leq 0, \quad (3.36e)$$

$$q(\zeta(t_0), \zeta(t_f)) = 0, \quad (3.36f)$$

$$\forall t \in [t_0, t_f].$$

Problem (3.36) formulates the ACT-SLAM problem over the full filtering state $\zeta(t) = (\mu(t), \text{vec}(\Sigma(t)))$ and control input $u(t)$. Notably, this problem closely mirrors problem (1.1), with the full filtering dynamics (over mean and covariance) imposed in equation (3.36b), path constraints imposed only over the mean state in equations (3.36c)-(3.36d) and boundary conditions imposed in equations (3.36e)-(3.36f). Here, $\zeta(t_0)$ and $\zeta(t_f)$ can be thought of as *priors* and *posteriors* in the context of the optimization horizon, respectively.

In this work, we opt to solve problem (3.36) using the CT-SCvx algorithm described in

section 1.3.2. Contrary to the approach in chapter 2, we opt to encode all path constraints continuously in time using the CTCS augmentation, rather than imposing them pointwise on a discrete time interval.

3.3.1 Co-design Configurations

The ACT-SLAM formulation presented in problem (3.36) enables a variety of candidate co-design modeling approaches for practical mission scenarios—we discuss several possible approaches here. One such approach is a direct specification of navigational requirements at the terminal point in time. Within aerospace contexts, these are typically modeled as 3σ error specifications. Consider, for example, a requirement that the terminal position in the inertial $+z$ element must fall within a 3σ uncertainty level ρ_{rz} , and denote index k to extract this position element from the covariance matrix $\Sigma(t)$. Then, we could specify the following as part of constraint (3.36e) such that

$$\sqrt{\Sigma_{k,k}(t_f)} \leq \frac{1}{3}\rho_{rz}. \quad (3.37)$$

Perhaps this approach may be used while satisfying some baseline mission-level objective, such as a minimum-fuel-usage objective as is typically seen for PDG scenarios

$$\mathcal{J}_{\text{minfuel}} = \int_{t_0}^{t_f} \|T(t)\|_2 dt. \quad (3.38)$$

If, however, there are no strict navigational requirements associated with the terminal uncertainty, one may instead wish to perform an *uncertainty-minimizing* maneuver. One technique to accomplish this is to apply the concept of differential entropy [67], which seeks to minimize the collective “surprise” associated with a probability distribution. It has been shown that, for Gaussian distributions, this term is proportional to the logarithmic determinant of the covariance of the distribution. We can formulate this as an objective

$$\mathcal{J}_{\text{entropy}} = \int_{t_0}^{t_f} \log \det \Sigma(t) dt. \quad (3.39)$$

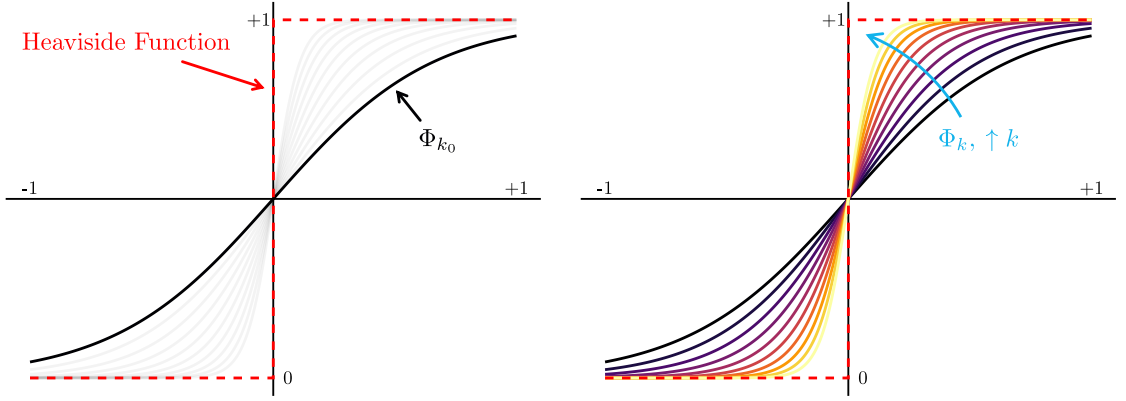


Figure 3.6: Illustrative diagram of the embedded numerical continuation process to provide an update law for the iterate-varying smoothing parameter κ_k .

Many mission design scenarios may also desire to combine these two types of configurations. For example, there may be certain navigational requirements which are hard-set and can be encoded as constraints, whereas other elements of the navigational solution must be optimized over to search for the best possible solution, as in [44]. This is not an exhaustive list of examples, either, but is meant to provide several highlighted options for the practitioner of these techniques. Going forward, we will use this differential entropy objective to produce results in section 3.4.

3.3.2 Embedded Numerical Continuation

With the ACT-SLAM problem established, we can now reassess the impact of the sharpening parameter κ embedded within the filtering dynamics in equation (3.35). Moreover, we would like to make use of an update law for this parameter, applied in tandem with the CT-SCvx iteration, such that the parameter can be systematically increased, and the corresponding step function approximation in the noise model can be tightened. We take direct inspiration from the *embedded numerical continuation* methodology presented in [80], where we conditionally update κ_k , indexed for the k th CT-SCvx iteration, according to a geometric

progression

$$\kappa_k = \begin{cases} \min(\alpha\kappa_{k-1}, \kappa_{\max}), & \Delta\mathcal{J}_k \in [\beta_L, \beta_U], \\ \kappa_{k-1}, & \text{else,} \end{cases} \quad (3.40)$$

where $\Delta\mathcal{J}_k = \mathcal{J}_k - \mathcal{J}_{k-1}$ and κ_0 represents the initial tightening parameter. We then apply an additional convergence criterion on top of criteria (1.20a)-(1.20b) which states that κ_k has converged to its maximum allowable tightness

$$\kappa_k = \kappa_{\max}. \quad (3.41)$$

An illustration of this technique is provided in figure 3.6. We refer the reader to [80] for more details, and for derivation of parameters $\alpha, \kappa_0, \kappa_{\max}, \beta_L, \beta_U$.³

3.4 Numerical Results

In this section, we consider a full-scale analysis of the ACT-SLAM algorithm applied to the lunar landing scenario mentioned in chapter 2. Specifically, we consider a descent scenario with three modeled landmarks located on a realistic digital elevation map (DEM), specifically corresponding to a designated Lunar south pole site of interest [1]. This is considered a reasonable number of landmarks, consistent with modern techniques [146], for the purpose of online SLAM in the context of lunar landing.

3.4.1 Scenario Parameters

The baseline scenario parameters (including vehicle configuration, path constraints and mean boundary conditions), as well as the (x, y) -components of the landmark positions (defined in a landing-site-relative *East-North-Up* (ENU) reference frame), can be found in [19]. The z -components of the landmarks are consequently taken from the Lunar site #1 DEM [1] at the referenced (x, y) coordinates. In terms of uncertainty priors, the covariance prior is set such that the vehicle position diagonal elements are equal to $(100)^2 \text{ m}^2$, and landmark position diagonal elements are equal to $(30)^2 \text{ m}^2$. For the purpose of this analysis, the process noise covariance matrix $W(t)$ is set to be zero. Additional model and algorithm parameters are provided in table 3.1.

³In [80], the parameters $\alpha > 1, \kappa_0, \kappa_{\max}$ are represented in expanded form with parameters $N_h, \epsilon_\kappa, \delta_0, \delta_1$.

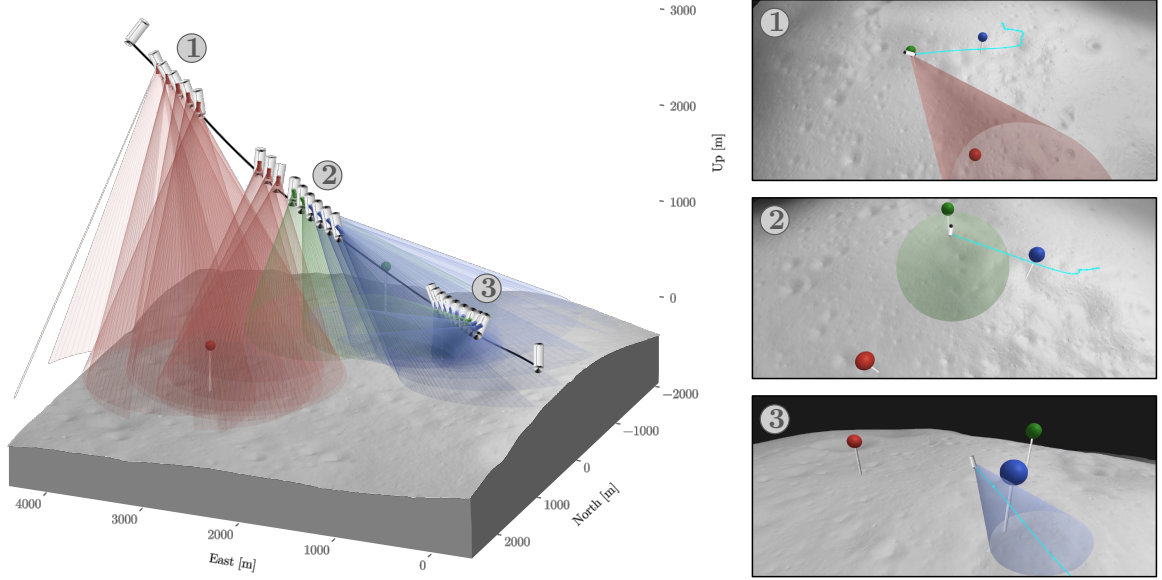


Figure 3.7: Left: Trajectory timelapse for the baseline PDG scenario using ACT-SLAM, where perceptive scans to each landmark are displayed with a 3 second delay between successive scans, and color-coded according to the **first**, **second** and **third** landmarks. Right: Snapshots of the baseline scenario in a high-fidelity visualization environment at different points in time.

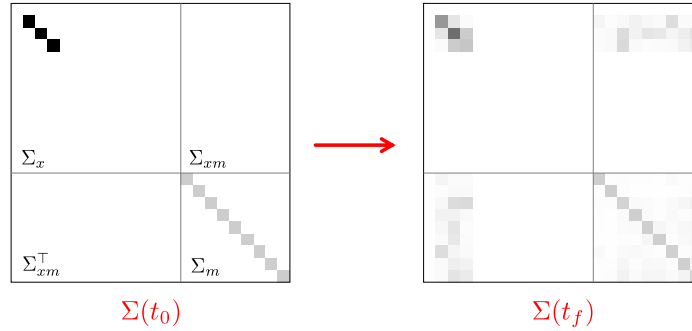


Figure 3.8: Example covariance evolution for the baseline scenario across the timeframe $t \in [0, t_f]$, where relative magnitudes between elements are represented by grayscale color intensity (*darker = higher magnitude*) under an element-wise square root mapping for visual clarity. Matrix partitions are demarcated by lines.

3.4.2 Baseline Scenario

Figure 3.7 illustrates an ACT-SLAM-generated solution for the baseline PDG scenario, with scans to each landmark displayed. As can be seen, the algorithm is able to obtain substantial

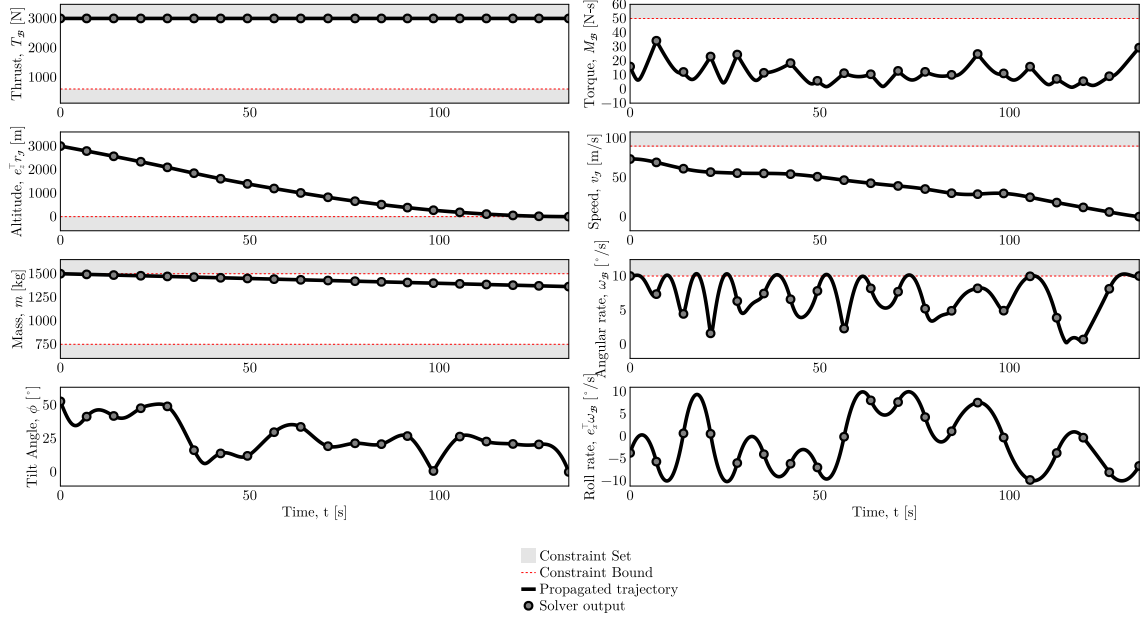


Figure 3.9: Vehicle trajectory signals and corresponding constraint sets associated with simulation of the baseline scenario.

coverage of each landmark, which aids in reducing vehicle uncertainty as well as landmark uncertainties over the full guidance horizon. The relative uncertainty changes can be visualized in the elements of the covariance matrix $\Sigma(t)$ as seen in figure 3.8, where diagonal elements correspond to direct navigational objectives or requirements. For this particular scenario, the landmark uncertainty priors were much smaller than the positional uncertainty priors, and so uncertainty reduction was mainly consolidated to the vehicle's position. This is consistent with expectations, and is a common observation made for terrain-relative navigation scenarios in contexts where terrain mapping can be performed *a-priori*, such that the vehicle localization portion of the SLAM problem is the primary focus for performance improvements. Figure 3.9 provides additional results for the trajectory of the lander in terms of the vehicle state and control input.

3.4.3 Model Validation

In light of the sigmoid smoothing approximation and measurement prediction assumption made in section 3.2, as well as inherent error associated with the EKBF's linearization-

based approximation of the covariance propagation, we desire to quantify the impacts of these approximations through statistical analysis. To facilitate this analysis, we can consider specification of two separate states: the *truth* state x_{truth} and the *navigational* state x_{nav} . We begin by sampling an initial truth state $x_{\text{truth},0}$ about the prior Gaussian distribution (μ_0, Σ_0) , and forward-simulate (integrate) this state according to the stochastic process model in equation (3.4). Simultaneously, we instantiate an EKBF filter based around equations (3.17a)-(3.17c). For validation purposes, we sample perceptive measurements $y(t)$ (lifting assumption 3.3), with FOV triggered as a step function based on the truth state $x_{\text{truth}}(t)$ at the current point in time (lifting the sigmoid approximation), to propagate and forward-simulate the EKBF model. Consequently, the navigational state is recorded as the mean of the filter $x_{\text{nav}}(t) = \mu(t)$. For analysis purposes, we consider the navigation-truth dispersion

$$\delta x \triangleq x_{\text{nav}} - x_{\text{truth}}. \quad (3.42)$$

Statistical analysis is performed through a Monte Carlo assessment over $n = 1000$ trials, distinguished by the sampling of state $x_{\text{truth},0}$ and model noise $w(t), v(t)$ according to their previously-defined distributions. Figure 3.10 provides the navigation-truth dispersion error histories across all trials for the vehicle position components, and compares them against the 3σ level sets of the Gaussian trajectory distribution provided by ACT-SLAM in the form of $\Sigma(t)$. Notably, the 3σ level set boundaries are only crossed in 1.1% of trials, corresponding to a 98.9% success rate—even in the cases where the boundaries are crossed, the magnitude of error is generally very small, suggesting high-precision approximation performance.

3.4.4 Literature Comparison

While sections 3.4.2 and 3.4.3 establish the viability of ACT-SLAM in isolation, we ultimately seek to understand how this technique compares to established methods in terms of active sensing guidance objectives. In the following subsection, we review two rivaling techniques and show how ACT-SLAM compares to them in terms of various performance benchmarks.

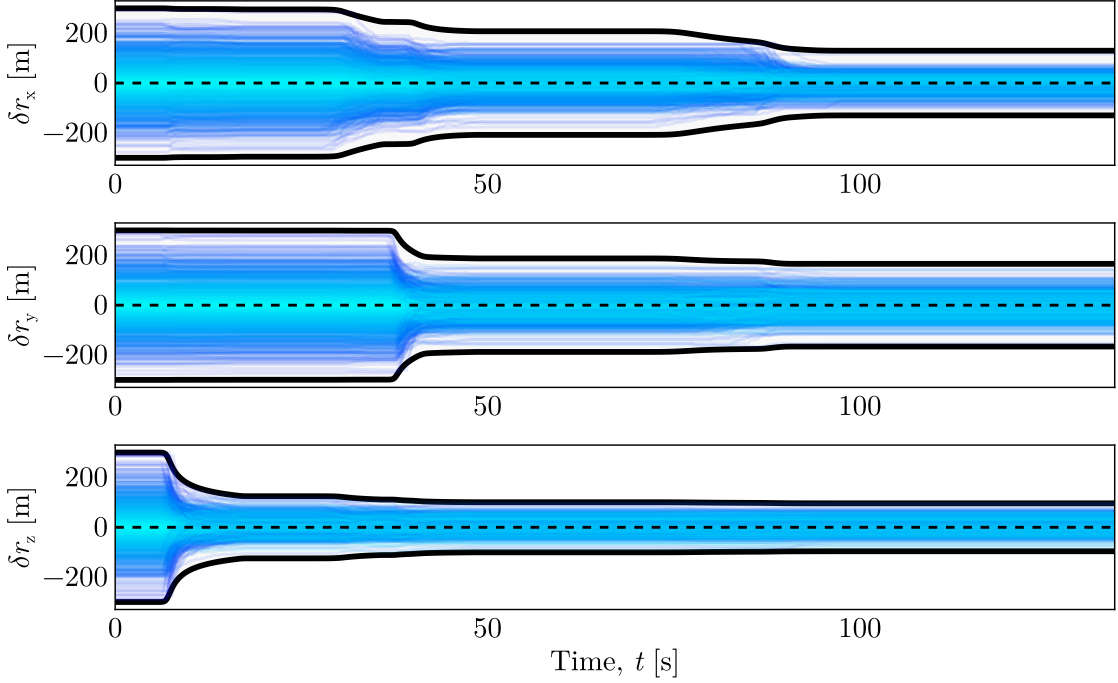


Figure 3.10: Monte Carlo analysis of ACT-SLAM model validation simulation over $n = 1000$ trials. Plots displaying navigation-truth dispersions are shown for each element of the vehicle position, where trajectories are color-coded according to the magnitude of the dispersion at $t = 0$. The 3σ level sets from the ACT-SLAM-output covariance are displayed by symmetric black curves.

Passive Perception Baseline

The first reference method concerns a passive perception algorithm which will serve as a baseline going forward. Borrowing from the modeling approach described in [133], we define a passive perception approach as one which does not opportunistically optimize for an information-theoretic objective, but independently computes a feasible trajectory for the scenario (in this case, under a separate objective), and modifies the trajectory to satisfy information-theoretic objectives *post hoc*.

We first compute a deterministic 6-DoF solution to the powered descent guidance problem using the CT-SCvx algorithm across only the vehicle mean state μ_x (neglecting mapping elements and covariance dynamics) using the minimum-fuel objective described in equation

(3.38) and all original constraints imposed on the vehicle mean state in the baseline scenario given by section 3.4.2. Next, we restrict our modeling considerations to a *non-gimbaled* lander configuration such that the thrust vector is fixed in the body frame along the longitudinal x -axis with $T_{\mathcal{B}}(t)/\|T_{\mathcal{B}}(t)\|_2 = e_1 = [1, 0, 0]^\top$. Under this condition, we remark that the system can be considered *roll-decoupled*, such that the lander can freely rotate (roll) about the $\mathcal{B}$ -frame x -axis without affecting translational degrees of freedom. Using this property, we modify the prior 6-DoF solution's rotation profile $\mathcal{R}(t)$ with roll adjustments that prioritize pointing towards the closest landmark at any time in the trajectory's history. To do this, we make use of the pointing direction projection technique described in [19]. Let us denote the provided minimum-fuel trajectory as $\{x^-(t), u^-(t)\}$, $t \in [t_0, t_f]$ using superscript $\square^-$. For each landmark, we denote the desired pointing direction to landmark j , expressed in the body frame, as $p_j(t) = \mathcal{R}^-(t)(m_j^-(t) - r^-(t))/\|m_j^-(t) - r^-(t)\|_2$, and the current boresight pointing direction as b . We can also define a projection matrix onto the $\mathcal{B}$ -frame y - z (roll) plane as

$$P \triangleq \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (3.43)$$

First, we must obtain the rotation matrix about the $\mathcal{B}$ -frame roll axis, $\mathcal{R}_j(t)$, which most-closely points $\mathcal{R}_j(t)b$ in the direction of $p_j(t)$. This can be done by rotational alignment of the pointing directions projected to the roll plane. Let us denote the signed angle between two (unit norm) vectors x, y about a rotation axis n as

$$\angle(x, y; n) \triangleq \text{atan2}((x \times y)^\top n, x^\top y). \quad (3.44)$$

Then, we can define $\mathcal{R}_j(t)$ by an axis-angle pair [63], such that $n(t) = \mathcal{R}(t)^\top e_1$ and $\theta(t) = \angle(Pb, Pp_j(t); n(t))$. Next, we can denote the unsigned (shortest-arc) angle between two (unit norm) vectors as

$$\angle_u(x, y) \triangleq \text{acos}(x^\top y). \quad (3.45)$$

Table 3.1: Supplementary scenario parameters.

Parameter	Value	Parameter	Value
ϵ_η	1e-4	N	20
ϵ_ν	1e-4	w_η	1e0
ϵ_{CTCS}	1e-4	w_ν	1e2
N_h	10	γ_1	100 m
ϵ_κ	0.01	γ_2	1e-5 m ⁻¹
δ_0	1		
δ_1	0.1		
β_L	-1e-3		
β_U	1e-1		

To determine the landmark we should point towards, we can select, for all times $t \in [t_0, t_f]$

$$j^\star = \underset{j \in [n_\ell]}{\operatorname{argmin}} \angle_{\mathbf{u}}(\mathcal{R}_j(t)b, p_j(t)). \quad (3.46)$$

Then, the provided trajectory is fitted with a new orientation profile $\mathcal{R}^+(t) = \mathcal{R}_{j^\star}(t)\mathcal{R}^-(t)$, which represents pointing towards the closest landmark in angular deviation space. Consequently, if $\angle_{\mathbf{u}}(\mathcal{R}_{j^\star}(t)b, p_j(t)) \leq \beta_{\max}$, then we have that landmark $j^\star$ is in field of view. We remark that this passive perception technique provides two salient benefits relative to ACT-SLAM: 1. computational efficiency due to only having to perform trajectory optimization on the reduced-order vehicle mean state space, and 2. preservation of the original minimum fuel objective in terms of optimality. Due to its computational efficiency, this solution is also used as an initial guess for problem (3.36). We also note that this technique can result in arbitrarily-large roll rates which may exceed the body rate constraints imposed on the scenario—for example, $\mathcal{R}_{j^\star}(t)$ is discontinuous in the general case when the tracked landmark $j^\star$ changes. For the purpose of comparison, we consider this an acceptable expression of the *upper bound* on landmark tracking performance (and consequently, in covariance reduction) that a passive perception technique would be capable of obtaining, such that any method that must project the roll rate onto the feasible body rate constraint space would logically exhibit worse performance.

Information-Aware Powered Descent Guidance

The second reference method considered is the information-aware powered descent guidance technique recently explored in [54], denoted as INFO-PDG. Consider the covariance matrix for the j th landmark

$$\Sigma^j = \begin{bmatrix} \Sigma_x^j & \Sigma_{xy}^j & \Sigma_{xz}^j \\ * & \Sigma_y^j & \Sigma_{yz}^j \\ * & * & \Sigma_z^j \end{bmatrix}. \quad (3.47)$$

INFO-PDG constructs a covariance dynamics law which is restricted to only the z -component covariance element of each landmark, $\Sigma_z^j(t)$, and models the dynamics as a smooth Gaussian conditioned on the FOV containment set $\mathcal{F}_j$, with an additional saturation condition to prevent the covariance elements from becoming negative. Using the conventions of this chapter, this dynamics model can be summarized as:

$$\dot{\Sigma}_z^j(t) = \begin{cases} -\delta \Phi_{-\kappa}(\max(0, F_j(\mu_x(t)))^2), & \Sigma_z^j(t) > 0, \\ 0, & \Sigma_z^j(t) \leq 0. \end{cases} \quad (3.48)$$

where $F_j(\mu_x(t)) = \cos \beta_{\max} - (\mathcal{R}(t)^\top b)^\top (m_j(t) - r(t)) / \|m_j(t) - r(t)\|_2$ denotes the FOV constraint set evaluation such that $F_j(\mu_x(t)) \leq 0 \iff \mu_x(t) \in \mathcal{F}_j$, and the composition $\max(0, F(\mu_x(t)))^2$ ensures that the covariance strictly decreases only when $\mu_x(t) \in \mathcal{F}_j$, driven by a rate of descent $\delta \in \mathbb{R}_+$. Consequently, the INFO-PDG objective is chosen as

$$J_{\text{info}} = \int_{t_0}^{t_f} \sum_{j=1}^{n_\ell} \Sigma_z^j(t) dt. \quad (3.49)$$

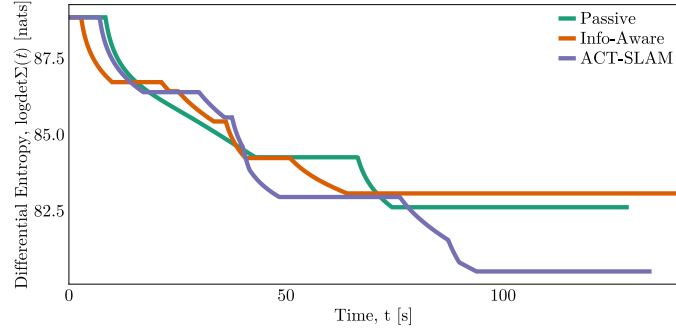


Figure 3.11: Differential entropy history over time between the passive perception, Info-PDG and ACT-SLAM methods.

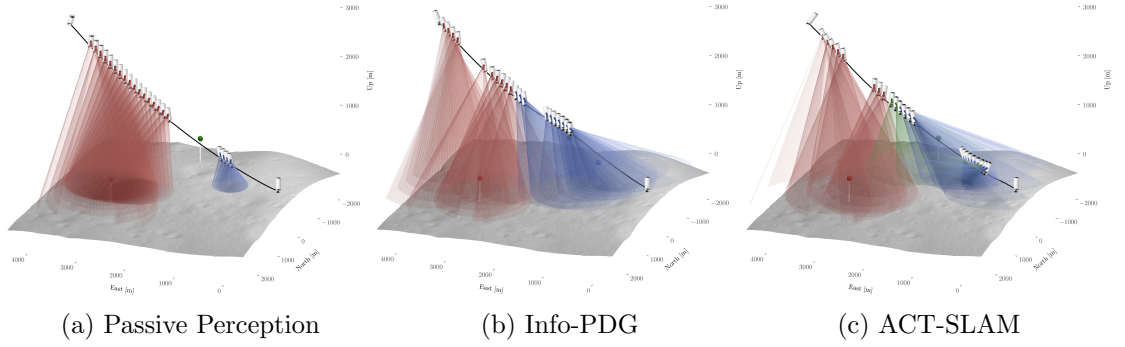


Figure 3.12: Trajectory timelapses for the passive perception, Info-PDG and ACT-SLAM methods, using the same visualization conventions as figure 3.7.

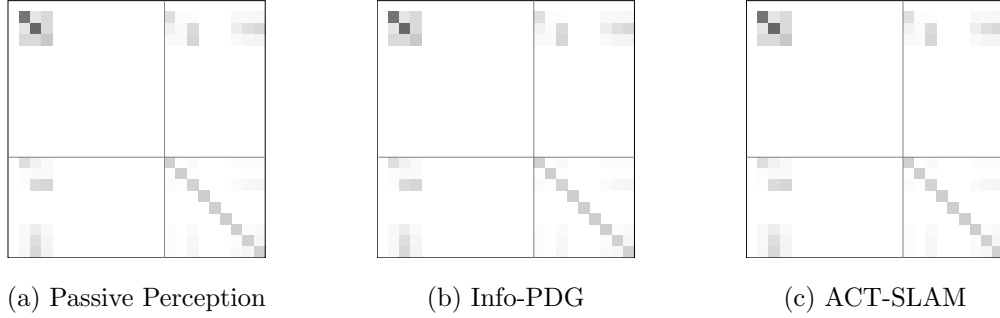


Figure 3.13: Visualization of the terminal covariance matrix, $\Sigma(t_f)$, for the passive perception, Info-PDG and ACT-SLAM methods, using the same visualization conventions as figure 3.8.

When drawing modeling comparisons to ACT-SLAM in terms of the covariance dynamics model in equation (3.31), we can make several important observations. First, ACT-SLAM considers a much larger state space encompassing all covariance elements and their cross-correlations in accordance with an EKF model, enabling a complete and rigorous expression of state estimation error at the cost of increased computational burden. This property also prevents the need for a switch statement as in equation (3.48), ensuring smoothness of the resulting dynamics function. Finally, ACT-SLAM does not require parameter tuning of δ and κ (the latter due to embedded numerical continuation) – this enables ACT-SLAM to be *tuning-free* by comparison and guarantees that sensor measurements are approximated to high numerical precision. In the case of INFO-PDG, these parameters are chosen as $\delta = 11$ and $\kappa = 1$ for the purpose of comparison.

Comparative Analysis

With both reference techniques established, we can now engage in a comparison between both of these methods and ACT-SLAM. Table 3.2 provides a summary of performance metrics used to compare between passive perception, INFO-PDG and ACT-SLAM. This analysis includes an additional metric introduced by [133] labeled *information gain* and characterized as the net difference in differential entropy over the maneuver duration, stated

as

$$\Delta I \triangleq \frac{1}{2} \log \det \Sigma(t_0) - \frac{1}{2} \log \det \Sigma(t_f) = \frac{1}{2} \log \frac{\det \Sigma(t_0)}{\det \Sigma(t_f)}. \quad (3.50)$$

Under this metric, we observe that ACT-SLAM achieves a +34.1% and +44.3% improvement over passive perception and INFO-PDG, respectively. We can also consult the original objective in equation (3.39) and observe +1.3% and +1.2% improvements, respectively (noting the difference in improvement magnitudes due to logarithmic compression). Notably, we normalize the objective by the time of maneuver, t_f , to improve the comparison – we seek to characterize how fast the differential entropy metric decreases *in relation to* maneuver duration, such that we don’t arbitrarily penalize a maneuver for having a longer duration. Figure 3.11 shows a time signal comparison between these methods in terms of differential entropy. In figure 3.12, we can see a visual comparison between the trajectories of each method and the sensor measurement scans taken for each landmark. Of particular note is the fact that, between all three methods, only ACT-SLAM manages to achieve visibility of all three landmarks. We contend that this is due to the fact that: 1. passive perception *greedily* prioritizes the closest landmark (in terms of angular deviation), and 2. INFO-PDG does not prioritize any landmark over another in its objective, provided the covariance associated with a landmark does not saturate to zero. ACT-SLAM, on the other hand, encodes an objective and dynamics law which results in up-weighting landmarks with higher covariances, prioritizing more mapping coverage and improving overall estimation performance. We note this as an important additional benefit of the ACT-SLAM method relative to its predecessors. Figure 3.13 displays the terminal covariance matrices associated with each method, further displaying the improvements in covariance reduction with respect to both the vehicle positional state and landmark states, in addition to more stable and uniform cross-correlations.

Referring back to table 3.2, we note that ACT-SLAM does exhibit worse performance in terms of the minimum fuel objective (expressed in terms of mass difference over the maneuver, $\Delta m = m(t_0) - m(t_f)$) and solve time, relative to the baseline passive perception method. This is consistent with expectations and highlights an important tradeoff for improved information acquisition. In light of these tradeoffs, we contend that these results

Table 3.2: Literature Comparison Summary.

Method	$\uparrow \Delta I$ (nats)	$\downarrow \int \log \det \Sigma(t) dt / t_f$ (nats/s)	$\downarrow \Delta m$ (kg)	$\downarrow$ Solve Time (s)
Passive [133]	3.11	84.45	128.03	1.05
INFO-PDG [54]	2.89	84.36	136.22	91.67
ACT-SLAM	4.17	83.40	136.82	96.12

present ACT-SLAM as a promising approach for optimal information-theoretic trajectory generation, suitable for detailed offline analysis and nominal trajectory generation in the context of active sensing guidance.

Chapter 4

GRAPH SUCCESSIVE CONVEXIFICATION

4.1 Introduction

In the last two chapters, we focused on techniques to explicitly characterize and satisfy perceptive objectives, framed through the lens of sensor pointing towards ground-based regions of interest. We now consider a philosophically distinct approach, where we do not assume any prior knowledge of our environment, and instead aim to passively acquire perceptive information while *deferring* the decision to commit to any particular terminal objective.

Formally, if we assume a collection of goal states are of interest, however the autonomous system does not currently have enough information to commit to any particular one, then a trajectory can be designed that defers the decision to proceed towards any particular goal state for a designated amount of time, or for a given ϵ -suboptimal threshold. This deferral requirement can be motivated in a multitude of ways. For example, a perception module may require prolonged visibility and exploration of the environment to determine goal objectives. Or perhaps a higher-level task planner requires extended time to decide on an objective due to computational limitations. We can alternatively view decision deferral in a contingency-aware context, where a singular goal state may be of interest, however contingency states must remain reachable for as long as possible in case the nominal target becomes unavailable.

We note that this approach is distinct from the related fields of robust and stochastic

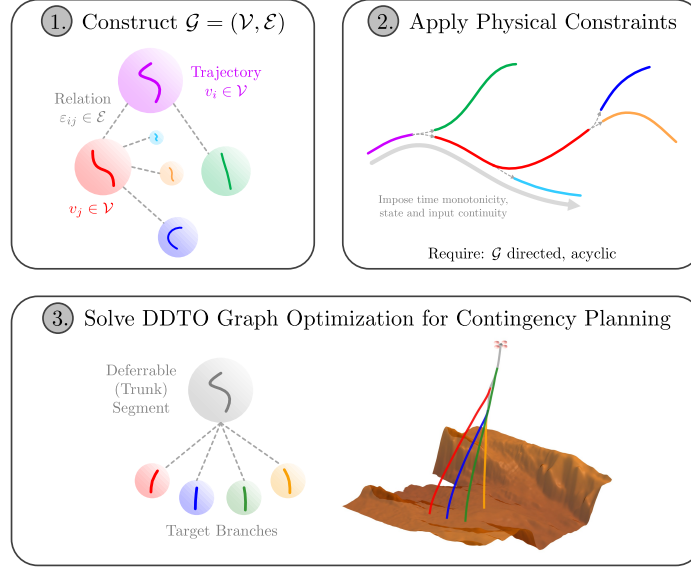


Figure 4.1: Conceptual illustration of GRAPH-SCvx problem modeling approach with application to DDTO problems.

trajectory optimization. In the presence of non-trivial uncertainties imposed on the system, many traditional approaches have focused on the usage of robust trajectory optimization to design for any possible uncertainty within a known, bounded set [64, 77]. Stochastic trajectory optimization takes a similar approach to this problem by instead considering common unbounded distributions, such as Gaussian distributions [36, 91, 113]. The commonality between these approaches is that the parameters of the uncertainty distribution are known *a priori*—in this work, we aim to lift that assumption entirely. This chapter’s contributions are based jointly on the work of [52] as well as [15].

4.1.1 Related Literature

The concept of deferrability has been well-explored in decision theory and cognitive psychology [10, 31, 136], motivated by findings that decision makers have a tendency to procrastinate choice selection for a variety of reasons. Consequently, the concept of decision deferral has contributed to autonomous systems research through application to air traffic management systems [150], error correction for inertial navigation systems [14], floorplanning algorithms for circuitry design [147] and active fault diagnosis [109].

Decision deferral in path planning

The deferred-decision trajectory optimization (DDTO) problem has been proposed as a tractable modeling approach towards modeling multiple-target path planning problems with deferrability objectives [38]. These methods construct a trajectory to each target, and seeks to maximize the duration over which the trajectories can remain identical in terms of their state. DDTO trajectories are permitted to be ϵ^j -suboptimal relative to a precomputed single-target trajectory to the j -th target (solved as a two-point boundary value problem).

A number of developments have been made towards modeling deferrability-driven trajectory optimization problems in the context of autonomous path planning. Hardy et al. [50] consider other agents in the environment (such as cars and pedestrians) as dynamic obstacles and make predictions on future trajectories of these agents. Then, a spline-based trajectory optimization problem is defined over a fixed time grid where trajectories towards multiple targets¹ are constrained to be identical over a fixed amount of time, treating one target as the primary objective, and others as contingency plans. Chen et al. [29] proposed an extension of the above formulation to a model predictive control (MPC) framework using a branching-based MPC approach introduced in [117].

This type of approach has similarly been explored in [7, 8, 28, 74, 121, 142], with notable distinctions in terms of probabilistic modeling of obstacle trajectory futures and risk-awareness. The commonality between all of these approaches, however, is a restriction to a discrete-time formulation of the optimal control problem coupled with a fixed deferrability window, as opposed to the DDTO formulation which *maximizes* over this window to prolong reachability to all targets (or objectives) for as long as possible under some suboptimality tolerance. Recently, Elango et al. [38] introduced a tractable approach to explicitly maximizing for deferrability using a multi-step quasiconvex optimization formulation solved with a bisection search. Notably, this approach requires a restrictive convex substructure (implying dynamics are linear, path constraints are convex, and the final maneuver time is fixed) and must still be posed in discrete-time. Kamath et al. [60] introduced a set-based dynamic programming

¹This work uses a soft constraint in the objective encourage reaching the goal state rather than a defined terminal condition.

approach which can solve the aforementioned problem closed-loop under bounded uncertainty, however is still restricted to convex substructure. DDTO has also been extended towards nonconvex problem definition and made amenable to a continuous-time formulation [39] by using the continuous-time successive convexification (CT-SCvx) optimization framework. While this framework provides the most general-form and expressive formulation for DDTO problems to date, its algorithmic implementation produces large solve times and $\mathcal{O}(n^2)$ model size complexity with respect to the number of targets, motivating a new approach to solving DDTO problems which is amenable to real-time path planning.

Perception-aware path planning for autonomous landing

The problem of fuel-efficient and safe landing in non-trivial, hazardous and unknown environments is of keen interest for the autonomous operation of planetary spacecraft [24, 135], quadrotors/unmanned aerial vehicles (UAVs) [41, 96] and aircraft [131]. Açıkmeşe et al. [2] introduced the concept of real-time, onboard retargeting/diverting towards a new landing site when an old one becomes unsafe through the use of convex optimization, coined the G-FOLD algorithm. This general approach has been studied further in the context of hazard avoidance for lunar-class landing missions [32, 118], and has been tested on flight hardware [3].

Other techniques have studied higher-frequency, opportunistic replanning, particularly in the context of UAV landing under dynamic obstacle avoidance. Falanga et al. [41] propose a fast retargeting technique using a polynomial trajectory generation method to perform quadrotor landing on a moving platform. Guo et al. [47] approach the same problem with the use of high-rate model-predictive control (MPC) with perception-in-the-loop.

The remainder of this chapter is structured as follows. Section 4.2 presents the continuous-time graph optimal control problem formulation. Section 4.3 presents techniques to convert this problem to a discrete-time optimal control problem on a normalized time grid. Section 4.4 shows how sequential convex programming can be used to solve the nonconvex discretized problem with convex approximations, resulting in the GRAPH-SCvx formulation. Section 4.5 adapts GRAPH-SCvx for the DDTO problem, resulting in the GRAPH-DDTO formulation.

Section 4.6 introduces the `ADAPTIVE-DDTO` to operationalize DDTO for perception-based maneuvers. Section 4.7 presents literature comparisons, analysis in a high-fidelity simulation environment and hardware tests onboard a quadrotor system.

4.1.2 Contributions

In this chapter, we present a unified formulation to the deferrability problem with rigorous ties to traditional optimal control theory. This work extends on the existing literature, particularly the deferred decision trajectory optimization (DDTO) framework [38, 39], by enabling flexible and numerically-efficient specification of DDTO problems under arbitrary nonconvex problem definition (in terms of path constraints and boundary conditions). This effort is conducted through introduction of a new approach to modeling multi-segment trajectory optimization problems through a graph-theoretic lens—the Graph Successive Convexification (`GRAPH-SCVX`) framework—while building on the continuous-time successive convexification (`CT-SCVX`) approach introduced in [37]. We note that aspects of this approach are largely an extension of the `CT-SCVX` method introduced in 1.3.2, however we will introduce it completely separately in this chapter due to the customization of the proposed method.

We present a unified framework for modeling and solving DDTO problems (`GRAPH-DDTO`) under the following core contributions:

1. Formulation of DDTO problems as a graph-theoretic problem, and introduction of the Graph Successive Convexification (`GRAPH-SCVX`) framework to solve these problems in real-time using sequential convex programming (SCP).
2. One-shot joint optimization to multiple deferrable targets rather than multi-step (lexicographic) optimization on sparse discretized time grids (`GRAPH-DDTO`).
3. Algorithmic and modeling advances customized to `GRAPH-DDTO`, including an efficient warmstart algorithm and a variable elimination technique for graph edges.
4. Introduction of the `ADAPTIVE-DDTO` algorithm [52] as a technique, wrapped around a given DDTO solver, to adaptively trigger DDTO recomputations based on target safety conditions.

5. Integration into a perception-in-the-loop ROS 2 stack with new simulation and flight hardware demonstrations, displaying advancements over the predecessor nonconvex DDTO formulation and against the broader perception-aware landing literature.

4.2 Problem Formulation

We begin by describing the baseline continuous-time optimal control problem for the graph trajectory optimization model, and presenting a transformation to convert it to a free-final-time problem with continuous-time constraint satisfaction guarantees using successive convexification techniques (GRAPH-SCVX).

4.2.1 Notation

We use the following notation throughout the remainder of this chapter. The set of real $n \times 1$ vectors is denoted as $\mathbb{R}^n$, and the set of real $n \times m$ matrices is denoted as $\mathbb{R}^{n \times m}$, and the set of non-negative real numbers is denoted as $\mathbb{R}_+$. The set of (one-indexed) natural numbers is denoted as $\mathbb{N}$. A concatenation of vectors $v_1, \dots, v_N \in \mathbb{R}^n$ is denoted as $(v_1, \dots, v_N) \in \mathbb{R}^{nN}$, equivalently stated as $(v_k)_{k \in [N]}$ and interpreted as a column vector. A horizontal concatenation of matrices $A_1, \dots, A_N \in \mathbb{R}^{n \times m}$ is denoted as $(A_1, \dots, A_N) \in \mathbb{R}^{n \times mN}$, equivalently stated as $(A_k)_{k \in [N]}$. A vector of natural numbers up to N is denoted as $[N] = (1, \dots, N)$. We define $|v|_+ = \max(0, v)$, applied element-wise if v is a vector, $\|v\|_p$ as the p -norm, $\partial f(v)$ as the subdifferential of f evaluated at v , $\otimes$ as the Kronecker product, and $\mathbf{1}$ as a vector of all ones (length determined by context). The gradient of a differentiable function $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ with respect to $v \in \mathbb{R}^l$, $l \leq n$, evaluated at $x \in \mathbb{R}^n$, can be expressed as $\nabla_v f(x)$.

4.2.2 Optimal Control Formulation

The graph trajectory optimization problem formulation begins with a definition of $n \in \mathbb{N}$ distinct trajectory segments. Moreover, for a segment j , we define a time interval $t^j \in [t_i^j, t_f^j] \subseteq \mathbb{R}^+$ with state $x^j \in \mathbb{R}^{n_x}$ and input $u^j \in \mathbb{R}^{n_u}$. Additionally, we consider state boundary conditions $x_i^j = x^j(t_i^j)$ and $x_f^j = x^j(t_f^j)$, with corresponding input boundary conditions u_i^j and u_f^j . All segments are defined under a global dynamics relationship $\dot{x}^j(t^j) = f(t^j, x^j(t^j), u^j(t^j))$.

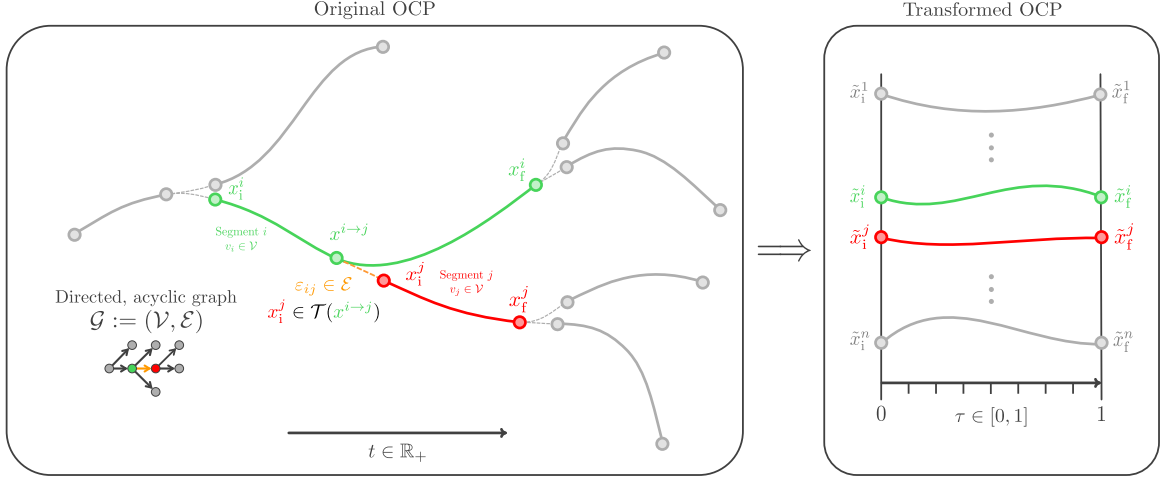


Figure 4.2: Graphical illustration of the GRAPH-SCvx modeling process, starting with the original OCP stated by problem (4.1), and converted to the transformed OCP stated by problem (4.7), where all trajectory segments are defined on the same normalized time grid.

In order to relate trajectory segments to each other, we construct a directed, acyclic graph (DAG) $\mathcal{G} := (\mathcal{V}, \mathcal{E})$, where vertices $\mathcal{V} = [n]$ indicate a segment $i \in \mathcal{V}$, and edges $\varepsilon_{ij} \in \mathcal{E}$ represent the stitching of a segment i to segment j . Each ε_{ij} contains stitching times $(t^{i \rightarrow j}, t_i^j)$ where $t^{i \rightarrow j} \in [t_i^i, t_f^i]$, such that $x^{i \rightarrow j} = x^i(t^{i \rightarrow j})$ is *stitched* to $x_i^j = x^j(t_i^j)$. More precisely, stitching is expressed by the constraint $x^j(t_i^j) \in \mathcal{T}(x^i(t^{i \rightarrow j}))$, where $\mathcal{T} : \mathbb{R}^{n_x} \rightarrow 2^{\mathbb{R}^{n_x}}$ represents the stitching constraint. To introduce the GRAPH-SCvx formulation, we consider the simple case of $\mathcal{T}$ representing the identity relation, such that $x^{i \rightarrow j} = x_i^j$. Notably, directly stitching as an equality constraint (hence referred to as *instantaneous* stitching) introduces a redundancy (duplication of a knot point) in the discretized problem; in appendix A, we introduce an alternative constraint formulation which removes this redundancy by instead modeling an additional trajectory segment between $x^{i \rightarrow j}$ and x_i^j , serving as a variable elimination (hence referred to as *phase-based* stitching). This approach yields an equivalent result from a modeling standpoint under certain assumptions (which we can show for the discrete-time case by proposition A.2 in appendix A.2), so we proceed with instantaneous stitching to simplify the introduction of the formulation, however we note that phase-based

stitching will be used in the experimental results.

Remark 4.1. *The choice of designating intermediate stitching time $t^{i \rightarrow j} \in [t_1^i, t_f^i]$, rather than stitching directly from the terminal time t_f^i , is made to improve model expressivity when discretizing the problem. In the case of DDTO, this is a consequence of multiple downstream segments needing to be connected to a shared segment at different discrete points. Restriction to time t_f^i would require an enumeration of additional segments and, consequently, an additional discrete point to be modeled per each downstream segment.*

We can now state the baseline continuous-time optimal control problem (OCP) as follows:

Graph Optimal Control Problem (4.1)

$$\begin{aligned} \underset{\substack{x^j(\cdot), u^j(\cdot) \\ t_1^j, t_f^j, t^{i \rightarrow j} \\ j \in \mathcal{V}, \varepsilon_{ij} \in \mathcal{E}}}{\text{minimize}} \quad & l(x_f) \end{aligned} \tag{4.1a}$$

$$\text{s. t.} \quad \dot{x}^j = f(t^j, x^j(t^j), u^j(t^j)), \quad j \in \mathcal{V}, \tag{4.1b}$$

$$g^j(t^j, x^j(t^j), u^j(t^j)) \leq 0, \quad j \in \mathcal{V}, \tag{4.1c}$$

$$h^j(t^j, x^j(t^j), u^j(t^j)) = 0, \quad j \in \mathcal{V}, \tag{4.1d}$$

$$p^j(t_1^j, x^j(t_1^j), t_f^j, x^j(t_f^j)) \leq 0, \quad j \in \mathcal{V}, \tag{4.1e}$$

$$q^j(t_1^j, x^j(t_1^j), t_f^j, x^j(t_f^j)) = 0, \quad j \in \mathcal{V}, \tag{4.1f}$$

$$\forall t^j \in [t_1^j, t_f^j], \quad j \in \mathcal{V}, \tag{4.1g}$$

$$x^j(t_1^j) = x^i(t^{i \rightarrow j}), \quad \varepsilon_{ij} \in \mathcal{E}, \tag{4.1h}$$

$$t_1^j = t^{i \rightarrow j}, \quad \varepsilon_{ij} \in \mathcal{E}, \tag{4.1i}$$

$$t^{i \rightarrow j} \in [t_1^i, t_f^i], \quad \varepsilon_{ij} \in \mathcal{E}. \tag{4.1j}$$

In problem (4.1), $x_f = (x_f^j)_{j \in \mathcal{V}}$, $l : \mathbb{R}^{n_x n} \rightarrow \mathbb{R}$ represents the Mayer form objective [105], $g^j : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_g}$ and $h^j : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_h}$ represent inequality and equality path constraints for $j \in \mathcal{V}$, and $p^j : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_p}$ and $q^j : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_q}$ represent inequality and equality boundary condition constraints for $j \in \mathcal{V}$. The constraint (4.1i) is

necessary to enforce time monotonicity between trajectory segments—making the branched system physically realizable in our formulation—and sets the requirement for the graph $\mathcal{G}$ to be directed and acyclic.

Remark 4.2. *If the vertex constraints (4.1b)-(4.1f) are concatenated, this problem closely resembles the canonical Mayer form optimal control problem, with the exception of the edge constraints (4.1h)-(4.1i) (which cannot be expressed directly by any of the vertex constraints). In practice, we leave these vertex constraints separated to allow for flexibility in node sparsity during discretization.*

4.2.3 Continuous-Time Constraint Satisfaction

We now consider the continuous-time constraint satisfaction (CTCS) formulation introduced in [37] to ensure path constraints (4.1c)-(4.1d) are satisfied at all points in time, rather than at a discrete nodal resolution, as is traditionally imposed in trajectory optimization. Moreover, we define the penalty function $\Lambda : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}$ such that $\Lambda(t^j, x^j, u^j) = 0, \forall t^j \in [t_i^j, t_f^j] \iff g^j(t^j, x^j, u^j) \leq 0, h^j(t^j, x^j, u^j) = 0, \forall t^j \in [t_i^j, t_f^j]$. One specific instantiation of this function is of the form

$$\Lambda(t, x^j(t), u^j(t)) = 1^\top |g^j(t, x^j(t), u^j(t))|_+^2 + 1^\top h^j(t, x^j(t), u^j(t))^2, \quad (4.2)$$

where $|x|_+ = \max(0, x)$. Then, for a trajectory segment $j \in \mathcal{V}$, we will define the penalty variable $\theta^j \in \mathbb{R}$ defined by the integration of the penalty function

$$\theta^j(t^j) = \int_{t_i^j}^{t^j} \Lambda(t, x^j(t), u^j(t)) dt. \quad (4.3)$$

Under this definition, the constraint satisfaction property can be obtained by solving for the following two-point boundary value problem

$$\dot{\theta}^j(t^j) = \Lambda(t^j, x^j(t^j), u^j(t^j)), \quad (4.4a)$$

$$\theta^j(t_i^j) = \theta^j(t_f^j). \quad (4.4b)$$

As will be shown in the following subsection, these conditions can be readily achieved by embedding the penalty variable θ^j in the segment state representation to satisfy (4.4a), and separately applying (4.4b) as an additional constraint in the resulting optimal control problem.

4.2.4 Generalized Time Dilation

We ultimately seek to optimize for all timing events, including segment transition times $t^{i \rightarrow j}$ and segment boundary times t_1^j, t_f^j . To do this, we consider the generalized time dilation formulation, introduced in [19, 37], wherein we define wall-clock time as a smooth, strictly monotonic mapping $t^j : [0, 1] \rightarrow \mathbb{R}_+$ such that $t^j(0) = t_1^j$ and $t^j(1) = t_f^j$. We will also define a normalized time $\tau \in [0, 1]$ and time derivative $s^j(t^j(\tau)) = dt^j(\tau)/d\tau > 0$, and consequently define an augmented state $\tilde{x}^j \in \mathbb{R}^{n_{\tilde{x}}}$, $n_{\tilde{x}} = n_x + 2$, and input $\tilde{u}^j \in \mathbb{R}^{n_{\tilde{u}}}$, $n_{\tilde{u}} = n_u + 1$, such that

$$\tilde{x}^j(\tau) = (x^j(t^j(\tau)), \theta^j(t^j(\tau)), t^j(\tau)), \quad (4.5a)$$

$$\tilde{u}^j(\tau) = (u^j(t^j(\tau)), s^j(t^j(\tau))). \quad (4.5b)$$

In fact, we can ultimately obtain the dynamics relationship $\dot{\tilde{x}}^j(\tau) = F(\tilde{x}^j(\tau), \tilde{u}^j(\tau))$ where $\dot{\square} = d\square/d\tau$ and

$$F(\tilde{x}^j, \tilde{u}^j) = s^j \begin{bmatrix} f(t^j, x^j, u^j) \\ \Lambda(t^j, x^j, u^j) \\ 1 \end{bmatrix}. \quad (4.6)$$

Consequently, we define the following augmented terms:

$$\begin{aligned} L(\tilde{x}_f) &\triangleq l(({}^xE\tilde{x}^j(1))_{j \in \mathcal{V}}), \\ P^j(\tilde{x}(0), \tilde{x}(1)) &\triangleq p^j({}^tE\tilde{x}(0), {}^xE\tilde{x}(0), {}^tE\tilde{x}(1), {}^xE\tilde{x}(1)), \\ Q^j(\tilde{x}(0), \tilde{x}(1)) &\triangleq q^j({}^tE\tilde{x}(0), {}^xE\tilde{x}(0), {}^tE\tilde{x}(1), {}^xE\tilde{x}(1)), \end{aligned}$$

where indexing matrices $({}^xE, {}^\theta E, {}^tE)$ extract components (x, θ, t) from $\tilde{x}$, respectively; indexing matrices $({}^uE, {}^sE)$ extract components (u, s) from $\tilde{u}$, respectively; and $\tilde{x}_f = (\tilde{x}_f^j)_{j \in \mathcal{V}}$. Since τ is now the independent variable, we can select $\tau^{i \rightarrow j} \in [0, 1]$ *a priori*, for a given

$\varepsilon_{ij} \in \mathcal{E}$, where we consequently have that $\tilde{x}^{i \rightarrow j} = \tilde{x}^i(\tau^{i \rightarrow j})$ and $\tilde{x}_1^j = \tilde{x}^j(\tau_1^j)$. Due to time dilation, the optimizer will automatically determine the corresponding segment transition time $t^{i \rightarrow j}$. We can now state the transformed continuous-time optimal control problem as follows:

Transformed Optimal Control Problem (4.7)

$$\underset{\substack{\tilde{x}^j(\cdot), \tilde{u}^j(\cdot) \\ j \in \mathcal{V}}}{\text{minimize}} \quad L(\tilde{x}_f) \quad (4.7a)$$

$$\text{s. t.} \quad \dot{\tilde{x}}^j(\tau) = F(\tilde{x}^j(\tau), \tilde{u}^j(\tau)), \quad j \in \mathcal{V}, \quad (4.7b)$$

$$P^j(\tilde{x}^j(0), \tilde{x}^j(1)) \leq 0, \quad j \in \mathcal{V}, \quad (4.7c)$$

$$Q^j(\tilde{x}^j(0), \tilde{x}^j(1)) = 0, \quad j \in \mathcal{V}, \quad (4.7d)$$

$${}^\theta E(\tilde{x}^j(1) - \tilde{x}^j(0)) = 0, \quad j \in \mathcal{V}, \quad (4.7e)$$

$${}^s E \tilde{u}^j(\tau) > 0, \quad j \in \mathcal{V}, \quad (4.7f)$$

$$\tilde{x}^i(\tau^{i \rightarrow j}) - \tilde{x}_1^j(\tau_1^j) = 0, \quad \varepsilon_{ij} \in \mathcal{E}, \quad (4.7g)$$

$$\forall \tau \in [0, 1]. \quad (4.7h)$$

This formulation enables us to design a tractable optimization problem which captures the modeling flexibility of problem 4.1 while being amenable to finite-dimensional optimization through the forthcoming transcription step.

4.3 Transcription

With problem (4.7) stated as a continuous-time OCP, we can now describe the steps to convert it to a finite-dimensional, discrete-time OCP that is amenable to numerical optimization.

4.3.1 Parameterization & Discretization

We first describe parameterization and discretization for an arbitrary trajectory before considering extension to the GRAPH-SCVx problem. Consider discretization of a trajectory $\tilde{x}(\tau), \tilde{u}(\tau)$, with $\tau \in [0, 1]$, onto a finite knot point grid of size $N \in \mathbb{N}$, with $0 = \tau_1 <$

$\dots < \tau_N = 1$. In doing so, we can then represent our segment states compactly with $\tilde{X} = (\tilde{x}_1, \dots, \tilde{x}_N)$. We define $\Gamma(\tau; N_{\tilde{u}}) : [0, 1] \rightarrow \mathbb{R}^{N_{\tilde{u}}}$ as vector of polynomial basis functions, where $N_{\tilde{u}} \in \mathbb{N}$ denotes the number of degrees of freedom of the input parameterization. Then, by defining $U \in \mathcal{U} \subseteq \mathbb{R}^{n_{\tilde{u}}N_{\tilde{u}}}$ as a finite-valued collection of control parameters, we can reformulate our continuous-time control signal to a finite-dimensional equivalent

$$\tilde{u}(\tau) = (\Gamma(\tau) \otimes I_{n_{\tilde{u}}})U. \quad (4.8)$$

Remark 4.3. In previous works [37], the parameterization U is split into components for u and s comprising $\tilde{u}$, due to the fact that the latter component requires the restriction $s(\tau) > s_{\min} > 0$. For notational simplicity, we do not make this distinction, however require that this constraint must still be captured by set $\mathcal{U}$.

Example 4.1 (First-order hold parameterization). Under a first-order hold parameterization, the number of control parameters is equal to the number of knot points, such that $N_{\tilde{u}} = N$, and the basis function $\Gamma(\tau; N) = (\Gamma_k(\tau))_{k=1}^N$ can be specified as follows:

$$\Gamma_k(\tau) = \begin{cases} \frac{\tau - \tau_{k-1}}{\tau_k - \tau_{k-1}}, & \tau \in [\tau_{k-1}, \tau_k], k \in \{2, \dots, N\}, \\ \frac{\tau_{k+1} - \tau}{\tau_{k+1} - \tau_k}, & \tau \in [\tau_k, \tau_{k+1}], k \in \{1, \dots, N-1\}, \\ 0, & \text{else.} \end{cases} \quad (4.9)$$

Going forward, we will assume a uniform first-order hold parameterization on all control parameterizations. We can now state the discretization of the dynamics constraint (4.7b) by defining the function $F_k : \mathbb{R}^{n_{\tilde{x}}} \times \mathbb{R}^{n_{\tilde{x}}} \times \mathbb{R}^{n_{\tilde{u}}N_{\tilde{u}}} \rightarrow \mathbb{R}^{n_{\tilde{x}}}$ such that

$$F_k(\tilde{x}_{k+1}, \tilde{x}_k, U; \Gamma) = \tilde{x}_{k+1} - \tilde{x}_k - \int_{\tau_k}^{\tau_{k+1}} F(\tilde{x}(\tau), (\Gamma(\tau) \otimes I_{n_{\tilde{u}}})U) d\tau, \quad (4.10)$$

where we enforce $F_k(\tilde{x}_{k+1}, \tilde{x}_k, U) = 0, \forall k \in [N-1]$, to satisfy the original continuous-time

dynamics across the optimization horizon. We make use of the inverse-free exact discretization method [37, 59] with multiple-shooting transcription to simultaneously linearize the dynamics model during discretization. Consider a discrete-time reference $\bar{Z} = (\bar{X}, \bar{U})$, with $\bar{X} = (\bar{x}_1, \dots, \bar{x}_N)$ as the reference state and $\bar{U}$ as the reference control parameterization. We can solve a collection of initial value problems (IVPs) on the system's integrated state ϕ and state transition matrices Φ_x, Φ_u over the interval $[\tau_k, \tau_{k+1})$, stated as

$$\dot{\phi}(\tau) = F(\phi(\tau), (\Gamma(\tau) \otimes I_{n_{\bar{u}}})\bar{U}), \quad (4.11a)$$

$$\dot{\Phi}_x(\tau) = A(\tau; \bar{Z}, \Gamma)\Phi_x(\tau), \quad (4.11b)$$

$$\dot{\Phi}_u(\tau) = A(\tau; \bar{Z}, \Gamma)\Phi_u(\tau) + \Gamma(\tau) \otimes B(\tau; \bar{Z}, \Gamma), \quad (4.11c)$$

$$\phi(\tau_k) = \bar{x}_k, \quad (4.11d)$$

$$\Phi_x(\tau_k) = I_{n_{\bar{x}}}, \quad (4.11e)$$

$$\Phi_u(\tau_k) = 0_{n_{\bar{x}} \times n_{\bar{u}} N_{\bar{u}}}. \quad (4.11f)$$

Where the continuous-time jacobians A, B are defined as

$$A(\tau; \bar{Z}, \Gamma) = \nabla_{\bar{x}} F(\phi(\tau), (\Gamma(\tau) \otimes I_{n_{\bar{u}}})\bar{U}), \quad (4.12a)$$

$$B(\tau; \bar{Z}, \Gamma) = \nabla_{\bar{u}} F(\phi(\tau), (\Gamma(\tau) \otimes I_{n_{\bar{u}}})\bar{U}). \quad (4.12b)$$

Finally, we can express the first-order linearization of equation (4.10) with the function $\Psi_k : \mathbb{R}^{n_{\bar{x}}} \times \mathbb{R}^{n_{\bar{x}}} \times \mathbb{R}^{n_{\bar{u}} N_{\bar{u}}^j} \rightarrow \mathbb{R}^{n_{\bar{x}}}$ such that

$$\Psi_k(\tilde{x}_{k+1}, \tilde{x}_k, U; \bar{Z}, \Gamma) = \tilde{x}_{k+1} - w_k(\bar{Z}, \Gamma) - A_k(\bar{Z}, \Gamma)(\tilde{x}_k - \bar{x}_k) - B_k(\bar{Z}, \Gamma)(U - \bar{U}), \quad (4.13)$$

where the terms

$$w_k(\bar{Z}, \Gamma) = \phi(\tau_{k+1}^-), \quad (4.14a)$$

$$A_k(\bar{Z}, \Gamma) = \Phi_x(\tau_{k+1}^-), \quad (4.14b)$$

$$B_k(\bar{Z}, \Gamma) = \Phi_u(\tau_{k+1}^-), \quad (4.14c)$$

represent the integrated solutions to equations (4.11a)-(4.11f) evaluated at final time τ_{k+1}^- . To improve notation going forwards, we will represent these terms in block form such that we have

$$0 = C_k(\bar{Z}, \Gamma)v_k - c_k(\bar{Z}, \Gamma), \quad (4.15a)$$

$$v_k = (\tilde{x}_{k+1}, \tilde{x}_k, U), \quad (4.15b)$$

$$C_k(\bar{Z}, \Gamma) = (I_{n_{\bar{x}} \times n_{\bar{x}}}, -A_k(\bar{Z}, \Gamma), -B_k(\bar{Z}, \Gamma)), \quad (4.15c)$$

$$c_k(\bar{Z}, \Gamma) = w_k(\bar{Z}, \Gamma) - A_k(\bar{Z}, \Gamma)\bar{x}_k - B_k(\bar{Z}, \Gamma)\bar{U}. \quad (4.15d)$$

4.3.2 Vertex Segments

With the machinery of our parameterization & discretization method established, we can now apply this approach towards the structure of `GRAPH-SCvx`. Consider discretization of a vertex segment $\tilde{x}^j(\tau), \tilde{u}^j(\tau)$ for $j \in \mathcal{V}$, with $\tau^j \in [0, 1]$, onto a finite grid of size $N^j \in \mathbb{N}$, with $0 = \tau_1^j < \dots < \tau_N^j = 1$ and $\tilde{X}^j = (\tilde{x}_1^j, \dots, \tilde{x}_N^j)$. We can also define control parameters $U^j \in \mathcal{U}^j \subseteq \mathbb{R}^{n_{\bar{u}}N_u^j}$ and basis function $\Gamma^j(\tau) : [0, 1] \rightarrow \mathbb{R}^{N_u^j}$ such that

$$\tilde{u}^j(\tau) = (\Gamma^j(\tau) \otimes I_{n_{\bar{u}}})U^j. \quad (4.16)$$

We can also state our segment-specific reference trajectory as $\bar{Z}^j = (\bar{X}^j, \bar{U}^j)$. Then, we can state the nonlinear discretization of our vertex dynamics function, along with its linearized counterpart terms, as

$$F_k^j(\tilde{x}_{k+1}^j, \tilde{x}_k^j, U^j) = F_k(\tilde{x}_{k+1}^j, \tilde{x}_k^j, U^j; \Gamma^j), \quad (4.17a)$$

$$\Psi_k^j(\tilde{x}_{k+1}^j, \tilde{x}_k^j, U^j) = \Psi_k(\tilde{x}_{k+1}^j, \tilde{x}_k^j, U^j; \bar{Z}^j, \Gamma^j), \quad (4.17b)$$

$$C_k^j(\bar{Z}^j) = C_k(\bar{Z}^j, \Gamma^j), \quad (4.17c)$$

$$c_k^j(\bar{Z}^j) = c_k(\bar{Z}^j, \Gamma^j). \quad (4.17d)$$

Assuming a uniform first-order hold control (FOH) parameterization, we can state $\Gamma^j(\tau) = \Gamma(\tau; N_u^j)$ under the form in example 4.1.

4.3.3 Edge Connections

Under certain control parameterizations, we may additionally require control continuity in addition to state continuity. Note that this does not necessarily imply continuous-time control continuity must hold such that $u^j(t_1^j) = u^i(t^{i \rightarrow j})$; for example, in the case of a discontinuous control parameterization, such as a zero-order hold (ZOH), this does not hold. In the case of our assumed uniform FOH parameterization, we can state the control continuity constraint according to example 4.2.

Example 4.2 (*Edge continuity under FOH parameterization*). Under a uniform FOH parameterization, we can state the control continuity constraint as

$$U_1^j = U_{k_i \rightarrow j}^i. \quad (4.18)$$

We will broadly state this joint relation between parent-child pairs of an edge under the constraint set $(U^i, U^j) \in \mathcal{U}^{ij}$ for an edge $\varepsilon_{ij} \in \mathcal{E}$.

4.3.4 Discrete-time Problem Formulation

We now have the necessary components to describe the discrete-time formulation of problem (4.7) as follows:

Discretized Optimal Control Problem (4.19)

$$\underset{\substack{\tilde{X}^j, U^j \\ j \in \mathcal{V}}}{\text{minimize}} \quad L(\tilde{x}_N) \quad (4.19a)$$

$$\text{s. t. } F_k^j(\tilde{x}_{k^j+1}^j, \tilde{x}_{k^j}^j, U^j) = 0, \quad j \in \mathcal{V}, \quad (4.19b)$$

$$\theta E(\tilde{x}_{k^j+1}^j - \tilde{x}_{k^j}^j) \leq \epsilon_\theta, \quad j \in \mathcal{V}, \quad (4.19c)$$

$$P^j(\tilde{x}_1^j, \tilde{x}_{N^j}^j) \leq 0, \quad j \in \mathcal{V}, \quad (4.19d)$$

$$Q^j(\tilde{x}_1^j, \tilde{x}_{N^j}^j) = 0, \quad j \in \mathcal{V}, \quad (4.19e)$$

$$U^j \in \mathcal{U}^j, \quad j \in \mathcal{V}, \quad (4.19f)$$

$$k^j = 1, \dots, N^j - 1, \quad j \in \mathcal{V}, \quad (4.19g)$$

$$\tilde{x}_1^j - \tilde{x}_{k_{i \rightarrow j}}^i = 0, \quad \varepsilon_{ij} \in \mathcal{E}, \quad (4.19h)$$

$$(U^i, U^j) \in \mathcal{U}^{ij}, \quad \varepsilon_{ij} \in \mathcal{E}, \quad (4.19i)$$

where $\tilde{x}_N = (\tilde{x}_{N^j}^j)_{j \in \mathcal{V}}$. We note that constraint (4.19c) is relaxed from equality by a constant $\epsilon_\theta \sim 1e^{-4}$ for satisfaction of linear independence constraint qualification (LICQ) – we refer the reader to [37] for more details. In the results, we show that this relaxation enables solutions which exhibit only very small path constraint violations.

4.4 Sequential Convex Programming

In this section, we consider application of a sequential convex programming (SCP) algorithm, where problem (4.19) is solved through an iteration such that nonconvex constraints are approximated as convex constraints through linearization, and convex constraints are preserved and imposed directly. Specifically, we consider the prox-linear method [34, 71] for convex-composite minimization, where problem (4.19) is converted to an unconstrained minimization problem via the ℓ_1 exact penalization.

4.4.1 Exact Penalization

When studying the structure of problem (4.19), we observe three classes of constraints that we can treat separately: 1. convex constraints; 2. nonconvex equality constraints;

and 3. nonconvex inequality constraints. Consider a concatenation of decision variables as $z = (\tilde{X}^1, \dots, \tilde{X}^n, U^1, \dots, U^n) \in \mathbb{R}^{n_z}$. To impose convex constraints, we leverage the set indicator function $\mathbb{1}_{\mathcal{Z}} : \mathbb{R}^{n_z} \rightarrow \{0, \infty\}$ over convex set $\mathcal{Z}$ such that $\mathbb{1}_{\mathcal{Z}}(z) = 0 \iff z \in \mathcal{Z}$. We define $\mathcal{Z} = \tilde{\mathcal{X}}^{\mathcal{V}} \times \tilde{\mathcal{X}}^{\mathcal{E}} \times \mathcal{U}^{\mathcal{V}} \times \mathcal{U}^{\mathcal{E}}$ as a closed convex set where

$$\tilde{\mathcal{X}}^{\mathcal{V}} := \left\{ (\tilde{X}^1, \dots, \tilde{X}^n) \left| \begin{array}{l} \theta E(\tilde{x}_{k^j+1}^j - \tilde{x}_{k^j}^j) \leq \epsilon_{\theta}, \\ k^j = 1, \dots, N^j - 1, \\ j = 1, \dots, n. \end{array} \right. \right\}, \quad (4.20a)$$

$$\tilde{\mathcal{X}}^{\mathcal{E}} := \left\{ (\tilde{X}^1, \dots, \tilde{X}^n) \left| \begin{array}{l} \tilde{x}_1^j - \tilde{x}_{k_{i \rightarrow j}}^i = 0, \\ \forall \varepsilon_{ij} \in \mathcal{E}. \end{array} \right. \right\}, \quad (4.20b)$$

$$\mathcal{U}^{\mathcal{V}} := \mathcal{U}^1 \times \dots \times \mathcal{U}^n, \quad (4.20c)$$

$$\mathcal{U}^{\mathcal{E}} := \Pi_{\varepsilon_{ij} \in \mathcal{E}} \mathcal{U}^{ij}. \quad (4.20d)$$

In the above, $\tilde{\mathcal{X}}^{\mathcal{V}}$ corresponds to convex vertex constraints (4.19c), $\tilde{\mathcal{X}}^{\mathcal{E}}$ corresponds to convex edge constraint (4.19h), $\mathcal{U}^{\mathcal{V}}$ corresponds to vertex control parameterization constraints (4.19f), and $\mathcal{U}^{\mathcal{E}}$ corresponds to edge control parameterization constraints (4.19i). We also require convexity of the objective function $\tilde{L}$. Next, we can define residual terms associated with nonconvex constraints as

$$R^{F_k^j}(z) = F_k^j(\tilde{x}_{k^j+1}^j, \tilde{x}_{k^j}^j, U^j), \quad (4.21a)$$

$$R^{P^j}(z) = P^j(\tilde{x}_1^j, \tilde{x}_{N^j}^j), \quad (4.21b)$$

$$R^{Q^j}(z) = Q^j(\tilde{x}_1^j, \tilde{x}_{N^j}^j). \quad (4.21c)$$

Then, we can express concatenated residual terms as

$$R_e = \left((R^{F_k^j})_{k \in [N^j-1]}^{j \in [n]}, (R^{Q^j})_{j \in [n]} \right), \quad (4.22a)$$

$$R_i = (R^{P^j})_{j \in [n]}. \quad (4.22b)$$

In the above, $R_e \in \mathbb{R}^{n_e}$ represents a concatenation of nonconvex equality constraints (4.19b) and (4.19e), and $R_i \in \mathbb{R}^{n_i}$ represents a concatenation of nonconvex inequality constraints

(4.19d). Then, given a weighting term $\gamma > 0$, we can state the ℓ_1 -penalized unconstrained minimization problem corresponding to problem (4.19) with the function $\Theta_\gamma : \mathbb{R}^{n_z} \rightarrow \mathbb{R}$ such that

$$\Theta_\gamma(z) = L_z(z) + \mathbb{1}_{\mathcal{Z}}(z) + \gamma(\|R_e(z)\|_1 + \mathbf{1}^\top |R_i(z)|_+), \quad (4.23)$$

where $L_z(z) = \tilde{L}(\tilde{x}_N)$. Our goal is to obtain the necessary conditions to compute a local minimizer of Θ_γ , noting that Θ_γ is nonsmooth due to ℓ_1 penalization. Specifically, we seek to obtain a stationary point $z^\star$ such that $0 \in \partial\Theta_\gamma(z^\star)$. Assume the gradients of $\tilde{L}, R_e, R_i$ are bounded on $\mathcal{Z}$. we define the convex subdifferential as follows

$$\partial|w| \triangleq \begin{cases} \{1\}, & w > 0, \\ [-1, 1], & w = 0, \\ \{-1\}, & w < 0, \end{cases} \quad (4.24a)$$

$$\partial|w|_+ \triangleq \begin{cases} \{1\}, & w > 0, \\ [0, 1], & w = 0, \\ \{0\}, & w < 0. \end{cases} \quad (4.24b)$$

Next, define $\mu \in \mathbb{R}^{n_z}$ and $\eta = (\eta_e, \eta_i) \in \mathbb{R}^{n_R}$ to express terms within the subdifferential of $\Theta_\gamma(z^\star)$ as

$$\mu \in \partial\mathbb{1}_{\mathcal{Z}}(z^\star), \quad (4.25a)$$

$$\eta_e \in \partial|R_e(z^\star)|, \quad (4.25b)$$

$$\eta_i \in \partial|R_i(z^\star)|_+. \quad (4.25c)$$

Then, from [30, Prop. 10.15, Thm 10.20], a point $z^\star$ is stationary if there exists some μ, η which satisfies

$$0 = \nabla_z L_z(z) + \mu^\top + \gamma \eta_e^\top \nabla_z R_e(z) + \gamma \eta_i^\top \nabla_z R_i(z), \quad (4.26)$$

Given that z^* and $\mathcal{Z}$ obey the conditions of [37, Lemma 13], we can use [37, Lemma 15] combined with equation (4.26) to show that, if z^* is a stationary point of Θ_γ and is feasible with respect to the constraints of problem (4.19), then it must also be a KKT point of problem (4.19). In other words, if γ is sufficiently large enough to produce a solution such that residuals $R_e(z^*), R_i(z^*) = 0$, then z^* is a KKT point of the original constrained problem. This property of exact penalization, among others, allows us to directly minimize for equation (4.23) to retrieve a solution to problem (4.19), which we will exploit further in the following subsection.

4.4.2 The Prox-linear Method

We now consider the prox-linear method [34, 71] for convex-composite minimization, summarizing the description in [37] for clarity, with modifications for our specific implementation. The prox-linear method finds a stationary point z^* of Θ_γ by iteratively minimizing a convex approximation $\Theta_\gamma^\rho : \mathbb{R}^{n_z} \rightarrow \mathbb{R}$ such that, in the case of ℓ_1 penalization, we have

$$\begin{aligned} \Theta_\gamma^\rho(z; \bar{z}) &= L_z(z) + \mathbb{1}_{\mathcal{Z}}(z) + \rho \|z - \bar{z}\|_2^2 \\ &\quad + \gamma \|R_e(\bar{z}) + \nabla_z R_e(\bar{z})(z - \bar{z})\|_1 \\ &\quad + \gamma \mathbf{1}^\top |R_i(\bar{z}) + \nabla_z R_i(\bar{z})(z - \bar{z})|_+, \end{aligned} \quad (4.27)$$

where $\bar{z}$ serves as a reference solution (either the previous iteration's solution for iterate $k \geq 2$, or an initial guess in the case of $k = 1$), and $\rho > 0$ serves as the proximal term weight. Notably, we must express the gradients and residuals associated with our dynamics terms as

$$\nabla_z R^{F_k^j}(\bar{z}) = C_k^j(\bar{Z}^j) M_k^j{}^\top, \quad (4.28a)$$

$$R^{F_k^j}(\bar{z}) = C_k^j(\bar{Z}^j) M_k^j{}^\top \bar{z} - c_k^j(\bar{Z}^j). \quad (4.28b)$$

where $M_k^j \in \mathbb{R}^{n_z \times (2n_{\bar{x}} + N_u^j)}$ represent allocation matrices, composed of sparse block identity matrices, satisfying $v_k^j = M_k^j{}^\top z$. Then, rather than minimizing (4.27) directly, we instead

solve the following constrained convex subproblem in practice:

$$\underset{z, \nu_e^-, \nu_e^+, \nu_i}{\text{minimize}} \quad L_z(z) + \gamma \mathbf{1}^\top (\nu_e^-, \nu_e^+, \nu_i) + \rho \|z - \bar{z}\|_2^2 \quad (4.29a)$$

$$\text{s. t.} \quad R_e(\bar{z}) + \nabla_z R_e(\bar{z})(z - \bar{z}) = \nu_e^+ - \nu_e^-, \quad (4.29b)$$

$$R_i(\bar{z}) + \nabla_z R_i(\bar{z})(z - \bar{z}) \leq \nu_i, \quad (4.29c)$$

$$\nu_e^-, \nu_e^+, \nu_i \geq 0, \quad (4.29d)$$

$$z \in \mathcal{Z}. \quad (4.29e)$$

In problem (4.29), slack variables ν_e^-, ν_e^+, ν_i are added to ensure smoothness of the objective and constraint functions. We denote a solution to problem (4.29) with the notation $z, \nu_e^-, \nu_e^+, \nu_i = \text{subproblem}(\bar{z})$. Algorithm 4.1 describes an overview of the prox-linear method specific to our implementation, where $k_{\max}$ represents the maximum number of iterations, and $\epsilon_\gamma, \epsilon_\rho > 0$ represent the convergence tolerances for the relaxed exact penalization and proximal terms, respectively.

Algorithm 4.1 Prox-linear Method

Require: $z_1, k_{\max}, \epsilon_\gamma, \epsilon_\rho, \gamma, \rho$

Ensure: z_k

```

1:  $k \leftarrow 1$ 
2: converged  $\leftarrow$  false
3: while  $k \leq k_{\max}$  and not converged do
4:    $z_{k+1}, \nu_e^-, \nu_e^+, \nu_i \leftarrow \text{subproblem}(z_k)$ 
5:   if  $\mathbf{1}^\top (\nu_e^-, \nu_e^+, \nu_i) \leq \epsilon_\gamma$  and  $\|z_{k+1} - z_k\|_2^2 \leq \epsilon_\rho$  then
6:     converged  $\leftarrow$  true
7:   end if
8:    $k \leftarrow k + 1$ 
9: end while
```

4.5 Deferred-Decision Trajectory Optimization

In this section, we consider the original continuous-time graph trajectory optimization problem (4.1), and assert the necessary structure to impose the deferred-decision trajectory optimization (DDTO) problem class. We consider an initial condition set $\mathcal{Z}^0 \subset \mathbb{R}^{n_x}$ and a set of candidate terminal condition (*target*) sets $\mathcal{Z}^j \subset \mathbb{R}^{n_x}$, $j \in [n_{\text{targ}}]$. The objective is to

maximize the amount of time such that multiple targets in the target collection are guaranteed to remain reachable (under nominal conditions, i.e. without process uncertainty), such that the decision to commit to a target in the reachable set may be *deferred*.

4.5.1 Graph Structure

The DDTO problem statement is broadly described by an arborescence (otherwise known as directed rooted tree) graph structure.

Definition 4.1 (Arborescence). *An arborescence defined by graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ contains a single source node $v_0 \in \mathcal{V}$ that has a unique directed path to every other node $v \in \mathcal{V} \setminus \{v_0\}$ in the graph.*

We consider a specific instantiation of this tree-like structure, containing one shared trunk segment, denoted by the index 0 with trajectory $(x^0(t^0), u^0(t^0))$, and $n_{\text{targ}} \in \mathbb{N}$ branch segments, each corresponding to a desired target and connected to the trunk segment, denoted by the index $j \in [n_{\text{targ}}]$ with trajectory $(x^j(t^j), u^j(t^j))$. This results in a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ such that $\mathcal{V} = (0, 1, \dots, n_{\text{targ}})$ and $\mathcal{E} = (\varepsilon_{01}, \dots, \varepsilon_{0n_{\text{targ}}})$. Stitching is performed such that the initial condition of each branch segment connects to an intermediate point of the trunk segment. For a target j , this corresponds to an edge ε_{0j} with trunk-relative deferral time $t^{0 \rightarrow j}$, and branch-relative connection time $t^{j \leftarrow 0} = t_1^j$. We stitch targets together in a predefined ordering characterized by $\lambda = (\lambda_1, \dots, \lambda_{n_{\text{targ}}})$, where $\lambda_j \in [n_{\text{targ}}]$ indexes a target, such that

$$t^{0 \rightarrow \lambda_1} \leq t^{0 \rightarrow \lambda_2} \leq \dots \leq t^{0 \rightarrow \lambda_{n_{\text{targ}}-1}} = t^{0 \rightarrow \lambda_{n_{\text{targ}}}} = t_f^0.$$

Under this restriction, the final two branches in the ordering shall always defer from the end of the trunk segment. We refer to this application of GRAPH-SCvx to the DDTO problem as GRAPH-DDTO going forward.

4.5.2 Objective & Constraints

We now discuss formulation of the objective and constraint functions specific to GRAPH-DDTO, formulated for the expression of the original OCP (4.1). The problem objective is to jointly maximize for a weighted sum of deferral times $t^{0 \rightarrow j}$, $\forall j \in [n_{\text{targ}}]$ under weighting parameters

$\alpha^j \in \mathbb{R}_+$. This objective can be considered *event-driven* and is evaluated at finitely-many intermediate times along the trunk; for simplicity, we opt to describe this objective directly on the discrete-time problem as

$$L_z(z) = - \sum_{j=1}^{n_{\text{targ}}} \alpha^j E^t \bar{x}_{k^0 \rightarrow j}^0. \quad (4.30)$$

Additionally, we would like to constrain the cumulative path to each target to be ϵ^j -suboptimal relative to some independent optimality metric, which we denote in Mayer form as $l_s(x_f^j)$ evaluated for a target j , with a reference optimality $\bar{l}_s^j$ obtained from a converged single-target solution corresponding to target j , computed *a priori*. Then, the suboptimality constraint is defined as

$$l_s(x_f^j) \leq (1 + \epsilon^j) \bar{l}_s^j, \quad j \in [n_{\text{targ}}], \quad (4.31)$$

where the above assumes $l_s(x_f^j), \bar{l}_s^j \geq 0$ (valid for most practical cases, such as a minimum-fuel reference objective). We would also like to impose global time-varying state $\mathcal{X}(t) \subseteq \mathbb{R}^{n_x}$ and input $\mathcal{U}(t) \subseteq \mathbb{R}^{n_u}$ path constraints. We can implicitly define path constraint functions g^j, h^j such that the following constraints

$$x^j(t^j) \in \mathcal{X}(t^j), \quad j \in \mathcal{V}, \quad (4.32a)$$

$$u^j(t^j) \in \mathcal{U}(t^j), \quad j \in \mathcal{V}, \quad (4.32b)$$

are satisfied for all $t^j \in [t_i^j, t_f^j]$. Additionally, we implicitly define boundary condition functions p^j, q^j such that the following constraints

$$x^0(t_i^0) \in \mathcal{Z}^0, \quad (4.33a)$$

$$x^j(t_f^j) \in \mathcal{Z}^j, \quad j \in \mathcal{V} \setminus \{0\}, \quad (4.33b)$$

are satisfied, in addition to suboptimality constraint (4.31), for t_i^0, t_f^j .

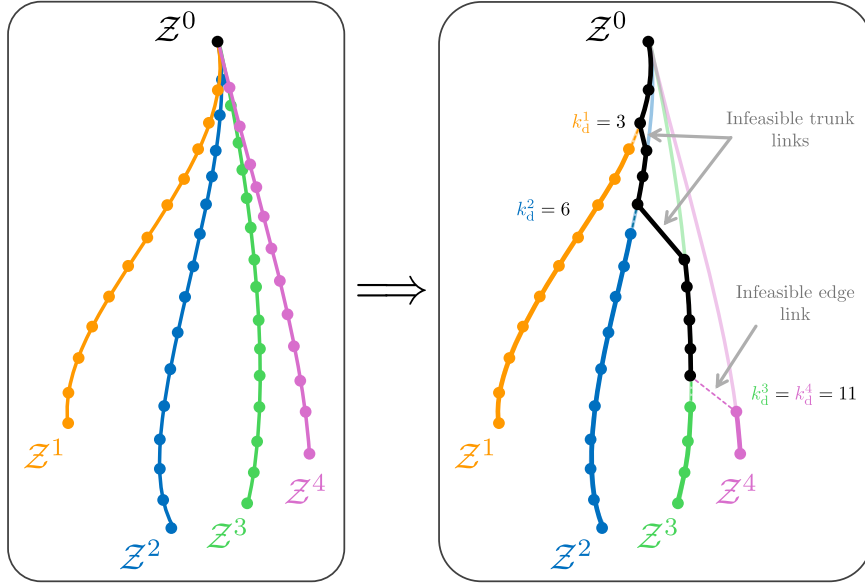


Figure 4.3: Illustration of the GRAPH-DDTO warmstart generation process described by algorithm 4.2. **Left:** four single-target solutions to terminal condition sets $\mathcal{Z}^1$ - $\mathcal{Z}^4$. **Right:** the converted warmstart for GRAPH-DDTO, where the black segment denotes the trunk and the colored segments denote the branches, with deferral indices $k_1^d, \dots, k_4^d$ and $\lambda = (1, 2, 3, 4)$.

4.5.3 Initialization

A key characteristic of the prox-linear method is the requirement of an initial guess to serve as the linearization reference for the first iterate of the algorithm. We seek to find a general-purpose formulation for an initial guess of the DDTO problem such that:

1. We can make efficient use of previously-computed information.
2. We preserve dynamic feasibility in the initial guess for as many dynamic links (knot point pairs) as possible.

We propose a candidate approach where we start by computing n_{targ} single-target solutions from $\mathcal{Z}^0$ to each target $\mathcal{Z}^j$, $j \in [n_{\text{targ}}]$. We recall that these solutions were already necessary to obtain reference optimalities $\bar{l}_s^j$ for constraint (4.31). To maintain consistency with GRAPH-SCvx, we use the CT-SCvx algorithm to generate single-target solutions, denoted as $\text{CT-SCvx}(\mathcal{Z}^j)$ for a solution to target j . For simplicity, we describe this ap-

proach assuming a uniform first-order hold control parameterization, but note it applies fundamentally to other parameterizations. We will denote the corresponding state solutions for single-target trajectories as $\bar{X}_s^j = (\bar{x}_{s,1}^j, \dots, \bar{x}_{s,N_s^j}^j)$ with corresponding control parameterizations $\bar{U}_s^j = (\bar{U}_{s,1}^j, \dots, \bar{U}_{s,N_s^j}^j)$. From these quantities, we would like to obtain the GRAPH-DDTO warmstart reference $\bar{X}^j = (\bar{x}_1^j, \dots, \bar{x}_{N^j}^j)$ with control parameterizations $\bar{U}^j = (\bar{U}_1^j, \dots, \bar{U}_{N^j}^j)$. In other words, we would like to convert n_{targ} single-target trajectories to $n_{\text{targ}} + 1$ trajectory segments corresponding to the trunk and branches of a GRAPH-DDTO problem.

The resulting warmstart approach is described in algorithm 4.2, with a corresponding visual illustration provided in figure 4.3. The algorithm proceeds through the targets in the deferral order parameterized by λ by extracting snippets of the single-target trajectories to form the trunk, and constructing the branches directly with the last N^j knot points of each single-target trajectory. We will denote $k_d^j = k_{0 \rightarrow j}$ going forward. Then, given single target solutions with grid sizes N_s^j , we require the grid size of each GRAPH-DDTO branch to be equal to $N^j = N_s^j - k_d^j + 1, \forall j \in [n_{\text{targ}}]$. This means that the full GRAPH-DDTO trajectory towards each target (trunk up to index k_d^j concatenated to the remaining portion of the branch) takes on the same concatenated grid size as its corresponding single-target solution, ensuring compatibility with our warmstart approach.

Algorithm 4.2 GRAPH-DDTO Warmstart

Require: $k_d^1, \dots, k_d^{n_{\text{targ}}}$
Ensure: $\bar{X}^j, \bar{U}^j, \bar{l}_s^j$
 1: $k_p \leftarrow 1$
 2: **for** $j \in (\lambda_1, \dots, \lambda_{n_{\text{targ}}})$ **do**
 3: $\bar{X}_s^j, \bar{U}_s^j, \bar{l}_s^j \leftarrow \text{CT-SCvx}(\mathcal{Z}^j)$
 4: $\bar{X}^j \leftarrow \bar{X}_{s, k_d^j: N_s^j}^j$
 5: $\bar{U}^j \leftarrow \bar{U}_{s, k_d^j: N_s^j}^j$
 6: **if** $j \neq \lambda_{n_{\text{targ}}}$ **then**
 7: $\bar{X}_{k_p: k_d^j}^0 \leftarrow \bar{X}_{s, k_p: k_d^j}^j$
 8: $\bar{U}_{k_p: k_d^j}^0 \leftarrow \bar{U}_{s, k_p: k_d^j}^j$
 9: $k_p \leftarrow k_d^j + 1$
 10: **end if**
 11: **end for**

To analyze this approach, we can define a feasible dynamical *link* as a pair of knot point states $\tilde{x}_1, \tilde{x}_2$ and control parameterization U which satisfies the relation $F(\tilde{x}_1, \tilde{x}_2, U) = 0$ according to the form of equation (4.10) (stated without additional notation for vertices and edges). Then, we can observe that this approach contains two salient properties: 1. only $(n_{\text{targ}} - 1)$ links in the output $\tilde{X}^j, \tilde{U}^j, j \in [n]$ are dynamically infeasible; and 2. the trunk initial guess always *chases* the next target for deferral, meaning only 1 of the aforementioned links corresponds to an edge of the graph. We have tested many different initialization techniques, including fixed-final-time quasiconvex DDTO [38], single-target initialization with the trunk naively shooting at only one target, and linear interpolations on state and control space to a geometric mean between the initial condition and targets $\{\mathcal{Z}^0, \mathcal{Z}^1, \dots, \mathcal{Z}^{n_{\text{targ}}}\}$. In our benchmarking, we have found algorithm 4.2 to consistently provide the most robust warmstart in terms of minimizing prox-linear iterations, and as such will be used to warmstart GRAPH-DDTO in all upcoming demonstrations.

4.6 Adaptive-DDTO

We now discuss an extension of DDTO modeling which considers an augmentation and operationalization of DDTO in online, real-time maneuvering of unknown and hazardous environments. In such environments, we may consider selecting a collection of target ter-

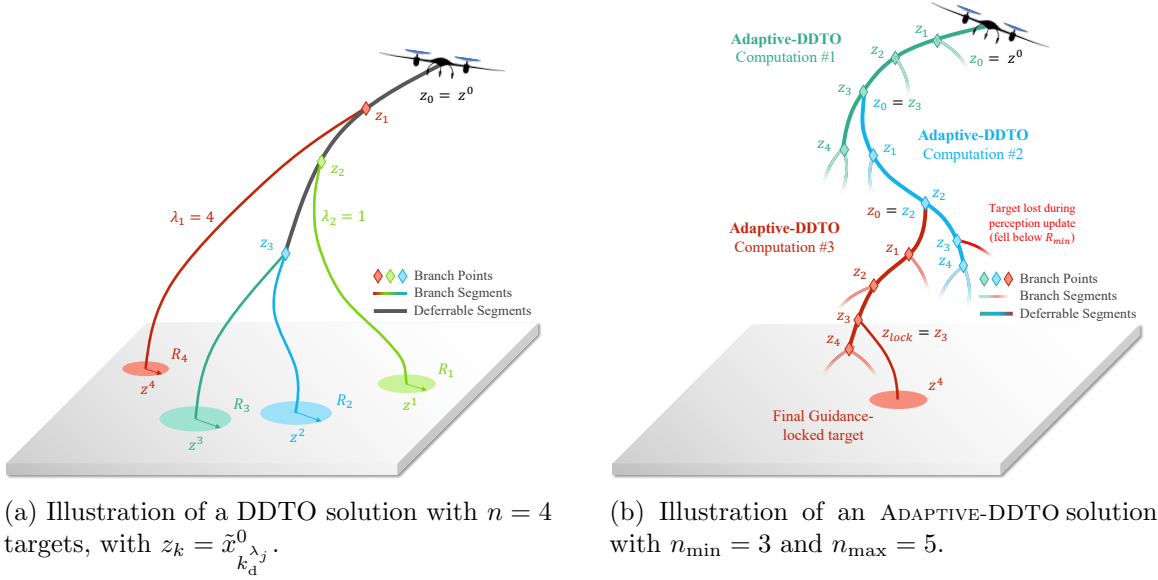


Figure 4.4: Comparison between DDTO and ADAPTIVE-DDTO in the quadrotor soft landing scenario.

minal conditions $\mathcal{Z}^j$ given information on the environment. For example, consider the autonomous soft landing problem; if we can parameterize safe subsets of the terrain, then we can select a collection of safe prospective targets and defer our decision to commit to any particular one. During said maneuver, targets may become unavailable for a number of reasons:

1. Updated perceptive information dictates the target is no longer safe.
2. The decision deferral (branch) point for a target has been reached, at which point either the target is deferred to (only the deferred target remains reachable) or the vehicle decides to remain on the deferrable (trunk) segment (all but the deferred target remains reachable).

Under these considerations, the viable target count may drop below an acceptable threshold or be eliminated entirely. We present the ADAPTIVE-DDTO algorithm [52, Algorithm 1] to ensure a bounded number of targets, $n_{\min} \leq n_{\text{targ}} \leq n_{\max}$, remain in the reachable set of the vehicle at all times during the maneuver. In this section, we will denote target

sets as pointwise equality constraints in position with zero terminal velocity, such that $\mathcal{Z}^j = \{(r, v) \in \mathbb{R}^3 \times \mathbb{R}^3 | r = z^j, v = 0\}$. Each target also has a safety radius, $R^j(t)$, associated with the feasible landing region at any point in time (tracked by a backend perception algorithm). These radii will be implicitly considered a (dynamically-updated) parameter of z^j to simplify notation going forward. Recall that DDTO requires specification of a target preference order $\lambda = (\lambda_1, \dots, \lambda_{n_{\text{targ}}})$; to determine this preference order, a desirability score is assigned to each target and represented by the evaluation function $g_{\text{des}} : \mathbb{R}^{n_x} \rightarrow [0, \infty)$. A proposed heuristic for this evaluation function is a weighted sum $g_{\text{des}}(z^j) = w_{\text{des}}^\top c_{\text{des}}$, where $c_{\text{des}} \in \mathbb{R}^4$ contains scores related to LiDAR point cloud density, proximity to vehicle, site clustering, and site radius $R^j(t)$, and $w_{\text{des}} \in \mathbb{R}^4$ contains corresponding tunable weights for each element of c_{des} .

The function $g_{\text{rem}} : \mathbb{R}^{n_x} \rightarrow \{0, 1\}$ determines if a target should be removed or not, and returns **true** if the target bounding radii falls below the minimum safe bounding radius, denoted $g_{\text{rem}}(z^j) = R^j < R_{\text{min}}$. The switching decision at a branch point is represented by a function $g_{\text{switch}} : \mathbb{R}^{n_x} \rightarrow \{0, 1\}$ and returns **true** if the desirability score associated with the target in consideration for deferral exceeds all other targets' scores, such that $g_{\text{switch}}(z^j) = g_{\text{des}}(z^j) > \max(\{g_{\text{des}}(z^i)\}_{i \in \{J/j\}})$.

Under certain conditions, it may be desirable to *lock* the guidance to the best available target $z^{j_{\text{lock}}}$, disabling ADAPTIVE-DDTO re-computations and applying control towards this target until the terminal condition $z^{j_{\text{lock}}}$ is reached. A reasonable locking condition based on a minimum “cutoff” altitude h_{cut} could be used, such that $g_{\text{lock}} : \mathbb{R} \rightarrow \{0, 1\}$ is defined with $g_{\text{lock}}(h) = h < h_{\text{cut}}$. To index into the altitude of $\tilde{x}_k^0$, the vector $e_h = [0, 0, 1, 0, 0, 0, 0, 0]^\top$ is used.

There are also several other useful functions that will be considered for the ADAPTIVE-DDTO formulation:

- $h_{\text{acquire}} : \mathbb{N} \rightarrow 2^{\mathbb{R}^3}$ returns $n_{\text{max}} - n$ new targets from the perception module (not already in the current set).
- $h_{\text{update}} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ calls from a perception module to update target z^j and associated

properties (radius and scoring metrics).

- $\text{sort_perm}_n : \mathbb{R}^n \rightarrow \mathbb{N}^n$ sorts the input collection of cardinality $n \in \mathbb{N}$ in monotonically decreasing order and returns the permutation indices that indicate the sorting order.
- $\text{comm_control} : (\mathbb{R}^{n_u^j}) \rightarrow \emptyset$ extracts and commands a parameterized control signal associated with the j th trajectory segment (no output since the control signal is realized on the vehicle's true state).

Algorithm 4.3 ADAPTIVE-DDTO

Require: $z^0, J, \{z^j\}_{j \in J}$

- 1: $j_{\text{lock}} \leftarrow -1$
- 2: $J_+ \leftarrow \{1, \dots, n_{\text{max}}\} \setminus J$
- 3: $\{z^j\}_{j \in J_+} \leftarrow h_{\text{acquire}}(|J_+|)$
- 4: $J \leftarrow J \cup J_+$
- 5: $\{\lambda_j\}_{j=1}^{n_{\text{max}}} \leftarrow \text{sort_perm}_n(\{g_{\text{des}}(z^j)\}_{j \in J})$
- 6: $\tilde{X}, U = \text{GRAPH-DDTO}(z^0, \{z^j\}_{j \in J}, \{\lambda_j\}_{j \in J}, \{\epsilon^j\}_{j \in J})$
- 7: **for** $k = 1, \dots, N^0$ **do**
- 8: **for** $j \in J$ **do**
- 9: $z^j \leftarrow h_{\text{update}}(z^j)$
- 10: **if** $g_{\text{rem}}(z^j)$ **then**
- 11: $J \leftarrow J \setminus \{j\}$
- 12: **end if**
- 13: **if** $k = k_{0 \rightarrow j}$ **and** $g_{\text{switch}}(z^j)$ **then**
- 14: $J \leftarrow \{j\}$
- 15: $j_{\text{lock}} \leftarrow j$
- 16: **else if** $k = k_{0 \rightarrow j}$ **then**
- 17: $J \leftarrow J \setminus \{j\}$
- 18: **end if**
- 19: **end for**
- 20: **if** $|J| < n_{\text{min}}$ **then**
- 21: $\tilde{X}, U \leftarrow \text{ADAPTIVE-DDTO}(\tilde{x}_k^0, J, \{z^j\}_{j \in J})$
- 22: **end if**
- 23: **if** $g_{\text{lock}}(e_h^\top \tilde{x}_k^0)$ **then**
- 24: $j_{\text{lock}} \leftarrow \arg\max(\{g_{\text{des}}(z^j)\}_{j \in J})$
- 25: **end if**
- 26: **if** $j_{\text{lock}} \neq -1$ **then**
- 27: $\text{comm_control}(U_{k-k_{0 \rightarrow j_{\text{lock}}}+1}^{j_{\text{lock}}})$
- 28: **else**
- 29: $\text{comm_control}(U_k^0)$
- 30: **end if**
- 31: **end for**

Algorithm 4.3 notably contains recursive calls to itself for *adaptive* re-computations when $n < n_{\text{min}}$. Here, z^0 represents the pointwise initial condition and $J \subseteq \{1, \dots, n_{\text{targ}}\}$ refers to a target indexing set. $\text{GRAPH-DDTO}()$ receives the initial condition z^0 and target parameters z^j, λ^j and $\epsilon^j, \forall j \in n_{\text{targ}}$, and returns the full graph trajectory bundle $\tilde{X}, U$. We note that, while we show loop logic executed at a frequency corresponding to the discretization grid

for notational simplicity, in practice these logical checks will be executed at a much higher frequency.

4.7 Experimental Results

In this section, we consider a multitude of tests that demonstrate the capabilities and performance of GRAPH-DDTO. We divide this investigation into three categories focused on quadrotor trajectory planning: 1. comparison of GRAPH-DDTO against a predecessor DDTO-based algorithm on challenging obstacle avoidance scenarios; 2. comparison of GRAPH-DDTO against several classes of landing divert maneuvers in the context of safe hazard-aware landing; and 3. hardware experiments for landing in environments with dynamic (moving) ground obstacles in a netted cage environment.

4.7.1 Implementation Details

In all forthcoming results, we leverage a Julia-based package designed to solve GRAPH-DDTO problems with optimizations for real-time performance. We make use of the JuMP optimization parser [35] and the Clarabel second-order cone program (SOCP) solver [46] to solve prox-linear convex subproblem (4.29). We leverage ForwardDiff, an autodifferentiation package, to obtain dynamics gradients (4.12a),(4.12b) and boundary condition gradients applied directly to residuals (4.21b),(4.21c). We also use OrdinaryDiffEq to solve the collection of IVPs described by equations (4.11a)-(4.11f), with CPU parallelization enabled and applied to batched dynamical links across $\mathcal{G}$. In practice, we make use of the trajectory stitching constraints introduced in appendix A to remove edge variable redundancies introduced in problem (4.1). Simulations were executed on a workstation with an AMD Ryzen 9 7950X3D CPU (16 cores, 32 threads, 5.8 GHz), 128 GB RAM, running Ubuntu 22.04 LTS. Julia was configured to use 16 threads. The code used to generate all results for section 4.7.2, and which serves as the GRAPH-DDTO backend for sections 4.7.3-4.7.4, exists as a Julia-based package and is provided at:

<https://github.com/bucknercsamuel/DDTOTOolbox.jl>

4.7.2 DDTO Method Comparison

We first seek to establish GRAPH-DDTO in relation to the state-of-the-art in nonconvex DDTO modeling, for which we consult the algorithm highlighted in [39, Algorithm 2]. This approach

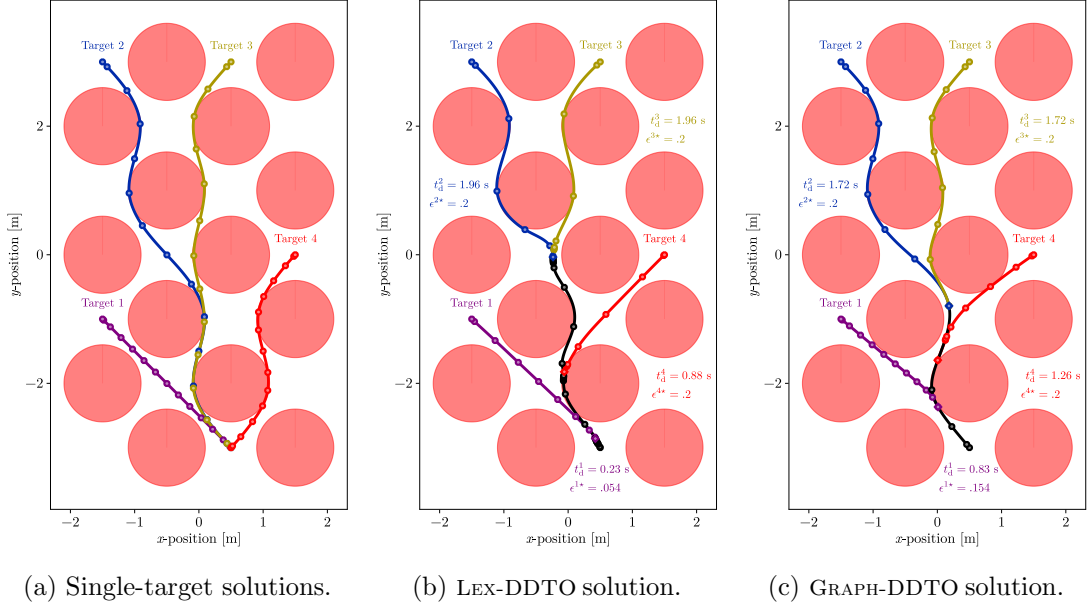


Figure 4.5: Comparison across three different trajectory bundles generated by 1. single-target planning; 2. LEX-DDTO; and 3. GRAPH-DDTO. Deferral times, $t_d^j = t^j(\tau_d^j)$, and suboptimality to each target with respect to the corresponding single-target solution, $\epsilon^{j\star}$, are displayed for the latter two cases. We use a global suboptimality upper bound of $\epsilon^j = 0.2$ and uniform GRAPH-DDTO deferral time weighting of $\alpha^j = 1$, $\forall j \in [n_{\text{targ}}]$.

considers a lexicographical optimization in recursive stages, where targets are dropped from the DDTO target set according to deferrability ordering λ in the transition to every subsequent stage. Additionally, this approach aims to convert the problem to a canonical single-target form such that the trunk and branch trajectories are directly concatenated on top of each other, resulting in a problem which can be solved directly with `CT-SCvx`. We refer to this approach as the LEX-DDTO method going forwards.

We conduct a detailed analysis of LEX-DDTO and GRAPH-DDTO by considering the static obstacle avoidance problem for quadrotors introduced by [127]. We consider state $x^j(t) \triangleq [r^j(t)^\top, v^j(t)^\top]^\top \in \mathbb{R}^6$ representing the concatenated position and velocity of the quadrotor, and control $u^j(t) \in \mathbb{R}^3$ representing the thrust produced by the vehicle. We define these vectors in an inertial $\{e_x, e_y, e_z\}$ basis. The dynamics can be described by a simple affine

double-integrator of the form $f(x^j(t)u^j(t)) = Ax^j(t) + Bu^j(t) + g$, where

$$A = \begin{bmatrix} 0_{3 \times 3} & I_3 \\ 0_{3 \times 3} & 0_{3 \times 3} \end{bmatrix}, B = \begin{bmatrix} 0_{3 \times 3} \\ I_3/m \end{bmatrix}, \quad (4.34)$$

where m represents the mass of the quadrotor vehicle and g represents the gravitational acceleration. We also impose a collection of path constraints on the vehicle, including: maximum vertical (z) and lateral (x - y) velocity magnitude bounds; minimum (nonconvex) and maximum thrust magnitude bounds; thrust direction restricted about a second-order cone relative to the e_z direction; a constant-altitude constraint to restrict vehicle motion to a 2D plane; and (nonconvex) obstacle avoidance constraints imposed on the position of the vehicle. Details on how each constraint is formed are provided in [127]. The boundary conditions represent an initial and terminal position, and zero velocity at both initial and terminal times. We configure 16 large circular obstacles such that the feasible state-space is highly restricted as a challenge case.

Figure 4.5 shows a direct comparison between LEX-DDTO and GRAPH-DDTO for a difficult obstacle avoidance scenario with $n_{\text{targ}} = 4$. We can observe that LEX-DDTO implicitly favors the last two targets to be deferred (targets 2 and 3 in this case), achieving a higher deferrability time for these targets. In contrast, GRAPH-DDTO exhibits greater deferrability for the first two targets, with a more even spread over deferral times due to equal weighting in the GRAPH-DDTO objective. When evaluating on said objective, such that we compute the cumulative deferral time $\sum_j t_d^j$, we see that LEX-DDTO achieves a total cumulative deferral time of 5.03 s, whereas GRAPH-DDTO achieves a time of 5.58 s, representing a +10.4% improvement in terms of the objective. We note that this objective difference is a result of a different deferral objective; lexicographic DDTO greedily optimizes for the target deferrability in the order of λ , whereas GRAPH-DDTO directly optimizes for the total deferral time (when $\alpha^j = 1, \forall j \in [n]$), exploring the pareto-optimal frontier of the solution space. We also observe maximum path constraint violation errors (on the nonlinear propagations of our trajectories) of $3.233e^{-4}$ and $1.437e^{-4}$ for LEX-DDTO and GRAPH-DDTO, respectively. While this value is slightly larger than our CTCS tolerance of $\epsilon_\theta = 1e^{-4}$, the extra gap

can be explained by the fact that SCP only converges to numerical tolerance of ϵ_γ and ϵ_ρ ; the fact that constraint violations fall within one order of magnitude provides a strong indicator of the quality of the solutions and empirically validates the CTCS relaxation in section 4.3.4.

Perhaps more interesting in the context of real-time planning, however, is the difference in cumulative solve time. GRAPH-DDTO was able to solve this problem in 2.07 s, however LEX-DDTO required 38.99 s, representing a $\times 18.8$ multiple increase in solve time. Table 4.1 provides an overview of the algorithm differences that led to this discrepancy. First, recall that LEX-DDTO solves a lexicographic sequence of optimization problems (stages), while reducing the target count by one for each new stage. This means the dimension of the decision variable of each stage is scaled by $(n, n-1, \dots, 3)$, resulting in $\mathcal{O}(n^2)$ composite complexity with respect to decision variables when considering all stages together. On the contrary, GRAPH-DDTO must only solve one optimization problem scaled by n . We note that, while this statement does not qualify the complexity of the underlying convex subproblem, we observe subproblem solve-times scaling closely to the decision variable size for these problem structures. Furthermore, when concatenating all trajectories together, evaluation of the dynamics jacobians during linearization takes on $\mathcal{O}(n^2)$ complexity, whereas considering trajectories independently only requires $\mathcal{O}(n)$ complexity. This discrepancy results from the fact that jacobians between different segments are trivially zero ($\nabla_{x^i, u^i} f(x^j, u^j) = 0, \forall i \neq j$); the latter can exploit that structure, while the former cannot without explicit customization of the discretization. Beyond these algorithm complexity differences, we also observe that LEX-DDTO must explicitly concatenate branches to the trunk due to lack of a stitching constraint formulation, resulting in n_{targ} redundant knot points (see proposition A.1 in the appendix). We note the impact of this modeling choice scales by the sparsity of the discretization grid. A further consequence of this concatenated approach is that the discretization & parameterization of all segments must match (such that $N^i = N^j, \forall i, j \in [n]$), resulting in extraneous trunk knot points and violating the requirements of the warmstarting technique in algorithm 4.2². To facilitate a more detailed analysis, we performed a Monte

²LEX-DDTO is warmstartable in principle but not with the identical simultaneous graph warmstart as in

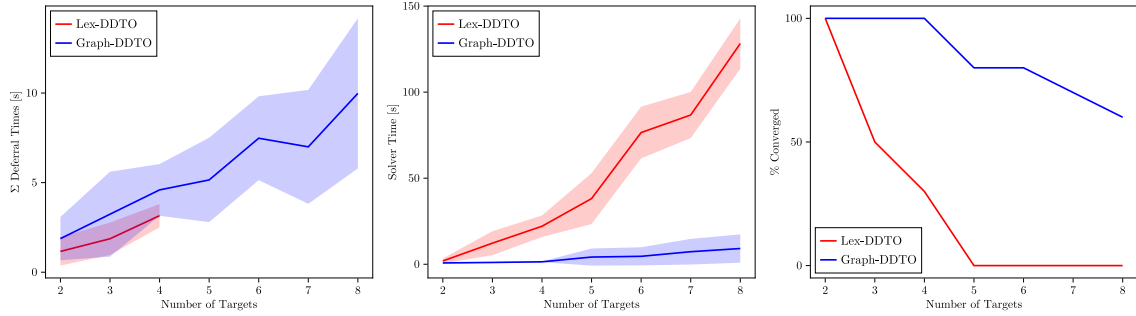


Figure 4.6: Monte Carlo comparison between LEX-DDTO and GRAPH-DDTO across multiple cumulative deferral times, algorithm solve time and percent convergence. For deferral and solve times, the mean and $1\text{-}\sigma$ uncertainty are displayed by a solid line and a transparent band about the line, respectively.

Table 4.1: DDTO algorithm comparison.

	LEX-DDTO	GRAPH-DDTO
Decision variable complexity	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
Linearization complexity	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
Minimal discretization	✗	✓
Flexible knot allocation	✗	✓
Warmstartable (alg. (4.2))	✗	✓

Carlo scanning over target counts $n_{\text{target}} = 2, \dots, 8$ with 10 trials of randomly-generated targets per each target count setpoint. To generate targets, we randomly sample in the feasible position space, set the target velocity to zero, and order the targets in terms of λ according to their distance from the initial condition (such that the closest target is deferred first). Figure 4.6 provides an overview of the results. When looking at cumulative solver time across all prox-linear iterates, we can see evidence of quadratic scaling for LEX-DDTO, whereas GRAPH-DDTO scales much more linearly. We can also observe that GRAPH-DDTO exhibits improved robustness in terms of prox-linear convergence, particularly as the target counts increase (we mark a run as unconverged if the algorithm exceeds $k_{\text{max}} = 100$ prox-linear iterates for any nonconvex problem in the algorithm loop). Finally, we observe a consistent algorithm 4.2.

Table 4.2: Ablation study on GRAPH-DDTO warmstart.

Method	No obstacles	Obstacles
Linear	0.35 ± 0.015 s	12.00 ± 7.40 s
Q _{CVX} -DDTO	0.42 ± 0.11 s	3.90 ± 5.25 s
Alg. 4.2	0.37 ± 0.06 s	2.92 ± 4.51 s

tently larger deferrability objective evaluation for GRAPH-DDTO relative to LEX-DDTO (we only consider converged solutions at this step, since unconverged solutions may not display accurate optimization metrics).

Ablation study: Graph-DDTO warmstart

To evaluate the benefits of using algorithm 4.2 to generate a warmstart for GRAPH-DDTO, we perform an ablation study on the total solve time of GRAPH-DDTO with this warmstart, relative to a linear-interpolation-based warmstart (standard for SCP methods), and a Q_{CVX}-DDTO warmstart [38, Algorithm 2]. We consider cases of both no obstacles and all obstacles present in this section’s obstacle avoidance problem, over 10 trials of randomly-generated targets with $n_{\text{targ}} = 4$. In the no-obstacles case, we observe very similar performance across all three methods, however in the challenge case with obstacles, we see a moderate improvement in solve time for algorithm 4.2 over Q_{CVX}-DDTO, and substantial improvement over a linear-interpolation-based warmstart. Thus, we conclude that algorithm 4.2 is the most versatile warmstart to use going forwards.

4.7.3 Trade Study: Hazardous Soft Landing

With GRAPH-DDTO’s strengths and real-time capabilities established in relation to the LEX-DDTO technique, we now move towards an important application of real-time deferred decision-making: soft landing of a quadcopter in a hazardous and unknown environment. As was mentioned in the introduction, an important motivation behind decision-deferral is to obtain information on the environment through perceptive sensing. To this end, we consider a quadrotor equipped with a LiDAR sensor which utilizes a perceptive backend to acquire

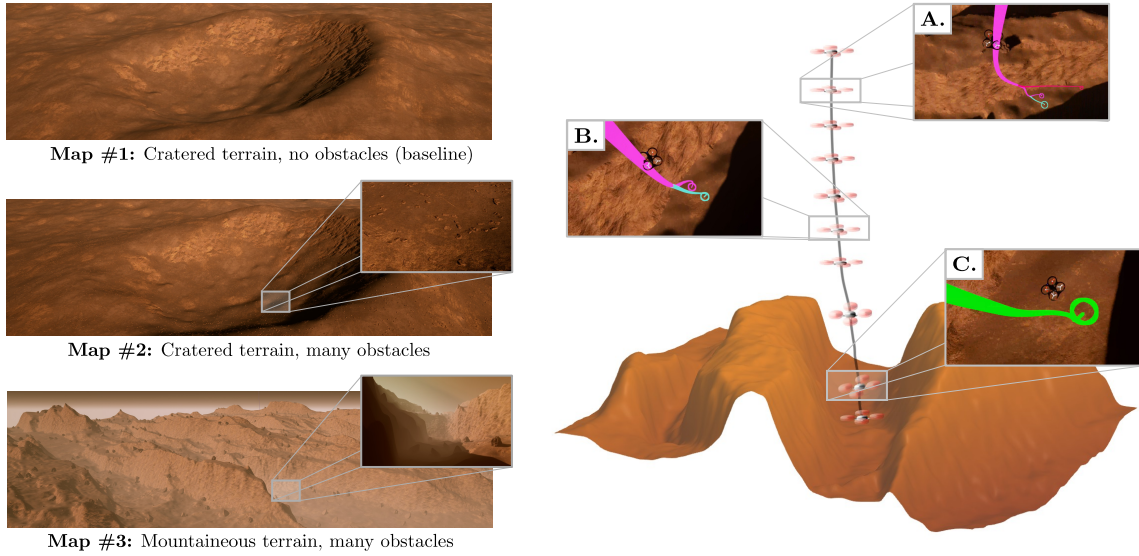


Figure 4.7: Left: visual clarification of the different demonstration maps tested on in AirSim. Right: Demonstration of an Adaptive-DDTO landing maneuver on map #3 with snapshots indicating: (A) initial descent with a full ($n_{\text{targ}} = 3$) GRAPH-DDTO solution; (B) midway through descent wherein a target has been lost due to exceeding its deferral time (branched off); (C) final descent locked to a single target.

and track multiple targets which have been deemed *safe*. The Hazard-Aware Landing Optimization (HALO) framework [52] is a joint planning & perception approach designed for quadrotor landing in hazardous and unknown environments. In this framework, there exist two core autonomy modules to safely execute the maneuver.

The first module, **Hazard-Aware Landing Site Selection** (HALSS), obtains the n_{targ} safest circular landing sites in the terrain environment by constructing a world-frame binary map (indicating safe & unsafe regions) and selecting centroid-radius pairs (z^j, R^j) , $j \in n_{\text{targ}}$ which maximize non-overlapping coverage of the safe region. Landing site radii are constantly tracked and updated as new perceptive measurements are made. HALSS is implemented to interface with the ADAPTIVE-DDTO module, described by algorithm 4.3. To determine GRAPH-DDTO's contribution to the perception-aware autonomous landing literature, we benchmark the algorithm, wrapped into ADAPTIVE-DDTO, against a class of single-target divert-based techniques.

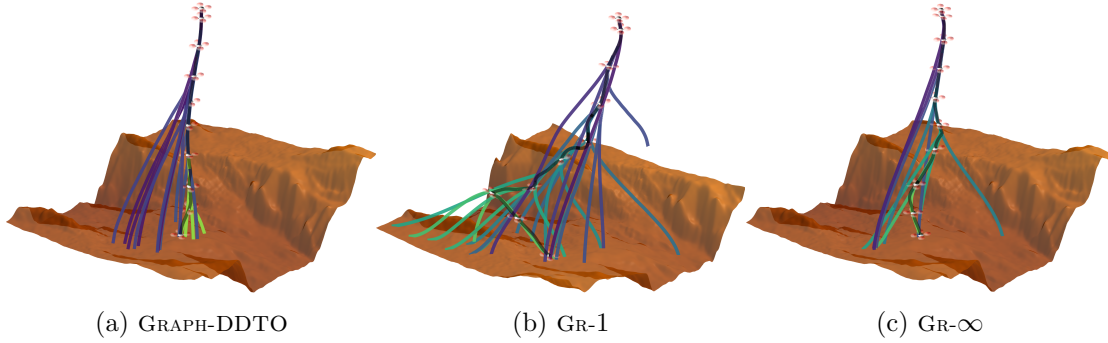


Figure 4.8: Comparison of descent profiles between GRAPH-DDTO and comparative methods for a selected problem scenario on map #3. Divert trajectories are color-coded according to temporal progress at the time they were computed relative to the overall maneuver duration.

Problem definition

Because the quadrotor is capable of descending at high velocities, we now consider a non-linear dynamics model which extends on the double integrator model in section 4.7.2 by including an aerodynamic drag term. This model is stated as follows:

$$\dot{r}^j(t) = v^j(t) \quad (4.35a)$$

$$\dot{v}^j(t) = \frac{u^j(t)}{m} - \frac{\rho_{\text{air}} C_d S_A \|v^j(t)\|_2 v^j(t)}{2m} + g \quad (4.35b)$$

Where $\rho_{\text{air}}, C_d, S_A \in \mathbb{R}$ are constants corresponding to the local air density, coefficient of drag and frontal surface area of the quadrotor, respectively. In terms of path constraints, in contrast to section 4.7.2, we do not impose obstacle avoidance or constant altitude constraints. Terminal conditions are specified as point-wise equality constraints such that $\mathcal{Z}^j = \{(r, v) \in \mathbb{R}^6 \mid r = z^j, v = 0_3\}$.

Real-time HALO implementation

To conduct a realistic and high-fidelity analysis, we present a real-time implementation of HALO built on top of the ROS 2 framework [79] with Julia’s PackageCompiler for fast startup. The provided software framework now operates in real-time, and hosts a number of updates to both the HALSS and ADAPTIVE-DDTO modules.

The first major update to HALSS, in relation to [52], is the inclusion of Gaussian-distributed range-dependent noise to LiDAR measurements to improve realism of the sensor model. In addition, we modified the world map constructor to be adaptively resizable, and gave HALSS a finite-horizon memory such that it will discard measurements beyond a certain backward-horizon window (circular buffer holding a fixed number of point cloud measurements). This feature is critical to discard higher-noise measurements obtained at a higher altitude, allowing the map to improve as the quadrotor descends. We also took the fine hazard detection technique [52, Section III.B] and used it to replace the previous Bayesian network-based coarse hazard detection at a lower binning resolution, such that the entire module no longer relies on a learning process. Finally, many of the processes in HALSS were optimized in implementation to enable the algorithm to run at ~ 20 hz for real-time target updates.

In terms of ADAPTIVE-DDTO, the primary update concerns wrapping GRAPH-DDTO into the DDTO solver call in line 8 of [52, Algorithm 1], replacing the previous quasiconvex DDTO solver from [38, Algorithm 2]. This replacement enables us to directly embed nonlinear dynamics and nonconvex path constraints into the problem description to produce higher-fidelity trajectories, and additionally enables all deferral events and terminal landing times to be directly optimized over.

The Gr-M algorithm

Given the new HALO framework wrapped around GRAPH-DDTO, we now seek to draw comparisons to established autonomous soft landing techniques in the literature. To do

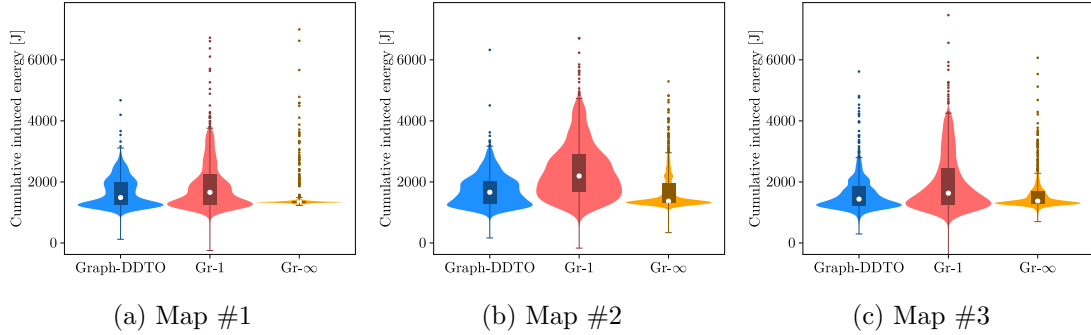


Figure 4.9: Monte Carlo statistics of induced energy between GRAPH-DDTO, GR-1 and GR- ∞ across all three demonstration maps. Data is visualized using violin plots characterizing the sampled probability distribution, along with quantile information displayed by box plot overlays and outliers (defined according to table 4.3) plotted with dots outside the box plot ranges.

this, we define a class of single-target retargeting techniques defined as the greedy- M , or GR- M , algorithm class.

Algorithm 4.4 provides an overview of the GR- M algorithm, which notably repurposes the functionalities of HALSS to acquire the single best target when requested through the function HALSS_TARGET(), returning the target position $z \in \mathbb{R}^3$ and safe radius $R \in \mathbb{R}$. The function CT-SCvx(z) computes a single-shot solution towards target z from the current quadrotor location to produce a trajectory $(x(t), u(t))$. During execution of the descent maneuver, the target safe radius is tracked with a call to HALSS_UPDATE(). If the target radius falls below the minimum threshold $R_{\min}$, or if an M -rate retargeting is scheduled, then a new target is obtained, with CT-SCvx called recursively to generate a new trajectory. Similarly to ADAPTIVE-DDTO, this process is repeated until the altitude r_z of the quadrotor falls below a cutoff altitude h_{cut} , at which point the current target is locked with no more retargeting diverts permitted.

This technique can be thought of as a simplification of the ADAPTIVE-DDTO algorithm (where only a single target is tracked), and is principally conditioned on the choice of the retargeting frequency M . In fact, we consider two special cases of GR- M as a reflection of

the related literature in section 4.1.1. $\text{GR-}\infty$ acts as a faithful real-time analog of the approach outlined by the G-FOLD algorithm and its successors [2, 32, 118], where diverts are only commanded when a target becomes unsafe (meaning the second condition on line 6 of algorithm 4.4 is never activated). GR-1 , on the other hand, represents an opportunistic diversion approach reflected by [41, 47], where diverts are commanded at the fastest frequency allowable (we found that single-target trajectories tend to solve at a rate of 1 – 3hz, so we upper bound the divert period at $M = 1\text{s}$). We note that these are not direct implementations of the aforementioned algorithms in the literature, but can be considered a controlled ablation study of the divert policy alone (all methods share identical HALSS backend, dynamics and proxlinear solver numerics). We do not draw direct comparisons to the original HALO framework, as it was not capable of performing landing in real-time (required simulation pauses during ADAPTIVE-DDTO divert calls), nor do we compare against LEX-DDTO as it was already benchmarked head-to-head in the previous section; the proceeding analysis is instead conducted to position HALO/ADAPTIVE-DDTO against the broader literature of single-target planning algorithms.

Algorithm 4.4 $\text{GR-}M$

Require: $M, h_{\text{cut}}, \Delta t$

```

1:  $t \leftarrow 0, t_p \leftarrow 0$ 
2:  $z, R \leftarrow \text{HALSS\_TARGET}()$ 
3:  $x(t), u(t) \leftarrow \text{CT-SCvx}(z)$ 
4: while  $r_z \geq h_{\text{cut}}$  do
5:    $R \leftarrow \text{HALSS\_UPDATE}()$ 
6:   if  $R \leq R_{\text{min}}$  or  $t - t_p \geq M$  then
7:      $z, R \leftarrow \text{HALSS\_TARGET}()$ 
8:      $x(t), u(t) \leftarrow \text{CT-SCvx}(z)$ 
9:      $t_p \leftarrow t$ 
10:  end if
11:   $t \leftarrow t + \Delta t$ 
12: end while
```

Table 4.3: Monte Carlo comparison across various metrics.

Algorithm	$\downarrow$ Induced Energy [J] ¹	$\uparrow$ Safe Radius [m] ¹	$\downarrow$ # Diverts ¹	$\uparrow$ % Success
Map #1				
GR-1	1987.52 $\pm$ 1091.71 (27)	11.92 $\pm$ 5.24 (8)	12.13 $\pm$ 2.80 (77)	78.15
GR- ∞	1568.58 $\pm$ 685.97 (77)	9.77 $\pm$ 4.65 (28)	2.15 $\pm$ 2.24 (75)	92.05
GRAPH-DDTO	1694.01 $\pm$ 884.68 (7)	6.83 $\pm$ 3.65 (28)	2.52 $\pm$ 2.64 (17)	93.82
Map #2				
GR-1	2457.18 $\pm$ 1197.61 (18)	5.38 $\pm$ 1.78 (10)	13.51 $\pm$ 3.15 (37)	61.04
GR- ∞	1765.79 $\pm$ 745.55 (44)	4.84 $\pm$ 1.87 (10)	3.79 $\pm$ 3.88 (22)	86.46
GRAPH-DDTO	1764.54 $\pm$ 679.09 (10)	3.74 $\pm$ 1.37 (13)	3.30 $\pm$ 1.01 (13)	90.83
Map #3				
GR-1	2057.18 $\pm$ 1272.44 (18)	5.39 $\pm$ 2.78 (11)	14.43 $\pm$ 5.92 (37)	75.41
GR- ∞	1645.62 $\pm$ 664.31 (61)	5.03 $\pm$ 2.40 (16)	4.00 $\pm$ 4.16 (30)	89.84
GRAPH-DDTO	1662.49 $\pm$ 671.62 (32)	3.76 $\pm$ 1.85 (13)	3.91 $\pm$ 2.87 (40)	90.45

¹ Data displayed as mean $\pm$ standard deviation (# outliers²)

² Outliers are defined relative to the interquartile range (IQR = $Q_3 - Q_1$), where Q_3, Q_1 are the 75th and 25th percentiles of the data distribution. The outlier set [27] is defined as $\mathcal{O} = \{d \in \mathcal{D} | d \leq Q_1 - 1.5\text{IQR}\} \cup \{d \in \mathcal{D} | d \geq Q_3 + 1.5\text{IQR}\}$, where d defines a datapoint in dataset $\mathcal{D}$.

Comparisons to Graph-DDTO

With the GR-1 and $\text{GR-}\infty$ algorithms established, we can now investigate how they compare to ADAPTIVE-DDTO with a GRAPH-DDTO backend. Figure 4.7 shows a demonstration of ADAPTIVE-DDTO and illustrates the three terrain environments (maps) we have selected to investigate for algorithm comparisons. Map #1 is chosen as a baseline and represents the simplest environment configuration with no small obstacles. Maps #2 and #3 represent challenge cases for landing, wherein the quadrotor must land in cluttered environments with many small obstacles (e.g. rocks and boulders) and significant elevation changes (such as craters and mountains). This data was extracted from the NASA HiRISE dataset representing DEMs from the Martian surface, and simulation is carried out using the Microsoft AirSim plugin to Unreal Engine 4.

In the interest of comparison, we would like to evaluate how these algorithms balance safety with energy efficiency, while noting that these two objectives are not necessarily uncorrelated. Moreover, the quadrotor could remain suspended in the air indefinitely if energy was not a concern, so the ability to perform an energy-efficient landing maneuver is paramount to ensuring energy allocation constraints are satisfied. While we do not have direct access to an onboard battery model, we can evaluate induced power using momentum theory [57], which is a standard first-order model for rotor power while neglecting other terms such as motor losses, blade profile and parasitic drag. This model can be stated as

$$P_i = \frac{\|T(t)\|_2^{3/2}}{\sqrt{2\rho_{\text{air}}A_r}}, \quad (4.36)$$

where A_r is the rotor disk area. If we integrate the induced power P_i over the duration of the maneuver, we can obtain the induced energy E_i of the maneuver, and for a given scenario, require that the induced energy falls below an energy budget $E_i \leq E_{\text{max}}$. We note that for both GR-1 and $\text{GR-}\infty$, cumulative thrust is imposed as the optimization objective since it can be parsed into standard SOCP solvers; for GRAPH-DDTO , cumulative thrust is imposed as the single-target objective, and the algorithm is constrained to be ϵ^j -suboptimal to each single-target trajectory, with $\epsilon^j = 0.05$, $\forall j \in [n_{\text{targ}}]$.

In an effort to capture safety considerations directly, we evaluate several additional performance metrics. First, we consider the radius of the largest available target landing site at the cutoff altitude h_{cut} , hence referred to as the *safe radius* of the maneuver. After this point, there is no further distinction between algorithms as the quadrotor must commit to a single trajectory with no further divers. From an operational perspective, we also consider the total number of divers required over a maneuver to be a point of contention, as each new divert incurs additional risk in terms of algorithm convergence and tracking performance.

Figure 4.8 displays a side-by-side demonstration of each algorithm for a selected scenario on map #3. We can see that GR-1 and GR- ∞ engage in larger and more numerous divers over the course of their respective maneuvers, incurring more lateral motion and, consequently, more energy expenditure. Moreover, these algorithms achieve induced energies of 3191.08 J and 1717.57 J, respectively, relative to GRAPH-DDTO which achieves a more desirable 1420.97 J, validating our observations. All algorithms otherwise behave similarly in terms of safe radius, and each require substantially more divers than GRAPH-DDTO (which only requires four divers), resulting in a safer and more energy-efficient maneuver.

Monte Carlo analysis

To provide a comprehensive analysis of our algorithm relative to its competitors, we perform comparisons across 500 Monte Carlo samples of initial position (initial velocity always starts at 0, i.e. hover condition), perturbed across the lateral boundaries of each terrain map at a fixed above ground level (AGL) altitude of 150 m³. Table 4.3 shows statistical data for all aforementioned metrics, as well as an analysis of the number of runs that were deemed successful⁴. We define a successful run as one which: 1. induced energy falls below $E_{\text{max}} = 2500$ J; 2. obtains a safe cutoff radius above the minimum allowable radius

³We encountered simulation runtime errors in some of the runs, so we filtered out their corresponding randomization seeds from the analysis. This left us with 453, 480 and 492 valid seeds for maps #1, #2 and #3, respectively.

⁴Statistical data is aggregated across all runs, not just successful runs

$R_{\min} = 1$ m; and 3. does not contain any instances where a divert maneuver failed to converge. We can consequently see that GRAPH-DDTO manages to achieve the highest success rate of the three algorithms across all tested maps.

Figure 4.9 provides a visualization of the induced energy sample probability distributions for each algorithm across all demonstration maps. From this figure, we can observe that GRAPH-DDTO: (i) tends to noticeably outperform GR-1 in terms of mean performance, and (ii) tends to noticeably outperform GR- ∞ in terms of outlier performance. We observe these improvements in spite of the fact that GRAPH-DDTO trajectories are necessarily ϵ^j -suboptimal in order to produce deferrable (trunk) segments. This deferrability is observed to implicitly lead to more energy-efficient motion in the case that the perception map (and thus safe landing site distribution) characteristics are changing rapidly, by avoiding unnecessary diverts over the course of the maneuver. From the data in table 4.3, we can observe that, across both challenge maps, GR-1 requires 31.50% more induced energy to complete a maneuver relative to GRAPH-DDTO on average, which we attribute to aggressive back-and-forth diverts at high resolve rates. On the other end, we observe that GR- ∞ tends to have almost the same mean induced energy as GRAPH-DDTO (within 1.1% on both maps), however contains $2.5\times$ as many outliers. This is due to the fact that GR- ∞ only diverts as-needed, achieving more stable performance, but may require large-scale diverts in the case that a newly-found unsafe target was placed under consideration for too long, leading to inferior worst-case performance and a less unimodal thrust distribution.

Across the other two safety metrics (safe radius and number of diverts), we observe that GRAPH-DDTO and GR- ∞ tend to perform similarly, with GR-1 exhibiting the most variability in terms of both metrics. Moreover, because GR-1 constantly seeks out the best available target, it exits the planning recursion with a higher safe radius at the cost of substantially more diverts. Ultimately, we observe that GRAPH-DDTO is able to achieve the highest success rate of the three algorithms across all tested maps when considering all three metrics, with GR- ∞ achieving the second-highest success rate consistently. We observe that the induced energy requirement is the most important metric that contributes to these success rates, followed by the number of diverts (which ultimately contributes to higher convergence failure

rates for GR-1). While the safe radius requirement is the least important metric (since the drone is able to consistently land within a relatively small safe radius of 1 m), it is still important to note the trade-off in terms of radius margin which could become more relevant under a more unstable landing configuration and environment.

Finally, in an attempt to obtain the most accurate comparison between algorithms, we note that differences in algorithm solve time are captured in the results due to the fact that the new HALO framework is real-time. This means that the commanded trajectory to track with a lower-level PID controller only updates when the solver obtains a new solution (this tracked trajectory refers to the trunk segment in the case of GRAPH-DDTO). We observe that GRAPH-DDTO takes, on average, 1.27 ± 0.66 s to converge (worst-case: 4.67 s), whereas GR-1 takes 0.40 ± 0.53 s (worst-case: 2.15 s) and GR- ∞ takes 0.88 ± 0.92 s (worst-case: 2.35 s), noting that GR- ∞ experiences much more solve-time variability due to larger diverts later in the maneuver.⁵ Thus, even in spite of GRAPH-DDTO’s increased computational overhead and latency, it is able to show improved performance over the other two techniques across most metrics.

Ablation study: energy budget

To ensure we are providing a fair analysis of our algorithm over a variety of energy budgets (representative of different landing scenarios), we perform an ablation study on the energy budget parameter $E_{\max}$, and study how this parameter affects the success rate of each algorithm on each map. Table 4.4 shows the results of this ablation study. We can observe that, when the energy budget is tight ($E_{\max} = 2000$ J), the tighter distribution and mean performance of GR- ∞ enable it to achieve a higher success rate than GRAPH-DDTO across all maps. However, when the energy budget is relaxed such that $E_{\max} = \{2500, 3000\}$ J, the degraded outlier performance of GR- ∞ causes this relation to flip such that GRAPH-DDTO achieves the best performance across all maps. GR-1 performs worse than both other algorithms across

⁵While these solve times may not be considered real-time in all applications, they are sufficient for the purposes of this work as replanning does not need to occur at high frequency in the context of landing maneuvers.

Table 4.4: Ablation study on $E_{\max}$.

Algorithm	$E_{\max} = 2000$ J	2500 J	3000 J
Map #1			
GR-1	68.21%	78.15%	86.53%
GR- ∞	88.08%	92.05%	95.14%
GRAPH-DDTO	75.94%	93.82%	98.45%
Map #2			
GR-1	40.21%	61.04%	77.29%
GR- ∞	75.62%	86.46%	91.46%
GRAPH-DDTO	72.71%	90.83%	97.08%
Map #3			
GR-1	64.63%	75.41%	82.52%
GR- ∞	84.35%	89.84%	93.90%
GRAPH-DDTO	79.47%	90.45%	95.12%

all three maps.

4.7.4 Hardware Test: Dynamic Obstacle Avoidance

In the HALO trade study, we demonstrated reliability of GRAPH-DDTO and ADAPTIVE-DDTO on a 500-sample AirSim study. We now establish that we can perform real-time closed-loop execution with hardware-in-the-loop using a collection of flight test demonstrations on an agile quadrotor platform in an indoor netted environment. This quadrotor is not readily equipped with a perceptive sensor, so we decided to consider an alternative form of hazardous and unknown environment: dynamic obstacles with unknown future trajectories. Moreover, we consider a collection of RC cars which are piloted by human operators to navigate the environment in an unpredictable manner.

Problem definition

We consider the same dynamical model introduced in section 4.7.3. The main distinction with this scenario concerns how targets z^j are acquired and tracked. We designate the feasible (safe) ground-plane set $\mathcal{S}$ as the intersection of the sets corresponding to: (i) the boundary of the test environment; (ii) the complement to the set of all RC car obstacles;

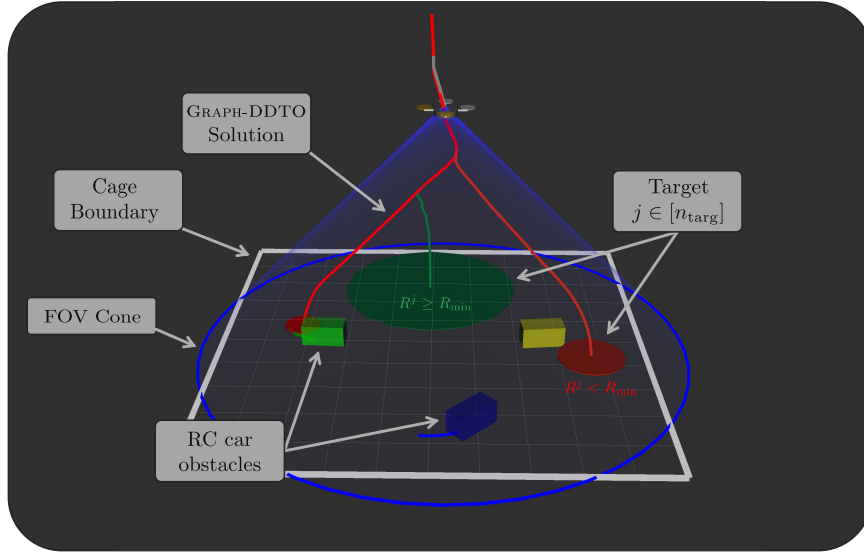
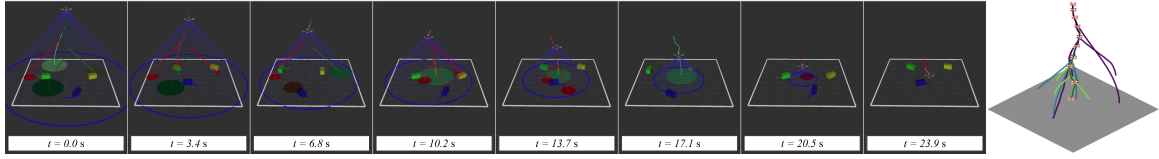
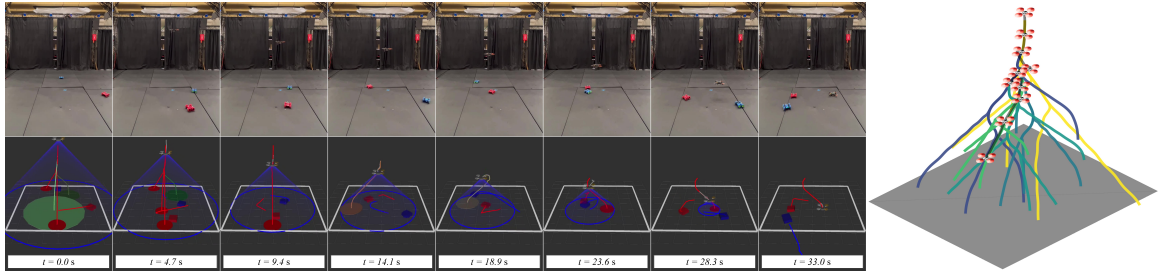


Figure 4.10: GUI-based illustration of the hardware test environment for autonomous quadrotor landing with dynamic obstacles.

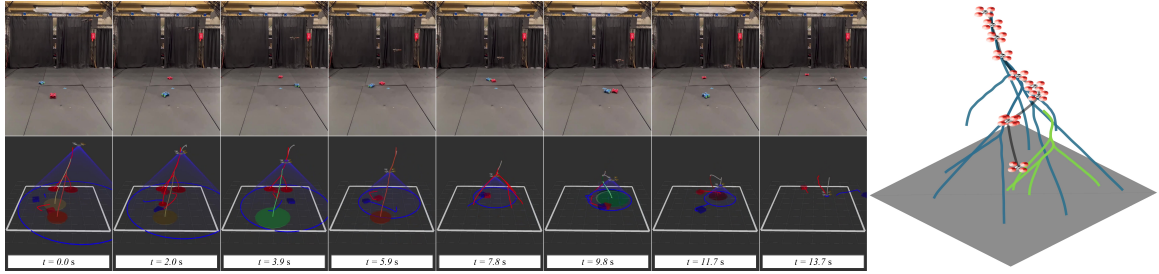
and (iii) the circular projection of a second-order cone emitting from the quadrotor along the $-e_z$ direction onto the ground plane. With the time-varying safe region $\mathcal{S}$ defined, we can now apply the ADAPTIVE-DDTO algorithm to perform safe landing. Ground targets are uniformly sampled on command from $\mathcal{S}$ during target acquisition, and the safe radii of all targets are constantly tracked by projecting their positions onto the boundary of $\mathcal{S}$ using a signed distance function. We note that the second-order cone is added as a test analog to the field-of-view of a perceptive sensor oriented in the $-e_z$ direction (which we hence refer to as the *FOV cone*), and forces the algorithm to converge on a localized region of the ground during descent. In terms of constraints, we also impose a maximum descent velocity $\|e_z^\top v^j(t)\|_2 \leq v_{d,\max}$ to prevent the quadrotor from descending too rapidly in the test environment and to simulate a more realistic landing scenario within the confines of the cage. Similar to the HALO implementation, we make use of the ROS 2 framework to host our targeting and planning modules, with trajectory packets sent to the quadrotor’s internal flight controller from an offboard computer (the same one used for all other experiments in this work), and state estimation packets received from an OptiTrack motion capture system. We demonstrate all forthcoming results using an rviz GUI; Figure 4.10 shows an illustration



(a) Simulated scenario of a fast descent with $v_{d,\max} = 0.2$ m/s.



(b) Flight test of a slow descent with $v_{d,\max} = 0.1$ m/s.



(c) Flight test of a fast descent with $v_{d,\max} = 0.2$ m/s.

Figure 4.11: Timelapses of dynamic obstacle avoidance droptests for quadrotor landing under various operating conditions. For flight tests, camera footage is shown in the top row, and the GUI perspective is shown in the bottom row; for simulated tests, only the GUI perspective is shown. On the far right of each timelapse, the full descent profile and trajectory bundles are displayed under the same conventions as in figure 4.8.

of the GUI display.

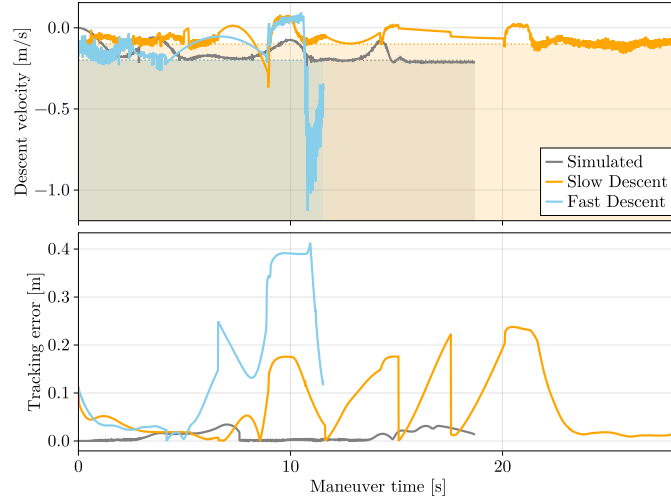


Figure 4.12: Flight test results in terms of both vertical descent velocity and planner tracking error across all three test scenarios for dynamic obstacle avoidance droptests.

Results & analysis

We consider three separate test scenarios under different operating conditions: 1. a fast descent ($v_{d,\max} = 0.2$ m/s) in a purely-simulated environment with three simulated RC cars driving in circular holding patterns; 2. a slow descent ($v_{d,\max} = 0.1$ m/s) in the flight test environment with two human-operated RC cars; and 3. a fast descent ($v_{d,\max} = 0.2$ m/s) in the flight test environment, once again with two human-operated RC cars. Figure 4.11 shows timelapses of each descent scenario. We observe that the quadrotor is able to make fairly stable landings in all three scenarios, despite the fact that the human operators were instructed to drive the RC cars in an unpredictable and chaotic manner. Figure 4.12 shows the drone’s measured descent velocity and tracking error during each test scenario; in the former, we also display the descent velocity constraint violation region dictated by $v_{d,\max}$. When looking at the descent velocity, we can observe that the quadrotor is able to maintain a stable descent velocity with mostly minimal constraint violations (noting that the constraints are satisfied to ϵ_θ tolerance in the trajectory optimization and only violated due to tracking and state estimation errors). We do observe that, in the fast descent scenario in the flight test environment, the quadrotor experiences a rather large violation

of the descent velocity constraint towards the end of the maneuver, which occurs due to divert planning latency under sequential planning. We note, however, that the software is behaving nominally in this mode of operation, and the quadrotor is ultimately able to perform a safe and stable landing regardless, despite the fact that this maneuver is an intentionally-aggressive stress test of a nominal landing scenario (the time of maneuver is only 12 seconds, whereas typical landing maneuvers were on the order of 25 – 30 seconds in HALO scenarios). In terms of tracking error, the error is maintained very close to zero for the simulated scenario due to the fact that the simulator may be paused during the trajectory optimization process (to serve as a control in relation to the hardware tests). In the flight test scenarios, we observe that tracking error is maintained mostly within a reasonably-tight 0.2m tolerance, with jumps observed due to new divers; we only see larger violation during towards the end of the fast descent scenario, due to the same reasoning as was observed for the descent velocity.

4.7.5 Numerical Parameters

In an effort to aid reproducibility of the results of this section, we provide all relevant numerical parameters used in the aforementioned experiments. Across all experiments, we use stopping tolerances of $\epsilon_\gamma = \epsilon_\rho = 5 \times 10^{-3}$, a CTCS relaxation of $\epsilon_\theta = 1 \times 10^{-4}$. We also uniformly apply an objective weighting update law to the $L_z(z)$ term, such that for any proxlinear iteration, the weight is reduced by a factor of 1.4 per each successive iteration. For GRAPH-DDTO, to determine the deferrability indices k_d^j for each stage j , we uniformly sample in the permutation order of λ up to $N^0/\sqrt{2}$ (rounded to the nearest integer), where N^0 represents the number of knot points corresponding to the trunk. Across all scenarios, we also apply a thrust pointing angular constraint relative to the vertical direction, e_z , of 45° . Multiple shooting of all experiments uses 10 stage points per knot interval with an RKF45 integration scheme. Clarabel and ForwardDiff.jl use all default parameters.

For the DDTO comparisons in section 4.7.2 specifically, in terms of algorithm parameters, we use $\rho = 2$, $\gamma = 50$, $k_{\max} = 100$, $N^j = 15$ and $\alpha^j = 1$. We consider a vehicle configuration with mass $m = 0.35$ kg and thrust bounds between 1 and 7 N, and time-of-flight bounds between 5 and 20 s. Cylindrical obstacles all have a radius of 0.6 m.

For the HALO scenarios in section 4.7.3 specifically, in terms of algorithm parameters, we use $\rho = 1$, $\gamma = 100$, $k_{\max} = 50$ and $N^j = 10$. The vehicle and aerodynamic configuration can be found from AirSim documentation [119], summarized as $m = 1 \text{ kg}$, $C_d = 0.325$, $S_A = 0.0198 \text{ m}^2$, $g = -9.81 e_z \text{ m/s}^2$, $\rho_{\text{air}} = 1.225 \text{ kg/m}^3$, with thrust bounds between 5 and 16.72 N and time-of-flight bounds between 5 and 60 s. We also constrain the magnitude of vehicle lateral velocity to a maximum of 10 m/s, and vertical velocity between -10 and 10^{-3} m/s. The ADAPTIVE-DDTO cutoff altitude h_{cut} is set to 30 m. In terms of the HALSS (perception) model, to model LiDAR sensor noise, we use an along-ray standard deviation of

$$\sigma(r) = 0.02 \exp(0.0002 r) \text{ m}, \quad (4.37)$$

where r is the range to the obstacle in meters. The LiDAR has points centrally along $-e_z$ with a 30° field of view half-angle, a maximum range of 500 m, 128 channels, 50 rotations per second and 10^6 points per second; we refer the reader to AirSim documentation [119] for more details. The HALSS circular buffer holds a maximum point cloud size of 10^5 points; we refer the reader to [52] for all other default HALSS parameters. The feedback controller uses AirSim's `moveOnPathAsync` to track the closest point on the tracked trajectory at a rate of 100 Hz.

Vehicle parameters associated with the flight quadrotor in section 4.7.4 are not able to be provided due to proprietary information.

Chapter 5

CONTACT-IMPLICIT SUCCESSIVE CONVEXIFICATION

5.1 Introduction

In previous chapters, we have explored the use of sequential convex programming to solve trajectory optimization problems pertaining to perception-aware planning, fulfilling the first objective of this dissertation’s defined *interaction-aware* exposition. We now turn our attention to the second objective, which is to consider *contact-rich* interactions. The prevailing approach to addressing this class of interactions in the literature is contact-implicit trajectory optimization (CITO). CITO methods are particularly attractive because they eliminate the need to predefine contact schedules. However, most existing CITO formulations rely on fine time discretization to capture contact events. This increases the number of decision variables and constraints, coupling solution quality tightly to the discretization resolution and creating an unfavorable accuracy-runtime tradeoff [101].

To address this limitation, we build on the sequential convex programming (SCP) approach, which iteratively approximates and solves nonconvex problems. While the usage of SCP has been addressed for CITO [97, 98], we instead consider continuous-time successive convexification (CT-SCvx) framework [37] introduced in section 1.3.2, which notably enforces constraints in continuous time and thus permits high-fidelity, feasible solutions over coarse discretization grids. Our key idea is to extend continuous-time satisfaction to contact constraints, specifically cross-complementarity constraints [9]. We introduce an integral form of cross-complementarity constraints expressed in continuous time, which ensures that contact mode changes occur at discretization nodes while eliminating the risk of missing contact

events between nodes. In conjunction with time-dilation [37], where nodes are allowed to stretch or shrink over time, the generality of the resulting contact mode sequence is ensured.

Furthermore, we investigate a variety of contact-implicit dynamics models, including whole-body (WB) models and single-rigid-body (SRB) models. SRB models [143] and the related centroidal dynamics model [99] act as reduced-order approximations of WB models, which trade off fidelity for computational efficiency. WB models, on the contrary, are able to capture the full range of motion of a system, are consistent with the maximum dissipation principle for stick-slip friction, and can model partially elastic impacts [123]. We also show how SRB models can be used to systematically warmstart WB models, resulting in greater computational efficiency than solving the full WB model from scratch. To handle complementarity constraints arising from all aforementioned models, we investigate multiple classes of homotopy schemes, including a backtracking update embedded into SCP iterates, and a nested-loop update. We implement the approach in stand-alone Python software that leverages JAX for GPU- or CPU-accelerated differentiation and linearization, together with a custom parser for the convex subproblems in SCP that avoids the overhead of general-purpose modeling tools such as CVXPY. We demonstrate the approach on two-dimensional legged robot models, including the Gymnasium HalfCheetah model [134]. This chapter encompasses the work presented in [17] as well as follow on work in preparation [18].

5.1.1 Related Literature

CITO has been studied extensively to generate locomotion solutions without prescribing contact schedules. Early approaches used direct collocation with complementarity constraints to enforce non-penetration and frictional contact [106], while contact-invariant optimization [89] handled contact through smooth penalties that encourage sticking and end-effector alignment, with dynamics enforced softly. These methods typically require fine time discretization to capture contact events accurately, yielding large-scale problems whose solution quality depends strongly on the grid resolution. To mitigate the risk of missing contact events between nodes, cross-complementarity constraints were proposed [9]. More recent formulations such as FESD-J [94] and moving finite elements [62] extend this idea by enabling exact switch detection for systems with impacts and Coulomb friction, ensur-

ing that contact events occur at finite-element boundaries. However, these are still inherently discrete-time constructions. Continuous-time trajectory optimization frameworks like CT-SCvx [37] enforce constraints in continuous time and thereby achieve accurate solutions with coarse discretization. The underlying solver, sequential convex programming (SCP), has been extended to CITO by [75, 97], however a continuous-time extension of SCP for CITO has not yet been explored.

In terms of contact-implicit dynamics models underpinning CITO formulations, both reduced-order (SRB) and full-order (WB) models have been explored extensively. In the case of SRB models, warmstarting the full mixed-integer formulation with learning-based approaches has shown to be effective [76]. Other approaches have instead reformulated the problem to avoid solving a mixed-integer formulation entirely [145] under additional modeling restrictions. However, these approaches often neglect whole-body effects and can lead to conservative solutions that underutilize the system’s full range of motion. The majority of recent works in the literature, therefore, fixate on WB approaches [68, 73]. The interplay between SRB and WB models has also received some attention recently. The Cafe-MPC framework [72] performs short-horizon planning with WB models, and long-horizon planning with SRB models, with descending model fidelity along the prediction horizon. Dynamic consensus between centroidal and WB models has also been explored [149], however not in the context of CITO (prescribed contact schedules). There has not been an effort up to this point to investigate how CITO-based SRB models can be used to warmstart CITO-based WB models.

5.1.2 Contributions

This chapter introduces a continuous-time CITO framework designed for locomotion tasks which asserts the following contributions:

1. The contact-implicit successive convexification (CI-SCvx) framework, an SCP-based approach for CITO that ensures continuous-time constraint satisfaction of path constraints.
2. A collection of contact-implicit multibody dynamics models, including several forms of

whole-body (WB) models and single-rigid-body (SRB) models, along with a structured warmstarting approach to convert SRB solutions into WB initializations.

3. Integral cross-complementarity constraints that ensure contact mode changes occur at discretization nodes. Combined with time-dilation [37], this preserves generality in contact mode sequences.
4. A backtracking homotopy scheme that progressively tightens the complementarity constraint relaxation during SCP iterations, ensuring reliable and fast convergence.
5. A stand-alone Python software that leverages JAX for parallelizable differentiation and linearization, together with a custom parser for convex subproblems in SCP that avoids the overhead of general-purpose modeling tools such as CVXPY.
6. Demonstrations on an assortment of practical locomotion tasks which assert the viability of SRB-based warmstarting, flexibility of ci-SCvx on difficult contact environments, and validate the approach in the MuJoCo-based Gymnasium HalfCheetah environment [134], with comparisons to a state-of-the-art baseline (MuJoCo-MPC).

The remainder of the chapter is outlined as follows. Section 5.2 introduces several forms of both whole-body and reduced-order contact-implicit multibody dynamics models. Section 5.3 introduces the theoretical foundation of integral cross-complementarity constraints. Section 5.4 presents contact-implicit trajectory optimization (CITO) formulated under our specific modeling methodology. Section 5.5 introduces the ci-SCvx framework, and section 5.6 provides numerical results.

5.2 Contact-Implicit Multibody Dynamics

We begin by defining the equations of motion and constraints which collectively define the dynamics of a multibody system which can make contact with a surface subject to Coulomb friction. We describe this dynamical model as *contact-implicit*, such that the mode of contact may be implicitly described by the system of equations without the use of combinatoric or mixed-integer representations. In terms of the multibody dynamics, we consider two types of dynamics models:

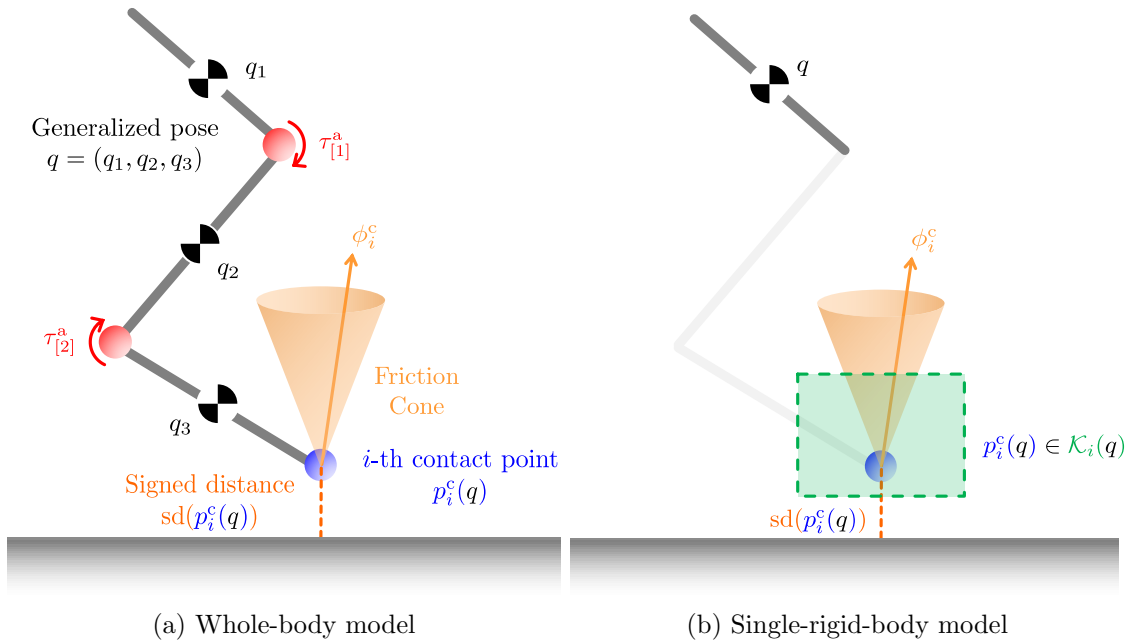


Figure 5.1: Illustration of the types of multibody models considered in this chapter.

1. **Whole-Body (WB):** explicitly models the state of each rigid body in the multibody system, either relatively (in the case of *minimal* coordinates) or absolutely (in the case of *maximal* coordinates). This representation is accompanied by a complete expression of joint torques used to produce relative motion of rigid bodies in relation to each other.
2. **Single-Rigid-Body (SRB):** only directly models the state of a designated *base* rigid body (for example, the torso of a quadruped), as well as the state of the points of contact, while neglecting intermediate rigid bodies. Consequently, rather than joint-level torque control, the impact of Coulomb friction is applied directly to the base.

In the following subsections, we explore representations of both of these models, and later show how the simplified SRB model can be used to warmstart a WB model.

5.2.1 Notation

We adopt the following notation for the remainder of this chapter. Concatenation of vectors $u \in \mathbb{R}^n$ and $v \in \mathbb{R}^m$ is $(u, v) \in \mathbb{R}^{n+m}$. The scalar components of $v \in \mathbb{R}^n$ are denoted by $v_{[i]}$, for $i = 1, \dots, n$. Matrices A and B can be concatenated horizontally as $[A \ B]$ or vertically as $\begin{bmatrix} A \\ B \end{bmatrix}$. The block-diagonal concatenation of A and B is $\text{blockdiag}(A, B)$. Given vectors $u, v \in \mathbb{R}^n$, the inequality $u \leq v$ is interpreted componentwise. A continuously differentiable function is called a $\mathcal{C}^1$ function. The Jacobian of a function $f(x)$ is $\nabla f(x)$, and the time derivative $d\Box/dt$ is denoted by $\dot{\Box}$. The Euclidean norm is $\|x\|$, the absolute-value function is $|x|$, The ramp (ReLU) function is denoted by $|x|_+$. Smooth pseudo-Huber approximations of the Euclidean norm, absolute value, and ramp functions with unit steepness parameter are denoted by $\|x\|_\sim$, $|x|_\sim$, and $|x|_\sim^+$. For vector arguments, $|x|_\sim^+$ is understood componentwise.

5.2.2 Whole-Body Model

We start with the whole-body (WB) model by letting $q \in \mathbb{R}^{n^q}$ denote the generalized position (pose) and $v \in \mathbb{R}^{n^q}$ the generalized velocity of the multibody system. The robot has n^c contact points with positions, expressed in the inertial (world) frame, given by $p_i^c(q) \in \mathbb{R}^{n^d}$ for $i = 1, \dots, n^c$. The Jacobian of the i th contact point (the contact Jacobian) is $J_i^c(q) = \nabla p_i^c(q)$. Contact occurs on a smooth surface $\mathcal{H}^c = \{p \in \mathbb{R}^{n^d} \mid h^c(p) = 0\}$, where h^c is a continuously differentiable function and $n^d \in \{2, 3\}$ is the dimension of the position space. At each contact point, we define a contact frame consisting of the unit surface normal $n_i^c(q) = \nabla \text{sd}(p_i^c(q))^\top \in \mathbb{R}^{n^d}$ and $(n^d - 1)$ unit tangent vectors given by the columns of $t_i^c(q) \in \mathbb{R}^{n^d \times (n^d - 1)}$. Together, the columns of $t_i^c(q)$ and $n_i^c(q)$ form a right-handed basis. The rotation matrix that maps vectors from the contact frame to the world frame is $R_i^c(q) = [t_i^c(q) \ n_i^c(q)]$.

Continuous-Time Dynamics

We start by describing the continuous-time dynamics of the multibody system. In this section, we consider maximal coordinate representations specifically. In the maximal representation, the generalized pose takes on the form $q = (q_1, \dots, q_{n^b})$, for n^b rigid bodies in the multibody system, where $q_j, j \in [n^b]$ represents the absolute pose of the j th rigid body. The

contact force at the i th contact point, represented in the contact frame, is $\phi_i^c = (\phi_i^{c,t}, \phi_i^{c,n})$, where $\phi_i^{c,n} \in \mathbb{R}$ is the normal component along $n_i^c(q)$ and $\phi_i^{c,t} \in \mathbb{R}^{n^d-1}$ is the tangential component. Let $c^j(q) = 0_{n^j}$ impose the kinematic constraints that connect links at their respective joints and restrict degrees of freedom according to joint type. The effect of joint reaction forces on the generalized acceleration $\dot{v}$ is captured by $\nabla c^j(q)^\top \phi^j$, where ϕ^j concatenates all joint reaction forces in the world frame. Then, the equations of motion for this system are described by

$$\dot{q} = J_q(q)v, \quad (5.1a)$$

$$M(q)\dot{v} = W(q, v, \tau^a, a^g) + J^c(q)^\top R^c(q)\phi^c + \nabla c^j(q)^\top \phi^j, \quad (5.1b)$$

where $J^c(q)^\top = [J_1^c(q)^\top \cdots J_{n^c}^c(q)^\top]$, $R^c(q) = \text{blockdiag}(R_1^c(q), \dots, R_{n^c}^c(q))$, and $\phi^c = (\phi_1^c, \dots, \phi_{n^c}^c)$. $M(q)$ is the mass matrix, J_q maps generalized velocities to generalized pose derivatives (identity in the case of $n^d = 2$), and $W(q, v, \tau^a, a^g)$ collects the generalized forces and torques due to gravity (with $a^g \in \mathbb{R}^{n^d}$ the gravitational acceleration expressed in the world frame), joint torques $\tau^a \in \mathbb{R}^{n^a}$, and joint stiffness and damping. To eliminate the joint reaction forces in (5.1b), we can enforce joint constraints at the acceleration level

$$c^j(q) = 0_{n^j} \implies \nabla c^j(q)v = 0_{n^j}, \quad (5.2a)$$

$$\implies \nabla c^j(q)\dot{v} + (\dots v^\top \nabla^2 c_{[i]}^j(q)v \dots) = 0_{n^j}, \quad (5.2b)$$

where $c_{[i]}^j(q)$ is the i th component of c^j , for $i = 1, \dots, n^j$. From (5.2b) and (5.1b) we solve for ϕ^j and substitute into (5.1b). Thus, we can reduce the form of our dynamics to a simple system of ordinary differential equations.

To model physically realistic contact, we require (i) a no-penetration condition expressed as a complementarity constraint between the normal contact force and signed distance to the contact surface, and (ii) a tangential contact force determined by the maximum dissipation

principle subject to a friction cone constraint

$$0 \leq \phi_i^{c,n} \perp \text{sd}(p_i^c(q)) \geq 0, \quad (5.3a)$$

$$\phi^{c,t} \in \underset{\phi^t}{\text{argmin}} \phi^{t\top} v_i^{c,t}(q, v) \quad \text{s.t.} \quad \|\phi^t\| \leq \mu_i \phi_i^{c,n}, \quad (5.3b)$$

for $i = 1, \dots, n^c$, where μ_i is the friction coefficient and $v_i^{c,t}(q, v) = t_i^c(q)^\top J_i^c(q)v$ is the tangential velocity at the i th contact. The first-order optimality conditions for (5.3b) are

$$0_{n^d-1} = v_i^{c,t}(q, v) + 2\gamma_i \phi_i^{c,t}, \quad (5.4a)$$

$$0 \leq \gamma_i \perp (\mu_i \phi_i^{c,n})^2 - |\phi_i^{c,t}|^2 \geq 0, \quad (5.4b)$$

where γ_i is the Lagrange multiplier and we denote $\gamma = (\gamma_1, \dots, \gamma_{n^c})$. The friction cone constraint is written in squared form to ensure differentiability at the boundary. When the i th contact is active (i.e., signed distance is zero), γ_i is a continuous function of time. Two issues arise with (5.4): (i) the $\|\cdot\|^2$ term causes numerical ill-conditioning, and (ii) when the i th contact is inactive (i.e., signed distance is positive), $\phi_i^c = 0_{n^d}$, so (5.4a) reduces to the non-physical case that the contact point's tangential velocity is zero, i.e., $t_i^c(q)^\top J_i^c(q)v = 0_{n^d-1}$. We address both by rewriting (5.4) as

$$0_{n^d-1} = \phi_i^{c,n} (v_i^{c,t}(q, v) + \gamma_i \nabla \|\phi_i^{c,t}\|_\sim^\top), \quad (5.5a)$$

$$0 \leq \gamma_i \perp |\mu_i \phi_i^{c,n}|_\sim - \|\phi_i^{c,t}\|_\sim \geq 0, \quad (5.5b)$$

using the smooth norm $\|\cdot\|_\sim$ and smooth absolute value $|\cdot|_\sim$. The multiplication of $\phi_i^{c,n}$ in (5.5a) enforces stationarity only when contact is active, i.e., normal contact force is positive. Henceforth, we denote $\lambda_i^c(q, v, \phi_i^c, \gamma_i) = v_i^{c,t}(q, v) + \gamma_i \nabla \|\phi_i^{c,t}\|_\sim^\top$ and $\rho_i^c(\phi_i^c, \gamma_i) = |\mu_i \phi_i^{c,n}|_\sim - \|\phi_i^{c,t}\|_\sim$.

We can generalize our description of the above system of equations by rewriting (5.1) in state-space form as $\dot{x} = f(x, u)$, with state $x = (q, v) \in \mathbb{R}^{n^x}$, control input $u = (\tau^a, \phi^c, \gamma) \in \mathbb{R}^{n^u}$, and dimensions $n^x = 2n^q$ and $n^u = n^a + n^c(n^d + 1)$. The state and control input are subject to path constraints $g(x(t), u(t)) \leq 0_{n^g}$ for all $t \in [0, t^f]$, which include, for each

$i = 1, \dots, n^c$, nonnegativity constraints on $\phi_i^{c,n}$, $\text{sd}(p_i^c(q))$, γ_i , and $\rho_i^c(\phi_i^c, \gamma_i)$, together with general constraints on state and input (e.g., obstacle avoidance, kinematic limits).

While we do not derive the explicit minimal coordinate representation in this section, we note that it can be obtained by a similar process to the one described above, but with the generalized pose represented as $q = (q_1, \dots, q_{n^b})$, where q_1 corresponds to the base link (w.l.o.g.) and is represented with an absolute pose, and $q_2, \dots, q_{n^b}$ correspond to the relative poses of the remaining links to the base link (typically represented as joint angles in the case of strictly rotational joints, as will be assumed through the remainder of this chapter). While the form of equations (5.1) will change in accordance with this representation change, we note that the contact-implicit constraints (5.3) retain the same form. We refer the interested reader to [61] for a detailed derivation of the minimal coordinate representation.

Impulsive Impact Model

We now consider impulsive impacts occurring at discrete instants, which are necessary to model to capture the instantaneous velocity changes that occur when contact is made. Let an impact occur at time $\hat{t}$. Each contact point may have an associated contact impulse denoted $\Phi_i^c = (\Phi_i^{c,t}, \Phi_i^{c,n})$ with tangential and normal components, respectively. Stack the contact impulses as $\Phi^c = (\Phi_1^c, \dots, \Phi_{n^c}^c)$, and let Φ^j denote the joint reaction impulse. The impact dynamics are

$$M(q(\hat{t}))(v(\hat{t}^+) - v(\hat{t}^-)) = J^c(q(\hat{t}))^\top R^c(q(\hat{t}))\Phi^c + \nabla c^j(q(\hat{t}))^\top \Phi^j, \quad (5.6a)$$

$$0 \leq \Phi_i^{c,n} \perp \text{sd}(p_i^c(q(\hat{t}))) \geq 0, \quad (5.6b)$$

$$0 = \Phi_i^{c,n} n_i^c(q(\hat{t}))^\top J_i^c(q(\hat{t}))(v(\hat{t}^+) + \varepsilon_i^c v(\hat{t}^-)), \quad (5.6c)$$

$$0 = \Phi_i^{c,n} \left(v_i^{c,t}(q(\hat{t}), v(\hat{t}^+)) + \Gamma_i \nabla \|\Phi_i^{c,t}\|_\sim^\top \right), \quad (5.6d)$$

$$0 \leq \Gamma_i \perp |\mu_i \Phi_i^{c,n}|_\sim - \|\Phi_i^{c,t}\|_\sim \geq 0, \quad (5.6e)$$

$$i = 1, \dots, n^c.$$

Conditions (5.6d)–(5.6e) play the same role as (5.5). They determine the tangential contact impulse via a stationarity condition (multiplied by normal contact impulse) and the friction

cone constraint, with Γ_i the associated Lagrange multiplier. Impact elasticity is governed by (5.6c), where ε_i^c is the coefficient of restitution at the i th contact.

We remove the explicit joint reaction impulse in (5.6a) by imposing the joint constraint post impact at the velocity level, $\nabla \mathcal{J}(q(\hat{t}^+))v(\hat{t}^+) = 0_{n^j}$. Solving the resulting constraint and (5.6a) yields an explicit expression for Φ^j , which we substitute back into (5.6a).

5.2.3 Single-Rigid-Body Model

We now introduce the single-rigid-body (SRB) formulation with a number of simplifications in relation to the WB model: (i) only the base body's pose, denoted q_b , is explicitly modeled, (ii) only sticking contact is considered, as slipping cannot produce physically-meaningful solutions with this type of model, and (iii) contacts must be considered purely inelastic, such that $\varepsilon_i^c = 0$. This effort is conducted to produce a proximal model which may be lower dimensional and less nonlinear than the equivalent WB representation. Much of this approach is based off the formulation provided in Winkler et al. [145].

In this section, we focus on the case of $n^d = 2$. Let $q = q_b = (p_b, \theta_b)$, where p_b represents the position of the root body, and θ_b represents the orientation of the root body. Similarly, let $v = (v_b, \omega_b)$, where v_b represents the velocity of the root body, and ω_b represents the angular velocity of the root body. We allow direct access to contact positions p_i^c as part of the system state, and treat contact velocities v_i^c as control variables. The contact force, ϕ_i^c , is expressed in the local contact frame. The terms t_i^c, n_i^c, R_i^c are described similarly to the WB model, although they operate directly on the contact position rather than the generalized coordinates due to our state-space representation, with $R_i^c(p_i^c) = [t_i^c(p_i^c), n_i^c(p_i^c)]$. The equations of motion for this system are described by

$$\dot{q} = v, \quad (5.7a)$$

$$m\dot{v}_b = \sum_{i=1}^{n^c} R_i^c(p_i^c)\phi_i^c - ma^g, \quad (5.7b)$$

$$I\dot{\omega}_b = \sum_{i=1}^{n^c} R_i^c(p_i^c)\phi_i^c \times (p_b - p_i^c), \quad (5.7c)$$

$$\dot{p}_i^c = v_i^c. \quad (5.7d)$$

We note that the mass m and moment of inertia I are expressed only for the base link, which implicitly assumes that all other link masses are negligible compared to that of the base link or that the other links are highly constrained such that they do not deviate much from a nominal generalized pose [145]. Next, we must impose a relation to maintain kinematic consistency between the base pose and contact points, such that

$$p_i^c \in \mathcal{K}_i(q) \quad (5.8)$$

Formally, the set-valued function $\mathcal{K}_i(q)$ must be designed such that any point within it has a corresponding feasible generalized pose (in maximal or minimal coordinates) expressed between base pose q and contact point p_i^c . This is often referred to as the *workspace* of p_i^c . One such way to find the workspace would be to walk down the kinematic chain (configuration graph) from root link to the link which an end effector is attached to, and compute the reachable set of each link in the chain relative to its predecessor for a given pose q . In the case where there are only joint-relative pose constraints (and no inertial-relative pose constraints), this set's definition is fixed under a pose transformation driven by q and can be precomputed. One restrictive option, explored by [145], considers a box constraint about a central point $\bar{p}_i^c$ with bounds $b \in \mathbb{R}^{n^d}$ such that

$$|\mathcal{R}(\theta_b)(p_i^c - p_b) - \bar{p}_i^c| \leq b \quad (5.9)$$

Where $\mathcal{R} : \mathbb{R} \rightarrow \text{SO}(2)$ represents a direction cosine matrix. We note that (5.7) and (5.9) represents an approximation to the WB dynamics expression (5.1). Then, the remainder of constraints defining the SRB model can be described by complementarity conditions as

$$0 \leq \phi_i^{c,n} \perp \|v_i^c\|_{\sim} \geq 0 \quad (5.10a)$$

$$0 \leq \phi_i^{c,n} \perp \text{sd}(p_i^c) \geq 0, \quad (5.10b)$$

$$\|\phi_i^{c,t}\|_2 \leq \mu_i \phi_i^{c,n}. \quad (5.10c)$$

Equation (5.10a) enforces the no-slip (stick-only) condition on contact velocity v_i^c when

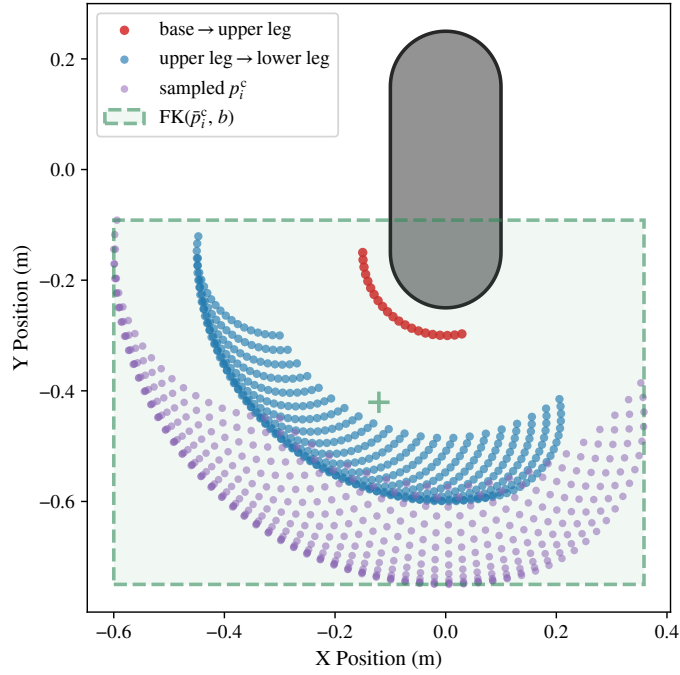


Figure 5.2: Illustration of how the forward-kinematic (FK) box constraint set is defined for the SRB model.

contact is made, acting in place of a true impulsive model while eliminating the need for explicit impulses and pre- and post-impact states. Equations (5.10b)-(5.10c) represent a simplification of (5.3) in the no-slip case (such that the maximum dissipation principle may be ignored, and contact forces only need to fall within the Coulomb friction cone (5.10c)).

Similarly to the whole-body case, we can rewrite (5.7) in state-space form with state $x = (q, v, p^c)$, control input $u = (\phi^c, v^c)$, where p^c, v^c stack p_i^c, v_i^c , respectively. We consequently have dimensions $n^x = 6 + 2n^c$ and $n^u = 4n^c$ when $n^d = 2$. The state and control input are subject to path constraints $g(x(t), u(t)) \leq 0_{n_g}$ for all $t \in [0, t^f]$, which include, for each $i = 1, \dots, n^c$, nonnegativity constraints on $\phi_i^{c,n}$, $\text{sd}(p_i^c)$, the direct expression of (5.10c), and the kinematic constraint (5.9), together with general constraints on state and input (of the same form as those for whole-body).

Defining the Forward-Kinematic Box

In contrast to [145], we develop a structured approach to defining the parameters $\bar{p}_i^c$, b for the forward-kinematic box constraint in equation 5.9. To do this, we recursively sample joint angles across their respective bounds along the kinematic chain from the base body to each contact point. We note that this process resembles a *reachability* problem expressed over the space of bodies from base to contact point, as opposed to reachability over time.

Figure 5.2 provides a visual example of this process for a simple three-link monopod. Starting from the base body's CoM, we obtain a collection of sampled points corresponding to joint angles of the base relative to the upper leg, subject to the relative orientation limits between these two bodies, resulting in the red points. We then repeat this process *for each* sampled point towards the next body in the chain (lower leg), resulting in the blue points. Since the contact point is directly attached to this body, we simply apply a translational offset from the lower leg's CoM to the contact point, resulting in the final sampled space of contact points p_i^c denoted as purple points. With this point cloud, we can then fit a box around the points parameterized by $\bar{p}_i^c$ and b parameterizing constraint (5.9).

5.3 Integral Cross-Complementarity

We seek a solution to (5.1), (5.6) (in the WB case) or (5.7) (in the SRB case) over a time horizon $[0, t^f]$. We discretize $[0, t^f]$ into a grid of N nodes, $0 = t_1, \dots, t_N = t^f$, and we isolate all contact mode switches to occur only at the grid nodes. These contact modes indicate either no contact, sticking contact or slipping contact (for WB models specifically).

First, we introduce the following continuous-time cross-complementarity condition. Let $a, b : [0, t^f] \rightarrow \mathbb{R}_+$ be piecewise continuous with any discontinuities confined to the grid. Specifically, for each $k = 1, \dots, N - 1$, a and b are continuous on $(t_k, t_{k+1}]$. They satisfy pointwise-in-time complementarity

$$0 \leq a(t) \perp b(t) \geq 0, \quad \forall t \in [0, t^f]. \quad (5.11)$$

To ensure switches between $a > 0, b = 0$ and $a = 0, b > 0$ happen *only* at the grid nodes,

we impose, for each $k = 1, \dots, N - 1$,

$$0 \leq a(t) \perp b(t') \geq 0, \quad \forall t, t' \in (t_k, t_{k+1}], \quad (5.12)$$

together with $0 \leq a(t_1) \perp b(t_1) \geq 0$. We then have the following continuous-time analogue of [94, Lem. 2].

Lemma 5.1. *Let (5.12) hold and fix $k \in \{1, \dots, N - 1\}$. If $a(t) > 0$ for some $t \in (t_k, t_{k+1}]$, then $b(t') = 0$ for all $t' \in (t_k, t_{k+1}]$. Conversely, if $b(t') > 0$ for some $t' \in (t_k, t_{k+1}]$, then $a(t) = 0$ for all $t \in (t_k, t_{k+1}]$.*

Proof. If $a(t) > 0$ at some $t \in (t_k, t_{k+1}]$ and $b(t') > 0$ at some $t' \in (t_k, t_{k+1}]$, then (5.12) is violated, which is a contradiction. $\square$

While (5.12) localizes switches to the grid nodes (a desirable property for both numerical reliability and physical realism), it is still a pointwise-in-time constraint and thus numerically intractable. Existing approaches [9, 62, 94] enforce discrete-time surrogates of (5.12) in aggregated forms, which risks inter-sample violations. To avoid this, we propose an integral reformulation, referred to as an *integral cross-complementarity* condition.

Proposition 5.1. *For each $k = 1, \dots, N - 1$, the constraint*

$$0 \leq \left(\int_{t_k}^{t_{k+1}} a(t) dt \right) \perp \left(\int_{t_k}^{t_{k+1}} b(t) dt \right) \geq 0, \quad (5.13)$$

is equivalent to (5.12).

Proof. ($\Rightarrow$) If (5.13) holds and $\int_{t_k}^{t_{k+1}} a(t) dt > 0$, then the complementarity of the integrals implies $\int_{t_k}^{t_{k+1}} b(t) dt = 0$. Since a and b are nonnegative and are continuous on $(t_k, t_{k+1}]$, it follows that $b(t) = 0$ for all $t \in (t_k, t_{k+1}]$. The argument is symmetric if $\int_{t_k}^{t_{k+1}} b(t) dt > 0$. Therefore, (5.12) holds. ($\Leftarrow$) If (5.12) holds, Lemma 5.1 implies that for each $k = 1, \dots, N - 1$, either $a(t) = 0$ or $b(t) = 0$ (or both) for all $t \in (t_k, t_{k+1}]$. Hence one (or both) of the integrals is zero, yielding (5.13). $\square$

In practice, we compute the integrals in (5.13) by integrating an auxiliary dynamical

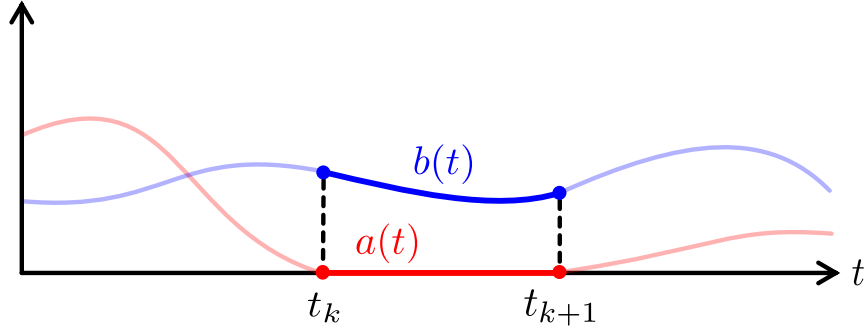


Figure 5.3: Illustration of a signal pair a, b satisfying integral cross-complementarity over a time interval $t \in [t_k, t_{k+1}]$ with $a(t) = 0, \forall t \in [t_k, t_{k+1}]$.

system

$$\dot{y} = (\dot{y}_{[1]}, \dot{y}_{[2]}) = (a, b), \quad (5.14)$$

and then impose the complementarity constraint

$$0 \leq y_{[1]}(t_{k+1}) - y_{[1]}(t_k) \perp y_{[2]}(t_{k+1}) - y_{[2]}(t_k) \geq 0. \quad (5.15)$$

If a or b are not inherently nonnegative, we enforce their nonnegativity through the path constraint function g .

Remark 5.1. *Discrete-time cross-complementarity formulations (e.g., FESD-J [94]) typically introduce additional cross-complementarity constraints to detect sign changes in the tangential velocity at each contact point. Our formulation obviates these by parameterizing the control input u on each interval $(t_k, t_{k+1}]$, $k = 1, \dots, N - 1$, with $\mathcal{C}^1$ basis functions. Consequently, λ_i^c is also $\mathcal{C}^1$ on $(t_k, t_{k+1}]$, which precludes nonsmoothness of the tangential velocity between nodes.*

In the case of the WB model, to ensure that contact mode switches occur only at the grid nodes, we impose integral cross-complementarity between the following pairs: (i) $\phi_i^{c,n}$ and $\text{sd}(p_i^c(q))$, (ii) $\phi_i^{c,n}$ and $\lambda_i^c(q, v, \phi_i^c, \gamma_i)$, and (iii) γ_i and $\rho_i^c(\phi_i^c, \gamma_i)$. In the case of the

SRB model, we must instead impose integral cross-complementarity between: (i) $\phi_i^{c,n}$ and $\|v_i^c\|_\sim$, and (ii) $\phi_i^{c,n}$ and $\text{sd}(p_i^c)$. To generalize this description, we define integrator states $y_i^{\text{WB}} \in \mathbb{R}^5$, $y_i^{\text{SRB}} \in \mathbb{R}^3$ with dynamics stated as

$$\dot{y}_i^{\text{WB}} = \begin{bmatrix} \phi_i^{c,n} \\ \text{sd}(p_i^c(q)) \\ \lambda_i^c(q, v, \phi_i^c, \gamma_i) \\ \gamma_i \\ \rho_i^c(\phi_i^c, \gamma_i) \end{bmatrix}, \quad \dot{y}_i^{\text{SRB}} = \begin{bmatrix} \phi_i^{c,n} \\ \|v_i^c\|_\sim \\ \text{sd}(p_i^c) \end{bmatrix}. \quad (5.16)$$

Then, for $y_i^{[M]}$, $M \in \{\text{WB}, \text{SRB}\}$, we seek to impose, for each $i = 1, \dots, n^c$,

$$0 \leq L^{[M]}(y_i^{[M]}(t_{k+1}) - y_i^{[M]}(t_k)) \perp R^{[M]}(y_i^{[M]}(t_{k+1}) - y_i^{[M]}(t_k)) \geq 0, \quad (5.17)$$

where $L^{[M]}, R^{[M]}$ are mixing matrices to avoid duplicating integrator states, and are defined as

$$L^{\text{WB}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \quad R^{\text{WB}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad (5.18a)$$

$$L^{\text{SRB}} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad R^{\text{SRB}} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (5.18b)$$

5.4 Contact-Implicit Trajectory Optimization Problem

We are now ready to formulate the contact-implicit trajectory optimization (CITO) problem for both rigid-body model classes. In contrast to methods from previous chapters, the impulse update (in the case of the WB model) necessitates a *hybrid optimal control* formulation; we will discuss this approach in more detail in the following sections.

5.4.1 Continuous-Time Dynamics

In the case of both the WB and SRB models, the continuous-time dynamics are described by equations (5.1) and (5.7), respectively, and can be described in a compressed form as $\dot{x} = f(x, u)$, where x is the state, u is the control input, and f is the dynamics function. We require a two-step augmentation process. First, we augment the state to include the complementarity-handling integrator states from integral cross-complementarity, such that $\hat{x} = (x, y)$, where $y = (y_1, \dots, y_{n^c})$ and $y_i = y_i^{[M]}$ is the integrator state for the i th contact point. We drop the superscript $[M]$ for brevity going forwards, where it is implied by the context of the problem. The dynamics may then be described as $\dot{\hat{x}} = \hat{f}(\hat{x}, u)$, where $\hat{f}$ is the augmented dynamics function.

Next, we augment both the state and control input according to the model augmentation process described in section 1.3.2. In doing this, we produce our final augmented state-space model on state $\tilde{x} = (\hat{x}, \theta)$, where θ maintains continuous-time constraint satisfaction of the path constraints $g(x(t), u(t)) \leq 0_{n^g}$ for all $t \in [0, t^f]$, and control input $\tilde{u} = (u, s)$, where s is the time dilation variable. Notably, inclusion of time dilation enables the optimizer to avoid constraining the contact mode sequence (e.g. gait cadence and pattern). The resulting model can be described by $\dot{\tilde{x}} = \tilde{f}(\tilde{x}, \tilde{u})$, where $\tilde{f}$ is the augmented dynamics function.

In this work, we opt for a separate discretization approach from the one described in section 1.3.2, such that the ordering between discretization and linearization is reversed. Moreover, we express our discretized (but not yet linearized) dynamics as

$$\tilde{F}(\tilde{x}^-, U, S) = \tilde{x}^- + \int_0^1 \tilde{f}(\tilde{x}(\tau), (\psi(\tau)U, S)) \, d\tau, \quad (5.19)$$

where $\psi(\tau) : [0, 1] \rightarrow \mathbb{R}^{n^u \times n^u N^u}$ is a matrix-valued polynomial basis function, and $U \in \mathbb{R}^{n_u N_u}$ and $S \in \mathbb{R}$ are corresponding coefficients.

5.4.2 Hybrid Optimal Control

We now formulate the CITO problem as a hybrid optimal control problem under non-linear complementarity constraints (specifically in the case of the WB model). We will first generalize the description of the impulse update from section 5.2.2 to the hybrid op-

timal control formulation. Recall that we define our discretization onto a grid of N nodes, $0 = t_1, \dots, t_N = t^f$, and we isolate all contact mode switches to occur only at the grid nodes. Then, we can assign pre- and post-impact augmented states at the grid nodes $\tilde{x}_1^-, \tilde{x}_1^+, \dots, \tilde{x}_N^-, \tilde{x}_N^+$, respectively. We also maintain coefficients parameterizing the augmented control input on each grid interval as $U_1, S_1, \dots, U_{N-1}, S_{N-1}$, respectively. Then, with pre- and post-impact states, we can define a generalized impulses between states as $V_1, \dots, V_N$, where $V_k = (\Phi_k^c, \Gamma_k)$. Then, we can express an impulse update with the expression

$$\tilde{x}_k^+ = \tilde{G}(\tilde{x}_k^-, V_k), \quad (5.20)$$

where (5.20) satisfies the relation (5.6a) for $x_k^- = x(t_k^-)$, $x_k^+ = x(t_k^+)$ and $V_k = V(t_k)$. Next, we can define the following functions to describe the complementarity conditions we would like to enforce for the impulse update

$$g_i^L(\tilde{x}^-, \tilde{x}^+, V) = \begin{bmatrix} \Phi_i^{c,n} \\ \Phi_i^{c,n} \\ \Phi_i^{c,n} \\ \Gamma_i \end{bmatrix}, \quad (5.21a)$$

$$g_i^R(\tilde{x}^-, \tilde{x}^+, V) = \begin{bmatrix} \text{sd}(p_i^c(q^-)) \\ n_i^c(q^-)^\top J_i^c(q^-)(v^+ + \varepsilon_i^c(v^-)) \\ v_i^{c,t}(q^-, v^+) + \Gamma_i \nabla \|\Phi_i^{c,t}\|_\sim^\top \\ |\mu_i \Phi_i^{c,n}|_\sim - \|\Phi_i^{c,t}\|_\sim \end{bmatrix}, \quad (5.21b)$$

such that, for each $i = 1, \dots, n^c$, we have the complementarity condition associated with the impulse update stated as

$$0 \leq g_i^L(\tilde{x}_k^-, \tilde{x}_k^+, V_k) \perp g_i^R(\tilde{x}_k^-, \tilde{x}_k^+, V_k) \geq 0. \quad (5.22)$$

5.4.3 Complementarity Relaxation

A key aspect of most existing approaches for CITO is a smooth relaxation of the complementarity constraints. Complementarity is inherently nonsmooth: for scalars a and b the

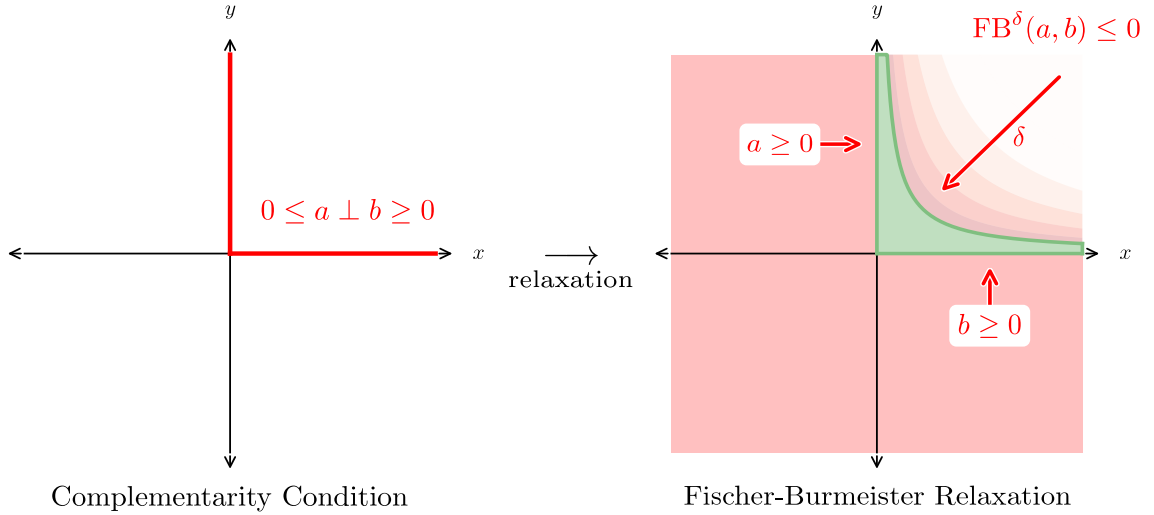


Figure 5.4: Illustration of the Fischer-Burmeister relaxation process

condition $0 \leq a \perp b \geq 0$ is equivalent to $\min(a, b) = 0$, i.e., feasible set is the nonnegative coordinate axes. We employ the well-known $\mathcal{C}^1$ Fischer–Burmeister function to relax a complementarity constraint

$$\text{FB}^\delta(a, b) \leq 0, \quad a \geq 0, \quad b \geq 0, \quad (5.23)$$

where $\text{FB}^\delta(a, b) = a + b - \sqrt{a^2 + b^2 + \delta}$ and $\delta > 0$ controls the tightness of the relaxation. As $\delta \rightarrow 0$ the feasible set of (5.23) approaches that of $\min(a, b) = 0$. Note, any other $\mathcal{C}^1$ nonlinear complementarity function could be used in place of Fischer–Burmeister. An illustration of the Fischer-Burmeister relaxation process is shown in Figure 5.4.

On the discretized grid, we can express the integral cross-complementarity condition as

$$0 \leq L(y_{i,k+1} - y_{i,k}) \perp R(y_{i,k+1} - y_{i,k}) \geq 0. \quad (5.24)$$

Additionally, we must ensure that we apply the relaxed constraint to ensure continuous-time constraint satisfaction for our path constraints $g(x(t), u(t)) \leq 0_{n_g}$ for all $t \in [0, t^f]$, via

constraint (1.19g). In our convention, this constraint can be stated as:

$$\theta_{k+1}^- - \theta_k^+ \leq \epsilon_{\text{CTCS}}. \quad (5.25)$$

Then, we will denote the expression

$$\Psi^{\delta, \epsilon_{\text{CTCS}}}(\tilde{x}_{k+1}^-, \tilde{x}_k^+) \leq 0 \quad (5.26)$$

as satisfying equations (5.24),(5.25) for all $i = 1, \dots, n^c$. We can do the same thing for the complementarity conditions associated with the impulse update by expressing

$$\Theta^\delta(\tilde{x}_k^+, \tilde{x}_k^-, V_k) \leq 0 \quad (5.27)$$

as satisfying equation (5.22) for all $i = 1, \dots, n^c$.

5.4.4 Nonconvex Optimization Problem

Finally, we can express the discretized hybrid CITO problem for the whole-body (WB) model as a nonconvex optimization problem in the following form:

WB-CITO Problem (5.28)

$$\underset{\tilde{x}_k^-, \tilde{x}_k^+, U_k, S_k, V_k}{\text{minimize}} \quad \sum_{k=1}^{N-1} L_k(\tilde{x}_k^+, U_k, S_k) \quad (5.28a)$$

$$\text{subject to} \quad \tilde{x}_{k+1}^- = \tilde{F}(\tilde{x}_k^+, U_k, S_k), \quad (5.28b)$$

$$\Psi^{\delta, \epsilon_{\text{CTCS}}}(\tilde{x}_{k+1}^-, \tilde{x}_k^+) \leq 0, \quad (5.28c)$$

$$U_k \in \mathcal{U}, S_k \in \mathcal{S}, \quad (5.28d)$$

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

$$\tilde{x}_k^+ = \tilde{G}(\tilde{x}_k^-, V_k), \quad (5.28e)$$

$$\Theta^\delta(\tilde{x}_k^+, \tilde{x}_k^-, V_k) \leq 0, \quad (5.28f)$$

$$k = 1, \dots, N,$$

$$\tilde{x}_1^- = \tilde{x}_i, \tilde{x}_N^+ = \tilde{x}_f. \quad (5.28g)$$

This problem encompasses continuous-time constraints on the discrete-time dynamics (5.28b), integral cross-complementarity and CTCS constraints (5.28c), and input constraints (5.28d). Here, $\mathcal{U}$ encodes joint torque bounds, nonnegativity of the multiplier γ_i and the contact force friction cone. Set $\mathcal{S}$ encodes strict positivity of the time dilation variable. In addition to continuous-time constraints, we also enforce the impulse update (5.28e), impulse complementarity conditions (5.28f), and boundary conditions (5.28g). We note that, in contrast to the form of the discretized proxlinear subproblem (1.19), constraints (5.28b), (5.28c), (5.28e), (5.28f) are maintained in their nonlinear form, and must be linearized before applying proxlinear machinery (as described in section 1.3.2).

The discretized CITO problem for the single-rigid-body (SRB) model can similarly be expressed as a nonconvex optimization problem in a slightly simplified form:

SRB-CITO Problem (5.29)

$$\underset{\tilde{x}_k, U_k, S_k}{\text{minimize}} \quad \sum_{k=1}^{N-1} L_k(\tilde{x}_k, U_k, S_k) \quad (5.29a)$$

$$\text{subject to} \quad \tilde{x}_{k+1} = \tilde{F}(\tilde{x}_k, U_k, S_k), \quad (5.29b)$$

$$\Psi^{\delta, \epsilon_{\text{CTCS}}}(\tilde{x}_{k+1}^-, \tilde{x}_k^+) \leq 0, \quad (5.29c)$$

$$U_k \in \mathcal{U}, S_k \in \mathcal{S}, \quad (5.29d)$$

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

$$\tilde{x}_1 = \tilde{x}_i, \tilde{x}_N = \tilde{x}_f. \quad (5.29e)$$

Notably, this problem does not contain hybrid mechanisms, therefore pre- and post-impact states are equivalent such that $\tilde{x}_k = \tilde{x}_k^- = \tilde{x}_k^+$, and the impulse update equations are not applicable.

5.4.5 Control Parameterization

For CITO problems, we must be deliberate with our choice of control parameterization as well. For the WB model, consider a choice $U = (U_{\tau^a}, U_{\phi^c}, U_{\gamma})$, where U_{τ^a} encodes parameterization of the joint torques, U_{ϕ^c} encodes parameterization of the contact forces, and U_{γ} encodes parameterization of the Lagrange multipliers for the maximum dissipation principle. Then, in order to ensure we can obtain a feasible solution for the CITO problem, we must ensure that the contact force parameterization U_{ϕ^c} is chosen such that the control may change instantaneously at the contact points. In practice, we achieve this by using a discontinuous first-order hold (dFOH) control parameterization, which simply superimposes a zero-order hold (ZOH) signal with a first-order hold (FOH) signal. The other two control signals, U_{τ^a} and U_{γ} , can be parameterized more arbitrarily; we opt for a FOH signal on each.

Similarly, in the case of the SRB model, we can choose a control parameterization $U = (U_{\phi^c}, U_{v^c})$, where U_{ϕ^c} encodes parameterization of the contact forces, and U_{v^c} encodes parameterization of the contact velocities. In this instance, we not only need a discontinuous

parameterization at the contact points for the contact forces U_{ϕ^c} , but also for the contact velocities U_{v^c} , since the latter enables us to encode a pseudo-impulse update. In both cases, we apply a dFOH control parameterization.

5.5 Sequential Convex Programming for CITO Problems

To solve the nonconvex problems (5.28),(5.29), we employ sequential convex programming (SCP), specifically the prox-linear method with exact penalization of nonconvex constraints, as is described in section 1.3.2. We refer to this collective approach as the *contact-implicit successive convexification* (CI-SCvx) framework. Applying SCP to problems with complementarity constraints requires special care. When using a relaxation such as (5.23), fixing the tightness parameter δ a priori often stalls the algorithm and prevents convergence. For reliable convergence, δ is typically driven toward zero via a homotopy over the SCP iterations [48, 80]. We propose two different homotopy schemes for δ : (i) an SCP-embedded approach with a backtracking capability (backtracking δ -homotopy), and (ii) a nested-loop approach (nested δ -homotopy).

5.5.1 Backtracking δ -Homotopy

We first describe the backtracking δ -homotopy scheme, in which δ is tightened progressively at each SCP (proxlinear) iterate, but can be temporarily relaxed to allow larger steps and broader search directions before committing to a smaller value. Algorithm 5.1 describes the approach. Let z_j , δ_j , J_j^{px} , and J_j^{ep} denote the subproblem solution, the tightness parameter, the proximal term, and the exact-penalty term, respectively, at SCP iterate j . Inputs to the algorithm are the initial tightness parameter δ_0 , an initial guess z_0 , tolerances $\delta^{\min}(< \delta_0)$, ϵ^{px} , ϵ^{ep} , a shrinkage factor $\alpha \in (0, 1)$, and a stall horizon N^{stall} . The update rule for the tightness parameter has three parts: (i) if either J_j^{px} or J_j^{ep} has not decreased relative to iterate $M = \max(j - N^{\text{stall}}, 1)$, set δ_{j+1} to the log-scale bisection between δ_j and δ_M , (ii) else, if either term has not decreased relative to the previous iterate, set $\delta_{j+1} = \delta_j$; (iii) otherwise, shrink the tightness parameter as $\delta_{j+1} = \alpha \delta_j$.

Algorithm 5.1 Backtracking δ -Homotopy

Require: $z_0, \delta_0, \delta^{\min}, \epsilon^{\text{px}}, \epsilon^{\text{ep}}, \alpha, N^{\text{stall}}$
Ensure: z_j

```

1:  $j \leftarrow 0, J_j^{\text{px}} \leftarrow 0, J_j^{\text{ep}} \leftarrow 0, \delta_j \leftarrow \delta_0$ 
2: while  $J_j^{\text{px}} > \epsilon^{\text{px}} \vee J_j^{\text{ep}} > \epsilon^{\text{ep}} \vee \delta_j > \delta^{\min}$  do
3:    $z_{j+1}, J_{j+1}^{\text{px}}, J_{j+1}^{\text{ep}} \leftarrow \text{solve\_subproblem}(z_j, \delta_j)$ 
4:    $M \leftarrow \max(j - N^{\text{stall}}, 1)$ 
5:   if  $J_{j+1}^{\text{px}} > J_M^{\text{px}} \vee J_{j+1}^{\text{ep}} > J_M^{\text{ep}}$  then
6:      $\delta_{j+1} \leftarrow \exp\left(\frac{1}{2} \log(\delta_j \delta_M)\right)$ 
7:   else if  $J_{j+1}^{\text{px}} > J_j^{\text{px}} \vee J_{j+1}^{\text{ep}} > J_j^{\text{ep}}$  then
8:      $\delta_{j+1} \leftarrow \delta_j$ 
9:   else
10:     $\delta_{j+1} \leftarrow \alpha \delta_j$ 
11:   end if
12:    $j \leftarrow j + 1$ 
13: end while

```

5.5.2 Nested δ -Homotopy

As an alternative to algorithm 5.1, we develop a nested (inner-outer) homotopy scheme for δ , which geometrically progresses δ after a full SCP solve repeatedly until the desired tolerance is achieved (without backtracking). We note that this approach closely resembles the nested homotopy scheme proposed in [92], since we are solving a full nonlinear complementarity problem at each outer iterate, and warmstarting the proceeding outer iterate with the previous outer iterate's solution. Algorithm 5.2 describes the approach. We assume the same variables as in algorithm 5.1, but without N^{stall} . The logic lends itself to a simpler description; an outer loop shrinks δ by a factor α until the desired tolerance is achieved, and an inner loop shrinks J_j^{px} and J_j^{ep} until the solver tolerances ϵ^{px} and ϵ^{ep} , respectively, are achieved.

Algorithm 5.2 Nested δ -Homotopy

Require: $z_0, \delta_0, \delta^{\min}, \epsilon^{\text{px}}, \epsilon^{\text{ep}}, \alpha$
Ensure: z_j

```

1:  $j \leftarrow 0, J_j^{\text{px}} \leftarrow 0, J_j^{\text{ep}} \leftarrow 0, \delta \leftarrow \delta_0$ 
2: while  $\delta > \delta^{\min}$  do
3:   while  $J_j^{\text{px}} > \epsilon^{\text{px}} \vee J_j^{\text{ep}} > \epsilon^{\text{ep}}$  do
4:      $z_{j+1}, J_{j+1}^{\text{px}}, J_{j+1}^{\text{ep}} \leftarrow \text{solve\_subproblem}(z_j, \delta)$ 
5:      $j \leftarrow j + 1$ 
6:   end while
7:    $\delta \leftarrow \alpha \delta$ 
8: end while
```

5.5.3 SRB to WB Warmstarting

In section 5.2, we defined contact-implicit dynamics for both the WB and SRB models, and in section 5.4, we described the CITO problem encompassing both of these dynamical models. While it is of interest to study each model class separately, we would also like to study the ability to warmstart a WB CITO problem from an SRB CITO problem, capitalizing on the lower-dimensional and less nonlinear nature of the latter. To this end, we define a warmstart procedure that maps an SRB CITO problem to a WB CITO problem, such that the initial guess for the WB CITO problem is the solution of the SRB CITO problem under some model error assumptions.

Figure 5.5 provides an overview of this approach. An optimized SRB-CITO problem provides a solution history of the base pose, $q_b(t)$, as well as the contact positions, $p_i^c(t)$, for each contact point $i = 1, \dots, n^c$, and their corresponding contact forces, $\phi_i^c(t)$. Notably, we forward-simulate this solution on a fine time grid such that we can treat the signal as approximately continuous. The first warmstarting step concerns using inverse kinematics (IK) [78, Chapter 6] to determine the full linkage state $q(t) = (q_1, \dots, q_{n^b})$, evaluated point-wise at a time t , which connects the base pose $q_b(t)$ to all contact positions $p_i^c(t)$. We note that this problem is not necessarily feasible; the imposed kinematic constraint (5.9) was formulated as an approximation of the true constraint (5.8) for practicality. The output of the IK feasibility problem will therefore project to the closest feasible point in the case that $p_i^c(t)$ is infeasible relative to $q(t)$. This is our first source of model error when transcribing

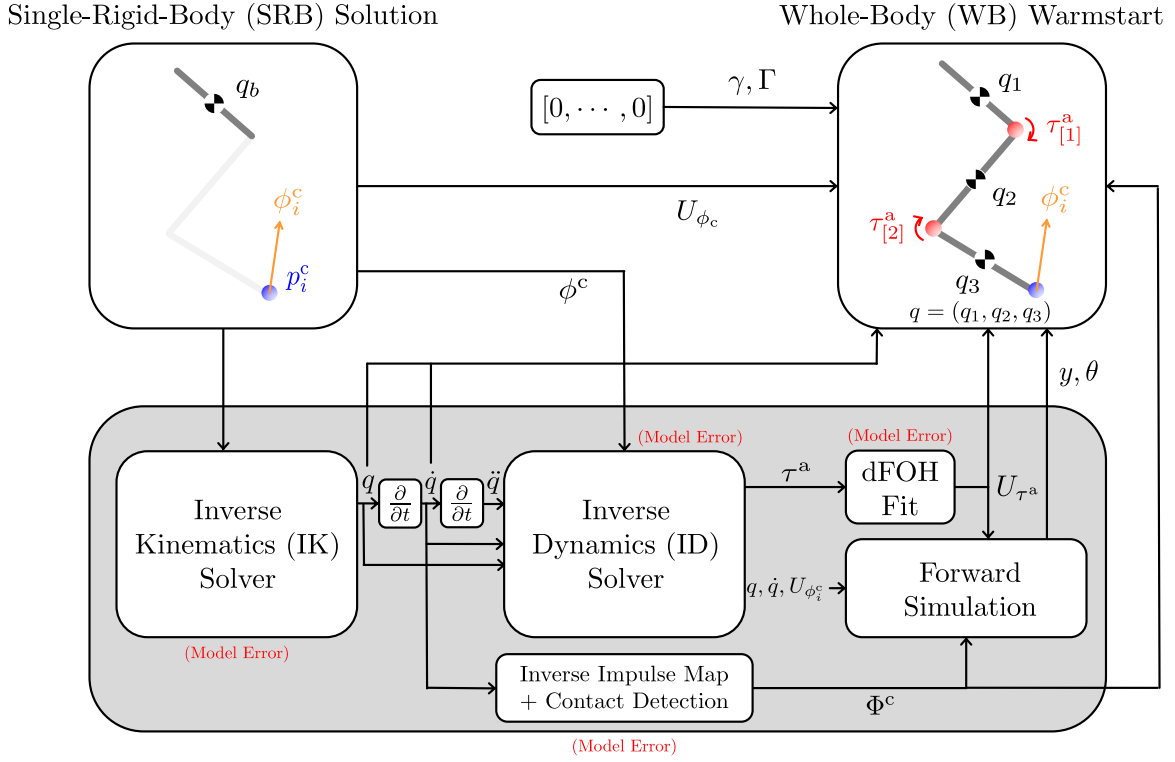


Figure 5.5: Illustration of the warmstart process from an SRB model to a WB model.

the SRB solution to a WB solution.

With a description of $q(t)$ at all times $t \in [0, t^f]$, we can then perform numerical differentiation to obtain time signals $\dot{q}(t), \ddot{q}(t)$. With these time signals, we can now perform the second major step of the warmstart procedure, which uses inverse dynamics (ID) [78, Chapter 8] to determine the joint torques $\tau^a(t)$ which explain the dynamics of the system as closely as possible. Moreover, we provide configuration signals $q(t), \dot{q}(t), \ddot{q}(t)$, as well as the contact forces $\phi^c(t)$, to solve equations (5.1) for $\tau^a(t)$ at each instant in time. Once again, we introduce model error in the case that our system is *underactuated*, meaning the number of independent joint torques and active (non-zero) contact forces is less than the number of configuration degrees of freedom. In this case, the ID problem may have no feasible solution, and so once again, we project to the closest feasible point in terms of configuration.

While the above steps enable us to obtain a continuous-time solution for the WB model,

we must also back out the contact impulses which explain the motion of the system. From $\dot{q}$, we can obtain the contact velocities $v^c(t)$ (or obtain them directly from the solution to the SRB model). At the optimization grid points, we detect if the post-impact velocity $\|v_i^c(\hat{t}^+)\|_2 \leq \epsilon_{\text{impulse}}$. If this is the case, we can then solve an inverse impulse problem on equation (5.6) to obtain the contact impulse $\Phi_i^c(\hat{t})$ which explains $q(\hat{t})$, $v^c(\hat{t}^-)$, and $v^c(\hat{t}^+)$. This introduces a third source of model error, because although we allowed the SRB model's $v^c(t)$ solution to be discontinuous as per section 5.4.5, we do not strictly constrain its degree of discontinuity.

The final source of model error arises from the fact that the ID solution's continuous joint torque solution $\tau^a(t)$ may not match the desired control parameterization (in this case, a dFOH parameterization). We perform a least-squares fit between the true signal and the closest feasible dFOH-parameterized signal. To complete the warmstart procedure, we can take our approximated signals $q(t)$, $\dot{q}(t)$, $U_{\phi^c}(t)$, $\tau^a(t)$, and forward-simulate a true WB solution across $t \in [0, t^f]$ to obtain integrator states y, θ . Because our SRB model is no-slip, the Lagrange multipliers associated with the maximum dissipation principle, $\gamma(t)$ and $\Gamma(t)$, are zero. The contact force parameters $U_{\phi^c}(t)$ can be fed directly from the SRB solution to the WB warmstart (assuming we have selected the same control parameterization for both).

5.5.4 Real-Time Implementation

In order to facilitate real-time implementation of `cr-SCvx`, we require a number of problem customizations encompassing an efficient transcription and canonicalization (convex subproblem parsing) pipeline. Figure 5.6 provides an overview of this process.

Our transcription step encompasses constructing and linearizing the contact-implicit dynamics equations (5.28b),(5.28e) in the case of WB, and (5.29b) in the case of SRB. We use the Jax library [13] to assemble efficient and differentiable implementations which parallelize across (i) grid intervals (multiple-shooting segments), (ii) decision variables at each node (state, control and impulse), and (iii) configuration elements within each continuous-time dynamics evaluation (such as joints and links in the case of WB models). We note that part (iii) is dependent on the specific type of model (for example, maximal-coordinate models enable more efficient parallelization of terms such as joint constraints and contact

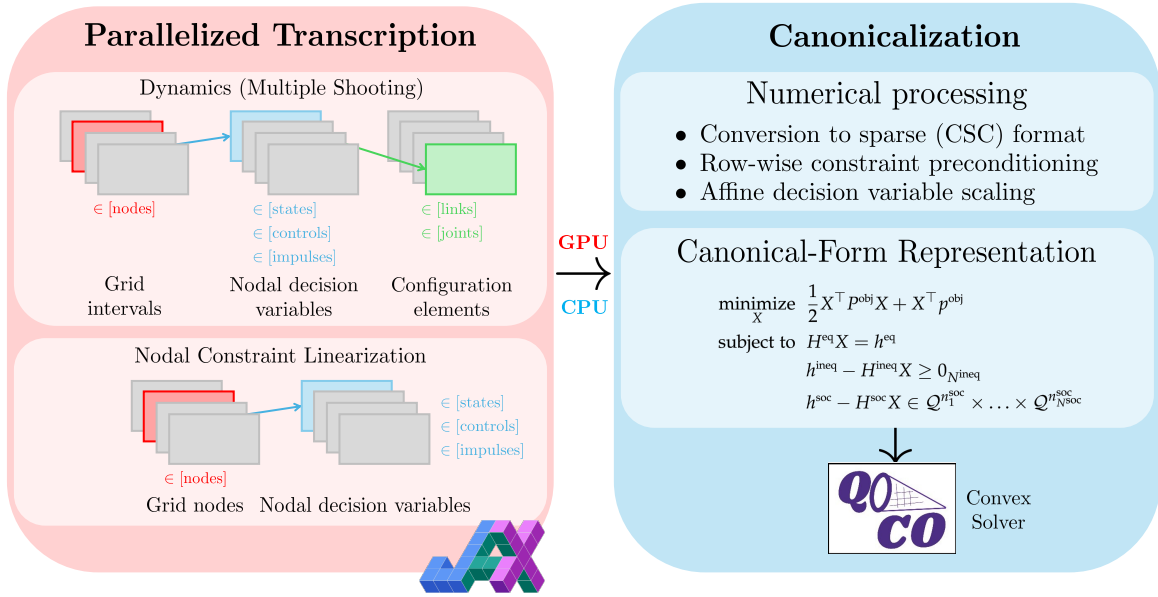


Figure 5.6: Illustration of the real-time implementation pipeline for ci-SCvx.

jacobians, as opposed to minimal-coordinate models, due to the fact that expression of link poses does not require forward kinematic rollouts). Transcription on each grid interval uses Internal Numerical Differentiation (IND) [110] with a custom Runge-Kutta-Fehlberg (RK45) integrator. We also allow for both CPU and GPU acceleration of the transcription step, and compile the resulting Jax functions for efficient and repeatable execution. Beyond transcription, we also evaluate the linearization of nodal constraints with similar efficiency; this includes constraints such as (5.28c),(5.28f) for the WB model, and (5.29c) for the SRB model.

We then transfer all quantities from the GPU (if applicable) to the CPU, and assemble the subproblem objective and constraints in a canonical-form representation such as the QOCO convex solver [26], allowing us to bypass general-purpose modeling tools, such as CVXPY, and accelerate the convex subproblem parsing step. Before this assembly, we apply several preprocessing steps to numerically condition the problem, including (i) conversion of all Jax-evaluated matrices to compressed sparse column (CSC) format SciPy arrays, (ii) row-wise constraint preconditioning, and (iii) affine scaling of the decision variables. We note that

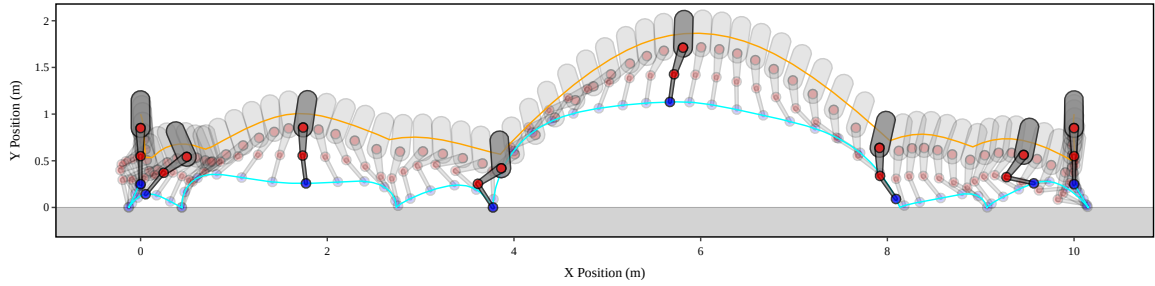
steps (ii) and (iii) can be considered dual and primal preconditioning, respectively, and are both important for numerical stability and convergence of the underlying SCP algorithm.

5.6 Experimental Results

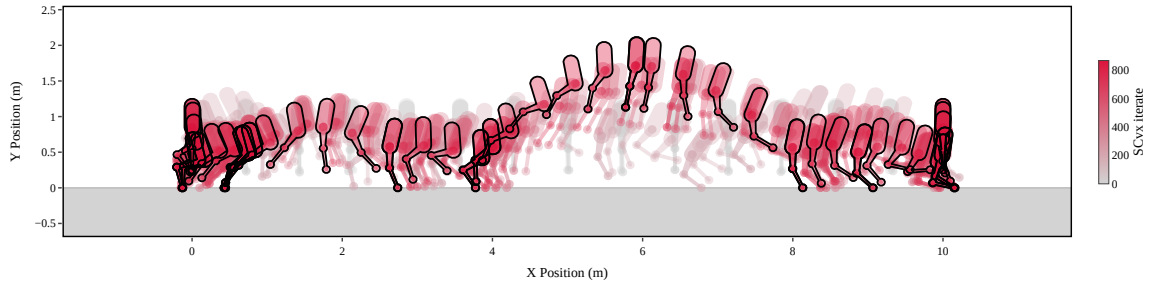
With the core methodology of `ci-SCvx` established, we now present experimental results across a variety of multibody configurations and environments which illustrate the versatility and effectiveness of our framework on practical problems in the robotics literature. We note that, when we are not generating an initial guess using the SRB→WB warmstart procedure, we perform an unstructured initialization as follows: (i) linearly interpolate (generalized) pose, (ii) set velocities, torques and Lagrange multipliers to zero, (iii) randomly sample contact forces from the friction cone such that their along-gravity component uniformly counteracts gravity across all contact points, and the remaining component may be selected randomly, and (iv) randomly sample contact impulses from the friction cone at $0.1\times$ the magnitude of the contact force. We note that this approach applies equivalently to either a WB or SRB model. A grid size of $N = 15$ and integration step count of 10 are used for all experiments, with grid interval wallclock time steps constrained between 0.05 and 0.5 seconds. Each scenario minimizes the time of maneuver t_f . We globally set $w_o = 10^{-3}$, $w_\eta = 1$, $w_\nu = 10^3$, $\delta_0 = 10^3$, $\delta^{\min} = 10^{-3}$, $N^{\text{stall}} = 10$, $\alpha = 0.85$, $\mu_i = 1$, and $\varepsilon_i^c = 0$, for $i = 1, \dots, n^c$. All results are visualized with the Plotly library in Python and forward-simulated with time steps of 0.01 seconds. We also note that, unless otherwise stated, it should be assumed that δ -homotopy is performed with algorithm 5.2 due to its robustness and ease-of-understanding.

5.6.1 Monoped Study

We start by investigating the performance of `ci-SCvx` on a simple three-link (Raibert-style) monoped model on a flat surface, while examining the impact of the SRB→WB warmstart procedure. We start by evaluating a WB model (minimal-coordinate) warmstarted by our unstructured initialization in figure 5.7. Figure 5.7a shows the gait produced by a minimal-coordinate WB model, and figure 5.7b showing its iteration history. We note that this solution converges in 37.53 seconds overall (in 873 SCP iterations), with early iterates being notably far away from the converged solution. When we examine the SRB→WB warmstart



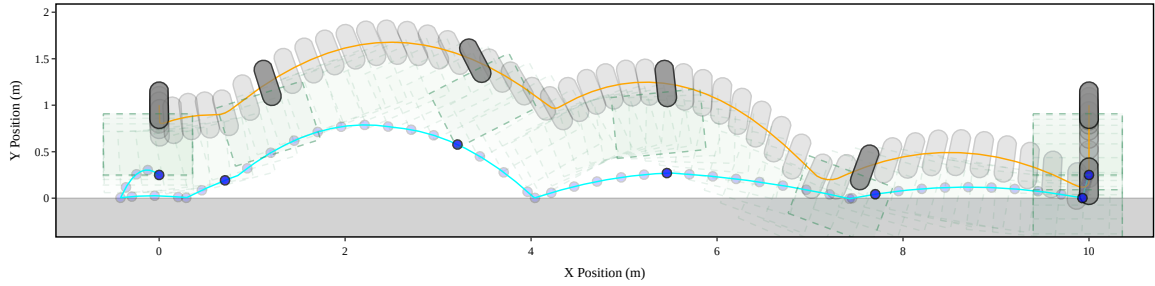
(a) Gait produced by the locomotion solution. The path of the center of mass (CoM) of the torso is shown by the orange trace, and the path of the single contact point (foot connected to the lower leg) is shown by the cyan trace.



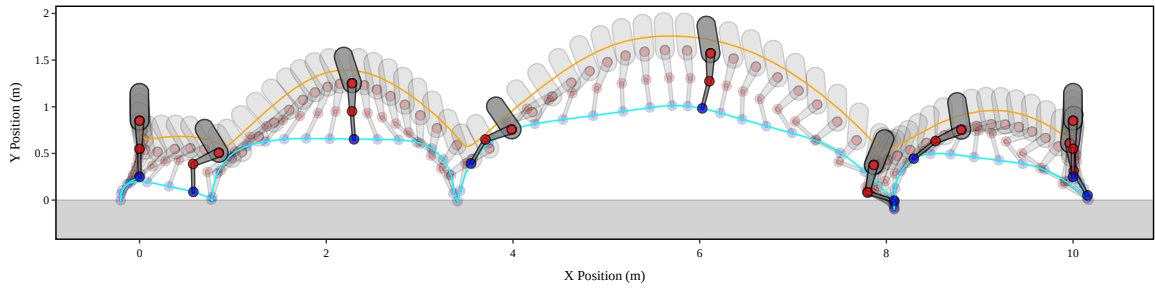
(b) SCP iteration history of the locomotion solution. 10 iterations are interpolated on the solution history and color coded from gray to red, with the converged solution bordered by a black outline.

Figure 5.7: Monoped locomotion solution on a flat surface generated by ci-SCvx with a minimal-coordinate WB model. The solution converges in 873 SCP iterations, taking 37.53 seconds to solve.

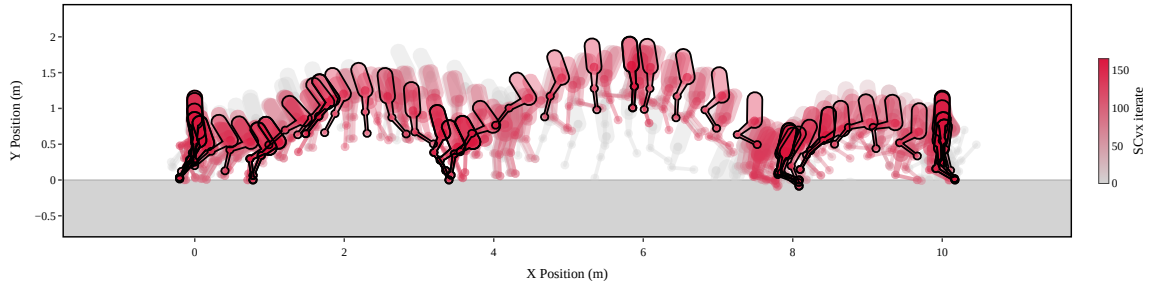
procedure in figure 5.8, we can see that the SRB model is able to converge much faster (only 340 SCP iterations), and the consequent WB warmstart lies much closer to the optimized solution of the WB model (as is shown in figure 5.8b), obtaining a solution in only 167 more SCP iterations. This process consequently enables us to converge in an elapsed time of 11.48 seconds, a $3.3\times$ speedup over the unstructured initialization, demonstrating the viability of this warmstart procedure. Figure 5.9 shows the contact forces and impulses for the warmstarted solution, showing that the solution takes five sustained no-slip contacts to get from the initial to terminal condition.



(a) Gait produced by the locomotion solution with an SRB model. The path of the center of mass (CoM) of the torso is shown by the orange trace, and the path of the free-moving single contact point is shown by the cyan trace, with its forward-kinematic constraint box shown by the green box.



(b) Gait produced by the locomotion solution with an SRB-warmstarted minimal-coordinate WB model, using the same visualization conventions of figure 5.7a.



(c) SCP iteration history of the locomotion solution with an SRB-warmstarted minimal-coordinate WB model, using the same visualization conventions of figure 5.7b.

Figure 5.8: Monopod locomotion solution on a flat surface generated by `ci-SCvx` with an SRB-warmstarted minimal-coordinate WB model. The solution converges in 340 SCP iterations (SRB) + 167 SCP iterations (WB), taking 11.48 seconds elapsed time to solve.

5.6.2 Biped Study

We now progress to analysis of a more complicated five-link biped model on a variety of surfaces. All tests are conducted with an SRB→minimal warmstart. Figure 5.10 shows the

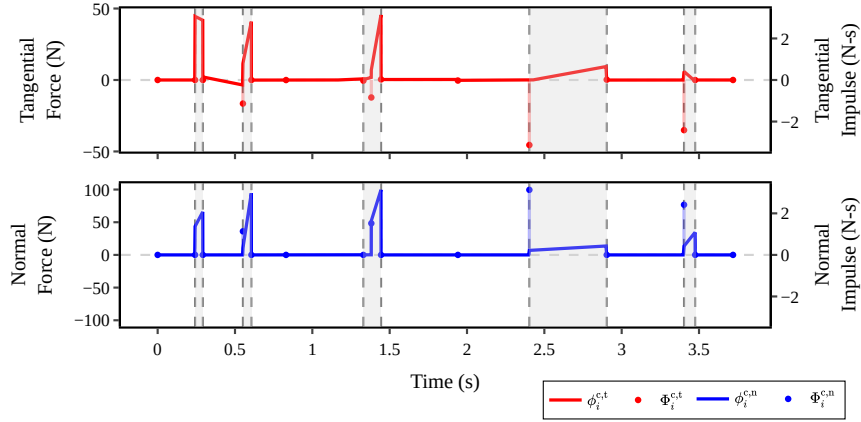
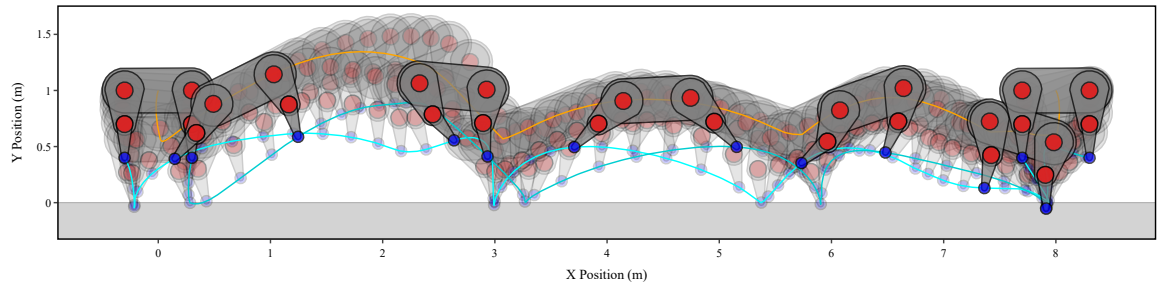


Figure 5.9: Time histories of the normal and tangential contact forces and impulses for the warmstarted monopod solution. Grid intervals with sustained contact are denoted by gray regions.

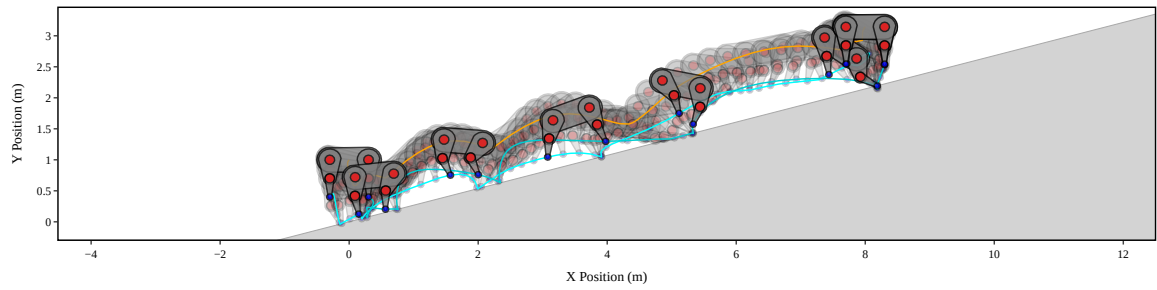
gaits produced by the biped on a variety of surfaces. We approximate the signed distance of the contact point to the contact surface as

$$\text{sd}(p_i^c(q)) \approx \frac{h^c(p_i^c(q))}{\|\nabla h^c(p_i^c(q))\|}. \quad (5.30)$$

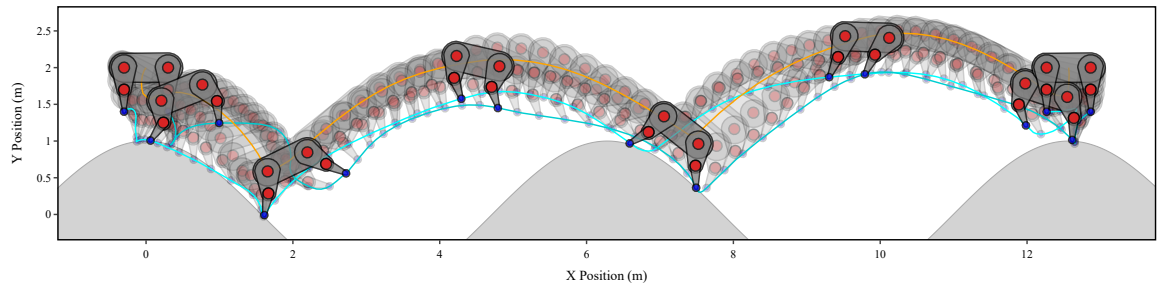
This choice (i) becomes arbitrarily accurate as $h^c(p_i^c(q)) \rightarrow 0$, (ii) is numerically stable far from the contact surface, and (iii) avoids solving a nonlinear optimization problem. The resulting plot shows that we can obtain capable gaits on a progression of increasingly-difficult surfaces to optimize for, starting with a flat surface and progressing towards a double-sinusoidal surface (superposition of two sinusoids with different frequencies and amplitudes). In the last case, we observe some ground penetration associated with the footholds (contact points) even in the fully converged (to numerical tolerance) solution, displaying an accuracy limitation of our approach, however we only experience this degree of violation in the most extreme environments (such as this one). We also observe an interesting phenomenon on the regular sinusoidal surface (figure 5.10c), where the biped’s hind foot performs prolonged slipping motions associated with each contact event. Figure 5.11 shows the time signals associated with the tangential velocity, Lagrange multiplier and friction cone evaluation for



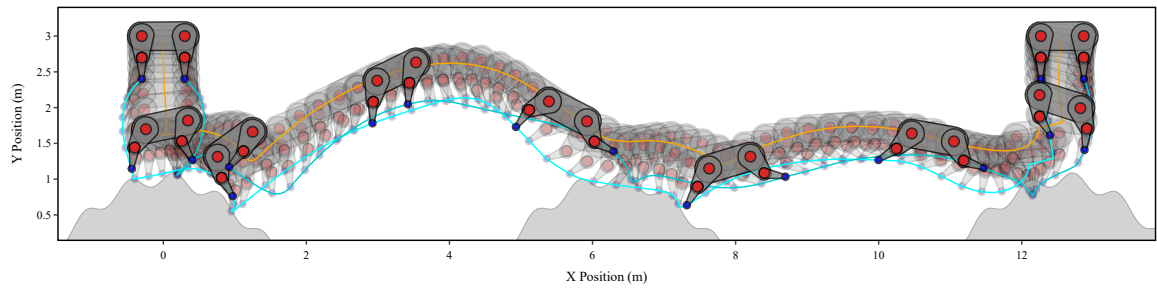
(a) Flat surface



(b) Sloped surface



(c) Sinusoidal surface



(d) Double-sinusoidal surface

Figure 5.10: Bipedal locomotion solutions on different surfaces. The path of the center of mass (CoM) of the torso is shown by the orange trace, and the path of each contact point (foot connected to each leg) is shown by the cyan traces.

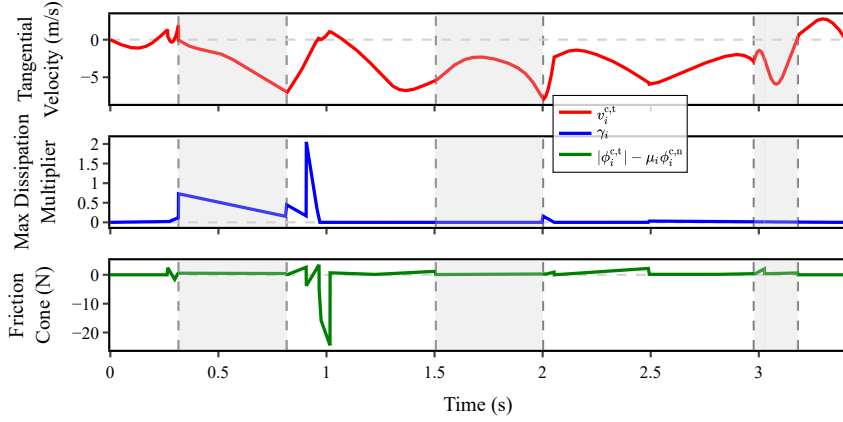


Figure 5.11: Time histories of tangential velocity, Lagrange multiplier and friction cone term for the hind foot of the biped on the sinusoidal surface (figure 5.10c).

the hind foot. From this plot, we can observe that the tangential velocity and Lagrange multiplier become non-zero across each contact event, with the friction cone being activated such that $|\mu_i \phi_i^{c,n}| - \|\phi_i^{c,t}\| \approx 0$. This result indicates that on the complex highly-sloped surface, an optimal solution may require the explicit use of slipping to perform efficient jumps, and that the cr-SCvx framework is able to accurately capture and optimize for this behavior.

5.6.3 Monte Carlo Analysis

The last two subsections present individual case studies of the monoped and biped models, respectively. We now move towards a systematic analysis of the performance of cr-SCvx across a variety of multibody models with a Monte Carlo comparison across 10 trials for each model. Specifically, we compare between (i) SRB, (ii) minimal-coordinate WB, (iii) maximal-coordinate WB, (iv) SRB→minimal warmstart, and (v) SRB→maximal warmstart. Figure 5.12 shows the solve time across the monoped and biped models between all five candidate multibody models on a flat surface. We observe the maximal-coordinate model performing the best in the monoped case. This can be explained by the fact that, in the case of a three-link monoped, there is not a substantive growth in the configuration space dimension for maximal-coordinate models yet.

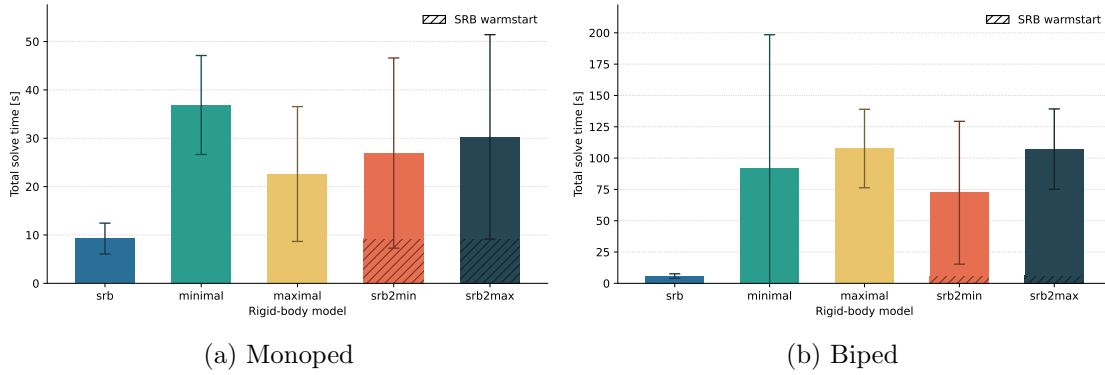


Figure 5.12: Monte Carlo analysis of the elapsed solve time across the monopod and biped models between all five candidate multibody models. The sample mean of each model’s elapsed solve time is displayed by a color-coded bar, with the error bars depicting one sample standard deviation.

When examining the solve time results for biped, however, we observe that our warmstart strategy is able to produce the best performance across WB solutions (specifically, SRB→minimal). In this instance, the benefits of an SRB warmstart become more important, and the underlying WB model becomes more nonlinear, such that the SRB model’s reduced-order complexity becomes more beneficial. Specifically, we see that the SRB→minimal warmstart solves 24.3% faster than the next best model (minimal-coordinate), and additionally solves with a much lower standard deviation, attributed to the solver stability introduced by an SRB warmstart. Figure 5.13 shows the objective value (minimum time) across all models, showing comparable performance that does tend to slightly favor the non-warmstarted maximal-coordinate model. This result displays that we do tend towards a tradeoff where an SRB warmstart may push us closer to a less-desirable local minimum, however in the context of CITO, we are primarily interested in finding a feasible solution to the contact-implicit dynamics equations in the shortest amount of time, so we accept this tradeoff going forwards, and investigate all other results with the SRB→minimal warmstart.

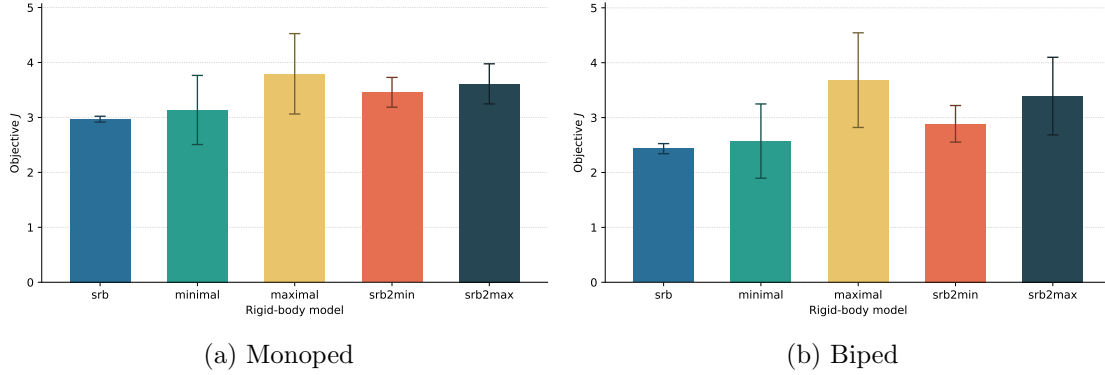


Figure 5.13: Monte Carlo analysis of the objective value across the monopod and biped models between all five candidate multibody models. The sample mean of each model’s objective value is displayed by a color-coded bar, with the error bars depicting one sample standard deviation.

Table 5.1: Half-Cheetah comparisons between CI-SCvx and MJPC.

Model	↓ Solve time [s]	J_η	J_ν	↓ ATE [m]	↓ $\int \ \tau^a\ _1$	↑ J_{gym}
SRB	5.3084	—	—	0.3056	605.0022	505.6226
WB (1/4)	11.4486	3.70×10^{-1}	2.24×10^{-2}	0.1480	810.5748	483.1663
WB (2/4)	16.8176	3.60×10^{-1}	3.14×10^{-2}	0.0975	652.1706	517.0439
WB (3/4)	23.1584	2.44×10^{-1}	3.10×10^{-2}	0.1349	586.3398	524.1444
WB (4/4)	28.1745	4.88×10^{-2}	5.80×10^{-2}	0.1667	617.2610	516.0035
WB (4/4, OL)	28.1745	4.88×10^{-2}	5.80×10^{-2}	3.4401	416.2654	−80.7163
MJPC	625.8882	—	—	—	643.2141	248.6145

Ablation study: δ -homotopy

We additionally perform an ablation study comparing the two δ -homotopy schemes, namely backtracking (algorithm 5.1) and nested (algorithm 5.2). From figure 5.14, we can observe that, in terms of solve time, the embedded backtracking homotopy scheme performs the best, both in terms of mean and standard deviation, displaying that our customized technique is able to outperform the extension from [92]. We do observe a tradeoff in terms of objective value, similar to what has been observed in the model-level analysis previously.

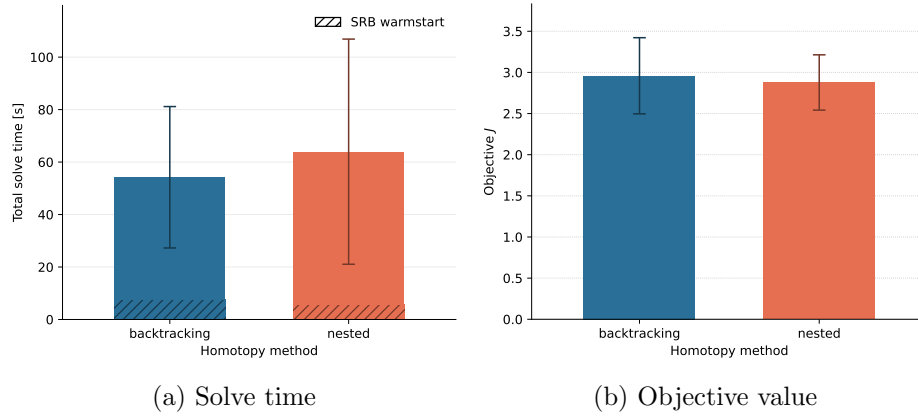
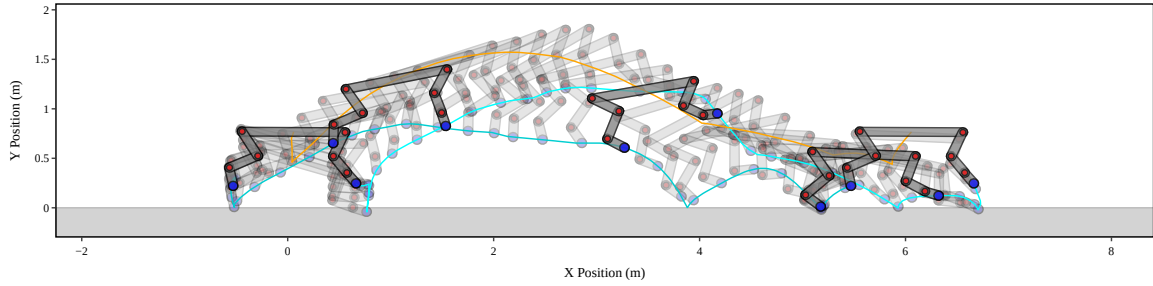


Figure 5.14: Ablation study of the δ -homotopy scheme on the biped model. The sample mean of each method’s solve time and objective value is displayed by a color-coded bar, with the error bars depicting one sample standard deviation.

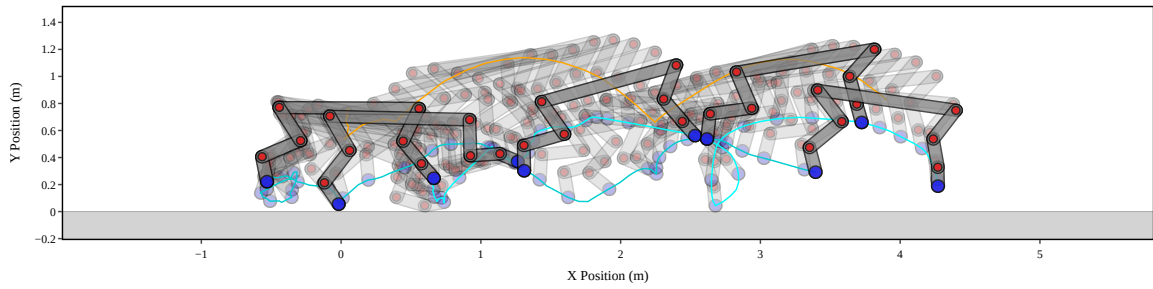
5.6.4 Half-Cheetah Study

The final component of our experimental results is a study of the performance of ci-SCvx in a high-fidelity simulator environment. Specifically, we use the Gymnasium HalfCheetah environment [134], which is a 2D planar seven-link model with two contact points (one extra limb per each leg relative to the previous biped configuration) with a MuJoCo backend. We compare the performance of ci-SCvx against the MuJoCo-based sampling model predictive control (MPC) framework MuJoCo-MPC (MJPC) [55]. For closed-loop analysis of ci-SCvx, we use the cross-entropy method (CEM) as a sampling-based tracker [103]. The environment objective of HalfCheetah is a weighted combination of maximizing forward velocity over the trajectory horizon, and minimizing the control effort in terms of the squared sum of joint torques, also over the trajectory horizon.

To enable ci-SCvx to produce practical solutions for this environment, we construct a two-point boundary value problem (TPBVP) where the terminal velocities are left free (unlike in previous scenarios, where the terminal velocities were constrained to be zero), and minimize for the time of maneuver as before. We utilize the SRB→minimal warmstart strategy to produce solutions as quickly as possible. For MJPC, we specify a reward function which



(a) CI-SCvx gait solution (SRB→minimal warmstart, fully-converged).



(b) MJPC gait solution.

Figure 5.15: HalfCheetah locomotion solutions on a flat surface across multiple tested configurations. The path of the center of mass (CoM) of torso is shown by the orange trace, and the path of the each contact point (foot connected to each leg) is shown by the cyan traces.

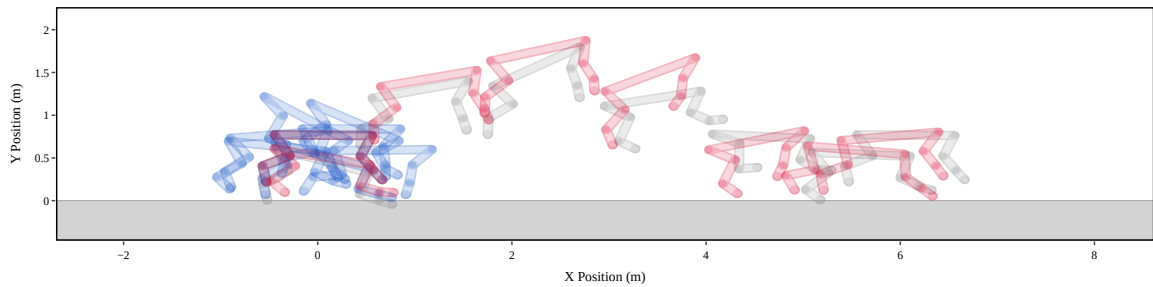


Figure 5.16: Comparison of open-loop (blue) and closed-loop (red) gait solutions for the HalfCheetah model on a flat surface when deployed in the MuJoCo simulator, against the optimized CI-SCvx solution directly propagated on its own dynamics model (gray).

balances the environment objective with a gait-phasing term that encourages alternating foot contacts to produce efficient motion. Sample rollouts are discarded if the torso exceeds

a critical pitch angle or drops below a critical height (indicating tip-over and slumping failure modes) to ensure MJPC cannot greedily prioritize poor trajectories. To compare between CI-SCvx and MJPC, we allow MJPC to run for the time of maneuver of the CI-SCvx solution, such that both methods are evaluated over the same time horizon, and the competing performance is evaluated in terms of environment reward which can be obtained over this time horizon.

Table 5.1 shows a detailed analysis of how CI-SCvx performs at different levels of convergence, and in comparison to MJPC. For each model, we analyze across: (i) elapsed solve time, (ii) proximal term objective weight (WB only), J_η , (iii) exact penalization objective weight (WB only), J_ν , (iv) average trajectory error (ATE) for the base body’s 2D position, (v) cumulative torque over the maneuver, and (vi) the HalfCheetah environment objective/reward, J_{gym} . We note that the ATE is computed as the average across all time steps of the Euclidean distance between the CI-SCvx-optimized trajectory and the trajectory produced by forward-simulating the control inputs (open- or closed-loop) in the MuJoCo simulator. We analyze CI-SCvx convergence at three levels: (i) SRB warmstart only, (ii) WB partially converged by a fraction ($x/4$) of the total number of SCP iterations during the WB solver phase (96 iterations in this case), and (iii) WB fully converged to numerical tolerance, along with an analysis of the open-loop (OL) response.

From table 5.1, we can derive a few important observations. First, the open-loop response of CI-SCvx is unable to produce a viable forward locomotion solution, resulting in $J_{\text{gym}} < 0$ and justifying the use of closed-loop tracking. Second, we observe that partially-converged solutions, even up to the point of just using the SRB warmstart, are able to be tracked with reasonably-small error ($\text{ATE} < 0.4$ m), however the best results are obtained with either the fully-converged or (3/4)-converged solutions, demonstrating that our optimizer can be used to produce practical motions with significantly-less solve-time expenditure than a fully-converged solution, suggesting its potential practicality towards implementations such as real-time iteration (RTI) MPC [93]. Third, our method is able to heavily outperform MJPC in terms of the environment objective, with a J_{gym} value $2.08\times$ higher, and a solve time $22.21\times$ faster, than MJPC. We note that this implementation of MJPC is not optimized

for speed, and that a more efficient implementation should be able to produce real-time performance, however we do not expect it to be able to outperform `ci-SCvx` in terms of the environment objective, as MJPC is a sampling-based method which does not explicitly optimize for the environment objective, and instead relies on a reward function to guide its search.

Figure 5.15 shows a comparison of the gaits produced by `ci-SCvx` and MJPC, showing that `ci-SCvx` produces substantially more horizontal distance, and consequently environment objective, over the optimized time horizon of $t_f = 1.70$ seconds. Figure 5.16 shows a comparison of the open-loop and closed-loop responses of `ci-SCvx` against the optimized solution, showing that the open-loop solution is unable to make sufficient forward progress without closed-loop tracking.

Chapter 6

CONCLUSIONS

This chapter concludes the dissertation by summarizing the key conclusions from each chapter and providing recommendations for future directions.

Chapter 2: Set-Based Viewplanning

The second chapter introduced the basic problem of set-based viewplanning, and introduced a new constraint formulation which can tractably solve the set containment problem. This chapter also introduced a sequential convex programming (SCP) framework to impose the resulting constraint set in mission-practical powered descent guidance scenarios, labeled constrained visibility guidance (CVG), and showed that highly-constrained set-based viewplanning problems could be solved tractably in real-time. One interesting direction for future work is to extend the set containment theory towards more general cases, such as: (i) p -norm field-of-view cone definitions with non-unitary aspect ratios, and (ii) non-spherical (e.g. ellipsoidal) set definition. Another interesting direction concerns viewing set containment from the perspective of robust optimal control (for example, assuming the landmark must fall within a bounded set) and applying this formulation to uncertain terrain environments.

Chapter 3: Active Simultaneous Localization & Mapping

The third chapter addressed a fundamental limitation with the previous chapter's approach in terms of parameterizing viewplanning (e.g. how long should you look at something, from what position, etc.). To address this, the chapter introduced a novel approach to the active simultaneous localization & mapping (SLAM) problem formulated as a continuous-time stochastic optimal control problem with a differentiable filtering model. The resulting ACT-SLAM framework was shown to precisely characterize vehicle and landmark uncertainties while optimizing for information gain favorably compared to relevant baselines. To

extend this work towards more generalized applications, perhaps those with a much larger number of landmarks, one interesting direction concerns the investigation of information filter parameterizations. Information filters have been studied at length in the SLAM literature [40, 140], as well as in the context of active SLAM [67, 90, 120, 133], however their application to continuous-time optimal control problems still remains an open problem. Additionally, the problem of handling landmark occlusions must be addressed in order to apply ACT-SLAM towards more generalized environments (such as cluttered building navigation).

Chapter 4: Graph Successive Convexification

The fourth chapter introduced an alternative approach towards unknown perceptive environments by instead *passively* perceiving the environment and proactively generating contingency plans as perceptive information becomes available. The resulting modeling approach, graph successive convexification (GRAPH-SCvx), was shown to be able to solve general non-convex deferred-decision trajectory optimization (DDTO) problems in real-time, in both high-fidelity simulation and hardware-in-the-loop (flight test) settings. One interesting future direction of this work is to lift the fixed graph structure of GRAPH-DDTO and instead optimize over all possible arborescence graph structures, such that the optimal graph topology for a given deferrability objective can be found.

Chapter 5: Contact-Implicit Successive Convexification

The final chapter addressed the other end of interaction-aware trajectory optimization by modeling and optimizing over contact interactions implicitly. The resulting contact-implicit successive convexification (CI-SCvx) framework was shown to produce high-fidelity trajectories for legged locomotion tasks in near-real-time, and was validated in high-fidelity environments against a relevant baseline (MuJoCo-MPC). The first immediate extension of this work concerns application of this framework towards more complex 3D locomotion tasks, such as quadrupedal gait planning, and even further towards manipulation tasks. Additionally, the framework could be extended to handle more complex contact interaction, such as contact between two sets rather than between a singular point and a set.

Appendix A

PHASE-BASED STITCHING

A.1 Modeling Phase-Based Stitching

We will reconsider the stitching constraint $\mathcal{T}$ discussed in section 4.2.2. Moreover, we consider the construction of two independent trajectories

$$(x(t^{(1)}), u(t^{(1)})), \quad t^{(1)} \in [t_i^{(1)}, t_f^{(1)}], \quad (\text{A.1a})$$

$$(y(t^{(2)}), v(t^{(2)})), \quad t^{(2)} \in [t_i^{(2)}, t_f^{(2)}], \quad (\text{A.1b})$$

where both trajectories obey the same dynamics relationship f such that $\dot{x}(t^{(1)}) = f(t^{(1)}, x(t^{(1)}), u(t^{(1)}))$ and $\dot{y}(t^{(2)}) = f(t^{(2)}, y(t^{(2)}), v(t^{(2)}))$.

Proposition A.1. *Consider the stitching relation $y(t_i^{(2)}) \in \mathcal{T}(x(t_f^{(1)}))$. If we choose $\mathcal{T}$ as the identity relation (instantaneous stitching), such that $y(t_i^{(2)}) = x(t_f^{(1)})$, this relation leads to a redundant knot point during discretization.*

Proof. Consider a discretization such that $\mathcal{X}_1 = (x_1, \dots, x_{N_x})$ and $\mathcal{X}_2 = (y_1, \dots, y_{N_y})$ with $y_1 = \mathcal{T}(x_{N_x}) = x_{N_x}$. If the discretized trajectories are concatenated such that $\mathcal{X} = (\mathcal{X}_1, \mathcal{X}_2)$, we observe a redundant trajectory knot point through the expansion $\mathcal{X} = (x_1, \dots, x_{N_x}, x_{N_x}, \dots, y_{N_y})$. □

If we instead choose $\mathcal{T}$ such that it expresses a continuous dynamical relationship between $x(t_f^{(1)})$ and $y(t_i^{(2)})$ (phase-based stitching), we can avoid this redundancy entirely. To do this, we will consider construction of a third trajectory $(z(t^{(3)}), w(t^{(3)}))$ with $t^{(3)} \in [t_i^{(3)}, t_f^{(3)}]$ which solves the two-point boundary value problem (TPBVP) between $x(t_f^{(1)})$ and $y(t_i^{(2)})$

such that

$$\dot{z}(t^{(3)}) = f(t^{(3)}, z(t^{(3)}), w(t^{(3)})), \quad (\text{A.2a})$$

$$z(t_i^{(3)}) = x(t_f^{(1)}), \quad (\text{A.2b})$$

$$z(t_f^{(3)}) = y(t_i^{(2)}). \quad (\text{A.2c})$$

Then, the stitching constraint set is defined such that $y(t_i^{(2)}) \in \mathcal{T}(x(t_f^{(1)}))$ with

$$\mathcal{T}_{\mathcal{Z}}(x) := \left\{ y \in \mathbb{R}^{n_x} \left| \begin{array}{l} (z(t), w(t)) \in \mathbb{R}^{n_x} \times \mathbb{R}^{n_u}, \\ \dot{z}(t) = f(t, z(t), w(t)), \\ z(t_i^{(3)}) = x, z(t_f^{(3)}) = y, \\ (t, z(t), w(t)) \in \mathcal{Z}, \\ t \in [t_i^{(3)}, t_f^{(3)}]. \end{array} \right. \right\}, \quad (\text{A.3})$$

where the set $\mathcal{Z}$ enables us to embed path constraints in the stitching constraint definition.

If we return to the continuous-time OCP formulation in problem (4.1), we can define path constraints on the j th segment as follows:

$$\mathcal{Z}^j \triangleq \{(t, z, w) \mid g^j(t, z, w) \leq 0, h^j(t, z, w) = 0\}.$$

Then, we can replace instantaneous stitching constraints (4.1h)-(4.1j) with the phase-based stitching constraint set, defined as

$$x^j(t_i^j) \in \mathcal{T}_{\mathcal{Z}^j}(x^i(t^{i \rightarrow j})), \quad (\text{A.4a})$$

$$t_i^j > t^{i \rightarrow j}, \quad (\text{A.4b})$$

for each $\varepsilon_{ij} \in \mathcal{E}$. Notably, constraint (A.4b) becomes necessary since the intermediate trajectory must correspond to a finite time duration. After the problem augmentation steps

in sections 4.2.3-4.2.4, we can define an augmented stitching constraint set as

$$\tilde{\mathcal{T}}^j(\tilde{x}) := \left\{ \tilde{y} \in \mathbb{R}^{n_{\tilde{x}}} \left| \begin{array}{l} (\tilde{z}(\tau), \tilde{w}(\tau)) \in \mathbb{R}^{n_{\tilde{x}}} \times \mathbb{R}^{n_{\tilde{u}}}, \\ \dot{\tilde{z}}(\tau) = F(\tilde{z}(\tau), \tilde{w}(\tau)), \\ \tilde{z}(0) = \tilde{x}, \tilde{z}(1) = \tilde{y}, \\ {}^\theta E(\tilde{z}(1) - \tilde{z}(0)) = 0, \\ {}^s E \tilde{w}(\tau) > 0, \\ \tau \in [0, 1]. \end{array} \right. \right\} \quad (\text{A.5})$$

Finally, the augmented constraint set which replaces (4.7g) can be stated as

$$\tilde{x}_i^j \in \tilde{\mathcal{T}}^j(\tilde{x}^{i \rightarrow j}), \quad (\text{A.6})$$

for each $\varepsilon_{ij} \in \mathcal{E}$, where constraint (A.4b) is implicit by setting ${}^s E \tilde{w}(\tau) > 0$.

A.2 Discretizing Phase-Based Stitching

To perform transcription of the phase-based stitching constraint defined by (A.5), we can make use of the discretization and parameterization definitions in section 4.3.1. Consider discretization of a stitching segment $\tilde{z}(\tau), \tilde{w}(\tau)$ onto a finite grid with *two* discrete knot points $\tau_1^z = 0, \tau_2^z = 1$ and $\tilde{Z} = (\tilde{z}_1, \tilde{z}_2)$. We can also define control parameters $W \in \mathcal{W} \subseteq \mathbb{R}^{2n_{\tilde{u}}}$ and basis function $\Gamma_w(\tau) : [0, 1] \rightarrow \mathbb{R}^2$ such that

$$\tilde{w}(\tau) = (\Gamma_w(\tau) \otimes I_{n_{\tilde{u}}})W. \quad (\text{A.7})$$

Then, to impose the dynamics and path constraints defined by equation (A.5), we must impose these constraints in discrete time such that

$$F_s(\tilde{z}_2, \tilde{z}_1, W) = F_1(\tilde{z}_2, \tilde{z}_1, W; \Gamma_w) = 0, \quad (\text{A.8a})$$

$${}^\theta E(\tilde{z}_2 - \tilde{z}_1) = 0, \quad (\text{A.8b})$$

$$W \in \mathcal{W}, \quad (\text{A.8c})$$

where $W \in \mathcal{W}$ captures the constraint ${}^sE\tilde{w}(\tau) > 0$. We now consider an edge $\varepsilon_{ij} \in \mathcal{E}$, and designate a stitching index $k_{i \rightarrow j}$ such that $\tilde{x}^{i \rightarrow j} = \tilde{x}_{k_{i \rightarrow j}}$. Under the definition of the stitching function, we can use a variable substitution for the boundary conditions, $\tilde{z}_1 = \tilde{x}_{k_{i \rightarrow j}}^i$ and $\tilde{z}_2 = \tilde{x}_1^j$, to restate equations (A.8a)-(A.8c) as

$$\begin{aligned} F^{ij}(\tilde{x}_1^j, \tilde{x}_{k_{i \rightarrow j}}^i, \phi^{ij}(U^i, U^j)) = \\ F_1(\tilde{x}_1^j, \tilde{x}_{k_{i \rightarrow j}}^i, \phi^{ij}(U^i, U^j); \Gamma^{ij}) = 0, \end{aligned} \quad (\text{A.9a})$$

$$\theta E(\tilde{x}_1^j - \tilde{x}_{k_{i \rightarrow j}}^i) = 0, \quad (\text{A.9b})$$

$$U_i \in \mathcal{U}^i, U^j \in \mathcal{U}^j, \quad (\text{A.9c})$$

where $\phi^{ij} : \mathcal{U}^i \times \mathcal{U}^j \rightarrow \mathcal{U}^{ij} \subseteq \mathbb{R}^{N_u^{ij}}$ is chosen to express the control parameterization across an edge segment, with a corresponding basis function $\Gamma^{ij}(\tau) : [0, 1] \rightarrow \mathbb{R}^{N_u^{ij}}$.

Example A.1 (*Phase-based edge continuity under FOH parameterization*). As an extension of example 4.1, under a first-order hold parameterization, we can state the parameterization terms as follows:

$$\phi^{ij}(U^i, U^j) = (U_{k_{i \rightarrow j}}^i, U_1^j), \quad (\text{A.10a})$$

$$\Gamma^{ij}(\tau) = \Gamma_1(\tau). \quad (\text{A.10b})$$

We can consequently describe the linearization of constraint (A.9a) as

$$\Psi^{ij}(\tilde{x}_1^j, \tilde{x}_{k_{i \rightarrow j}}^i, \phi^{ij}(U^i, U^j)) = \Psi_1(\tilde{x}_1^j, \tilde{x}_{k_{i \rightarrow j}}^i, \phi^{ij}(U^i, U^j); \bar{Z}^{ij}, \Gamma^{ij}), \quad (\text{A.11})$$

along with terms

$$C^{ij}(\bar{Z}^{ij}) = C_1(\bar{Z}^{ij}, \Gamma^{ij}), \quad (\text{A.12a})$$

$$c^{ij}(\bar{Z}^{ij}) = c_1(\bar{Z}^{ij}, \Gamma^{ij}), \quad (\text{A.12b})$$

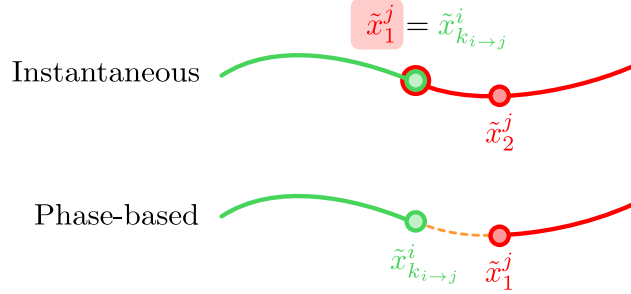


Figure A.1: Illustration of how variable elimination works with the non-redundant stitching formulation. In the redundant case, post-discretization, $\tilde{x}_1^j$ is a redundant variable. In the non-redundant case, $\tilde{x}_1^j$ is able to serve the same utility as the second discretization point, $\tilde{x}_2^j$, in the redundant case.

where $\bar{Z}^{ij} = (\bar{x}_{k_i \rightarrow j}^i, \bar{x}_1^j, \bar{U}_{k_i \rightarrow j}^i, \bar{U}_1^j)$. Notably, because of the variable substitution combined with two-point discretization, we are able to completely eliminate the edge segment trajectory $\tilde{Z}$ in the discrete-time formulation, allowing us to address the redundancy consideration in proposition A.1. The intuition behind this idea is enabling a dynamical mapping between two points rather than effectively copying a variable through a direct equality constraint. Then, rather than imposing constraint (4.19h) in problem (4.19), we can instead impose the discrete-time dynamics relationship defined over edges $\varepsilon_{ij} \in \mathcal{E}$ stated as

$$F^{ij}(\tilde{x}_1^j, \tilde{x}_{k_i \rightarrow j}^i, \phi^{ij}(U^i, U^j)) = 0, \quad (\text{A.13a})$$

$${}^\theta E(\tilde{x}_1^j - \tilde{x}_{k_i \rightarrow j}^i) \leq \epsilon_\theta, \quad (\text{A.13b})$$

$${}^t E(\tilde{x}_1^j - \tilde{x}_{k_i \rightarrow j}^i) > 0. \quad (\text{A.13c})$$

To prove discrete-time equivalence of instantaneous and phase-based stitching constraints, consider fixing an edge $\varepsilon_{ij} \in \mathcal{E}$ with parent stitch index $m = k_i \rightarrow j$, and assume both phase-based and instantaneous techniques are using first-order hold (see examples 4.1, 4.2, and A.1).

Proposition A.2. *Eliminating the child knot point through equality (4.19h) yields a cost-*

preserving bijection between the feasible sets of the instantaneous and phase-based transcriptions of problem (4.19), such that the two define the same problem and consequently yield the same optimal value.

Proof. Denote a phase-based trajectory as $(\tilde{X}^j, U^j)$ and an instantaneous trajectory as $(\check{X}^j, \check{U}^j)$. The strict equality (4.19h) constrains $\check{x}_1^j = \check{x}_m^i$, and the FOH parameterization constrains $\check{U}_1^j = \check{U}_m^i$. Then, under first-order hold, the child's first knot point interval defect becomes

$$F_1^j(\check{x}_2^j, \check{x}_1^j, \check{U}^j) = F_1(\check{x}_2^j, \check{x}_m^i, (\check{U}_m^i, \check{U}_2^j); \Gamma_1). \quad (\text{A.14})$$

The bijective map $\Phi : (\check{X}^j, \check{U}^j) \mapsto (\tilde{X}^j, U^j)$, defined by $\tilde{x}_k^j = \check{x}_{k+1}^j$ and $U_k^j = \check{U}_{k+1}^j$ for $k \in [N^j - 1]$, produces

$$F^{ij}(\tilde{x}_1^j, \tilde{x}_m^i, \phi^{ij}(U^i, U^j)), \quad (\text{A.15})$$

which exactly recovers the phase-based defect (A.13a). Every remaining phase-based constraint (A.13c), (A.13b) carries over under Φ without change, and the Mayer objective (4.19a) depends only on the terminal knots, which Φ preserves. Hence Φ is a cost-preserving bijection between the feasible sets of the two transcriptions. $\square$

Using proposition A.2, we can now state that the discrete-time phase-based and instantaneous stitching constraints are equivalent. An illustration of this equivalence is provided in figure A.1.

A.3 Implementing Phase-Based Stitching in SCP

To embed the discretized phase-based stitching constraints defined by equations (A.13a)-(A.13c) in the SCP algorithm defined in section 4.4, we simply need to redefine some variables. Moreover, to impose constraint (A.13b) as a convex constraint, we first replace convex set (4.20b) with a set defined as

$$\tilde{\mathcal{X}}_s^{\mathcal{E}} := \left\{ (\tilde{X}^1, \dots, \tilde{X}^n) \left| \begin{array}{l} \theta E(\tilde{x}_1^j - \tilde{x}_{k_i \rightarrow j}^i) \leq \epsilon_\theta, \\ tE(\tilde{x}_1^j - \tilde{x}_{k_i \rightarrow j}^i) \geq \Delta t_{\min}, \\ \forall \varepsilon_{ij} \in \mathcal{E}, \end{array} \right. \right\}, \quad (\text{A.16})$$

where $\Delta t_{\min}$ lifts (A.13c) to a non-strict inequality for numerical purposes. Then, to impose the nonconvex constraint (A.13a), we must define a residual term for the constraint as

$$R^{F^{ij}}(z) = F^{ij}(\tilde{x}_1^j, \tilde{x}_{k_i \rightarrow j}^i, \phi^{ij}(U^i, U^j)), \quad (\text{A.17})$$

and consequently replace the concatenated equality constraint residual term (4.22a) with a new term

$$R_{e,s} = \left((R^{F_k^j})_{k \in [N^j-1]}^{j \in [n]}, (R^{Q^j})_{j \in [n]}, (R^{F^{ij}})_{\varepsilon_{ij} \in \mathcal{E}} \right). \quad (\text{A.18})$$

Then, all remaining steps throughout sections 4.4.1-4.4.2 proceed with $\tilde{\mathcal{X}}^{\mathcal{E}} = \tilde{\mathcal{X}}_s^{\mathcal{E}}$ and $R_e = R_{e,s}$. Notably, the linearization of penalty term (A.17) is defined by terms

$$\nabla_z R^{F^{ij}}(\bar{z}) = C^{ij}(\bar{Z}^{ij}) M^{ij\top}, \quad (\text{A.19a})$$

$$R^{F^{ij}}(\bar{z}) = C^{ij}(\bar{Z}^{ij}) M^{ij\top} \bar{z} - c^{ij}(\bar{Z}^{ij}), \quad (\text{A.19b})$$

where $M^{ij} \in \mathbb{R}^{n_z \times (2n_{\tilde{x}} + N_u^{ij})}$ represent allocation matrices, similar to M_k^j in section 4.4.2, satisfying $v^{ij} = M^{ij\top} z$. Additionally, the GRAPH-DDTO warmstart defined by algorithm 4.2 must also reflect a slight indexing change, with lines 4-5 of the algorithm instead stated as

$$\begin{aligned} \bar{X}^j &\leftarrow \bar{X}_{s, k_d^j+1:N_s^j}^j, \\ \bar{U}^j &\leftarrow \bar{U}_{s, k_d^j+1:N_s^j}^j, \end{aligned}$$

which causes the branch sizes of the warmstart to instead be $N_s^j - k_d^j$.

BIBLIOGRAPHY

- [1] High-resolution lola topography for lunar south pole sites. <https://pgda.gsfc.nasa.gov/products/78>. Accessed: 2025-05-26.
- [2] B Acikmese, J Casoliva, JM Carson, and L Blackmore. G-fold: A real-time implementable fuel optimal large divert guidance algorithm for planetary pinpoint landing. *Concepts and Approaches for Mars Exploration*, 1679:4193, 2012.
- [3] Behcet Acikmese, M Aung, Jordi Casoliva, Swati Mohan, Andrew Johnson, Daniel Scharf, David Masten, Joel Scotkin, Aron Wolf, and Martin W Regehr. Flight testing of trajectories computed by g-fold: Fuel optimal large divert guidance algorithm for planetary landing. In *AAS/AIAA spaceflight mechanics meeting*, page 386, 2013.
- [4] Behçet Açıkmeşe and Lars Blackmore. Lossless convexification of a class of optimal control problems with non-convex control constraints. *Automatica*, 47(2):341–347, 2011.
- [5] Behçet Açıkmeşe, John M Carson, 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, 2013.
- [6] Behcet Acikmese and Scott R Ploen. Convex programming approach to powered descent guidance for mars landing. *Journal of Guidance, Control, and Dynamics*, 30(5):1353–1366, 2007.
- [7] 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. IEEE, July 2019.
- [8] John P Alsterda and J Christian Gerdes. Contingency model predictive control for linear time-varying systems. *arXiv preprint arXiv:2102.12045*, 2021.
- [9] B.T. Baumrucker and L.T. Biegler. MPEC strategies for optimization of a class of hybrid dynamic systems. *Journal of Process Control*, 19(8):1248–1256, September 2009.
- [10] Sudeep Bhatia and Timothy L Mullett. The dynamics of deferred decision. *Cogn. Psychol.*, 86:112–151, 2016.
- [11] Lars Blackmore, Behçet Açıkmeşe, and John M Carson III. Lossless convexification of control constraints for a class of nonlinear optimal control problems. *Systems & Control Letters*, 61(8):863–870, 2012.

- [12] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
- [13] James Bradbury, Roy Frostig, Peter Hawkins, et al. JAX: composable transformations of Python+NumPy programs, 2018.
- [14] 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 vol. 1, 1999.
- [15] Samuel C Buckner and Behcet Acikmese. Graph-scvx: Graph successive convexification for contingency-aware planning. Manuscript in preparation, 2026.
- [16] Samuel C Buckner, John M Carson, Breanna J Johnson, and Behcet Acikmese. Active continuous-time simultaneous localization & mapping for powered descent guidance maneuvers. In *AIAA SciTech 2026 Forum*, 2026. AIAA 2026-0969.
- [17] Samuel C Buckner and Purnanand Elango. Gpu-accelerated continuous-time successive convexification for contact-implicit legged locomotion. In *2026 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2026.
- [18] Samuel C Buckner, Purnanand Elango, and Behcet Acikmese. Contact-implicit successive convexification for multibody systems. Manuscript in preparation, 2026.
- [19] Samuel C Buckner, Joshua Shaffer, John M Carson, Breanna J Johnson, Ronald R Sostaric, and Behcet Acikmese. Constrained visibility guidance for 6-dof powered descent maneuvers with terrain scanning using sequential convex programming. In *AIAA SciTech 2024 Forum*, 2024. AIAA 2024-1759.
- [20] Richard H Byrd, Nicholas IM Gould, Jorge Nocedal, and Richard A Waltz. An algorithm for nonlinear optimization using linear programming and equality constrained subproblems. *Mathematical Programming*, 100(1):27–48, 2003.
- [21] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332, 2017.
- [22] Grace E Calkins, David C Woffinden, and Zachary R Putnam. Robust trajectory optimization for guided powered descent and landing. In *AAS/AIAA Astrodynamics Specialists Conference*, number AAS 22-660, 2022.
- [23] Jan Carius, René Ranftl, Vladlen Koltun, and Marco Hutter. Trajectory optimization with implicit hard contacts. *IEEE Robotics and Automation Letters*, 3(4):3316–3323, 2018.
- [24] John M Carson, Michelle M Munk, Ronald R Sostaric, Jay N Estes, Farzin Amzajerjian, James B Blair, David K Rutishauser, Carolina I Restrepo, Alicia M Dwyer-Cianciolo, George Chen, et al. The splice project: Continuing nasa development of

- gn&c technologies for safe and precise landing. In *AIAA SciTech 2019 forum*, page 0660, 2019.
- [25] Runqi Chai, Al Savvaris, Antonios Tsourdos, and Senchun Chai. Overview of trajectory optimization techniques. In *Design of Trajectory Optimization Approach for Space Maneuver Vehicle Skip Entry Problems*, pages 7–25. Springer, 2019.
 - [26] Govind M Chari and Behçet Açikmeşe. QOCO: A quadratic objective conic optimizer with custom solver generation. *arXiv preprint arXiv:2503.12658*, 2025.
 - [27] Chris Chatfield. Exploratory data analysis. *European journal of operational research*, 23(1):5–13, 1986.
 - [28] Kuo Chen, Xinyan Huang, Xunjie Chen, and Jingang Yi. Contingency model predictive control for bipedal locomotion on moving surfaces with a linear inverted pendulum model. *arXiv preprint arXiv:2404.12455*, 2024.
 - [29] 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.
 - [30] Francis Clarke. *Functional analysis, calculus of variations and optimal control*, volume 264. Springer, 2013.
 - [31] R Dhar. The effect of decision strategy on deciding to defer choice. *J. Behav. Decis. Mak.*, 9(4):265–281, December 1996.
 - [32] Javier A Doll, Samuel T Wagner, Matthew Fritz, Jiann-Woei Jang, Jeanette M Harper, Kyle W Smith, Behcet Acikmese, Samuel M Pedrotty, and Gavin F Mendeck. Performance analysis of a dual quaternion guidance algorithm applicable during lunar approach with a hazard avoidance maneuver. In *AIAA SciTech 2024 Forum*, page 1757, 2024.
 - [33] Alexander Domahidi, Eric Chu, and Stephen Boyd. Ecos: An socp solver for embedded systems. In *2013 European control conference (ECC)*, pages 3071–3076. IEEE, 2013.
 - [34] Dmitriy Drusvyatskiy and Adrian S Lewis. Error bounds, quadratic growth, and linear convergence of proximal methods. *Mathematics of Operations Research*, 43(3):919–948, 2018.
 - [35] Iain Dunning, Joey Huchette, and Miles Lubin. Jump: A modeling language for mathematical optimization. *SIAM review*, 59(2):295–320, 2017.
 - [36] Kazuya Echigo, Oliver Sheridan, Samuel Buckner, and Behçet Açikmeşe. Dispersion sensitive optimal control: A conditional value-at-risk-based tail flattening approach via sequential convex programming. *IEEE Transactions on Control Systems Technology*, 2024.
 - [37] Purnanand Elango, Dayou Luo, Abhinav G. Kamath, Samet Uzun, Taewan Kim, and Behçet Açikmeşe. Continuous-time successive convexification for constrained trajectory optimization. *Automatica*, 180:112464, October 2025.

- [38] Purnanand Elango, Selahattin Burak Sarsilmaz, and Behcet Acikmese. Deferring decision in multi-target trajectory optimization. In *AIAA SciTech 2022 Forum*, page 1583, 2022.
- [39] Purnanand Elango, Selahattin Burak Sarsilmaz, and Behcet Acikmese. Deferred-decision trajectory optimization. *arXiv preprint arXiv:2502.06623*, 2025.
- [40] Ryan Eustice, Matthew Walter, and John Leonard. Sparse extended information filters: Insights into sparsification. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3281–3288. IEEE, 2005.
- [41] Davide Falanga, Alessio Zanchettin, Alessandro Simovic, Jeffrey Delmerico, and Davide Scaramuzza. Vision-based autonomous quadrotor landing on a moving platform. In *2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, pages 200–207. IEEE, 2017.
- [42] Matthew Fritz, Javier Doll, Kari C Ward, Gavin Mendeck, Ronald R Sostaric, Sam Pedrotty, Chris Kuhl, Behcet Acikmese, Stefan R Bieniawski, Lloyd Strohl, et al. Post-flight performance analysis of navigation and advanced guidance algorithms on a terrestrial suborbital rocket flight. In *AIAA SCITECH 2022 Forum*, page 0765, 2022.
- [43] David K Geller. Linear covariance techniques for orbital rendezvous analysis and autonomous onboard mission planning. *Journal of Guidance, Control, and Dynamics*, 29(6):1404–1414, 2006.
- [44] David K Geller and Daniel P Christensen. Linear covariance analysis for powered lunar descent and landing. *Journal of Spacecraft and Rockets*, 46(6):1231–1248, 2009.
- [45] Rafal Goebel, Ricardo G Sanfelice, and Andrew R Teel. Hybrid dynamical systems. *IEEE control systems magazine*, 29(2):28–93, 2009.
- [46] Paul J Goulart and Yuwen Chen. Clarabel: An interior-point solver for conic programs with quadratic objectives. *arXiv preprint arXiv:2405.12762*, 2024.
- [47] Kaiyang Guo, Pan Tang, Hui Wang, Defu Lin, and Xiaoxi Cui. Autonomous landing of a quadrotor on a moving platform via model predictive control. *Aerospace*, 9(1):34, 2022.
- [48] Jonas Hall, Armin Nurkanović, Florian Messerer, and Moritz Diehl. A sequential convex programming approach to solving quadratic programs and optimal control problems with linear complementarity constraints. *IEEE Control Systems Letters*, 6:536–541, 2021.
- [49] S-P Han and Olvi L Mangasarian. Exact penalty functions in nonlinear programming. *Mathematical programming*, 17(1):251–269, 1979.
- [50] Jason Hardy and Mark Campbell. Contingency planning over probabilistic obstacle predictions for autonomous road vehicles. *IEEE Trans. Rob.*, 29(4):913–929, August 2013.

- [51] Gage W Harris, Ping He, and Ossama O Abdelkhalik. Control co-design optimization of spacecraft trajectory and system for interplanetary missions. *Journal of Spacecraft and Rockets*, 61(4):939–952, 2024.
- [52] Christopher R Hayner, Samuel C Buckner, Daniel Broyles, Evelyn Madewell, Karen Leung, and Behçet Açikmeşe. Halo: Hazard-aware landing optimization for autonomous systems. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3261–3267. IEEE, 2023.
- [53] Christopher R Hayner, John M Carson Iii, Behçet Açikmeşe, and Karen Leung. Continuous-time line-of-sight constrained trajectory planning for 6-degree of freedom systems. *IEEE Robotics and Automation Letters*, 2025.
- [54] Christopher R Hayner, Natalia Pavlasek, Karen Leung, Behcet Acikmese, and John M Carson. Information-aware powered descent guidance for entry, descent and landing. In *AIAA SCITECH 2025 Forum*, page 1896, 2025.
- [55] Taylor Howell, Nimrod Gileadi, Saran Tunyasuvunakool, Kevin Zakka, Tom Erez, and Yuval Tassa. Predictive sampling: Real-time behaviour synthesis with MuJoCo. *arXiv preprint arXiv:2212.00541*, 2022.
- [56] Rolf Johansson, Michel Verhaegen, and Chun Tung Chou. Stochastic theory of continuous-time state-space identification. *IEEE Transactions on Signal Processing*, 47(1):41–51, 1999.
- [57] Wayne Johnson. *Helicopter theory*. Courier Corporation, 2012.
- [58] Abhinav G Kamath, Purnanand Elango, Taewan Kim, Skye Mceowen, Yue Yu, John M Carson, Mehran Mesbahi, and Behcet Acikmese. Customized real-time first-order methods for onboard dual quaternion-based 6-dof powered-descent guidance. In *AIAA SciTech 2023 Forum*, page 2003, 2023.
- [59] Abhinav G Kamath, Purnanand Elango, Yue Yu, Skye Mceowen, Govind M Chari, John M Carson III, and Behçet Açikmeşe. Real-time sequential conic optimization for multi-phase rocket landing guidance. *IFAC-PapersOnLine*, 56(2):3118–3125, 2023.
- [60] Abhinav G Kamath, Abraham P Vinod, Purnanand Elango, Stefano Di Cairano, and Avishai Weiss. Robust optimal control for autonomous precision landing via set-based dynamic programming. In *AIAA SCITECH 2026 Forum*, page 0970, 2026.
- [61] Thomas R Kane and David A Levinson. *Dynamics, theory and applications*. McGraw Hill, 1985.
- [62] Saif R. Kazi, Kexin Wang, and Lorenz Biegler. On optimal control of hybrid dynamical systems using complementarity constraints. *Journal of Process Control*, 153:103492, September 2025.
- [63] Soohwan Kim and Minkyoun Kim. Rotation representations and their conversions. *Ieee Access*, 11:6682–6699, 2023.

- [64] 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.
- [65] Yoonsoo Kim, Mehran Mesbahi, Gurkirpal Singh, and Fred Y Hadaegh. On the convex parameterization of constrained spacecraft reorientation. *IEEE Transactions on Aerospace and Electronic Systems*, 46(3):1097–1109, 2010.
- [66] Shumon Koga, Arash Asgharivaskasi, and Nikolay Atanasov. Active exploration and mapping via iterative covariance regulation over continuous se (3) trajectories. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2735–2741. IEEE, 2021.
- [67] Shumon Koga, Arash Asgharivaskasi, and Nikolay Atanasov. Active slam over continuous trajectory and control: A covariance-feedback approach. In *2022 American Control Conference (ACC)*, pages 5062–5068. IEEE, 2022.
- [68] Vince Kurtz, Alejandro Castro, Aykut Özgün Önel, and Hai Lin. Inverse dynamics trajectory optimization for contact-implicit model predictive control. *The International Journal of Robotics Research*, 45(1):23–40, 2026.
- [69] Nikolay Kyurkchiev and Svetoslav Markov. Sigmoid functions: some approximation and modelling aspects. *LAP LAMBERT Academic Publishing, Saarbrücken*, 4:34, 2015.
- [70] Unsik Lee and Mehran Mesbahi. Spacecraft reorientation in presence of attitude constraints via logarithmic barrier potentials. In *Proceedings of the 2011 American Control Conference*, pages 450–455. IEEE, 2011.
- [71] Adrian S Lewis and Stephen J Wright. A proximal method for composite minimization. *Mathematical Programming*, 158:501–546, 2016.
- [72] He Li and Patrick M Wensing. Cafe-mpc: A cascaded-fidelity model predictive control framework with tuning-free whole-body control. *IEEE Transactions on Robotics*, 41:837–856, 2024.
- [73] Jiayun Li, Dejian Gong, and Georgia Chalvatzaki. Impact: An implicit active-set augmented lagrangian for fast contact-implicit trajectory optimization. *arXiv preprint arXiv:2605.09127*, 2026.
- [74] Tong Li, Lu Zhang, Sikang Liu, and Shaojie Shen. MARC: Multipolicy and Risk-Aware contingency planning for autonomous driving. *IEEE Robotics and Automation Letters*, 8(10):6587–6594, October 2023.
- [75] Yulin Li, Haoyu Han, Shucheng Kang, Jun Ma, and Heng Yang. On the surprising robustness of sequential convex optimization for contact-implicit motion planning. *arXiv preprint arXiv:2502.01055*, 2025.

- [76] Xuan Lin, Feng Xu, Alexander Schperberg, and Dennis Hong. Learning near-global-optimal strategies for hybrid non-convex model predictive control of single rigid body locomotion. *arXiv preprint arXiv:2207.07846*, 2022.
- [77] Brett T Lopez, Jean-Jacques E Slotine, and Jonathan P How. Dynamic tube mpc for nonlinear systems. In *2019 American Control Conference (ACC)*, pages 1655–1662. IEEE, 2019.
- [78] Kevin M. Lynch and Frank C. Park. *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, 2017.
- [79] Steven Macenski, Tully Foote, Brian Gerkey, Chris Lalancette, and William Woodall. Robot operating system 2: Design, architecture, and uses in the wild. *Science robotics*, 7(66):eabm6074, 2022.
- [80] Danylo Malyuta and Behçet Açıkmeşe. Fast homotopy for spacecraft rendezvous trajectory optimization with discrete logic. *Journal of Guidance, Control, and Dynamics*, 46(7):1262–1279, 2023.
- [81] Danylo Malyuta, Taylor Reynolds, Michael Szmuk, Behcet Acikmese, and Mehran Mesbahi. Fast trajectory optimization via successive convexification for spacecraft rendezvous with integer constraints. In *AIAA Scitech 2020 Forum*, page 0616, 2020.
- [82] Danylo Malyuta, Taylor Reynolds, Michael Szmuk, Mehran Mesbahi, Behcet Acikmese, and John M Carson. Discretization performance and accuracy analysis for the rocket powered descent guidance problem. In *AIAA Scitech 2019 Forum*, page 0925, 2019.
- [83] 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.
- [84] Yuanqi Mao, Michael Szmuk, and Behçet Açıkmeşe. Successive convexification of non-convex optimal control problems and its convergence properties. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 3636–3641. IEEE, 2016.
- [85] Yuanqi Mao, Michael Szmuk, Xiangru Xu, and Behçet Açıkmeşe. Successive convexification: A superlinearly convergent algorithm for non-convex optimal control problems. *arXiv preprint arXiv:1804.06539*, 2018.
- [86] Peter S Maybeck. *Stochastic models, estimation, and control*, volume 3. Academic press, 1982.
- [87] Skye Mceowen, Daniel J Calderone, Aman Tiwary, Jason SK Zhou, Taewan Kim, Purnanand Elango, and Behçet Açıkmeşe. Autotuned primal–dual successive convexification for reentry guidance. *Journal of Guidance, Control, and Dynamics*, 48(9):2020–2040, 2025.

- [88] Skye Mceowen, Carlos M Morales, Breanna J Johnson, Daniel J Calderone, Edgerton M Cook, John M Carson, and Behcet Acikmese. Auto-tuned successive convexification for entry guidance with continuous-time constraint satisfaction. In *AIAA SCITECH 2026 Forum*, page 0971, 2026.
- [89] Igor Mordatch, Emanuel Todorov, and Zoran Popović. Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics*, 31(4):1–8, July 2012.
- [90] Beipeng Mu, Matthew Giamou, Liam Paull, Ali-akbar Agha-mohammadi, John Leonard, and Jonathan How. Information-based active slam via topological feature graphs. In *2016 IEEE 55th Conference on decision and control (Cdc)*, pages 5583–5590. IEEE, 2016.
- [91] Yashwanth Kumar Nakka and Soon-Jo Chung. Trajectory optimization of chance-constrained nonlinear stochastic systems for motion planning under uncertainty. *IEEE Transactions on Robotics*, 39(1):203–222, 2022.
- [92] Armin Nurkanović and Moritz Diehl. Nosnoc: A software package for numerical optimal control of nonsmooth systems. *IEEE Control Systems Letters*, 6:3110–3115, 2022.
- [93] Armin Nurkanović, Andrea Zanelli, Sebastian Albrecht, and Moritz Diehl. The advanced step real time iteration for nmpc. In *2019 IEEE 58th Conference on decision and control (CDC)*, pages 5298–5305. IEEE, 2019.
- [94] Armin Nurkanović, Jonathan Frey, Anton Pozharskiy, and Moritz Diehl. FESD-J: Finite Elements with Switch Detection for numerical optimal control of rigid bodies with impacts and Coulomb friction. *Nonlinear Analysis: Hybrid Systems*, 52:101460, May 2024.
- [95] Kazuhide Okamoto and Panagiotis Tsiotras. Optimal stochastic vehicle path planning using covariance steering. *IEEE Robotics and Automation Letters*, 4(3):2276–2281, 2019.
- [96] Helen Oleynikova, Zachary Taylor, Roland Siegwart, and Juan Nieto. Safe local exploration for replanning in cluttered unknown environments for microaerial vehicles. *IEEE Robotics and Automation Letters*, 3(3):1474–1481, 2018.
- [97] Aykut Özgün Önoğlu, Radu Corcodel, Philip Long, and Taşkın Padır. Tuning-free contact-implicit trajectory optimization. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1183–1189. IEEE, 2020.
- [98] Aykut Özgün Önoğlu, Philip Long, and Taşkın Padır. Contact-implicit trajectory optimization based on a variable smooth contact model and successive convexification. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2447–2453. IEEE, 2019.
- [99] David E. Orin, Ambarish Goswami, and Sung-Hee Lee. Centroidal dynamics of a humanoid robot. *Autonomous Robots*, 35(2–3):161–176, June 2013.

- [100] Fernando Palacios-Gomez, L Lasdon, and Michael Engquist. Nonlinear optimization by successive linear programming. *Management science*, 28(10):1106–1120, 1982.
- [101] Amir Patel, Stacey Leigh Shield, Saif Kazi, Aaron M Johnson, and Lorenz T Biegler. Contact-implicit trajectory optimization using orthogonal collocation. *IEEE Robotics and Automation Letters*, 4(2):2242–2249, 2019.
- [102] Benedetto Piccoli. Hybrid systems and optimal control. In *Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No. 98CH36171)*, volume 1, pages 13–18. IEEE, 1998.
- [103] Cristina Pinneri, Shambhuraj Sawant, Sebastian Blaes, Jan Achterhold, Joerg Stueckler, Michal Rolinek, and Georg Martius. Sample-efficient cross-entropy method for real-time planning. In *Conference on Robot Learning*, pages 1049–1065. PMLR, 2021.
- [104] Julio A Placed, Jared Strader, Henry Carrillo, Nikolay Atanasov, Vadim Indelman, Luca Carlone, and José A Castellanos. A survey on active simultaneous localization and mapping: State of the art and new frontiers. *IEEE Transactions on Robotics*, 39(3):1686–1705, 2023.
- [105] Lev Semenovich Pontryagin. *Mathematical theory of optimal processes*. Routledge, 2018.
- [106] Michael Posa, Cecilia Cantu, and Russ Tedrake. A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33(1):69–81, October 2013.
- [107] Ciro Potena, Daniele Nardi, and Alberto Pretto. Effective target aware visual navigation for uavs. In *2017 European Conference on Mobile Robots (ECMR)*, pages 1–7. IEEE, 2017.
- [108] Michael JD Powell and YJMP Yuan. A recursive quadratic programming algorithm that uses differentiable exact penalty functions. *Mathematical programming*, 35(3):265–278, 1986.
- [109] 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.
- [110] Rien Quirynen. *Numerical simulation methods for embedded optimization*. PhD thesis, KU Leuven, 2017.
- [111] Taylor Reynolds, Michael Szmuk, Danylo Malycha, Mehran Mesbahi, Behçet Açikmeşe, and John M Carson. A state-triggered line of sight constraint for 6-dof powered descent guidance problems. In *AIAA scitech 2019 forum*, page 0924, 2019.
- [112] Taylor P Reynolds, Michael Szmuk, Danylo Malycha, Mehran Mesbahi, Behçet Açikmeşe, 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, 2020.

- [113] Jack Ridderhof, Kazuhide Okamoto, and Panagiotis Tsiotras. Nonlinear uncertainty control with iterative covariance steering. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 3484–3490. IEEE, 2019.
- [114] Peter A Ruymgaart and Tsu T Soong. *Mathematics of Kalman-Bucy Filtering*, volume 14. Springer Science & Business Media, 2013.
- [115] Luca Scardovi, Marco Baglietto, and Thomas Parisini. Active state estimation for nonlinear systems: A neural approximation approach. *IEEE Transactions on Neural Networks*, 18(4):1172–1184, 2007.
- [116] Klaus Schittkowski and C Zillober. Sequential convex programming methods. In *Stochastic programming: numerical techniques and engineering applications*, pages 123–141. Springer, 1995.
- [117] P O M Scokaert and D Q Mayne. Min-max feedback model predictive control for constrained linear systems. *IEEE Trans. Automat. Contr.*, 43(8):1136–1142, August 1998.
- [118] Joshua Shaffer, Chris Owens, Theresa Klein, Andrew D Horchler, Samuel C Buckner, Breanna J Johnson, John M Carson, and Behcet Acikmese. Implementation and testing of convex optimization-based guidance for hazard detection and avoidance on a lunar lander. In *AIAA SciTech 2024 forum*, page 1584, 2024.
- [119] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and service robotics: Results of the 11th international conference*, pages 621–635. Springer, 2017.
- [120] Robert Sim and Nicholas Roy. Active exploration planning for slam using extended information filters. In *Proc. 20th Conf. Uncertainty in AI*, volume 2, page 1, 2004.
- [121] Pantelis Sopasakis, Domagoj Herceg, Alberto Bemporad, and Panagiotis Patrinos. Risk-averse model predictive control. *Automatica*, 100:281–288, 2019.
- [122] Cyrill Stachniss, John J Leonard, and Sebastian Thrun. Simultaneous localization and mapping. *Springer handbook of robotics*, pages 1153–1176, 2016.
- [123] David E. Stewart. *Dynamics with Inequalities: Impacts and Hard Constraints*, volume 123 of *Other Titles in Applied Mathematics*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2011.
- [124] Michael Szmuk and Behcet Acikmese. Successive convexification for 6-dof mars rocket powered landing with free-final-time. In *2018 AIAA Guidance, Navigation, and Control Conference*, page 0617, 2018.
- [125] Michael Szmuk, Behcet Acikmese, and Andrew W Berning. Successive convexification for fuel-optimal powered landing with aerodynamic drag and non-convex constraints. In *AIAA Guidance, Navigation, and Control Conference*, page 0378, 2016.
- [126] Michael Szmuk, Utku Eren, and Behcet Acikmese. Successive convexification for mars

- 6-dof powered descent landing guidance. In *AIAA Guidance, Navigation, and Control Conference*, page 1500, 2017.
- [127] Michael Szmuk, Danylo Malychuk, Taylor P Reynolds, Margaret Skye Mceowen, and Behçet Açıkmeşe. Real-time quad-rotor path planning using convex optimization and compound state-triggered constraints. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7666–7673. IEEE, 2019.
 - [128] Michael Szmuk, Carlo Alberto Pascucci, Daniel Dueri, and Behçet Açıkmeşe. Convexification and real-time on-board optimization for agile quad-rotor maneuvering and obstacle avoidance. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4862–4868. IEEE, 2017.
 - [129] 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, 2020.
 - [130] Koki Tanaka and Genya Ishigami. Modeling of lidar measurement uncertainty for rover path planning. In *Proc. of The International Symposium on Artificial Intelligence, Robotics and Automation in Space*, 2016.
 - [131] Huseyin Emre Tekaslan and Ella M Atkins. Gradient-guided search for autonomous contingency landing planning. *Drones*, 9(9):642, 2025.
 - [132] Sebastian Thrun. Probabilistic robotics. *Communications of the ACM*, 45(3):52–57, 2002.
 - [133] Lorenzo Ticozzi and Panagiotis Tsiotras. Factor graph-based active slam for spacecraft proximity operations. *arXiv preprint arXiv:2501.10950*, 2025.
 - [134] Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, et al. Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*, 2024.
 - [135] Nikolas Trawny, Joel Benito, Brent E Tweddle, Charles F Bergh, Garen Khanoyan, Geoffrey Vaughan, Jason Zheng, Carlos Villalpando, Yang Cheng, Daniel P Scharf, et al. Flight testing of terrain-relative navigation and large-divert guidance on a vtol rocket. In *AIAA SPACE 2015 conference and exposition*, page 4418, 2015.
 - [136] Amos Tversky and Eldar Shafir. Choice under conflict - the dynamics of deferred decision. *Psychol. Sci.*, 1992.
 - [137] Samet Uzun, Behçet Acikmese, and John M Carson. Sequential convex programming for 6-dof powered descent guidance with continuous-time compound state-triggered constraints. In *AIAA SciTech 2025 Forum*, page 1895, 2025.
 - [138] Samet Uzun, Behçet Açıkmeşe, and Stefano Di Cairano. Motion planning for information acquisition via continuous-time successive convexification. *IEEE Control Systems Letters*, 9:751–756, 2025.

- [139] Samet Uzun, Purnanand Elango, Pierre-Loic Garoche, and Behcet Acikmese. Optimization with temporal and logical specifications via generalized mean-based smooth robustness measures. *arXiv preprint arXiv:2405.10996*, 2024.
- [140] Matthew R Walter, Ryan M Eustice, and John J Leonard. Exactly sparse extended information filters for feature-based slam. *The International Journal of Robotics Research*, 26(4):335–359, 2007.
- [141] Ke Wang, Yong Wang, Bingjun Liu, and Junlan Chen. Quantification of uncertainty and its applications to complex domain for autonomous vehicles perception system. *IEEE Transactions on Instrumentation and Measurement*, 72:1–17, 2023.
- [142] 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.
- [143] Patrick M. Wensing, Michael Posa, Yue Hu, Adrien Escande, Nicolas Mansard, and Andrea Del Prete. Optimization-Based Control for Dynamic Legged Robots. *IEEE Transactions on Robotics*, 40:43–63, 2024.
- [144] James Williams, David Woffinden, and Zachary R Putnam. Mars entry guidance and navigation analysis using linear covariance techniques for the safe and precise landing–integrated capabilities evolution (splice) project. In *AIAA Scitech 2020 Forum*, page 0597, 2020.
- [145] Alexander W. Winkler, C. Dario Bellicoso, Marco Hutter, and Jonas Buchli. Gait and Trajectory Optimization for Legged Systems Through Phase-Based End-Effector Parameterization. *IEEE Robotics and Automation Letters*, 3(3):1560–1567, July 2018. Conference Name: IEEE Robotics and Automation Letters.
- [146] David Woffinden, Shane Robinson, James Williams, and Zachary R Putnam. Linear covariance analysis techniques to generate navigation and sensor requirements for the safe and precise landing integrated capabilities evolution (splice) project. In *AIAA SciTech 2019 forum*, page 0662, 2019.
- [147] 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.
- [148] Yue Yu, Purnanand Elango, Ufuk Topcu, and Behçet Açıkmeşe. Proportional–integral projected gradient method for conic optimization. *Automatica*, 142:110359, 2022.
- [149] Ziyi Zhou and Ye Zhao. Accelerated admm based trajectory optimization for legged locomotion with coupled rigid body dynamics. In *2020 American control conference (ACC)*, pages 5082–5089. IEEE, 2020.
- [150] 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)*, Aviation Technology, Integration, and Operations (ATIO) Conferences. American Institute of Aeronautics and Astronautics, 2009.