Deterministic and Chance-Constrained Real-Time Motion Planning Using Reachability Analysis

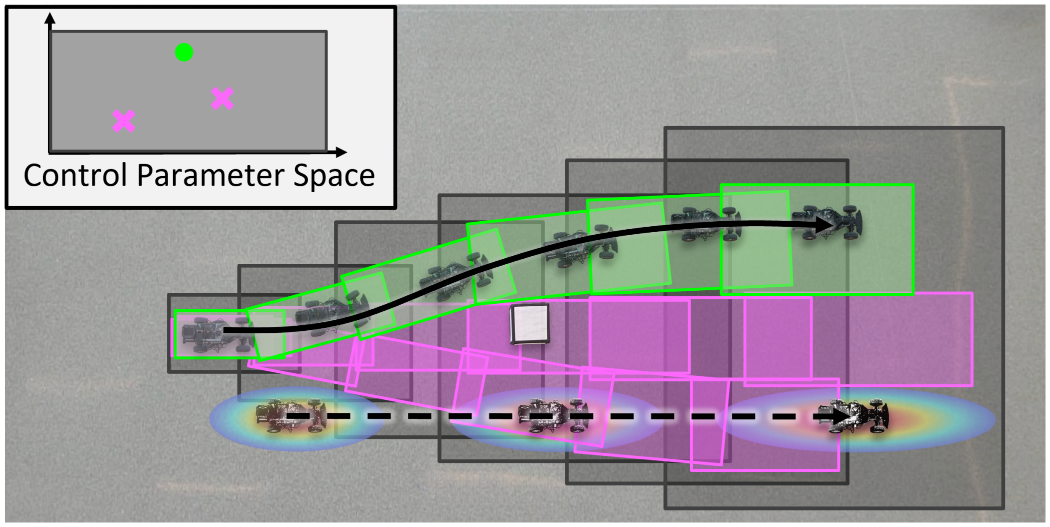
by

Jinsun Liu

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Robotics)
in the University of Michigan
2023

Doctoral Committee:

Associate Professor Ram Vasudevan, Chair
Professor Jessy Grizzle
Associate Professor Necmiye Ozay
Assistant Professor Katherine Skinner


Control Parameter Space

Jinsun Liu

jinsunl@umich.edu

ORCID iD: 0000-0003-2067-4144

© Jinsun Liu 2023

ACKNOWLEDGMENTS

First and foremost, I am deeply grateful to my advisor, Ram Vasudevan, for his guidance and support throughout my PhD journey. When I was a master's student, I had no plans of pursuing a PhD, but Ram's wisdom, enthusiasm, endless ideas and clear explanations convinced me that doing research was fun, so I made the best decision in my time at Michigan and said yes immediately when Ram asked me if I wanted to pursue for a PhD on a Monday afternoon in October 2016. His mentorship has significantly shaped my mind and improved my abilities in all aspects. Together, we have tackled many challenging problems, and I would not have made it without his continued support and encouragement, especially during the tough days. I could not have asked for a better mentor - accessible, humorous, and dedicated. Ram, I keep saying that you are my life changer, and I really mean it.

Next, I would like to express my gratitude to my fantastic committee for their invaluable support and guidance. Prof. Jessy Grizzle, it was your question on approximation error using a simplified model during my qualification exam that sparked the inception of this thesis. Since then I have been thinking hard to answer the question convincingly, and I hope this thesis has made it happen. In fact, it was you who invited Ram to give a guest lecture in your hybrid control class and let me met my future advisor, so I am very luck to have you as a witness of my whole PhD journey. Prof. Necmiye Ozay, I am sincerely indebted to you for the training and instruction you provided during my time in your group. You were the first one who taught me optimization, hybrid systems and zonotopes, and all these mathematical tools play key roles in this thesis. And Prof. Katherine Skinner, I would like to thank you for your valuable feedback and detailed suggestions on my proposal and thesis, which helped me to make the work more complete.

Third, I would like to express my appreciation to my collaborators, Prof. Andrew Lamperski, Challen Enninful Adu, Zhenyu Gan, Vishrut Kaushik, Lucas Lymburner, Joshua Mangelson, Shanker Mohan, Hyongju Park, Hansen Qin, Yifei Shao, Lena Trang and Pengcheng Zhao for their invaluable work. I had a great experience working with each of them and cannot stress enough how much I value their support. I want to extend a special thanks to Prof. Andrew Lamperski for providing me with my first research opportunity and sparking my interest in tackling challenging problems. I would also give special thanks to

Challen Enninful Adu and Lucas Lymburner. Challen, I cannot be more thankful for your support and respect during my hardest time for graduation. I look forward to see your future accomplishment in your PhD, and I guarantee that I will share all my knowledge with you. Lucas, my favourite 'grandson', this thesis literally could not be finished without you. You are an incredibly intelligent and talented individual, and I feel so fortunate to have you in my team.

Fourth, I would like to thank my other friends and colleagues in ROAHM Lab and FCAV: Cyrus Anderson, Zachary Brei, Fan Bu, Yizhou Chen, Shannon Danforth, Jorge Vilchis Dominguez, Sidhartha Dey, Parker Ewen, Xun Fu, Patrick Holmes, Seth Isaacson, Miracle Kanu-Asiegbu, Shreyas Kousik, Yong Seok Kwon, Hannah Larson, Adam Li, Zehui Lu, Aohan Mei, S.R. Mamikandasriram, Jon Michaux, Matthew Porter, Daphna Raz, Elena Shrestha, Nils Smit-Anseeuw, Sean Vaskov, Danial Yan, Ming-yuan Yu, Bohao Zhang, Junming Zhang, Yiting Zhang, and all others whose names I may have missed. I am grateful for the stimulating work environment and the knowledge that I have gained from working with all of you. I cherished the fun times we shared and I hope that our friendship will continue to last. I want to give special thanks to Fan Bu, Junming Zhang, and Pengcheng Zhao for all the time we spent together in and out of the lab. Your presence has brought infinitely joy and fulfillment to my PhD journey, and I value our bond of brotherhood deeply. I also want to extend my appreciation to Prof. Talia Moore for her thoughtful discussion on biology and gracious support throughout my PhD.

Fifth, I would like to express my gratitude to many others. Thanks to Jiaxin Li and Aibo Gong for being close friends and roommates. I am very lucky to always have them as my backup and share my sorrow and joy. Thanks to Yukai Gong, Ray Zhang, Dan Zhao and Hao Luo for those interesting poker hours when I needed to escape from work. Thanks to my friends Yuxin Chen, Shihao Cheng, Zhe Du, Bruce Huang, Bowen Liu, Jiyuan Song, Jianyang Tang, Zhongda Wang, Yijun Wu, Liang Zhu, Xinping Zuo, who made these years enjoyable and delightful, and I wish them all the best in their future endeavors. And finally thanks Lu Gan for her consolation and accompany during my toughest days.

Lastly, I would like to say thank you to my parents Qian Liu and Chun Yang for their endless love and encouragement. You have always protected me and given me the largest freedom to chase my dream, even though sometimes I may not know what really I want. It has been years since we last met due to expired visa and the pandemic, and I really miss you. Hope what I have achieved can make you proud, and I love you both.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

ABSTRACT

Due to their limited sensing horizon, robots construct trajectories in a receding-horizon fashion, where a trajectory defined over a finite time horizon is computed while the robot tracks a previously planned trajectory. This trajectory is constructed by applying an optimization or sampling based method wherein collision checking is performed against obstacles at discrete time instances. Unfortunately this presents an undesirable tradeoff between real-time performance and safety. Reachability-based Trajectory Design (RTD) circumvents this tradeoff by leveraging offline pre-computation of parameterized over-approximations of the robot behavior using Forward Reachable Sets (FRS) thereby achieving safety and real-time operation.

To accomplish this objective, RTD represents the full order dynamics of the robot using a reduced-order model, which enables it to apply polynomial-based reachability analysis offline to the reduced-order model while conservatively bounding the difference between the two models. The result is a parameterized over-approximation of the full order robot behavior which can then be used for real-time trajectory design without sacrificing safety. However, RTD suffers from a pair of shortcomings: first, representing the full order dynamics using a reduced order model can introduce undue conservatism that makes it challenging to construct safe, dynamic motion; and second, RTD deals with probabilistic models of the surrounding environment by requiring that any possible behavior (even one with exceedingly small probability) is safe thereby introducing more conservatism. This thesis focuses on illustrating RTD on a walking robot model and addressing the two issues of RTD.

The first contribution of this thesis generalizes the RTD framework to bipedal robots for flat ground walking using the idea of templates and anchors, where 'templates' are simplified descriptions of the behavior of the full-order models as 'anchors'. Reachability analysis is performed on the template model under the assumption that the difference between the template and anchor can be bounded. Offline-computed polynomial reachable sets are then incorporated into a Model Predictive Control framework to select controllers that result in safe walking on the biped in an online fashion. This method is validated in simulation on a 5-link planar model and in the real-world on Cassie and Digit.

The second contribution of this thesis improves the RTD framework for autonomous vehicles by designing a novel robust, partial feedback linearization controller and performing zonotope-based reachability analysis on the closed-loop system. Because only the full-order model is involved in the computation of the reachable set, the outer approximation of the FRS using zonotopes is much tighter than the original polynomial-based RTD approaches. The proposed method is implemented on a full-size vehicle model in simulation, and on a 1/10th race car model in real experiments.

The third contribution of this thesis extends the RTD framework to deal with uncertainty. Consider the challenge of performing motion planning where one is given a Probability Density Function (PDF) description of an obstacle's state. The problem of bounding the risk of collision while performing motion planning can be cast as a chance-constrained program. To address this challenge, this portion of the thesis develops a numerical scheme that conservatively approximates the integral of a PDF over FRS and its gradient with respect to a control parameterization. Using this information, one can formulate a chance-constrained RTD approach to real-time risk-averse motion planning. The proposed method is evaluated on a full-size vehicle model with uncertain obstacle observations in simulation.

CHAPTER 1

Introduction

Real-time, safe motion planning is of great interest in robotics because any practical deployment of robots in the real-world must deal with real-time updates regarding the status of the environment. Because of the limited horizon of sensors deployed on robots, real-time motion planning algorithm usually operate in a receding-horizon fashion. Although roboticists have made tremendous strides in developing algorithms for receding horizon robot motion planning, ensuring that these algorithms can operate in real-time in uncertain environments while providing meaningful safety guarantees remains challenging because of the difficulties in robot behavior prediction, robot safety ensuring and real-time performance. And this is particularly challenging as perception algorithms are imperfect.

The remainder of this section gives an overview of state-of-arts methods to ensure safety during motion planning. Depending on how perception information is incorporated, motion planning algorithms can be divided into two categories: deterministic and stochastic. Deterministic motion planning algorithms either assume the knowledge of the environment is perfect or conservatively treat all possible area in which a crash might happen as unsafe region. On the other hand, stochastic methods model perception in a probabilistic fashion while trying to minimize the risk of a crash. We begin by describing several types of deterministic motion planning algorithms and then summarize how stochastic approaches balance robot safety with uncertain environments.

To generate safe motion plan in real-time while satisfying the robot dynamics, it is helpful to have accurate predictions of robot behavior over the time horizon in which planning is occurring. Because robot dynamics are typically nonlinear, closed-form solutions of robot trajectories are usually not computable; thus, planning methods utilize approximations to the solutions of robot's equations of motion. For instance, sampling-based planning methods compute approximate solutions to the robot dynamic model by applying different forms of discretization to try to construct collision-free paths [46, 60]. Sampling-based method generate approximate solutions to a robot's equations of motion by numerically integrating the robot dynamics or an approximation to them. To account for the inaccurate integration, obstacles are usually buffered by some small amount [25, 59]. However, selecting the correct amount to buffer by to account exactly for integration error can be challenging. As a result, most users of such sampling-based motion planning algorithms typically rely on heuristics to achieve real-time performance.

The idea of discretizing robot dynamics to generate an approximate solution is also utilized in Nonlinear Model Predictive Control (NMPC). In this instance, collision-free trajectories are numerically constructed through optimization algorithms that enforce safety at discrete time instances rather than continuous time intervals [27, 103]. Solving this optimization problem in real-time for high-dimensional robot models can be challenging as the required number of decision variables within the optimization problem to construct accurate representations of the solution to the equations of motion may be large. As a result, applying NMPC can require that the user make an undesirable trade between real-time performance and robot safety [125].

To account for inaccuracies that may arise due to following a trajectory generating by applying NMPC or a sampling-based motion planning algorithm, researchers have proposed the idea of applying robust feedback control algorithms. One particularly popular approach in recent years utilizes a Control Barrier Function (CBF) to modify a controller built to track a trajectory in real-time to ensure that the robot stays within a safe set. Because this can be done by solving a quadratic program, such methods have been successfully applied to ensure the safety of pre-computed trajectories for autonomous vehicles and walking robots [2, 4, 10, 74]. However, this pre-computation can make the CBF-based method scenario-specific; thus, the performance can be limited in real applications when environment information is collected at run-time [21, 77, 96].

To address some of these challenges, researchers have proposed to pre-compute over-approximations to a set of solutions of a robot's equations of motion, which is called a *Forward Reachable Set* (FRS) [34,56,92,120]. These FRS-es can be represented as level sets of polynomials or zonotopes and can be constructed offline by applying convex optimization techniques [67] or linearizing the robot dynamics while accounting for linearization error and computing the reachable set of a linear system under a disturbance [6]. During online planning, robot safety can then be enforced directly using the offline-computed FRS. Unfortunately constructing such FRS representations that are overapproximative but not overly conservative for high dimensional robotic systems can be numerically challenging.

The aforementioned methodologies are deterministic because they each assume perfect of the environment. However in real applications, it may be impossible to perfectly know the state of the system being controlled or the state of the surrounding environment. Because deterministic approaches deal with this challenge by bounding the estimation error and requiring the robot to avoid much more conservative unsafe area, such conservatism may render the online planning infeasible which can result in a large decline in the performance of the robot [91]. To balance the tradeoff between safety and maneuverability in uncertain environments, stochastic methods for risk-aware motion planning have been proposed [19, 86]. In this instance, instead of guaranteeing that an obstacle is avoided, these stochastic methods impose safety by restricting the probability of the risk that an unsafe event occurs. This is usually done by utilizing chance constraints [112, 121], Conditional Value-at-Risk (CVaR) constraints [38] or Entropic Value-at-Risk (EVaR) constraints [24]. In particular, a chance constraint directly limits the probability of risk within an online optimization problem, while constraints involving CVaR and its tightest upper bound, EVaR, regulate safety by limiting the expectation of the distance to the safe region by using a worst-case quantile representation of a distribution. Experiments have shown stochastic methods can regulate the risk of unsafe performance and maintain a balance between robot safety and maneuverability [105]. However evaluating chance constraints usually require strong assumptions on probability distributions that are difficult to meet in real application (i.e., assuming the cumulative probability distribution function is available and can be evaluated at run-time). Meanwhile, CVaR and EVaR constraints potentially make real-time planning challenging because they require applying sampling techniques or mixed integer optimization.

1.1 Reachability-based Trajectory Design

The Reachability-based Trajectory Design (RTD) framework [52] is a deterministic planning strategy for real-time safe motion planning that circumvents the undesirable tradeoff between real-time performance and safety. As a reachability-based planning method, RTD achieves safe real-time motion planning by first performing offline reachability analysis on a reduced-order model of the full-order robot dynamics with the difference between the two bounded conservatively. At run-time, the parameterized polynomial reachable sets are optimized over to ensure robot safety. Although RTD has been applied to autonomous vehicles, drones, and manipulators [43, 51, 106], there are two challenges that restrict the broad applicability of RTD.

1. Because a simplified model with bounded error is used to bound the potential behavior of the full-order robot model, the introduced conservatism can make it challenging to construct safe dynamic motions.

2. Because RTD is a deterministic method, it requires that any possible behavior of the

robot is safe in uncertain environment.

1.2 Contributions and Outlines

This thesis focuses on illustrating the generality of the RTD framework and addressing its aforementioned two issues. In particular, this thesis develops a real-time motion planning in the deterministic and chance-constrained setting. It begins by describing how classical RTD can be extended to a planar bipedal robot for flat ground walking without falling using the idea of template and anchor [30] in Chapter 2[1]. This is done by defining a set of outputs which are functions of the robot state and can be used to determine whether a gait can be safely tracked. These outputs enable the construction of a simplified (template) model whose behavior can be used to guarantee the safety of the full-order (anchor) robot model. Offline polynomial reachable sets are constructed for these outputs using the simplified model. Such reachable sets are then used online in a Model Predictive Control framework to achieve safe walking. The method is validated in simulation on a 5-link planar walking robot model named RABBIT and in the real-world on Cassie and Digit.

Chapter 3[2] presents a controller-oriented planning strategy named REFINE which extends the reachability-based approach developed in [52] using Feedback Linearization and Zonotopes on autonomous vehicles. REFINE drastically improves the tightness of reachable set computation by designing a robust partial feedback linearization controller that is able to deal with model inaccuracy. This controller makes offline reachability analysis over the closed loop dynamics of the full-order vehicle model possible using zonotopes. Because the full-order dynamic model is used for reachability analysis, conservatism introduced by model simplification as in classic RTD or as described in Chapter 2 is explicitly avoided. During online planning, control synthesis is performed in a receding horizon fashion by solving optimizations in which zonotope reachable sets are used to check against collisions. Experimental results on a full-size vehicle model and a 1/10th race car robot show that the proposed REFINE framework drastically improves the tightness of reachable sets and thereby enables far more maneuverability during online operation.

In Chapter 4, a probabilistic framework named Risk-RTD is provided to achieve real-time risk-aware motion planning. As the probabilistic extension of the traditional RTD method [52], Risk-RTD shares the same offline reachability analysis with REFINE and benefit from the tightness of zonotope reachable sets, but enforces robot safety by bounding the probability of any collision from above as chance constraints at all time during

[1]This chapter was previously published in ICRA 2020 [62].
[2]This chapter is under review at TRO [61].

online planning. To avoid making strong assumptions on probability estimation of the environment for chance constraint evaluation as in [105], a closed-form over-approximation on the probability of a collision is derived given the obstacle estimation satisfying arbitrary and twice-differentiable probabilistic distribution from an exponential family. Such over-approximation of collision probability is constructed in a way that ensures the existence of its analytical derivative which can improve the computation speed of online optimization. In addition, the proposed computations of both probability over-approximation and its derivative are parallelizable. As a result, Risk-RTD is able to perform real-time, chance-constrained motion planning. The tightness of proposed probability over-approximation is validated through numerical experiments, and simulation on a full-size vehicle model shows that the proposed Risk-RTD framework allows the ego vehicle to travel through crowded traffic more aggressively with a higher success rate when compared with deterministic motion planning methods.

In Chapter 5, the thesis contributions are summarized and potential future directions are provided.

CHAPTER 2

Real-Time Safe Control for Bipedal Robots

2.1 Introduction

[1] Legged robots are an ideal system to perform locomotion on unstructured terrains. Unfortunately designing controllers for legged systems to operate safely in such situations has proven challenging. To robustly traverse such environments, an ideal control synthesis technique for legged robotic systems should satisfy several requirements.

First, since uncertainties and disturbances may appear during operation, any algorithm for control synthesis should run in real-time. Second, since modeling contact can be challenging, any control synthesis technique should be able to accommodate model uncertainty. Third, since the most appropriate controller may be a function of the environment and given task, a control synthesis algorithm should optimize over as rich a family of control inputs at run-time as possible. Finally, since falling can be costly both in time and expense, a control synthesis technique should be able to guarantee the satisfactory behavior of any constructed controller. As illustrated in Fig. 2.1, this chapter presents an optimization-based algorithm to design gaits for legged robotic systems while satisfying each of these requirements.

We begin by summarizing related work with an emphasis on techniques that are able to make guarantees on the safety of the designed controller. For instance, the Zero-Moment Point approach [109] characterizes the stability of a legged robot with planar feet by defining the notion of the Zero-Moment Point and requiring that it remains within the robot's base of support. Though this requirement can be used to design a controller that can avoid falling at run-time, the gaits designed by the ZMP approach are static and energetically expensive [55] [114, Section 10.8].

In contrast, the Hybrid Zero Dynamics approach, which relies upon feedback linearization to drive the actuated degrees of freedom of a robot towards a lower dimensional mani-

[1]This chapter was previously published in 2020 IEEE International Conference on Robotics and Automation (ICRA) [62].

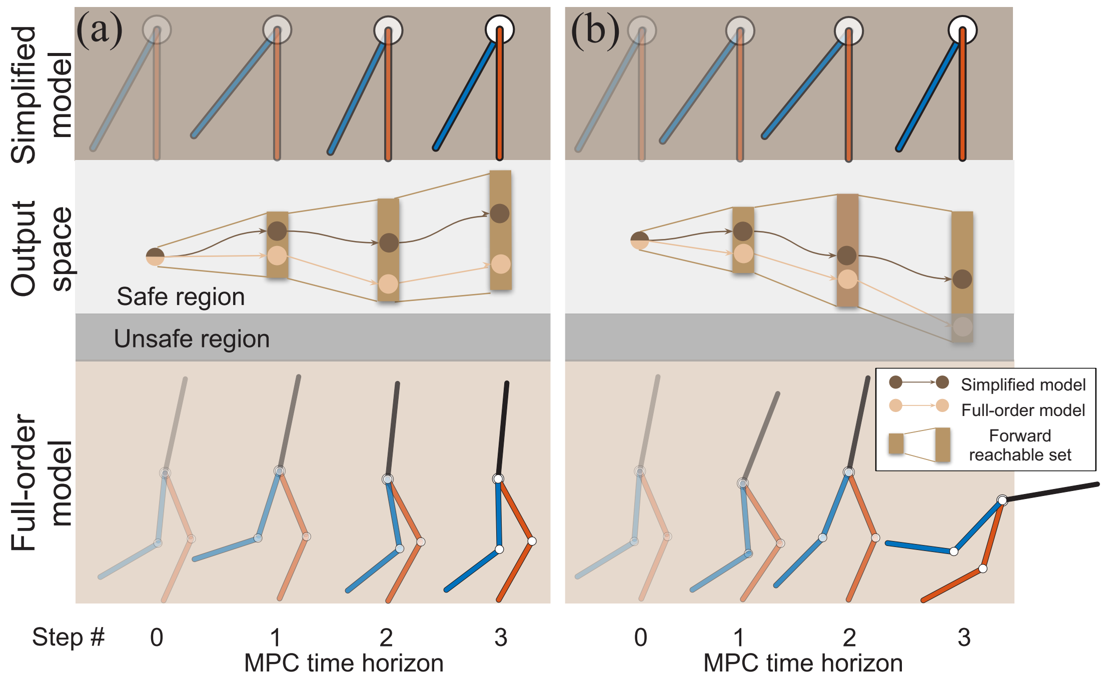


Figure 2.1: This chapter proposes a method to design gaits that are certified to be tracked by a full-order robot model (bottom row sub-figures) for N-steps without falling over. To construct this method, this chapter defines a set of outputs that are functions of the state of the robot and a chosen gait (middle row sub-figures). If the outputs associated with a particular gait satisfy a set of inequality constraints (depicted as the safe region drawn in light gray in the middle row sub-figures), then the gait is proven to be safely tracked by the legged system without falling. Due to the high-dimensionality of the robot's dynamics, forward propagating these outputs via the robot's dynamics for N-steps to design a gait that is certified to be tracked safely is intractable. To address this challenge, this chapter constructs a template model (top row sub-figures) whose outputs are sufficient to predict the behavior of the anchor's outputs. In particular, if all of the points in a bounded neighborhood of the forward reachable set of the outputs of the template model remain within the safe region, then the anchor is certified to behave safely. This chapter illustrates how this can be incorporated into a MPC framework to design safe gaits in real-time.

fold, is able to synthesize a controller which generates gaits that are more dynamic. Though this approach can generate safety preserving controllers for legged systems in real-time in the presence of model uncertainty [9, 45, 73, 75, 76], it is only able to prove that the gait associated with a synthesized control is locally stable. As a result, it is non-trivial to switch between multiple constructed controllers while preserving any safety guarantee. Recent work has extended the ability of the hybrid zero dynamic approach beyond a single neighborhood of any synthesized gait [11, 71, 94, 108]. These extensions either assume full-actuation [11] or ignore the behavior of the legged system off the lower dimensional manifold [71, 94, 108].

Rather than designing controllers for legged systems, other techniques have focused on characterizing the limits of safe performance by using Sums-of-Squares (SOS) optimization [80]. These approaches use semi-definite programming to identify the limits of safety in the state space of a system as well as associated controllers for hybrid systems [85, 93]. These safe sets can take the form of *reachable sets* [50, 93] or *invariant sets* in state space [84, 85, 115]. However, the representation of each of these sets in state space restricts the size of the problem that can be tackled by these approaches and as a result, these SOS-based approaches have been primarily applied to reduced models of walking robots: ranging from spring mass models [127], to inverted pendulum models [50, 101] and to inverted pendulum models with an offset torso mass [84]. Unfortunately the differences between these simple models and real robots makes it challenging to extend the safety guarantees to more realistic real-world models.

This chapter addresses the shortcomings of prior work by making the following four contributions. First, in Section 2.3.1, we describe a set of outputs that are functions of the state of the robot, which can be used to determine whether a particular gait can be safely tracked by a legged system without falling. In particular, if a particular gait's outputs satisfy a set of inequality constraints that we define, then we show that the gait can be safely tracked by the legged system without falling. To design gaits over N-steps that do not fall over, one could begin by forward propagating these outputs via the robot's dynamics for N-steps. Unfortunately performing this computation can be intractable due to the high-dimensionality of the robot's dynamics. To address this challenge, our second contribution, in Section 2.3.2, leverages the anchor and template framework to construct a simple model (template) whose outputs are sufficient to predict the behavior of the full model's (anchor's) outputs [30] under the assumption that the modeling error between the anchor and template can be bounded. Third, in Section 2.4.1, we develop an offline method to compute a gait parameterized forward reachable set that describes the evolution of the outputs of the simple model.

Similar to recently developed work on motion planning for ground and aerial vehicles [40, 51, 52, 66], one can then require that all possible outputs in the forward reachable set satisfy a family of conditions that we define to ensure that the robot does not fall over during the N-steps. Finally, in Section 2.4.2, we describe how to incorporate these conditions in a Model Predictive Control (MPC) framework that are sufficient to ensure N-step walking safely. Note, to simplify exposition, this chapter focuses on an example implementation on a 14-dimensional model of the robot RABBIT that is described in Section 2.2. The remainder of this chapter is organized as follows. Section 2.5 demonstrates the performance of the proposed approach on walking examples in simulation and on hardware. Section 2.6 concludes the chapter.

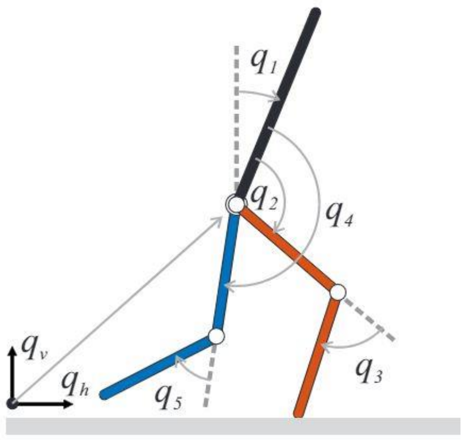


Figure 2.2: Illustration of the RABBIT model. Stance leg and swing leg are shown in red and blue respectively.

2.2 Preliminaries

This section introduces the notation, the dynamic model of the RABBIT robot, and a Simplified Biped Model (SBM) that are used throughout the remainder of this paper. The following notation is adopted in this manuscript. All sets are denoted using calligraphic capital letters. Let $\mathbb{R}$ denote the set of real numbers, and let $\mathbb{N}_+$ denote the collection of all non-negative integers. Give a set $\mathcal{A} \subset \mathbb{R}^n$ for some $n \in \mathbb{N}_+$, let $C^1(\mathcal{A})$ denote the set of all differentiable continuous functions from $\mathcal{A}$ to $\mathbb{R}$ whose derivative is continuous and let $\lambda_{\mathcal{A}}$ denote the Lebesgue measure which is supported on $\mathcal{A}$.

2.2.1 RABBIT Model (Anchor)

This chapter considers the walking motion of a planar 5-link model of RABBIT [20] as shown in Figure 2.2. The walking motion of the RABBIT model consists of alternating phases of *single stance* (one leg in contact with the ground) and *double stance* (both legs in contact with the ground). While in single stance, the leg in contact with the ground is called the *stance leg*, and the non-stance leg is called the *swing leg*. The double stance phase is instantaneous. The configuration of the robot at time t is

$$q(t) := [q_h(t), q_v(t), q_1(t), q_2(t), q_3(t), q_4(t), q_5(t)]^\intercal \in \mathcal{Q} \subset \mathbb{R}^7 \tag{2.1}$$

where $(q_h(t), q_v(t))$ are Cartesian position of the robot hip; $q_1(t)$ is the torso angle relative to the upright direction; $q_2(t)$ and $q_4(t)$ are the hip angles relative to stance and swing leg, respectively; and $q_3(t)$ and $q_5(t)$ are the knee angles. The joints (q_2, q_3, q_4, q_5) are actuated, and q_1 is an underactuated degree of freedom. Let $\theta(q) := q_1 + q_2 + q_3/2$ denote the *stance leg angle*, and let $\phi(q) := q_1 + q_4 + q_5/2$ denote the *swing leg angle*. We refer to the configuration when the robot hip is right above the stance foot, i.e. $\theta = \pi$, as *mid-stance*. We refer to the motion between the i-th and $(i+1)$-st swing leg foot touch down with the ground as the *i-th step*.

Using the method of Lagrange, we can obtain a continuous dynamic model of the robot during swing phase:

$$\dot{a}(t) = f(a(t), u(t)) \tag{2.2}$$

where $a(t) = [q^\intercal(t), \dot{q}^\intercal(t)]^\intercal \in T\mathcal{Q} \subset \mathbb{R}^{14}$ denotes the tangent bundle of $\mathcal{Q}$, $u(t) \in U$, U describes the permitted inputs to the system, and t denotes time. We model the RABBIT as a hybrid system and describe the instantaneous change using the notation of a *guard* and a *reset map*. That is, suppose $(\theta(q(t)), c_{\text{foot}}(q(t)))$ denotes the stance leg angle and the vertical position of the swing foot relative to the stance foot, respectively, given a configuration $q(t)$ at time t. The guard $\mathcal{G}$ is $\{(b, b') \in T\mathcal{Q} \mid \pi/2 < \theta(b) < 3\pi/2, c_{\text{foot}}(b) = 0 \text{ and } \dot{c}_{\text{foot}}(b, b') < 0\}$. Notice the force of the ground contact imposes a holonomic constraint on stance foot position, which enables one to obtain a reset map: [114, Section 3.4.2]:

$$\dot{q}^+(t) = \Delta\big(\dot{q}^-(t)\big), \tag{2.3}$$

where Δ describes the relationship between the pre-impact and post impact velocities. More details about the definition and derivation of this hybrid model can be found in [114, Section 3.4].

To simplify exposition, this chapter at run-time optimizes over a family of reference gaits that are characterized by their average velocity and step length. These reference gaits are described by a vector of *control parameters* $P(i) = \big(p_1(i), p_2(i)\big) \in \mathcal{P}$ for all $i \in \mathbb{N}$, where $p_1(i)$ denotes the average horizontal velocity and $p_2(i)$ denotes the step length between the i-th and $(i+1)$-st mid-stance position. Note $\mathcal{P}$ is compact. These reference gaits are generated by solving a finite family of nonlinear optimization problems using FROST in which we incorporate $p_1(i)$, $p_2(i)$, and periodicity as constraints, and minimize the average torque squared over the gait period [41]. Each of these problems yields a reference trajectory parameterized by $P(i)$ and interpolation is applied over these generated gaits to generate a continuum of gaits. Given a control parameter, a control input into the RABBIT model is generated by tracking the corresponding reference trajectories using a classical PD controller.

Next, we define a solution to the hybrid model as a pair $(\mathcal{I}, a)$, where $\mathcal{I} = \{I_i\}_{i=0}^N$ is a *hybrid time set* with I_i being intervals in $\mathbb{R}$, and $a = \{a_i(\cdot)\}_{i=0}^N$ is a finite sequence of functions with each element $a_i(\cdot) : I_i \to TQ$ satisfying the dynamics (2.2) over I_i where $N \in \mathbb{N}$ [64, Definitions 3.3, 3.4, 3.5]. Denote each $I_i := [\tau_i^+, \tau_{i+1}^-]$ for all $i < N$. τ_i corresponds to the time of the transition between $(i-1)$-th to i-th step. We let τ_i^- correspond to the time just before the transition and and τ_i^+ correspond to the time just after the transition. Since transitions are assumed to be instantaneous, $\tau_i = \tau_i^- = \tau_i^+$ if all values exist. When a transition never happens during the i-th step, we denote $\tau_{i-1}^- = +\infty$. Note when $\tau_{i+1} < \infty$, $a_i(\tau_{i+1}^-) \in \mathcal{G}$ and $a_{i+1}(\tau_{i+1}^+) \in \Delta(a_i(\tau_{i+1}^-))$.

2.2.2 Simplified Biped Model (Template)

As we show in Section 2.5, performing online optimization with the full RABBIT model is intractable due to the size of its state space. In contrast, performing online optimization with the *Simplified Biped Model* (SBM) adopted from [116] is tractable. This model consists of a point-mass M and two mass-less legs each with a constant length l. The configuration of the SBM at time t is described by the stance leg angle, $\hat{\theta}$, and the swing leg angle, $\hat{\phi}$. The guard is the set of configurations when $\hat{\theta} + \hat{\phi} = 2\pi$. The swing leg swings immediately to a specified step length. During the swing phase, one can use the method of Lagrange to describe the evolution of the configuration as a function of the current configuration and the input. Subsequent to the instantaneous double stance phase, an impact with the ground happens with a coefficient of restitution of 0. We denote a hybrid execution of the SBM as a pair $(\hat{\mathcal{I}}, \hat{a})$ where $\hat{\mathcal{I}} = \{\hat{I}_i\}_{i=0}^N$ is a hybrid time set with $\hat{I}_i := [\hat{\tau}_i^+, \hat{\tau}_{i+1}^-]$ and $\hat{a} = \{\hat{a}_i(\cdot)\}_{i=0}^N$ is a finite sequence of solutions to the SBM's equations of motion.

2.3 Outputs to Describe Successful Walking

During online optimization, we want to optimize over the space of parameterized inputs while introducing a constraint to guarantee that the robot does not fall over. This section first formalizes what it means for the RABBIT model to walk successfully without falling over. Unfortunately due to the high-dimensionality of the RABBIT model, implementing this definition directly as a constraint during online optimization is intractable. To address this problem, Section 2.3.1 defines a set of outputs that are functions of the state of RABBIT and proves that the value of these outputs can determine whether RABBIT is able to walk successfully. Subsequently in Section 2.3.2 we define a corresponding set of outputs that are functions of the state of the SBM and illustrate how their values can be used to determine whether RABBIT is able to walk successfully.

To define successful walking on RABBIT, we begin by defining the time during step i

at which mid-stance occurs (i.e., the largest time t at which $\theta(q(t)) = \pi$ during I_i) as

$$t_i^{MS} := \begin{cases} +\infty, & \text{if } \theta(q(t)) < \pi \quad \forall t \in I_i, \\ -\infty, & \text{if } \theta(q(t)) > \pi \quad \forall t \in I_i, \\ \max\{t \in I_i \mid \theta(q(t)) = \pi\}, & \text{otherwise.} \end{cases} \tag{2.4}$$

Note if mid-stance is never reached during step i, then the mid-stance time is defined as plus or minus infinity depending upon if the hip-angle remains less than π or greater than π during step i, respectively. Using this definition, we formally define successful walking for the RABBIT model as:

Definition 1. *The RABBIT model* walks successfully *in step $i \in \mathbb{N}$ if*

1. $t_i^{MS} \neq \pm\infty$,
2. $\pi/2 < \theta(q(t)) < 3\pi/2$ *for all $t \in I_i$, and*
3. $\tau_{i+1}^- < +\infty$.

To understand this definition, note that the first requirement ensures that mid-stance is reached, the second requirement ensures that the hip remains above the ground, and the final requirement ensures that the swing leg actually makes contact with the ground. Though satisfying this definition ensures that RABBIT takes a step, enforcing this condition directly during optimization can be cumbersome due to the high dimensionality of the RABBIT dynamics.

2.3.1 Outputs to Describe Successful RABBIT Walking

This subsection defines a set of discrete outputs that are functions of the state of RABBIT model and illustrates how they can be used to predict failure. We begin by defining another time variable t_i^0:

$$t_i^0 := \begin{cases} \tau_i^+, & \text{if } \dot{\theta}(q(t), \dot{q}(t)) < 0 \quad \forall t \in I_i, \\ \tau_{i+1}^-, & \text{if } \dot{\theta}(q(t), \dot{q}(t)) > 0 \quad \forall t \in I_i, \\ \max\{t \in I_i \mid \dot{\theta}(q(t), \dot{q}(t)) = 0\}, & \text{otherwise.} \end{cases} \tag{2.5}$$

Note t_i^0 is defined to be the last time in I_i when a sign change of $\dot{\theta}$ occurs; when a sign change does not occur, t_i^0 is defined as an endpoint of I_i associated with the sign of $\dot{\theta}$.

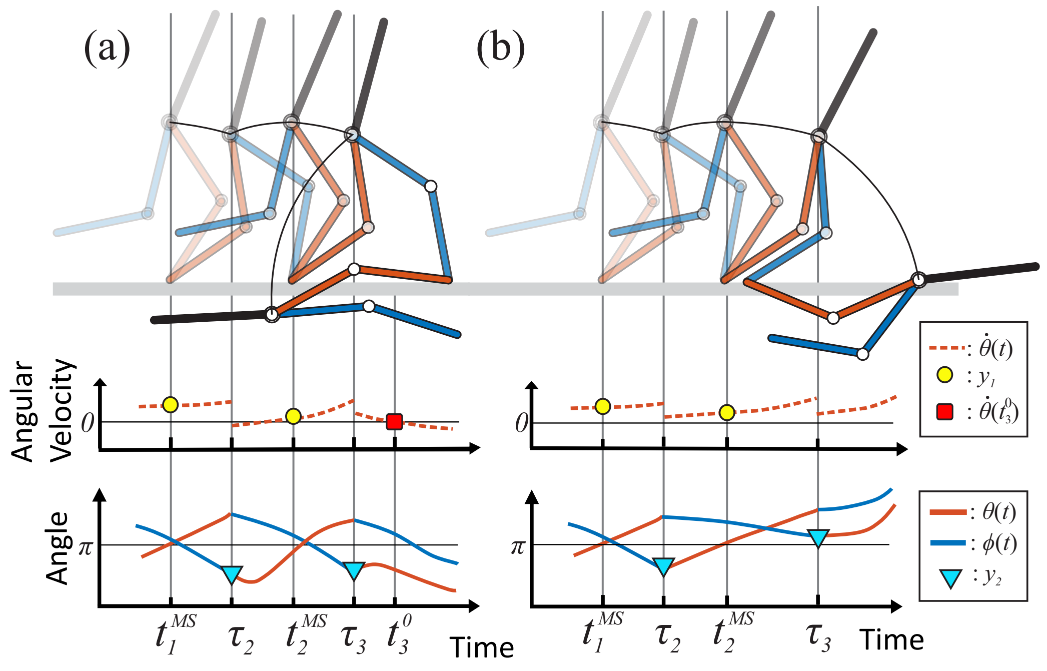


Figure 2.3: An illustration of how the values of the outputs can be used to determine whether the robot walks safely. To ensure that the robot does not fall backwards, one can require that $y_1(i) \geq 0$ (left column). In particular if $y_1(i) < 0$, then $t_i^{MS} = +\infty$ which implies that the robot is falling backwards. To ensure that the robot does not fall forward, one can require that $y_2(i) \leq \pi$ (right column).

We first define an output, $y_1 : \mathbb{N} \to \mathbb{R}$ that can be used to ensure that $t_i^{MS} \neq +\infty$:

$$y_1(i) := \begin{cases} \dot{\theta}(q(t_i^{MS}), \dot{q}(t_i^{MS})), & \text{if } t_i^{MS} \neq \pm\infty, \\ -\sqrt{2\mathrm{g}(l_{\mathrm{st}}(t_i^0) - q_v(t_i^0))}/l_{\mathrm{st}}(t_i^0), & \text{if } t_i^{MS} = +\infty, \\ 1 & \text{if } t_i^{MS} = -\infty, \end{cases} \tag{2.6}$$

where g is gravity and $l_{\mathrm{st}}(t_i^0)$ is the stance leg length at time t_i^0. Note that $y_1(i)$ is the hip angular velocity when the mid-stance position is reached during the i-th step. When the mid-stance position is not reached, $-y_1(i)$ represents the additional hip angular velocity needed to reach the mid-stance position. In particular, notice $t_i^{MS} \neq +\infty$ whenever $y_1(i) \geq 0$
.

Next, we define an output $y_2 : \mathbb{N} \to \mathbb{R}$ that can be used to ensure that $t_i^{MS} \neq -\infty$:

$$y_2(i) := \begin{cases} \phi(q(\tau_{i+1}^-)), & \text{if } \tau_{i+1}^- < +\infty, \\ 2\pi, & \text{otherwise.} \end{cases} \tag{2.7}$$

Note, $y_2(i)$ is the swing leg angle at touch-down at the end of the i-th step; if touch-down does not occur, $y_2(i)$ is defined as 2π. Recall $\phi(q(\tau_{i+1}^-)) = \theta(q(\tau_{i+1}^+))$, so if $y_2(i) \leq \pi$, it then follows from (2.4) and (2.7) that $t_{i+1}^{MS} \neq -\infty$ and $\tau_{i+1}^{-} < +\infty$. Fig. 2.3 illustrates the behavior of y_1 and y_2.

We now define our last two outputs $y_3, y_4 : \mathbb{N} \to \mathbb{R} \cup \{-\infty, +\infty\}$ that can be used to ensure that the hip stays above the ground:

$$y_3(i) := \begin{cases} \inf\{\theta(q(t)) \mid t \in [t_i^{MS}, t_{i+1}^{MS}]\}, & \text{if } t_{i+1}^{MS}, t_i^{MS} \in \mathbb{R}, \\ -\infty, & \text{otherwise.} \end{cases} \tag{2.8}$$

$$y_4(i) := \begin{cases} \sup\{\theta(q(t)) \mid t \in [t_i^{MS}, t_{i+1}^{MS}]\}, & \text{if } t_{i+1}^{MS}, t_i^{MS} \in \mathbb{R}, \\ +\infty, & \text{otherwise.} \end{cases} \tag{2.9}$$

Finally, we let $\mathcal{Y} := \mathbb{R} \times \mathbb{R} \times (\mathbb{R} \cup \{-\infty, +\infty\}) \times \mathbb{R}$.

The outputs are defined based on the observation that the hip usually has forward speed (e.g. moving forward, rather than falling backwards) at mid-stance and appears between thee two legs at touch-down when the RABBIT model walks safely. Specifically, y_1 represents the hip angular velocity at mid-stances and y_2 represents the swing leg angle at touch-downs. y_3 and y_4 are defined as the maximum and minimum stance leg angles between adjacent mid-stances, which are used to indicate whether the hip hits the ground.

Using these definitions, we can prove the following theorem that constructs a sufficient condition to ensure successful walking by RABBIT.

Theorem 2. *Suppose that the 0-th step can be successfully completed (i.e. τ_0^+ and t_0^{MS} are finite, $\inf\{\theta(q(t)) \mid t \in [\tau_0^+, t_0^{MS}]\} > \pi/2$, and $\sup\{\theta(q(t)) \mid t \in [\tau_0^+, t_0^{MS}]\} < 3\pi/2$)). Suppose $y_1(i) \geq 0$, $y_2(i) \leq \pi$, $y_3(i) > \pi/2$ and $y_4(i) < 3\pi/2$ for each $i \in \{0, \cdots, N\}$, then the robot walks successfully at the i-th step for each $i \in \{0, \cdots, N\}$.*

Proof. Notice $y_1(i) \geq 0 \Rightarrow t_i^{MS} \neq +\infty$ and $y_2(i) \leq \pi \Rightarrow t_{i+1}^{MS} \neq -\infty$ for each $i \in \{1, \cdots, N\}$. By induction we have t_i^{MS} is finite $\forall i \in \{1, \cdots, N\}$. $y_2(i) \leq \pi < 2\pi$ implies that $\tau_{i+1}^{-} < +\infty$. By using the definitions of y_3 and y_4, one has that the robot walks successfully in the i-th step based on Definition 1. $\square$

2.3.2 Approximating Outputs Using the SBM

Finding an analytical expression describing the evolution of each of the outputs can be challenging. Instead we define corresponding outputs

$$\hat{y}(i) := \big(\hat{y}_1(i), \hat{y}_2(i), \hat{y}_3(i), \hat{y}_4(i)\big) \in \mathcal{Y} \tag{2.10}$$

for SBM. Importantly, the dynamics of each of these corresponding outputs can be succinctly described.

As we did for the RABBIT model, consider the following set of definitions for the SBM:

$$\hat{t}_i^{MS} := \begin{cases} +\infty, & \text{if } \hat{\theta}(t) < \pi \quad \forall t \in \hat{I}_i, \\ -\infty, & \text{if } \hat{\theta}(t) > \pi \quad \forall t \in \hat{I}_i, \\ \max\{t \in \hat{I}_i \mid \hat{\theta}(t) = \pi\}, & \text{otherwise.} \end{cases} \tag{2.11}$$

$$\hat{t}_i^{0} := \begin{cases} \hat{\tau}_i^{+}, & \text{if } \dot{\hat{\theta}}(t) < 0 \quad \forall t \in \hat{I}_i, \\ \hat{\tau}_{i+1}^{-}, & \text{if } \dot{\hat{\theta}}(t) > 0 \quad \forall t \in \hat{I}_i, \\ \max\{t \in \hat{I}_i \mid \dot{\hat{\theta}}(t) = 0\}, & \text{otherwise.} \end{cases} \tag{2.12}$$

$$\hat{y}_1(i) := \begin{cases} \dot{\hat{\theta}}(\hat{t}_i^{MS}), & \text{if } \hat{t}_i^{MS} \neq \pm\infty \\ -\sqrt{2\mathrm{g}(l(1+\cos(\hat{\theta}(\hat{t}_i^{0}))))/l}, & \text{if } \hat{t}_i^{MS} = +\infty \\ 1 & \text{if } \hat{t}_i^{MS} = -\infty, \end{cases} \tag{2.13}$$

$$\hat{y}_2(i) := \begin{cases} \hat{\phi}(\hat{\tau}_{i+1}^{-}), & \text{if } \hat{\tau}_{i+1}^{-} < +\infty, \\ 2\pi, & \text{otherwise.} \end{cases} \tag{2.14}$$

$$\hat{y}_3(i) := \begin{cases} \inf\{\hat{\theta}(t) \mid t \in [\hat{t}_i^{MS}, \hat{t}_{i+1}^{MS}]\}, & \text{if } \hat{t}_{i+1}^{MS}, \hat{t}_i^{MS} \in \mathbb{R}, \\ -\infty, & \text{otherwise.} \end{cases} \tag{2.15}$$

$$\hat{y}_4(i) := \begin{cases} \sup\{\hat{\theta}(t) \mid t \in [\hat{t}_i^{MS}, \hat{t}_{i+1}^{MS}]\}, & \text{if } \hat{t}_{i+1}^{MS}, \hat{t}_i^{MS} \in \mathbb{R}, \\ +\infty, & \text{otherwise.} \end{cases} \tag{2.16}$$

The discrete-time dynamics of each of these outputs of SBM can be described by the following difference equations:

$$\begin{aligned} \hat{y}_1(i+1) &= f_{\hat{y}_1}\big(\hat{y}_1(i), P(i)\big) \\ \hat{y}_2(i) &= f_{\hat{y}_2}\big(P(i)\big) \\ \hat{y}_3(i) &= f_{\hat{y}_3}\big(\hat{y}_1(i), P(i)\big) \\ \hat{y}_4(i) &= f_{\hat{y}_4}\big(\hat{y}_1(i), P(i)\big) \end{aligned} \tag{2.17}$$

for each $i \in \mathbb{N}$, $\hat{y}(i) \in \mathcal{Y}$, and $P(i) \in \mathcal{P}$. Such functions $f_{\hat{y}_1}$, $f_{\hat{y}_2}$, $f_{\hat{y}_3}$ and $f_{\hat{y}_4}$ can be generated using elementary mechanics [2].

[2]A derivation can be found in appendix.

To describe the gap between the discrete signals y and $\hat{y}$ we make the following assumption:

Assumption 3. *For any sequence of control parameters, $\{P(i)\}_{i\in N}$, and corresponding sequences of outputs, $\{y_1(i), y_2(i), y_3(i), y_4(i)\}_{i\in\mathbb{N}}$ and $\{\hat{y}_1(i), \hat{y}_2(i), \hat{y}_3(i), \hat{y}_4(i)\}_{i\in\mathbb{N}}$, generated by the RABBIT dynamics and* (2.17)*, respectively, there exists* bounding functions $\underline{B}_1$, $\overline{B}_1 : \mathbb{R}\times\mathcal{P}\to\mathbb{R}$, $\overline{B}_2 : \mathcal{P}\times\mathbb{R}\times\mathcal{P}\to\mathbb{R}$, *and* $\underline{B}_3$, $\overline{B}_4 : \mathbb{R}\times\mathcal{P}\to\mathbb{R}$ *satisfying*

$$\underline{B}_1\big(y_1(i), P(i)\big) \le y_1(i+1) - \hat{y}_1(i+1) \le \overline{B}_1\big(y_1(i), P(i)\big) \tag{2.18}$$

$$y_2(i) - \hat{y}_2(i) \le \overline{B}_2\big(P(i-1), y_1(i), P(i)\big) \tag{2.19}$$

$$y_3(i) - \hat{y}_3(i) \ge \underline{B}_3\big(y_1(i), P(i)\big) \tag{2.20}$$

$$y_4(i) - \hat{y}_4(i) \le \overline{B}_4\big(y_1(i), P(i)\big). \tag{2.21}$$

In other words, if $y_1(i) = \hat{y}_1(i)$, then $\overline{B}_1$, $\underline{B}_1$, $\underline{B}_2$, $\underline{B}_3$, and $\overline{B}_4$ bound the maximum possible difference between $(y_1(i+1), y_2(i), y_3(i), y_4(i))$ and $(\hat{y}_1(i+1), \hat{y}_2(i), \hat{y}_3(i), \hat{y}_4(i))$. Though we do not describe how to construct these bounding functions here due to space limitations, one could apply SOS optimization to generate them [95] or bound the dynamics of the system [51]. Note, constructing such a bound precisely using SOS optimization can be challenging due to the high-dimensionality of the full-order model. However, several papers have proposed techniques that have begun to address these scaling challenges while applying SOS optimization [3, 54, 79, 99]. To simplify further exposition, we define the following:

$$\begin{aligned}\mathcal{B}(y_1(i), P(i)) := [&f_{\hat{y}_1}\big(y_1(i), P(i)\big) + \underline{B}_1\big(y_1(i), P(i)\big),\\ &f_{\hat{y}_1}\big(y_1(i), P(i)\big) + \overline{B}_1\big(y_1(i), P(i)\big)]\end{aligned} \tag{2.22}$$

for all $(y_1(i), P(i)) \in \mathbb{R}\times\mathcal{P}$. In particular, it follows from (2.18) that for any sequence of control parameters, $\{P(i)\}_{i\in N}$, and corresponding sequences of outputs, $\{y_1(i)\}_{i\in\mathbb{N}}$ generated by the RABBIT dynamics, $y_1(i+1) \in \mathcal{B}\big(y_1(i), P(i)\big)$ for all $i \in \mathbb{N}$.

2.4 Enforcing N-Step Safe Walking

This section proposes an online MPC framework to design a controller for the RABBIT model that can ensure successful walking for N-step. In fact, when $N = 1$ one can directly apply Theorem 2 and Assumption 3 to generate the following inequality constraints over $y_1(i)$, $P(i-1)$ and $P(i)$ to guarantee walking successfully from the i-th to the $(i+1)$-th mid-stance:

$$f_{\hat{y}_1}\big(y_1(i), P(i)\big) + \underline{B}_1\big(y_1(i), P(i)\big) \geq 0, \tag{2.23}$$
$$f_{\hat{y}_2}\big(P(i)\big) + \overline{B}_2\big(P(i-1), y_1(i), P(i)\big) \leq \pi, \tag{2.24}$$
$$f_{\hat{y}_3}\big(y_1(i), P(i)\big) + \underline{B}_3\big(y_1(i), P(i)\big) > \pi/2, \tag{2.25}$$
$$f_{\hat{y}_4}\big(y_1(i), P(i)\big) + \overline{B}_4\big(y_1(i), P(i)\big) < 3\pi/2. \tag{2.26}$$

Unfortunately, to construct a similar set of constraints when $N > 1$, one has to either compute $(y_1(i+M), y_2(i+M), y_3(i+M), y_4(i+M))$ for each $1 \leq M \leq N$, which can be computationally taxing, or one can apply (2.18) recursively to generate an outer approximation to $y_1(i+M)$ for each $1 \leq M \leq N$ and then apply the remainder of Assumption 3 to generate an outer approximation to $y_2(i+M), y_3(i+M),$ and $y_4(i+M)$ for each $1 \leq M \leq N$. In the latter instance, one would need the entire set of possible values for the outputs to satisfy conditions in Theorem 2 from the i-th step to the $(i+N)$-th step to ensure N-step safe walking. This requires introducing set inclusion constraints that can be cumbersome to enforce at run-time. To address these challenges, Section 2.4.1 describes how to compute in an offline fashion, an N-step *Forward Reachable Set* (FRS) that captures all possible outcomes for the output y_1 from a given initial state and set of control parameters for up to N steps. Subsequently, Section 2.4.2 illustrates how to write down N-step successful walking conditions on outputs (y_2, y_3, y_4) and set up an MPC framework to update gait parameters for RABBIT using nonlinear program with set inclusion constraints.

2.4.1 Forward Reachable Set

Letting $\mathcal{Y}_1 \subset \mathbb{R}$ be compact, we define the *N-step FRS of the output y_1*:

Definition 4. *The N-step FRS of the output beginning from $\big(y_1(i), P(i)\big) \in \mathcal{Y}_1 \times \mathcal{P}$ for $i \in \mathbb{N}$ and for $N \in \mathbb{N}$ is defined as*

$$\begin{aligned}\mathcal{W}_N\big(y_1(i), P(i)\big) := \bigcup_{n=i+1}^{i+N} \Big\{ & y_1(n) \in \mathcal{Y}_1 \mid \exists P(i+1), \ldots, \\ & P(n-1) \in \mathcal{P} \textit{ such that } \forall j \in \{i, \ldots, i+n-1\}, \\ & y_1(j+1) \textit{ is generated by the RABBIT} \\ & \textit{dynamics from } y_1(j) \textit{ under } P(j) \Big\}\end{aligned} \tag{2.27}$$

In other words, given a fixed output $y_1(i)$ and the current control parameter $P(i)$, the FRS $\mathcal{W}_N$ captures all the outputs $y_1(j)$ that can be reached within N steps, provided that all subsequent control parameters are contained in a set $\mathcal{P}$. Since $\mathcal{W}_N$ is the union of all possible y_1 within the next N steps, it follows that:

Lemma 5.

$$\mathcal{W}_M\big(y_1(i), P(i)\big) \subseteq \mathcal{W}_N\big(y_1(i), P(i)\big) \quad \forall 1 \leq M \leq N \tag{2.28}$$

As a result of Lemma 5, to predict the behavior of RABBIT system over N steps, it is unnecessary to compute distinct FRS-es for each of the next N steps. Instead one only needs to compute a single FRS.

To compute an outer approximation of the FRS, inspired by [39], one can solve the following infinite-dimensional linear problem over the space of functions:

$$\begin{aligned}
\inf_{w_N, v_1, \cdots, v_N} \int_{\mathcal{Y}_1 \times \mathcal{P} \times \mathcal{Y}_1} w_N(x_1, x_2, x_3)\, d\lambda_{\mathcal{Y}_1 \times \mathcal{P} \times \mathcal{Y}_1} & & \text{(FRSopt)} \\
\text{s.t.} \quad v_1(x_1, x_2, x_3) \geq 0, & & \forall x_3 \in \mathcal{B}(x_1, x_2) \\
& & \forall (x_1, x_2) \in \mathcal{Y}_1 \times \mathcal{P} \\
v_{\zeta+1}(x_1, x_2, x_4) \geq v_\zeta(x_1, x_2, x_3), & & \forall \zeta \in \{1, 2, \cdots, N-1\} \\
& & \forall x_4 \in \mathcal{B}(x_3, x_5) \\
& & \forall (x_1, x_2, x_5) \in \mathcal{Y}_1 \times \mathcal{P} \times \mathcal{P} \\
w_N(x_1, x_2, x_3) \geq 0, & & \forall (x_1, x_2, x_3) \in \mathcal{Y}_1 \times \mathcal{P} \times \mathcal{Y}_1 \\
w_N(x_1, x_2, x_3) \geq v_\zeta(x_1, x_2, x_3) + 1, & & \forall \zeta = 1, 2, \cdots, N \\
& & \forall (x_1, x_2, x_3) \in \mathcal{Y}_1 \times \mathcal{P} \times \mathcal{Y}_1
\end{aligned}$$

where the sets $\mathcal{Y}_1$ and $\mathcal{P}$ are given, and the infimum is taken over an $(N+1)$-tuple of continuous functions $(w_N, v_1, \cdots, v_N) \in \big(C^1(\mathcal{Y}_1 \times \mathcal{P} \times \mathcal{Y}_1; \mathbb{R})\big)^{N+1}$. Note that only the SBM's dynamics appear in this program via $\mathcal{B}(\cdot, \cdot)$.

Next, we prove that the FRS is contained in the 1-superlevel set of all feasible w's in (FRSopt):

Lemma 6. *Let $(w_N, v_1, \cdots, v_N)$ be feasible functions to (FRSopt), then the following condition is true for all $\big(y_1(i), P(i)\big) \in \mathcal{Y}_1 \times \mathcal{P}$:*

$$\mathcal{W}_N\big(y_1(i), P(i)\big) \subseteq \big\{x_3 \in \mathcal{Y}_1 \mid w_N\big(y_1(i), P(i), x_3\big) \geq 1\big\}. \tag{2.29}$$

Proof. Let $(w_N, v_1, \cdots, v_N)$ be feasible functions to the optimization (FRSopt). Substitute an arbitrary $y_1(i) \in \mathcal{Y}_1$ and $P(i) \in \mathcal{P}$ into x_1 and x_2, respectively. Suppose $\mu \in \mathcal{W}_N\big(y(i), P(i)\big)$, then there exists a natural number $n \in [i+1, i+N]$ and a sequence of control parameters $P(i+1),\cdots,P(n-1)\in\mathcal{P}$, such that $y_1(j+1)\in\mathcal{B}\big(y_1(j),P(j)\big)$ for all $i\le j\le n-1$ and $\mu=y_1(n)$.

We prove the result by induction. Let $x_3=y_1(i+1)\in\mathcal{B}\big(y_1(i),P(i)\big)$. It then follows from the first constraint of (FRSopt) that $v_1\big(y_1(i),P(i),y_1(i+1)\big)\ge 0$. Now, suppose $v_\zeta\big(y_1(i),P(i),y_1(i+\zeta)\big)\ge 0$ for some $1<\zeta\le n-i-1$. In the second constraint of (FRSopt), let $x_3=y_1(i+\zeta)$, $x_4=y_1(i+\zeta+1)\in\mathcal{B}\big(y_1(i+\zeta),P(i+\zeta)\big)$, and $x_5=P(i+\zeta)\in\mathcal{P}'$, then $v_{\zeta+1}\big(y_1(i),P(i),y_1(i+\zeta+1)\big)\ge 0$. By induction, we know $v_N\big(y_1(i),P(i),y_1(n)\big)\ge 0$. Using the fourth constraint of (FRSopt), let $x_3=\mu=y_1(n)$, and we get $w_N\big(y_1(i),P(i),\mu\big)\ge 1$. Therefore $\mu\in\big\{x_3\in\mathcal{Y}_1 \mid w_N\big(y_1(i),P(i),x_3\big)\ge 1\big\}$. $\square$

Though we do not describe it here due to space restrictions, a feasible polynomial solution to (FRSopt) can be computed offline by making compact approximation of $\mathcal{Y}_1$ and applying Sums-of-Squares programming [70, 126].

2.4.2 N-step Successful Walking and MPC

To ensure safe walking through N-steps beginning at step i, we require several set inclusions to be satisfied during online optimization based on Theorem 2. First, we require that $\mathcal{W}_N\big(y_1(i),P(i)\big)\subseteq[0,\infty)$, which sufficiently guarantee $y_1(i)\ge 0$ for each $i\le N$. Since we cannot compute $\mathcal{W}_N\big(y_1(i),P(i)\big)$ exactly, from Lemma 6 we instead can require that the 1-superlevel set of w_N is a subset of $[0,\infty)$.

With the help of the FRS, N-step successful walking conditions on (y_2,y_3,y_4) can be guaranteed in a fashion similar to (2.24), (2.25), (2.26) if

$$f_{\hat{y}_2}\big(P(i+M)\big)+\overline{B}_2\big(P(i+M-1),y_1(i+M),P(i+M)\big)\le\pi, \tag{2.30}$$

$$f_{\hat{y}_3}\big(y_1(i+M),P(i+M)\big)+\underline{B}_3\big(y_1(i+M),P(i+M)\big)>\pi/2, \tag{2.31}$$

$$f_{\hat{y}_4}\big(y_1(i+M),P(i+M)\big)+\overline{B}_4\big(y_1(i+M),P(i+M)\big)<3\pi/2. \tag{2.32}$$

hold for all $y_1(i+M)\in\mathcal{W}_M(y_1(i),P(i))$, $1\le M\le N$. Applying Lemma 5, one can instead enforce (2.30), (2.31), (2.32) for all $y_1(i+M)\in\mathcal{W}_N(y_1(i),P(i))$ to avoid computing $\mathcal{W}_M(y_1(i),P(i))$ for each $1\le M<N$.

We then use a MPC framework to select gait parameter for RABBIT by solving the following nonlinear program:

$$
\begin{aligned}
\min_{\substack{P(i)\\ \vdots \\ P(i+N-1)}} \quad & r\left(y(i), P(i), P(i+1), \cdots, P(i+N-1)\right) && \text{(OL)}\\
\text{s.t.} \quad & P(i), P(i+1), \cdots, P(i+N-1) \in \mathcal{P} \\
& f_{\hat{y}2}\big(P(i)\big) + \overline{B}_2\big(P(i-1), y_1(i), P(i)\big) \le \pi \\
& f_{\hat{y}3}\big(y_1(i), P(i)\big) + \underline{B}_3\big(y_1(i), P(i)\big) > \pi/2 \\
& f_{\hat{y}4}\big(y_1(i), P(i)\big) + \overline{B}_4\big(y_1(i), P(i)\big) < 3\pi/2 \\
& \mathcal{W}_N(y_1(i), P(i)) \subseteq [0, \infty) \\
& f_{\hat{y}_2}\big(P(i+M)\big) + \overline{B}_2\big(P(i+M-1), y_1(i+M), P(i+M)\big) \le \pi, \\
& \qquad\qquad \text{if } y_1(i+M) \in \mathcal{W}_N(y_1(i), P(i)),\ 1 \le M \le N-1 \\
& f_{\hat{y}_3}\big(y_1(i+M), P(i+M)\big) + \underline{B}_3\big(y_1(i+M), P(i+M)\big) > \pi/2, \\
& \qquad\qquad \text{if } y_1(i+M) \in \mathcal{W}_N(y_1(i), P(i)),\ 1 \le M \le N-1 \\
& f_{\hat{y}_4}\big(y_1(i+M), P(i+M)\big) + \overline{B}_4\big(y_1(i+M), P(i+M)\big) < 3\pi/2, \\
& \qquad\qquad \text{if } y_1(i+M) \in \mathcal{W}_N(y_1(i), P(i)),\ 1 \le M \le N-1
\end{aligned}
$$

where $r \in C^1(\mathcal{Y} \times \mathcal{P}^N; \mathbb{R})$ is any user specified cost function. Note that the last four constraints in (OL) are set inclusion constraints. These can be difficult to implement these directly. Howeveer, one can conservatively represent these set inclusion constraints by the 0-super level set of a set of polynomials by using the generalized S-procedure described in Section 2.6.3 of [16] and SOS optimization [70,126]. Note that this set of polynomial functions to conservatively represent these set inclusion constraints can be generated offline.

Notice that (OL) is solved at the i-th mid-stance and only the optimal $P(i)$ is applied to the RABBIT and the problem is then solved again for the $(i+1)$-st step. The constraints of (OL) lead to the following theorem:

Theorem 7. *Suppose that RABBIT is at the i-th mid-stance, then tracking the gait parameters associated with any feasible solution to (OL) ensures that RABBIT can walk successfully for the next N-steps.*

2.5 Results

To illustrate that the proposed method is able to guarantee safe walking performance online, we test the proposed method in simulation on RABBIT and in the real-world on Cassie and Digit.

2.5.1 Simulation

We evaluate the performance of our method when it is tasked with tracking a randomly generated speed sequence. In each trial, the RABBIT model is required to track a randomly generated speed sequence that holds still for the first 4 steps and changes to a different value starting from the 5th step, i.e. a step function. The speed sequence is restricted to be in a range $[0.2, 2]$. We repeat this experiment on 300 randomly generated speed sequences. The space of control parameter is restricted to be $\mathcal{P} = [0.25, 2] \times [0.15, 0.7]$ on which the gait library is generated. The RABBIT model is initialized with the gait whose speed is closest to the initial value of the desired speed sequence in each trial. The control parameter can only be updated at the mid-stance of each step. Our MATLAB implementation of the experiments can be found online [3].

In the proposed method, the cost function of (FRSopt) is set to be the weighted Euclidean norm of the difference between the predicted speeds and the desired speeds within the next 3 steps. Note $N = 3$. We compute an outer approximation to the (FRSopt) using the commercial solver MOSEK on a machine with 144 64-bit 2.40GHz Intel Xeon CPUs and 1 Terabyte memory. To create the bounding functions that satisfy Assumption 3, we employ simulation. In particular, we initialized the RABBIT model randomly twenty thousand times and varied the control parameter randomly for 5 steps and observed the output. We repeated the process with the SBM model using the same control parameter sequence and calculated the difference of the output between the two models on each of the twenty thousand trials. Finally, we applied SOS optimization to bound the difference from above and below to generate the error bounding polynomial that satisfied the conditions in Assumption 3.

We compare our method with a naïve method and the direct method using the same speed tracking sequences. The naïve method uses the SBM model to update gaits in an MPC framework without enforcing walking successful conditions. The direct method uses the full-order dynamics of the RABBIT model to design a controller by solving an optimal control problem via FROST [41].

Fig. 2.4 illustrates the performance of the naïve method and the method proposed in this chapter on one of the 300 trials. Note in particular that the gait generated by the naïve method is unable to be followed by the full-order RABBIT model. On the other hand, as shown in Fig. 2.4, the method proposed in this chapter is able to generate a gait that can satisfy the safety requirements described in Theorem 2. This results in a controller which can track the synthesized gait without falling over.

[3] https://github.com/pczhao/TA_GaitDesign.git

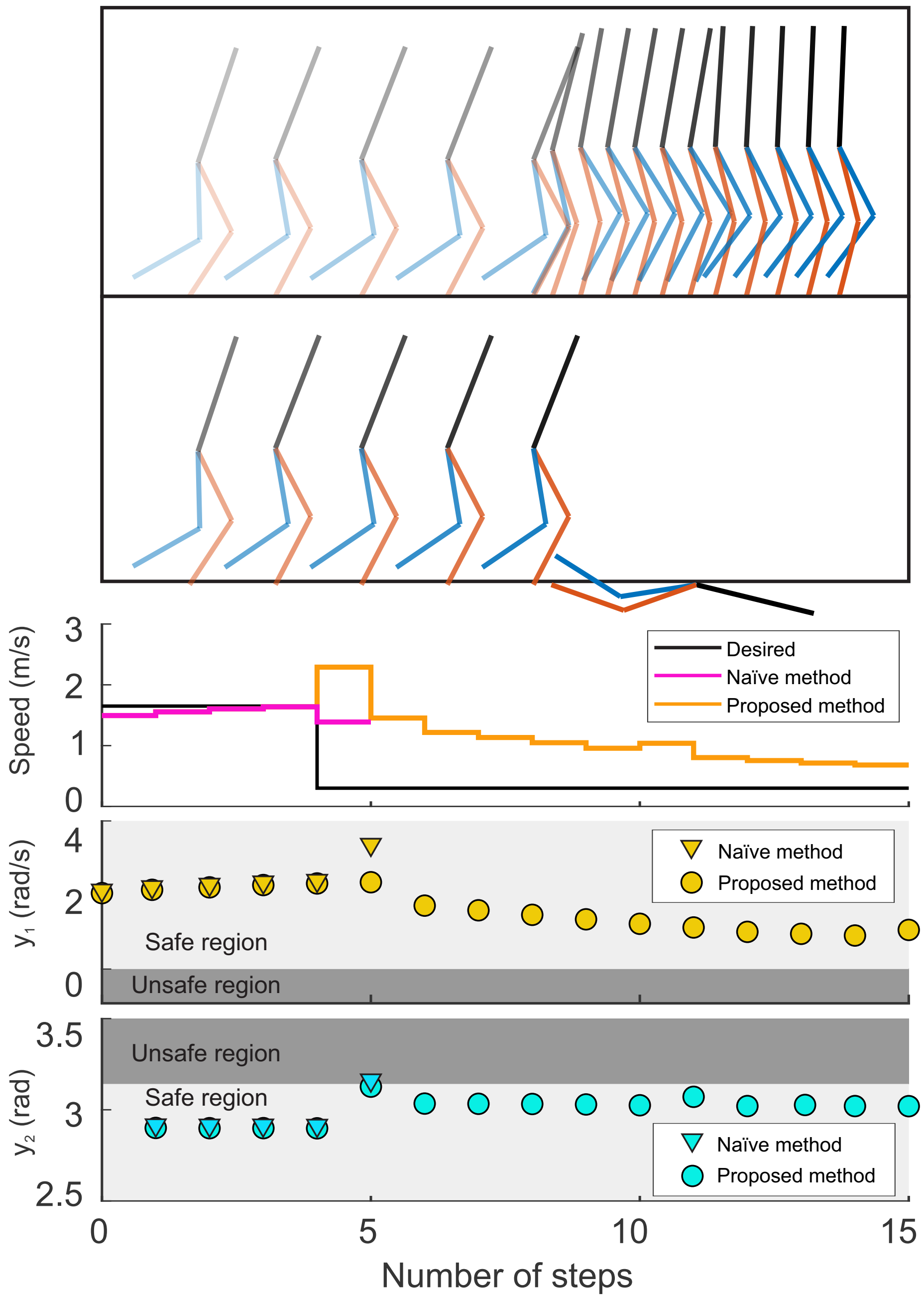


Figure 2.4: An illustration of the performance of the method proposed in this chapter (top) and the naïve method (second from top). Note that the rapid change in the desired speed (third from top) results in a gait which cannot be tracked by just considering a SBM model without successful walking constraints. By ensuring that the outputs satisfy the inequality constraints proposed in Theorem 2 (bottom two sub-figures), the proposed method is able to safely track the synthesized gaits. Note the naïve method violates the y_2 constraint proposed in Theorem 2 on Step 5.

Fig. 2.5 compares the speed tracking performance of another trial, where both methods generate gaits that can be followed by the RABBIT model. Notice naïve method achieves a lower speed tracking error of 0.3681 in Euclidean norm, while the proposed method

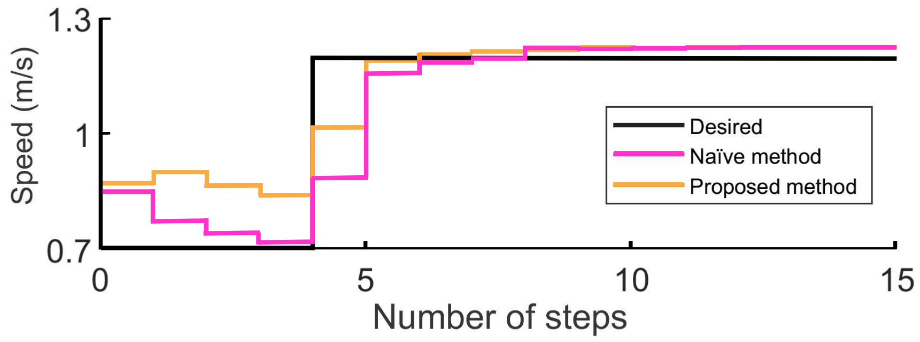


Figure 2.5: A comparison of speed tracking performance of the proposed method and the naïve method. Although both methods generate gaits that can be followed by the RABBIT model without falling over, the tracking error of naïve method is lower (0.3681) compared to the proposed method (0.3912).

achieves a slightly higher tracking error of 0.3912. This is because the additional safety constraints in the MPC prevents rapid transition from a low-speed gait to a high-speed gait, therefore generating higher costs.

Across all 300 trials the computation time of the naïve method is 0.01 seconds, the direct method is 93.12 seconds, and the proposed method is 0.11 seconds. Moreover, the RABBIT model falls 2% of the time with the naïve method, but never falls with the proposed method or the direct method. Therefore the proposed method is able to guarantee walking performance online. The average speed tracking error (computed in Euclidean norm) of the naïve method is 0.8719, and the proposed method is 0.9808.

2.5.2 Real-World Experiments

The proposed method is tested in real-world experiments on bipedal robots Cassie and Digit to achieve no-falling and collision-free walking. Safety of both bipedal robots in the sagittal plane are enforced using the proposed method, while obstacle avoidance in the horizontal plan is achieved by using techniques in [52] with a differential drive robot model [68] as its simplified model.

Because Cassie doesn't have on-board sensors to detect the environment, we use the motion capture system to specify the locations of Cassie itself and surrounding white cardboard boxes as obstacles. On the other hand, because Digit comes with a VLP-16 lidar, it runs A-LOAM as an advanced implementation of LOAM [124] to perform simultaneous localization and mapping in real-time, thus fully autonomy is achieved. As shown in Figure 2.6 and 2.7, the proposed method is able to prevent Cassie and Digit from falling while they are tasked to navigate themselves through obstacles without any collision.

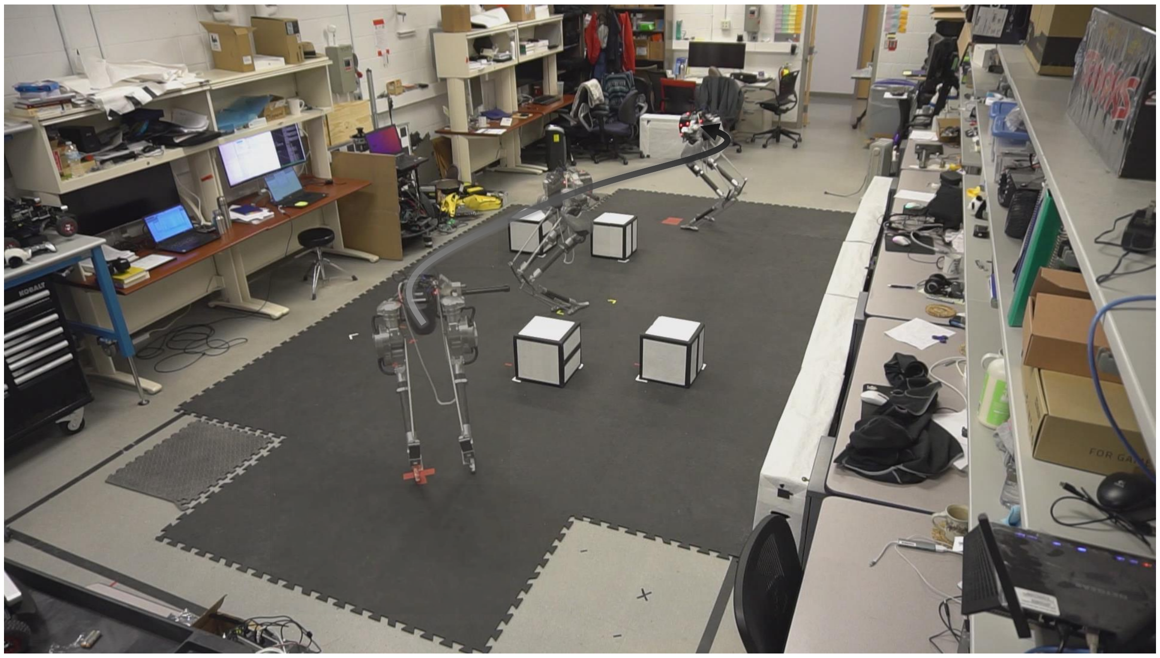


Figure 2.6: An illustration of proposed method on Cassie.

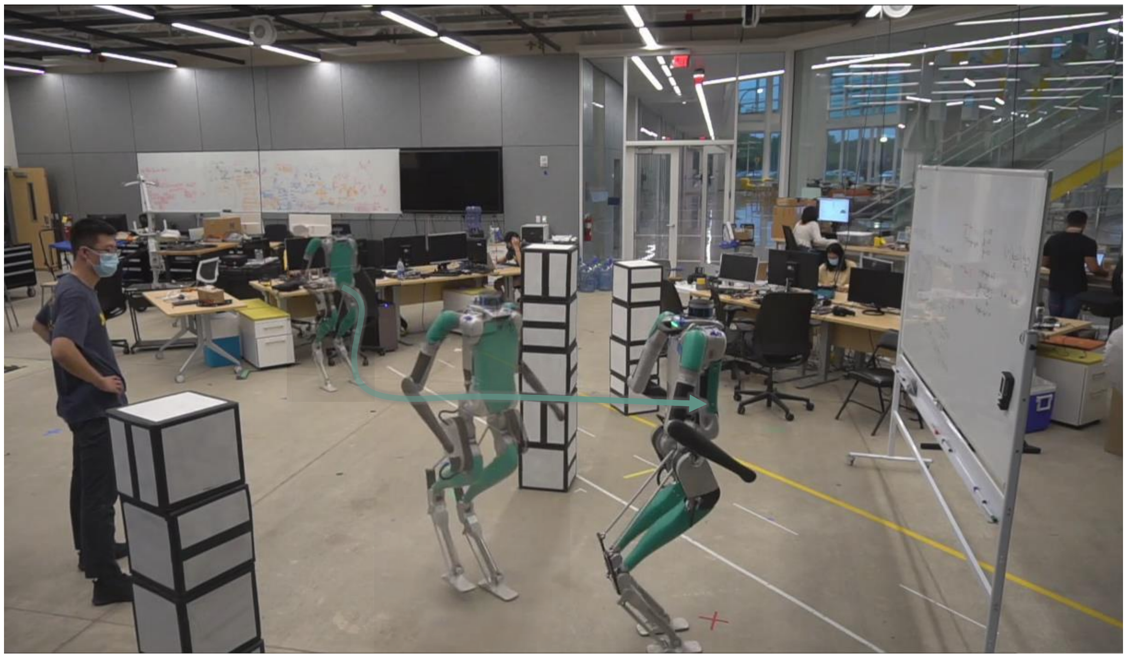


Figure 2.7: An illustration of proposed method on Digit.

2.6 Conclusion

This chapter develops a method to generate safety-preserving controllers for full-order (anchor) models by performing reachability analysis on simpler (template) models while bounding the modeling error. The method is illustrated on a 5-link, 14-dimenstional RABBIT model, and is shown to allow the robot to walk safely while utilizing controllers designed in a real-time fashion. Real-world experiments on Cassie and Digit show the proposed method is able to prevent robots from falling during walking. Though this method enables real-time motion planning, it still suffers from the same conservatism of using a simplified model as the original RTD framework. Thus in the next chapter, we aim for computing tighter forward reachable sets using the full-order robot model.

CHAPTER 3

REFINE: Reachability-based Trajectory Design Using Robust Feedback Linearization and Zonotopes

3.1 Introduction

[1] Autonomous vehicles are expected to operate safely in unknown environments with limited sensing horizons. Because new sensor information is received while the autonomous vehicle is moving, it is vital to plan trajectories using a receding-horizon strategy in which the vehicle plans a new trajectory while executing the trajectory computed in the previous planning iteration. It is desirable for such motion planning frameworks to satisfy three properties: First, they should ensure that any computed trajectory is dynamically realizable by the vehicle. Second, they should operate in real time so that they can react to newly acquired environmental information collected. Finally, they should verify that any computed trajectory when realized by the vehicle does not give rise to collisions. This paper develops an algorithm to satisfy these three requirements by designing a robust, partial feedback linearization controller and performing zonotope-based reachability analysis on a full-order vehicle model.

We begin by summarizing related works on trajectory planning and discuss their potential abilities to ensure safe performance of the vehicle in real-time. To generate safe motion plan in real-time while satisfying vehicle dynamics, it is critical to have accurate predictions of vehicle behavior over the time horizon in which planning is occurring. Because vehicle dynamics are nonlinear, closed-form solutions of vehicle trajectories are incomputable and approximations to the vehicle dynamics are utilized. For example, sampling-based methods typically discretize the system dynamic model or state space to explore the environment and find a path, which reaches the goal location and is optimal with respect to a

[1]This chapter is under review at TRO [61].

user-specified cost function [46,60]. To model vehicle dynamics during real-time planning, sampling-based methods apply online numerical integration and buffer obstacles to compensate for numerical integration error [25, 57, 59]. Ensuring that a numerically integrated trajectory can be dynamically realized and be collision-free can require applying fine time discretization. This typically results in an undesirable trade-off between these two properties and real-time operation. Similarly, Nonlinear Model Predictive Control (NMPC) uses time discretization to generate an approximation of solution to the vehicle dynamics that is embedded in optimization program to compute a control input that is dynamically realizable while avoiding obstacles [27, 44, 103, 118]. Just as in the case of sampling-based methods, NMPC also suffers from the undesirable trade-off between safety and real-time operation.

To avoid this undesirable trade-off, researchers have begun to apply reachability-based analysis. Traditionally reachability analysis was applied to verify that a pre-computed trajectory could be executed safely [8, 82]. More recent techniques apply offline reachable set analysis to compute an over-approximation of the Forward Reachable Set (FRS), which collects all possible behaviors of the vehicle dynamics over a fixed-time horizon. Unfortunately computing this FRS is challenging for systems that are nonlinear or high dimensional. To address this challenge, these reachability-based techniques have focused on pre-specifying a set of maneuvers and simplifying the dynamics under consideration. For instance, the funnel library method [66] computes a finite library of funnels for different maneuvers and over approximates the FRS of the corresponding maneuver by applying Sums-of-Squares (SOS) Programming. Computing a rich enough library of maneuvers and FRS to operate in complex environments can be challenging and result in high memory consumption. To avoid using a finite number of maneuvers, a more recent method called Reachability-based Trajectory Design (RTD) was proposed [52] that considers a continuum of trajectories and applies SOS programming to represent the FRS of a dynamical system as a polynomial level set. This polynomial level set representation can be formulated as functions of time for collision checking [53, 106, 107]. Although such polynomial approximation of the FRS ensures strict vehicle safety guarantees while maintaining online computational efficiency, SOS optimization still struggles with high dimensional systems. As a result, RTD still relies on using a simplified, low-dimensional nonlinear model that is assumed to bound the behavior of a full-order vehicle model. Unfortunately it is difficult to ensure that this assumption is satisfied. More troublingly, this assumption can make the computed FRS overly conservative because the high dimensional properties of the full-order model are treated as disturbances within the simplified model.

These aforementioned reachability-based approaches still pre-specify a set of trajecto-

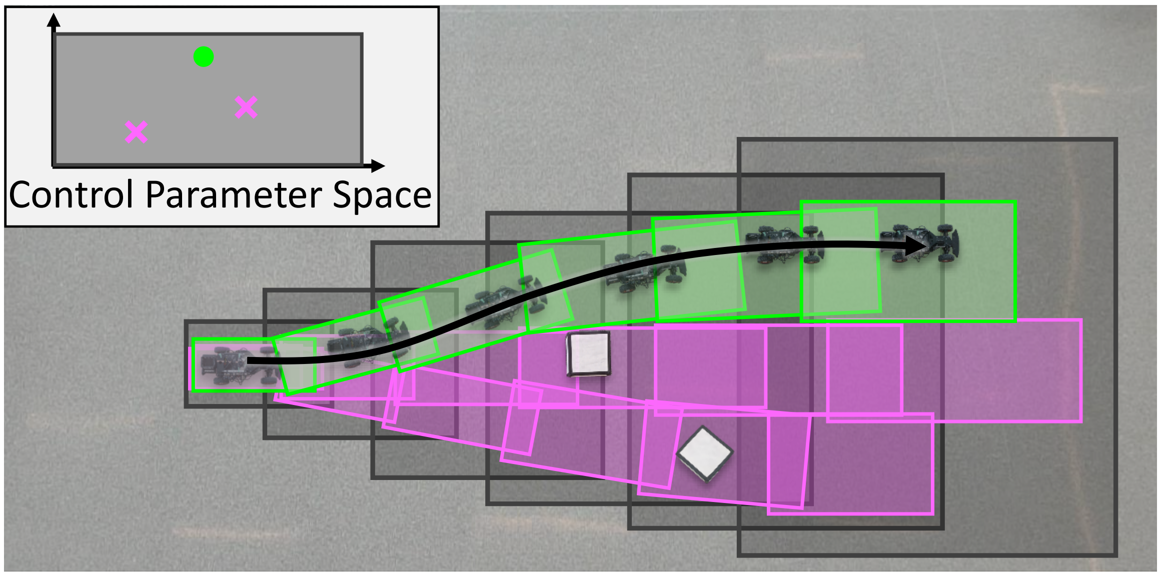


Figure 3.1: REFINE first designs a robust controller to track parameterized reference trajectories by feedback linearizing a subset of vehicle states. REFINE then performs offline reachability analysis using a closed-loop full-order vehicle dynamics to construct a control-parameterized, zonotope reachable sets (shown as grey boxes) that over-approximate all possible behaviors of the vehicle model over the planning horizon. During online planning, REFINE computes a parameterized controller that can be safely applied to the vehicle by solving an optimization problem, which selects subsets of pre-computed zonotope reachable sets that are guaranteed to be collision free. In this figure, subsets of grey zonotope reachable sets corresponding to the control parameter shown in green ensures a collision-free path while the other two control parameters shown magenta might lead to collisions with white obstacles.

ries for the offline reachability analysis. To overcome this issue, recent work has applied a Hamilton-Jacobi-Bellman based-approach [15] to pose the offline reachability analysis as a differential game between a full-order model and a simplified planning model [40]. The reachability analysis computes the tracking error between the full-order and planning models, and an associated controller to keep the error within the computed bound at run-time. At run-time, one buffers obstacles by this bound, then ensures that the planning model can only plan outside of the buffered obstacles. This approach can be too conservative in practice because the planning model is treated as if it is trying to escape from the high-fidelity model.

To address the limitations of existing approaches, this paper proposes a real-time, receding-horizon motion planning algorithm named REchability-based trajectory design using robust Feedback lInearization and zoNotopEs (REFINE) depicted in Figure 3.1 that builds on the reachability-based approach developed in [52] by using feedback linearization and zonotopes. This papers contributions are three-fold: First, a novel parameterized robust controller that partially linearizes the vehicle dynamics even in the presence of modeling error. Second, a method to perform zonotope-based reachability analysis on a closed-loop, full-order vehicle dynamics to compute a control-parameterized, over-approximate Forward Reachable Sets (FRS) that describes the vehicle behavior. Because reachability analysis is applied to the full-order model, potential conservativeness introduced by using a simplified model is avoided. Finally, an online planning framework that performs control synthesis in a receding horizon fashion by solving optimization problems in which the offline computed FRS approximation is used to check against collisions. This control synthesis framework applies to All-Wheel, Front-Wheel, or Rear-Wheel-Drive vehicle models.

The rest of this chapter is organized as follows: Section 3.2 describes necessary preliminaries and Section 3.3 describes the dynamics of Front-Wheel-Drive vehicles. Section 3.4 explains the trajectory design and vehicle safety in considered dynamic environments. Section 3.5 formulates the robust partial feedback linearization controller. Section 3.6 describes Reachability-based Trajectory Design and how to perform offline reachability analysis using zonotopes. Section 3.7 formulates the online planning using an optimization program, and in Section 3.8 the proposed method is extended to various perspectives including All-Wheel-Drive and Rear-Wheel-Drive vehicle models. Section 3.9 describes how the proposed method is evaluated and compared to other state of the art methods in simulation and in hardware demo on a 1/10th race car model. And Section 3.10 concludes the chapter.

3.2 Preliminaries

This section defines notations and set representations that are used throughout the remainder of this manuscript. Sets and subspaces are typeset using calligraphic font. Subscripts are primarily used as an index or to describe an particular coordinate of a vector.

Let $\mathbb{R}$, $\mathbb{R}_+$ and $\mathbb{N}$ denote the spaces of real numbers, real positive numbers, and natural numbers, respectively. Let $0_{n_1 \times n_2}$ denote the n_1-by-n_2 zero matrix. The Minkowski sum between two sets $\mathcal{A}$ and $\mathcal{A}'$ is $\mathcal{A} \oplus \mathcal{A}' = \{a + a' \mid a \in \mathcal{A},\ a' \in \mathcal{A}'\}$. The power set of a set $\mathcal{A}$ is denoted by $P(\mathcal{A})$. Given vectors $\alpha, \beta \in \mathbb{R}^n$, let $[\alpha]_i$ denote the i-th element of α, let $\texttt{sum}(\alpha)$ denote the summation of all elements of α, let $\|\alpha\|$ denote the Euclidean norm of α, let $\texttt{diag}(\alpha)$ denote the diagonal matrix with α on the diagonal, and let $\texttt{int}(\alpha, \beta)$ denote the n-dimensional box $\{\gamma \in \mathbb{R}^n \mid [\alpha]_i \le [\gamma]_i \le [\beta]_i,\ \forall i = 1, \dots, n\}$. Given $\alpha \in \mathbb{R}^n$ and $\epsilon > 0$, let $\mathcal{B}(\alpha, \epsilon)$ denote the n-dimensional closed ball with center α and radius ϵ under the Euclidean norm. Given arbitrary matrix $A \in \mathbb{R}^{n_1 \times n_2}$, let $A^\top$ be the transpose of A, let $[A]_{i:}$ and $[A]_{:i}$ denote the i-th row and column of A for any i respectively, and let $|A|$ be the matrix computed by taking the absolute value of every element in A.

Next, we introduce a subclass of polytopes, called zonotopes, that are used throughout this work:

Definition 8. *A* zonotope $\mathcal{Z}$ *is a subset of* $\mathbb{R}^n$ *defined as*

$$\mathcal{Z} = \left\{ x \in \mathbb{R}^n \mid x = c + \sum_{k=1}^{\ell} \beta_k g_k, \quad \beta_k \in [-1,1] \right\} \tag{3.1}$$

with center $c \in \mathbb{R}^n$ *and* ℓ generators $g_1, \ldots, g_\ell \in \mathbb{R}^n$. *For convenience, we denote* $\mathcal{Z}$ *as* $<c,\ G>$ *where* $G = [g_1, g_2, \ldots, g_\ell] \in \mathbb{R}^{n \times \ell}$.

Note that an n-dimensional box is a zonotope because

$$\texttt{int}(\alpha, \beta) = <\frac{1}{2}(\alpha + \beta),\ \frac{1}{2}\texttt{diag}(\beta - \alpha)>. \tag{3.2}$$

By definition the Minkowski sum of two arbitrary zonotopes $\mathcal{Z}_1 = <c_1,\ G_1>$ and $\mathcal{Z}_2 = <c_2,\ G_2>$ is still a zonotope as $\mathcal{Z}_1 \oplus \mathcal{Z}_2 = <c_1 + c_2,\ [G_1, G_2]>$. Finally, one can define the multiplication of a matrix A of appropriate size with a zonotope $\mathcal{Z} = <c,\ G>$ as

$$A\mathcal{Z} = \left\{ x \in \mathbb{R}^n \mid x = Ac + \sum_{k=1}^{\ell} \beta_k A g_k,\ \beta_k \in [-1,1] \right\}. \tag{3.3}$$

Note in particular that $A\mathcal{Z}$ is equal to the zonotope $<Ac,\ AG>$.

3.3 Vehicle Dynamics

This section describes the vehicle models that we used in both high-speed and low-speed scenarios throughout this manuscript for autonomous navigation with safety concerns.

3.3.1 Vehicle Model

The approach described in this chapter can be applied to a front-wheel-drive (FWD), rear-wheel drive (RWD), or all-wheel drive (AWD) vehicle models. However, to simplify exposition, we focus on how the approach applies to FWD vehicles and describe how to extend the approach to AWD or RWD vehicles in Section 3.8.3. To simplify exposition, we attach a body-fixed coordinate frame in the horizontal plane to the vehicle as shown in Fig. 3.2. This body frame's origin is the center of mass of the vehicle, and its axes are aligned with the longitudinal and lateral directions of the vehicle. Let

$z^{\text{hi}}(t) = [x(t), y(t), h(t), u(t), v(t), r(t)]^\top \in \mathbb{R}^6$ be the states of the vehicle model at time t, where $x(t)$ and $y(t)$ are the position of vehicle's center of mass in the world frame, $h(t)$ is the heading of the vehicle in the world frame, $u(t)$ and $v(t)$ are the longitudinal and lateral speeds of the vehicle in its body frame, $r(t)$ is the yaw rate of the vehicle center of mass, and $\delta(t)$ is the steering angle of the front tire. To simplify exposition, we assume vehicle weight is uniformly distributed and ignore the aerodynamic effect while modeling the flat ground motion of the vehicles by the following dynamics [48, Chapter 10.4]:

$$\dot{z}^{\text{hi}}(t) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{h}(t) \\ \dot{u}(t) \\ \dot{v}(t) \\ \dot{r}(t) \end{bmatrix} = \begin{bmatrix} u(t)\cos h(t) - v(t)\sin h(t) \\ u(t)\sin h(t) + v(t)\cos h(t) \\ r(t) \\ \frac{1}{m}\big(F_{\text{xf}}(t) + F_{\text{xr}}(t)\big) + v(t)r(t) \\ \frac{1}{m}\big(F_{\text{yf}}(t) + F_{\text{yr}}(t)\big) - u(t)r(t) \\ \frac{1}{I_{\text{zz}}}\big(l_{\text{f}}F_{\text{yf}}(t) - l_{\text{r}}F_{\text{yr}}(t)\big) \end{bmatrix} \tag{3.4}$$

where l_{f} and l_{r} are the distances from center of mass to the front and back of the vehicle, I_{zz} is the vehicle's moment of inertia, and m is the vehicle's mass. Note: l_{f}, l_{r}, I_{zz} and m are all constants and are assumed to be known. The tire forces along the longitudinal and lateral directions of the vehicle at time t are $F_{\text{xi}}(t)$ and $F_{\text{yi}}(t)$ respectively, where the 'i' subscript can be replaced by 'f' for the front wheels or 'r' for the rear wheels. Note the ignored aerodynamic effect is accounted for as dynamics computational error later in this section.

To describe the tire forces along the longitudinal and lateral directions, we first define the *wheel slip ratio* as

$$\lambda_{\text{i}}(t) = \begin{cases} \dfrac{r_{\text{w}}\omega_{\text{i}}(t) - u(t)}{u(t)} & \text{during braking} \\ \\ \dfrac{r_{\text{w}}\omega_{\text{i}}(t) - u(t)}{r_{\text{w}}\omega_{\text{i}}(t)} & \text{during acceleration} \end{cases} \tag{3.5}$$

where the 'i' subscript can be replaced as described above by 'f' for the front wheels or 'r' for the rear wheels, r_{w} is the wheel radius, $\omega_{\text{i}}(t)$ is the tire-rotational speed at time t, braking corresponds to whenever $r_{\text{w}}\omega_{\text{i}}(t) - u(t) < 0$, and acceleration corresponds to whenever

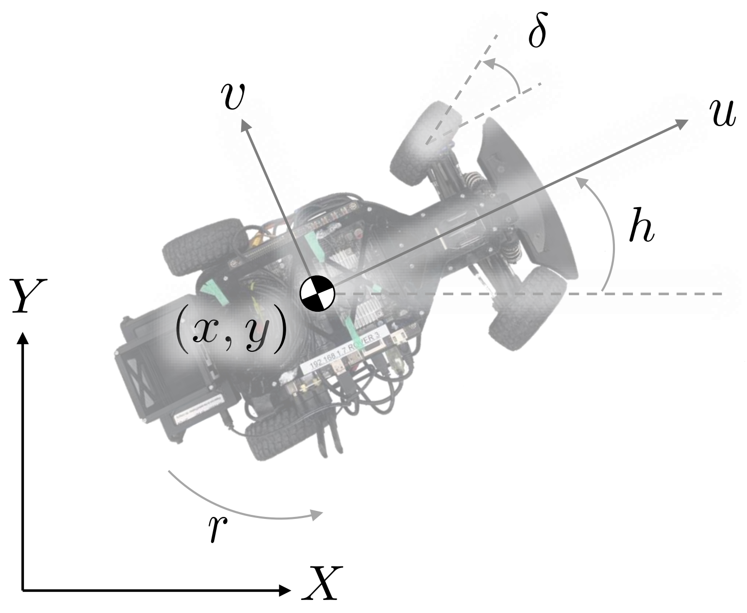


Figure 3.2: Vehicle model with the global frame shown in black and body frame in gray.

$r_{\mathrm{w}}\omega_{\mathrm{i}}(t) - u(t) \geq 0$. Then the longitudinal tire forces [102, Chapter 4] are computed as

$$F_{\mathrm{xf}}(t) = \frac{mgl_{\mathrm{r}}}{l}\mu(\lambda_{\mathrm{f}}(t)), \tag{3.6}$$

$$F_{\mathrm{xr}}(t) = \frac{mgl_{\mathrm{f}}}{l}\mu(\lambda_{\mathrm{r}}(t)), \tag{3.7}$$

where g is the gravitational acceleration constant, $l = l_{\mathrm{f}} + l_{\mathrm{r}}$, and $\mu(\lambda_{\mathrm{i}}(t))$ gives the surface-adhesion coefficient and is a function of the surface being driven on [102, Chapter 13.1]. Note that in FWD vehicles, the longitudinal rear wheel tire force has a much simpler expression:

Remark 9 ([48])**.** *In a FWD vehicle, $F_{xr}(t) = 0$ for all t.*

For the lateral direction, define *slip angles* of front and rear tires as

$$\alpha_{\mathrm{f}}(t) = \delta(t) - \frac{v(t) + l_{\mathrm{f}}r(t)}{u(t)}, \tag{3.8}$$

$$\alpha_{\mathrm{r}}(t) = -\frac{v(t) - l_{\mathrm{r}}r(t)}{u(t)}, \tag{3.9}$$

then the lateral tire forces [102, Chapter 4] are real-valued functions of the slip angles:

$$F_{\mathrm{yf}}(t) = c_{\alpha\mathrm{f}}(\alpha_{\mathrm{f}}(t)), \tag{3.10}$$

$$F_{\mathrm{yr}}(t) = c_{\alpha\mathrm{r}}(\alpha_{\mathrm{r}}(t)). \tag{3.11}$$

Note $\mu,\ c_{\alpha\mathrm{f}}$ and $c_{\alpha\mathrm{r}}$ are all nonlinear functions, but share similar characteristics. In particular, they behave linearly when the slip ratio and slip angle are close to zero, but saturate when the magnitudes of the slip ratio and slip angle reach some critical values of λ^{cri} and α^{cri}, respectively, then decrease slowly [102, Chapter 4, Chapter 13]. As we describe in Section 3.8.2, during trajectory optimization we are able to guarantee that $\mu,\ c_{\alpha\mathrm{f}}$ and $c_{\alpha\mathrm{r}}$ operate in the linear regime. As a result, to simplify exposition until we reach Section 3.8.2, we make the following assumption:

Assumption 10. *The absolute values of the the slip ratio and angle are bounded below their critical values (i.e., $|\lambda_f(t)|, |\lambda_r(t)| < \lambda^{cri}$ and $|\alpha_f(t)|, |\alpha_r(t)| < \alpha^{cri}$ hold for all time).*

Assumption 10 ensures that the longitudinal tire forces can be described as

$$\begin{aligned} F_{\mathrm{xf}}(t) &= \frac{mgl_{\mathrm{r}}}{l}\bar{\mu}\lambda_{\mathrm{f}}(t), \\ F_{\mathrm{xr}}(t) &= \frac{mgl_{\mathrm{f}}}{l}\bar{\mu}\lambda_{\mathrm{r}}(t), \end{aligned} \tag{3.12}$$

and the lateral tire forces can be described as

$$\begin{aligned} F_{\mathrm{yf}}(t) &= \bar{c}_{\alpha\mathrm{f}}\alpha_{\mathrm{f}}(t), \\ F_{\mathrm{yr}}(t) &= \bar{c}_{\alpha\mathrm{r}}\alpha_{\mathrm{r}}(t), \end{aligned} \tag{3.13}$$

with constants $\bar{\mu}, \bar{c}_{\alpha\mathrm{f}}, \bar{c}_{\alpha\mathrm{r}} \in \mathbb{R}$. Note $\bar{c}_{\alpha\mathrm{f}}$ and $\bar{c}_{\alpha\mathrm{r}}$ are referred to as *cornering stiffnesses*.

Note that the steering angle of the front wheel, δ, and the tire rotational speed, ω_{i}, are the inputs that one is able to control. In particular for an AWD vehicle, both ω_{f} and ω_{r} are inputs; whereas, in a FWD vehicle only ω_{f} is an input. When we formulate our controller in Section 3.5 for a FWD vehicle we begin by assuming that we can directly control the front tire forces, F_{xf} and F_{yf}. We then illustrate how to compute δ and ω_{f} when given F_{xf} and F_{yf}. For an AWD vehicle, we describe in Section 3.8.3, how to compute δ, ω_{f}, and ω_{r}.

In fact, as we describe in Section 3.5, our approach to perform control relies upon estimating the rear tire forces and controlling the front tire forces by applying appropriate tire speed and steering angle. Unfortunately, in the real-world our state estimation and models for front and rear tire forces may be inaccurate and aerodynamic-drag force could also affect vehicle dynamics [102, Section 4.2]. To account for the inaccuracy, we extend the vehicle dynamic model in (3.4) by introducing a time-varying affine modeling error $\Delta_u, \Delta_v, \Delta_r$ into the dynamics of u, v, and r:

$$\dot{z}^{\text{hi}}(t) = \begin{bmatrix} u(t)\cos h(t) - v(t)\sin h(t) \\ u(t)\sin h(t) + v(t)\cos h(t) \\ r(t) \\ \frac{1}{m}\big(F_{\text{xf}}(t) + F_{\text{xr}}(t)\big) + v(t)r(t) + \Delta_u(t) \\ \frac{1}{m}\big(F_{\text{yf}}(t) + F_{\text{yr}}(t)\big) - u(t)r(t) + \Delta_v(t) \\ \frac{1}{I_{zz}}\big(l_{\text{f}}F_{\text{yf}}(t) - l_{\text{r}}F_{\text{yr}}(t)\big) + \Delta_r(t) \end{bmatrix}. \tag{3.14}$$

Note, we have abused notation and redefined $\dot{z}^{\text{hi}}$ which was originally defined in (3.4). For the remainder of this paper, we assume that the dynamics $\dot{z}^{\text{hi}}$ evolves according to (3.14). To ensure that this definition is well-posed (i.e. their solution exists and is unique) and to aid in the development of our controller as described in Section 3.5, we make the following assumption:

Assumption 11. $\Delta_u, \Delta_v, \Delta_r$ *are all square integrable functions and are bounded (i.e., there exist real numbers* $M_u, M_v, M_r \in [0, +\infty)$ *such that* $\|\Delta_u(t)\|_\infty \leq M_u$, $\|\Delta_v(t)\|_\infty \leq M_v$, $\|\Delta_r(t)\|_\infty \leq M_r$ *for all t).*

Note in Section 3.9.3.3, we explain how to compute $\Delta_u, \Delta_v, \Delta_r$ using real-world data.

3.3.2 Low-Speed Vehicle Model

When the vehicle speed lowers below some critical value $u^{\text{cri}} > 0$, the denominator of the wheel slip ratio (3.5) and tire slip angles (3.8) and (3.9) approach zero which makes applying the model described in (3.4) intractable. As a result, in this work when $u(t) \leq u^{\text{cri}}$ the dynamics of a vehicle are modeled using a steady-state cornering model [33, Chapter 6], [14, Chapter 5], [22, Chapter 10]. Note that the critical velocity u^{cri} can be found according to [49, (5) and (18)].

The steady-state cornering model or low-speed vehicle model is described using four states, $z^{\text{lo}}(t) = [x(t), y(t), h(t), u(t)]^\top \in \mathbb{R}^4$ at time t. This model ignores transients on lateral velocity and yaw rate. Note that the dynamics of x, y, h and u are the same as in the high speed model (3.14); however, the steady-state corning model describes the yaw rate and lateral speed as

$$v^{\text{lo}}(t) = l_{\text{r}} r^{\text{lo}}(t) - \frac{m l_{\text{f}}}{\bar{c}_{\alpha\text{r}} l} u(t)^2 r^{\text{lo}}(t) \tag{3.15}$$

$$r^{\text{lo}}(t) = \frac{\delta(t) u(t)}{l + C_{\text{us}} u(t)^2} \tag{3.16}$$

with understeer coefficient

$$C_{\text{us}} = \frac{m}{l}\left(\frac{l_{\text{r}}}{\bar{c}_{\alpha\text{f}}} - \frac{l_{\text{f}}}{\bar{c}_{\alpha\text{r}}}\right). \tag{3.17}$$

As a result, $\dot{z}^{\text{lo}}$ satisfies the dynamics of the first four states in (3.4) except with r^{lo} taking the role of r and v^{lo} taking the role of v.

Notice when $u(t) = v(t) = r(t) = 0$ and the longitudinal tire forces are zero, $\dot{u}(t)$ could still be nonzero due to a nonzero $\Delta_u(t)$. To avoid this issue, we make a tighter assumption on $\Delta_u(t)$ without violating Assumption 11:

Assumption 12. *For all t such that $u(t) \in [0, u^{cri}]$, $|\Delta_u(t)|$ is bounded from above by a linear function of $u(t)$ (i.e.,*

$$|\Delta_u(t)| \le b_u^{pro} \cdot u(t) + b_u^{off}, \text{ if } u(t) \in [0, u^{cri}], \tag{3.18}$$

where b_u^{pro} and b_u^{off} are constants satisfying $b_u^{pro} \cdot u^{cri} + b_u^{off} \le M_u$). In addition, $\Delta_u(t) = 0$ if $u(t) = 0$.

As we describe in detail in Section 3.5.3, the high-speed and low-speed models can be combined together as a hybrid system to describe the behavior of the vehicle across all longitudinal speeds. In short, when u transitions past the critical speed u^{cri} from above at time t, the low speed model's states are initialized as:

$$z^{\text{lo}}(t) = \pi_{1:4}(z^{\text{hi}}(t)) \tag{3.19}$$

where $\pi_{1:4} : \mathbb{R}^6 \to \mathbb{R}^4$ is the projection operator that projects $z^{\text{hi}}(t)$ onto its first four dimensions via the identity relation. If u transitions past the critical speed from below at time t, the high speed model's states are initialized as

$$z^{\text{hi}}(t) = [z^{\text{lo}}(t)^\top, v^{\text{lo}}(t), r^{\text{lo}}(t)]^\top. \tag{3.20}$$

3.4 Trajectory Design and Safety

This section describes the space of trajectories that are optimized over at run-time within REFINE, how this chapter defines safety during motion planning via the notion of not-at-fault behavior, and what assumptions this chapter makes about the environment surrounding the ego-vehicle.

3.4.1 Trajectory Parameterization

Each trajectory plan is specified over a compact time interval. Without loss of generality, we let this compact time interval have a fixed duration t_f. Because REFINE performs receding-horizon planning, we make the following assumption about the time available to construct a new plan:

Assumption 13. *During each planning iteration starting from time t_0, the ego vehicle has t_{plan} seconds to find a control input. This control input is applied during the time interval $[t_0 + t_{plan}, t_0 + t_{plan} + t_f]$ where $t_f \geq 0$ is a user-specified constant. In addition, the state of the vehicle at time $t_0 + t_{plan}$ is known at time t_0.*

In each planning iteration, REFINE chooses a trajectory to be followed by the ego vehicle. These trajectories are chosen from a pre-specified continuum of trajectories, with each uniquely determined by a *trajectory parameter* $p \in \mathcal{P}$. Let $\mathcal{P} \subset \mathbb{R}^{n_p}$, $n_p \in \mathbb{N}$ be a n-dimensional box $\texttt{int}(\underline{p}, \overline{p})$ where $\underline{p}, \overline{p} \in \mathbb{R}^{n_p}$ indicate the element-wise lower and upper bounds of p, respectively. We define these desired trajectories as follows:

Definition 14. *For each $p \in \mathcal{P}$, a* desired trajectory *is a function for the longitudinal speed, $u^{des}(\cdot, p) : [t_0 + t_{plan}, t_0 + t_{plan} + t_f] \to \mathbb{R}$, a function for the heading, $h^{des}(\cdot, p) : [t_0 + t_{plan}, t_0 + t_{plan} + t_f] \to \mathbb{R}$, and a function for the yaw rate, $r^{des}(\cdot, p) : [t_0 + t_{plan}, t_0 + t_{plan} + t_f] \to \mathbb{R}$, that satisfy the following properties.*

1. *For all $p \in \mathcal{P}$, there exists a time instant $t_m \in [t_0 + t_{plan}, t_0 + t_{plan} + t_f)$ after which the desired trajectory begins to brake (i.e., $|u^{des}(t, p)|$, $|h^{des}(t, p)|$ and $|r^{des}(t, p)|$ are non-increasing for all $t \in [t_m, t_0 + t_{plan} + t_f]$).*

2. *The desired trajectory eventually comes to and remains stopped (i.e., there exists a $t_{stop} \in [t_0 + t_{plan}, t_0 + t_{plan} + t_f]$ such that $u^{des}(t, p) = h^{des}(t, p) = r^{des}(t, p) = 0$ for all $t \geq t_{stop}$).*

3. *u^{des} and h^{des} are piecewise continuously differentiable [88, Chapter 6, §1.1] with respect to t and p.*

4. *The time derivative of the heading function is equal to the yaw rate function (i.e., $r^{des}(t,p) = \frac{\partial}{\partial t} h^{des}(t,p)$ over all regions that $h^{des}(t,p)$ is continuously differentiable with respect to t).*

The first two properties ensure that a fail safe contingency braking maneuver is always available and the latter two properties ensure that the tracking controller described in Section 3.5 is well-defined. Note that sometimes we abuse notation and evaluate a desired trajectory for $t > t_0 + t_{\text{plan}} + t_{\text{f}}$. In this instance, the value of the desired trajectory is equal to its value at $t_0 + t_{\text{plan}} + t_{\text{f}}$.

3.4.2 Not-At-Fault

In dynamic environments, avoiding collision may not always be possible (e.g. a parked car can be run into). As a result, we instead develop a trajectory synthesis technique which ensures that the ego vehicle is not-at-fault [91]:

Definition 15. *The ego vehicle is* not-at-fault *if it is stopped, or if it is never in collision with any obstacles while it is moving.*

In other words, the ego vehicle is not responsible for a collision if it has stopped and another vehicle collides with it. One could use a variant of not-at-fault and require that when the ego-vehicle comes to a stop it leave enough time for all surrounding vehicles to come safely to a stop as well. The remainder of the paper can be generalized to accommodate this variant of not-at-fault; however, in the interest of simplicity we use the aforementioned definition.

Remark 16. *Under Assumption 10, neither longitudinal nor lateral tire forces saturate (i.e., drifting cannot occur). As a result, if the ego vehicle has zero longitudinal speed, it also has zero lateral speed and yaw rate. Therefore in Definition 15, the ego vehicle being stopped is equivalent to its longitudinal speed being* 0.

3.4.3 Environment and Sensing

To provide guarantees about vehicle behavior in a receding horizon planning framework and inspired by [107, Section 3], we define the ego vehicle's footprint as:

Definition 17. *Given $\mathcal{W} \subset \mathbb{R}^2$ as the world space, the ego vehicle is a rigid body that lies in a rectangle $\mathcal{O}^{ego} := \texttt{int}([-0.5L, -0.5W]^T, [0.5L, 0.5W]^T) \subset \mathcal{W}$ with width $W > 0$, length $L > 0$ at time $t = 0$. Such $\mathcal{O}^{ego}$ is called the* footprint *of the ego vehicle.*

In addition, we define the dynamic environment in which the ego vehicle is operating within as:

Definition 18. *An* obstacle *is a set $\mathcal{O}_i(t) \subset \mathcal{W}$ that the ego vehicle cannot intersect with at time t, where $i \in \mathcal{I}$ is the index of the obstacle and $\mathcal{I}$ contains finitely many elements.*

The dependency on t in the definition of an obstacle allows the obstacle to move as t varies. However if the i-th obstacle is static, then $\mathcal{O}_i(t)$ remains constant at all time. Assuming that the ego vehicle has a maximum speed ν^{ego} and all obstacles have a maximum speed ν^{obs} for all time, we then make the following assumption on planning and sensing horizon.

Assumption 19. *The ego vehicle senses all obstacles within a sensor radius $S > (t_f + t_{plan}) \cdot (\nu^{ego} + \nu^{obs}) + 0.5\sqrt{L^2 + W^2}$ around its center of mass.*

Assumption 19 ensures that any obstacle that can cause a collision between times $t \in [t_0 + t_{\text{plan}}, t_0 + t_{\text{plan}} + t_{\text{f}}]$ can be detected by the vehicle [107, Theorem 15]. Note one could treat sensor occlusions as a obstacles that travel at the maximum obstacle speed [122, 123].

3.5 Controller Design and Hybrid System Vehicle Model

This section describes the control inputs that we use to follow the desired trajectories and describes the closed-loop hybrid system vehicle model. Recall that the control inputs to the vehicle dynamics model are the steering angle of the front wheel, δ, and the tire rotational speed, ω_{i}. Section 3.5.1 describes how to select front tire forces to follow a desired trajectory and Section 3.5.2 describes how to compute a steering angle and tire rotational speed input from these computed front tire forces. Section 3.5.3 describes the closed-loop hybrid system model of the vehicle under the chosen control input. Note that this section focuses on the FWD vehicle model.

3.5.1 Robust Controller

Because applying reachability analysis to linear systems generates tighter approximations of the system behavior when compared to nonlinear systems, we propose to develop a feedback controller that linearizes the dynamics. Unfortunately, because both the high-speed and low-speed models introduced in Section 3.3 are under-actuated (i.e., the dimension of control inputs is smaller than that of system state), our controller is only able to partially feedback linearize the vehicle dynamics. Such controller is also expected to be robust such that it can account for computational errors as described in Assumptions 11 and 12.

We start by introducing the controller on longitudinal speed whose dynamics appears in both high-speed and low-speed models. Recall $\|\Delta_u(t)\|_\infty \leq M_u$ in Assumption 11. Inspired by the controller developed in [35], we set the longitudinal front tire force to be

$$\begin{aligned} F_{\text{xf}}(t) = -mK_u(u(t) - u^{\text{des}}(t,p)) + m\dot{u}^{\text{des}}(t,p) + \\ -F_{\text{xr}}(t) - mv(t)r(t) + m\tau_u(t,p), \end{aligned} \tag{3.21}$$

where

$$\tau_u(t,p) = -\big(\kappa_u(t,p)M_u + \phi_u(t,p)\big)e_u(t,p), \tag{3.22}$$

$$\kappa_u(t,p) = \kappa_{1,u} + \kappa_{2,u}\int_{t_0}^{t} \|u(s) - u^{\text{des}}(s,p)\|^2 ds, \tag{3.23}$$

$$\phi_u(t,p) = \phi_{1,u} + \phi_{2,u}\int_{t_0}^{t} \|u(s) - u^{\text{des}}(s,p)\|^2 ds, \tag{3.24}$$

$$e_u(t,p) = u(t) - u^{\text{des}}(t,p), \tag{3.25}$$

with user-chosen constants $\kappa_{1,u}, \kappa_{2,u}, \phi_{1,u}, \phi_{2,u} \in \mathbb{R}_+$. Note in (3.21) we have suppressed the dependence on p in $F_{\text{xf}}(t)$ for notational convenience. Using (3.21), the closed-loop dynamics of u become:

$$\begin{aligned} \dot{u}(t) = \tau_u(t,p) + \Delta_u(t) + \dot{u}^{\text{des}}(t,p) + \\ -K_u\left(u(t) - u^{\text{des}}(t,p)\right). \end{aligned} \tag{3.26}$$

The same control strategy can be applied to vehicle yaw rate whose dynamics only appear in the high-speed vehicle model. Let the lateral front tire force be

$$\begin{aligned} F_{\text{yf}}(t) = -\frac{I_{\text{zz}}K_r}{l_{\text{f}}}\left(r(t) - r^{\text{des}}(t,p)\right) + \frac{I_{\text{zz}}}{l_{\text{f}}}\dot{r}^{\text{des}}(t,p) + \\ -\frac{I_{\text{zz}}K_h}{l_{\text{f}}}\left(h(t) - h(0) - h^{\text{des}}(t,p)\right) + \frac{l_{\text{r}}}{l_{\text{f}}}F_{\text{yr}}(t) + \frac{I_{\text{zz}}}{l_{\text{f}}}\tau_r(t,p), \end{aligned} \tag{3.27}$$

where

$$\tau_r(t,p) = -\left(\kappa_r(t,p)M_r + \phi_r(t,p)\right)e_r(t,p) \tag{3.28}$$

$$\kappa_r(t,p) = \kappa_{1,r} + \kappa_{2,r}\int_{t_0}^{t}\left\|\begin{bmatrix} r(s) \\ h(s)\end{bmatrix} - \begin{bmatrix} r^{\text{des}}(s,p) \\ h^{\text{des}}(s,p)\end{bmatrix}\right\|^2 ds \tag{3.29}$$

$$\phi_r(t,p) = \phi_{1,r} + \phi_{2,r}\int_{t_0}^{t}\left\|\begin{bmatrix} r(s) \\ h(s)\end{bmatrix} - \begin{bmatrix} r^{\text{des}}(s,p) \\ h^{\text{des}}(s,p)\end{bmatrix}\right\|^2 ds \tag{3.30}$$

$$e_r(t,p) = \begin{bmatrix} K_r & K_h\end{bmatrix}\begin{bmatrix} r(t) - r^{\text{des}}(t,p) \\ h(t) - h^{\text{des}}(t,p)\end{bmatrix} \tag{3.31}$$

with user-chosen constants $\kappa_{1,r}, \kappa_{2,r}, \phi_{1,r}, \phi_{2,r} \in \mathbb{R}_+$. Note in (3.27) we have again suppressed the dependence on p in $F_{\text{yf}}(t)$ for notational convenience. Using (3.27), the closed-loop dynamics of r become:

$$\begin{aligned}\dot{r}(t) =& \tau_r(t,p) + \Delta_r(t) + \dot{r}^{\text{des}}(t,p) + \\ & - K_r\big(r(t) - r^{\text{des}}(t,p)\big) + \\ & - K_h\big(h(t) - h^{\text{des}}(t,p)\big).\end{aligned} \tag{3.32}$$

Using (3.27), the closed-loop dynamics of v become:

$$\begin{aligned}\dot{v}(t) = \frac{1}{m}\Bigg(&\frac{l}{l_{\text{f}}}F_{\text{yr}}(t) + \frac{I_{\text{zz}}}{l_{\text{f}}}\Big(\tau_r(t,p) + \dot{r}^{\text{des}}(t,p) + \\ & -u(t)r(t) + \Delta_v(t) - K_r\big(r(t) - r^{\text{des}}(t,p)\big) + \\ & -K_h\big(h(t) - h^{\text{des}}(t,p)\big)\Big)\Bigg).\end{aligned} \tag{3.33}$$

Because u^{des}, r^{des}, and h^{des} depend on trajectory parameter p, one can rewrite the closed loop high-speed and low-speed vehicle models as

$$\dot{z}^{\text{hi}}(t) = f^{\text{hi}}(t, z^{\text{hi}}(t), p), \tag{3.34}$$

$$\dot{z}^{\text{lo}}(t) = f^{\text{lo}}(t, z^{\text{lo}}(t), p), \tag{3.35}$$

where dynamics of x, y and h are stated as the first three dimensions in (3.4), closed-loop dynamics of u is described in (3.26), and closed-loop dynamics of v and r in the high-speed model are presented in (3.33) and (3.32). Note that the lateral tire force could be defined to simplify the dynamics on v instead of r, but the resulting closed loop system may differ. Controlling the yaw rate may be easier in real applications, because r can be directly measured by an IMU unit.

3.5.2 Extracting Wheel Speed and Steering Inputs

Because we are unable to directly control tire forces, it is vital to compute wheel speed and steering angle such that the proposed controller described in (3.21) and (3.27) is viable. Under Assumption 10, wheel speed and steering inputs can be directly computed in closed form. The wheel speed to realize longitudinal front tire force (3.21) can be derived from (3.5) and (3.12) as

$$\omega_{\text{f}}(t) = \begin{cases} \left(\dfrac{lF_{\text{xf}}(t)}{\bar{\mu}mgl_{\text{r}}} + 1\right)\dfrac{u(t)}{r_{\text{w}}} & \text{during braking,} \\[2ex] \dfrac{u(t)}{\left(1 - \frac{lF_{\text{xf}}(t)}{\bar{\mu}mgl_{\text{r}}}\right)r_{\text{w}}} & \text{during acceleration.} \end{cases} \tag{3.36}$$

Similarly according to (3.8) and (3.13), the steering input

$$\delta(t) = \frac{F_{\text{yf}}(t)}{\bar{c}_{\alpha\text{f}}} + \frac{v(t) + l_{\text{f}}r(t)}{u(t)} \tag{3.37}$$

achieves the lateral front tire force in (3.27) when $u(t) > u^{\text{cri}}$.

Notice lateral tire forces does not appear in the low-speed dynamics, but one is still able to control the lateral behavior of the ego vehicle. Based on (3.15) and (3.16), yaw rate during low-speed motion is directly controlled by steering input $\delta(t)$ and lateral velocity depends on yaw rate. Thus to achieve desired behavior on the lateral direction, one can set the steering input to be

$$\delta(t) = \frac{r^{\text{des}}(t)(l + C_{\text{us}}u(t)^2)}{u(t)}. \tag{3.38}$$

3.5.3 Augmented State and Hybrid Vehicle Model

To simplify the presentation throughout the remainder of the paper, we define a hybrid system model of the vehicle dynamics that switches between the high and low speed vehicle models when passing through the critical longitudinal velocity. In addition, for computational reasons that are described in subsequent sections, we augment the initial condition of the system to the state vector while describing the vehicle dynamics. In particular, denote $z_0 = [(z_0^{\text{pos}})^\top, (z_0^{\text{vel}})^\top]^\top \in \mathcal{Z}_0 \subset \mathbb{R}^6$ the initial condition of the ego vehicle where $z_0^{\text{pos}} = [x_0, y_0, h_0]^\top \in \mathbb{R}^3$ gives the value of $[x(t), y(t), h(t)]^\top$ and $z_0^{\text{vel}} = [u_0, v_0, r_0]^\top \in \mathbb{R}^3$

gives the value of $[u(t), v(t), r(t)]^\top$ at time $t = 0$. Then we augment the initial velocity condition $z_0^{\text{vel}} \in \mathbb{R}^3$ of the vehicle model and trajectory parameter p into the vehicle state vector as $z^{\text{aug}}(t) = [x(t), y(t), h(t), u(t), v(t), r(t), (z_0^{\text{vel}})^\top, p^\top]^\top \in \mathbb{R}^9 \times \mathcal{P} \subset \mathbb{R}^{9+n_p}$. Note the last $3 + n_p$ states are static with respect to time. As a result, the dynamics of the augmented vehicle state during high-speed and low-speed scenarios can be written as

$$\dot{z}^{\text{aug}}(t) = \begin{cases} \begin{bmatrix} f^{\text{hi}}(t, z^{\text{hi}}(t), p) \\ 0_{(3+n_p)\times 1} \end{bmatrix}, \text{ if } u(t) > u^{\text{cri}}, \\ \\ \begin{bmatrix} f^{\text{lo}}(t, z^{\text{lo}}(t), p) \\ 0_{(5+n_p)\times 1} \end{bmatrix}, \text{ if } u(t) \le u^{\text{cri}}, \end{cases} \tag{3.39}$$

which we refer to as the *hybrid vehicle dynamics model*. Notice when $u(t) \le u^{\text{cri}}$, assigning zero dynamics to v and r in (3.39) does not affect the evolution of the vehicle's dynamics because the lateral speed and yaw rate are directly computed via longitudinal speed as in (3.15) and (3.16).

Because the vehicle's dynamics changes depending on u, it is natural to model the ego vehicle as a hybrid system HS [63, Section 1.2]. The hybrid system has z^{aug} as its state and consists of a high-speed mode and a low-speed mode with dynamics in (3.39). Instantaneous transition between the high and low speed models within HS are described using the notion of a *guard* and *reset map*. The guard triggers a transition and is defined as $\{z^{\text{aug}}(t) \in \mathbb{R}^9 \times \mathcal{P} \mid u(t) = u^{\text{cri}}\}$. Once a transition happens, the reset map maintains the last $3 + n_p$ dimensions of $z^{\text{aug}}(t)$, but resets the first 6 dimensions of $z^{\text{aug}}(t)$ via (3.19) if $u(t)$ approaches u^{cri} from above and via (3.20) if $u(t)$ approaches u^{cri} from below.

We next prove that for desired trajectory defined as in Definition 14 under the controllers defined in Section 3.5, the vehicle model eventually comes to a stop. To begin note that experimentally, we observed that the vehicle quickly comes to a stop during braking once its longitudinal speed becomes $u(t) \le 0.15$[m/s]. Thus we make the following assumption:

Assumption 20. *Suppose* $u(t) = 0.15$ *for some* $t \ge t_{stop}$. *Then under the control inputs* (3.21) *and* (3.27) *and while tracking any desired trajectory as in Definition 14, the ego vehicle takes at most* t_{fstop} *seconds after* t_{stop} *to come to a complete stop.*

We use this assumption to prove that the vehicle can be brought to a stop within a specified amount of time in the following lemma whose proof can be found in Appendix B:

Lemma 21. *Let $\mathcal{Z}_0 \subset \mathbb{R}^6$ be a compact subset of initial conditions for the vehicle dynamic model and $\mathcal{P}$ be a compact set of trajectory parameters. Let $\Delta_u(t)$ be bounded for all t as in Assumptions 11 and 12 with constants M_u, b_u^{pro} and b_u^{off}. Let z^{aug} be a solution to the hybrid vehicle dynamics model* (3.39) *beginning from $z_0 \in \mathcal{Z}_0$ under trajectory parameter $p \in \mathcal{P}$ while applying the control inputs* (3.21) *and* (3.27) *to track some desired trajectory satisfying Definition 14. Assume the desired longitudinal speed satisfies the following properties: $u^{des}(0,p) = u(0)$, $u^{des}(t,p)$ is only discontinuous at time t_{stop}, and $u^{des}(t,p)$ converges to u^{cri} as t converges to t_{stop} from below. If K_u, $\kappa_{1,u}$ and $\phi_{1,u}$ are chosen such that $\frac{M_u}{\kappa_{1,u}M_u+\phi_{1,u}} \in (0.15, u^{cri}]$ and $\frac{(b_u^{off})^2}{4(\kappa_{1,u}M_u+\phi_{1,u}-b_u^{pro})} < 0.15^2 K_u$ hold, then for all $p \in \mathcal{P}$ and $z_0 \in \mathcal{Z}_0$ satisfying $u(0) > 0$, there exists t_{brake} such that $u(t) = 0$ for all $t \geq t_{brake}$.*

Note, the proof of Lemma 21 includes an explicit formula for t_{brake} in (B.19). This lemma is crucial because it specifies the length of time over which we should construct FRS, so that we can verify that not-at-fault behavior can be satisfied based on Definition 15 and Remark 16.

3.6 Computing and Using the FRS

This section describes how REFINE operates at a high-level. It then describes the offline reachability analysis of the ego vehicle as a state-augmented hybrid system using zonotopes and illustrates how the ego vehicle's footprint can be accounted for during reachability analysis.

REFINE conservatively approximates a control-parameterized FRS of the full-order vehicle dynamics. The FRS includes all behaviors of the ego vehicle over a finite time horizon and is mathematically defined in Section 3.6.1. To ensure the FRS is a tight representation, REFINE relies on the controller design described in Section 3.5. Because this controller partially linearizes the dynamics, REFINE relies on a zonotope-based reachable set representation which behave well for nearly linear systems.

During online planning, REFINE performs control synthesis by solving optimization problems in a receding horizon fashion, where the optimization problem computes a trajectory parameter to navigate the ego vehicle to a waypoint while behaving in a not-at-fault manner. As in Assumption 13, each planning iteration in REFINE is allotted $t_{\text{plan}} > 0$ to generate a plan. As depicted in Figure 3.3, if a particular planning iteration begins at time t_0, its goal is to find a control policy by solving an online optimization within t_{plan} seconds so that the control policy can be applied during $[t_0 + t_{\text{plan}}, t_0 + t_{\text{plan}} + t_{\text{f}}]$. Because any trajectory in Definition 14 brings the ego vehicle to a stop, we partition

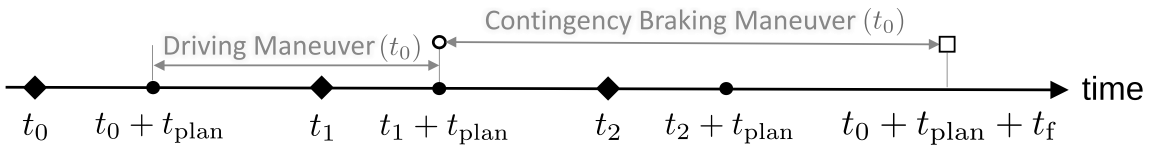


Figure 3.3: An illustration of 3 successive planning/control iterations. t_{plan} seconds are allotted to compute a planned trajectory. Each plan is of duration t_{f} and consists of a driving maneuver of duration t_{m} and a contingency braking maneuver. Diamonds denote the time instances where planning computations begin and $t_2 - t_1 = t_1 - t_0 = t_{\text{m}}$. Filled-in circles denote the instances where feasible driving maneuvers are initiated. If the planning phase between $[t_1, t_1 + t_{\text{plan}}]$ is infeasible, the contingency braking maneuver whose feasibility is verified during the planning phase between $[t_0, t_0 + t_{\text{plan}}]$ is applied.

$[t_0 + t_{\text{plan}}, t_0 + t_{\text{plan}} + t_{\text{f}}]$ into $[t_0 + t_{\text{plan}}, t_0 + t_{\text{plan}} + t_{\text{m}})$ during which a driving maneuver is tracked and $[t_0 + t_{\text{plan}} + t_{\text{m}}, t_0 + t_{\text{plan}} + t_{\text{f}}]$ during which a contingency braking maneuver is activated. Note t_{m} is not necessarily equal to t_{stop}. As a result of Lemma 21, by setting t_{f} equal to t_{brake} one can guarantee that the ego vehicle comes to a complete stop by t_{f}.

If the planning iteration at time t_0 is feasible (i.e., not-at-fault), then the entire feasible planned driving maneuver is applied during $[t_0 + t_{\text{plan}}, t_0 + t_{\text{plan}} + t_{\text{m}})$. Meanwhile another planning iteration will start at time $t_0 + t_{\text{plan}} + t_{\text{m}} - t_{\text{plan}} = t_0 + t_{\text{m}}$, and the same planning procedure repeats. However, if the planning iteration starting at time t_0 is infeasible, then the braking maneuver, whose safe behavior was verified in the previous planning iteration, can be applied starting at $t_0 + t_{\text{plan}}$ to bring the ego vehicle to a stop in a not-at-fault manner. To ensure real-time performance, $t_{\text{plan}} \le t_{\text{m}}$. To simplify notation, we reset time to 0 whenever a feasible control policy is about to be applied, i.e., $t_0 + t_{\text{plan}} \equiv 0$.

3.6.1 Offline FRS Computation

The *Forward Reachable Set (FRS)* of the ego vehicle is defined as

$$\begin{aligned}\mathcal{F}_{xy}([0,t_{\text{f}}]) = \Bigg\{ (x,y) \in \mathcal{W} \,\Bigg|\, & \exists t \in [0,t_{\text{f}}], p \in \mathcal{P}, z_0 = \begin{bmatrix} z_0^{\text{pos}} \\ z_0^{\text{vel}} \end{bmatrix} \in \mathcal{Z}_0 \text{ s.t.} \\ & \begin{bmatrix} x \\ y \end{bmatrix} = \pi_{xy}(z^{\text{aug}}(t)), z^{\text{aug}} \text{ is a solution of } HS \text{ with } z^{\text{aug}}(0) = \begin{bmatrix} z_0 \\ z_0^{\text{vel}} \\ p \end{bmatrix} \Bigg\}, \end{aligned} \tag{3.40}$$

where $\pi_{xy} : \mathbb{R}^{9+n_p} \to \mathbb{R}^2$ is the projection operator that outputs the first two coordinates from its argument. $\mathcal{F}_{xy}([0,t_{\text{f}}])$ collects all possible behavior of the ego vehicle while following the dynamics of HS in the xy-plane over time interval $[0,t_{\text{f}}]$ for all possible $p \in \mathcal{P}$ and initial condition $z_0 \in \mathcal{Z}_0$. Computing $\mathcal{F}_{xy}([0,t_{\text{f}}])$ precisely is numerically challenging because the ego vehicle is modeled as a hybrid system with nonlinear dynamics, thus we aim to compute an outer-approximation of $\mathcal{F}_{xy}([0,t_{\text{f}}])$ instead.

To outer-approximate $\mathcal{F}_{xy}([0,t_{\text{f}}])$, we start by making the following assumption:

Assumption 22. *The initial condition space $\mathcal{Z}_0 = \{0_{3\times1}\}\times\mathcal{Z}_0^{vel}$ where $\mathcal{Z}_0^{vel} = int(\underline{z_0^{vel}}, \overline{z_0^{vel}}) \subset \mathbb{R}^3$ is a 3-dimensional box representing all possible initial velocity conditions z_0^{vel} of the ego vehicle.*

Because vehicles operate within a bounded range of speeds, this assumption is trivial to satisfy. Notice in particular that $\mathcal{Z}_0^{\text{vel}}$ is a zonotope $<c_0^{\text{vel}},\ G_0^{\text{vel}}>$ where $c_0^{\text{vel}} = \frac{1}{2}(\underline{z_0^{\text{vel}}} + \overline{z_0^{\text{vel}}})$ and $G_0^{\text{vel}} = \frac{1}{2}\texttt{diag}(\overline{z_0^{\text{vel}}} - \underline{z_0^{\text{vel}}})$. We assume a zero initial position condition z_0^{pos} in the first three dimensions of $\mathcal{Z}_0$ for simplicity, and nonzero z_0^{pos} can be dealt with via coordinate transformation online as described in Section 3.7.1.

Recall that because $\mathcal{P}$ is a compact n-dimensional box, it can also be represented as a zonotope as $<c_p,\ G_p>$ where $c_p = \frac{1}{2}(\underline{p}+\overline{p})$ and $G_p = \frac{1}{2}\texttt{diag}(\overline{p}-\underline{p})$. Then the set of initial conditions for $z^{\text{aug}}(0)$ can be represented as a zonotope $\mathcal{Z}_0^{\text{aug}} = <c_{z^{\text{aug}}},\ G_{z^{\text{aug}}}> \subset \mathcal{Z}_0\times\mathcal{Z}_0^{\text{vel}}\times\mathcal{P}$ where

$$c_{z^{\text{aug}}} = \begin{bmatrix} 0_{3\times1} \\ c_0^{\text{vel}} \\ c_0^{\text{vel}} \\ c_p \end{bmatrix},\ G_{z^{\text{aug}}} = \begin{bmatrix} 0_{3\times3} & 0_{3\times n_p} \\ G_0^{\text{vel}} & 0_{3\times n_p} \\ G_0^{\text{vel}} & 0_{3\times n_p} \\ 0_{n_p\times3} & G_p \end{bmatrix}. \tag{3.41}$$

Observe that by construction each row of $G_{z^{\text{aug}}}$ has at most 1 nonzero element. Without loss of generality, we assume G_0^{vel} and G_p has no zero rows. If there was a zero row it would mean that the corresponding dimension can only take one value and does not need to be augmented or traced in z^{aug} for reachability analysis.

Next we pick a time step $\Delta_t \in \mathbb{R}_+$ such that $t_{\text{f}}/\Delta_t \in \mathbb{N}$, and partition the time interval $[0,t_{\text{f}}]$ into t_{f}/Δ_t *time segments* as $T_j = [(j-1)\Delta_t, j\Delta_t]$ for each $j \in \mathcal{J} = \{1,2,\cdots,t_{\text{f}}/\Delta_t\}$. Finally we use an open-source toolbox CORA [7], which takes HS and the initial condition space $\mathcal{Z}_0^{\text{aug}}$, to over-approximate the FRS in (3.40) by a collection of zonotopes $\{\mathcal{R}_j\}_{j\in\mathcal{J}}$ over all time intervals where $\mathcal{R}_j \subset \mathbb{R}^{9+n_p}$. As a direct application of Theorem 3.3, Proposition 3.7 and the derivation in Section 3.5.3 in [6], one can conclude the following theorem:

Theorem 23. *Let $\mathcal{R}_j \subset \mathbb{R}^{9+n_p}$ be the zonotopes computed by CORA under the hybrid vehicle dynamics model beginning from $\mathcal{Z}_0^{aug}$. Let z^{aug} be a solution to hybrid system HS starting from an initial condition in $\mathcal{Z}_0^{aug}$. Then $z^{aug}(t) \in \mathcal{R}_j$ for all $j \in \mathcal{J}$ and $t \in T_j$ and*

$$\mathcal{F}_{xy}([0,t_f]) \subset \bigcup_{j\in\mathcal{J}} \pi_{xy}(\mathcal{R}_j). \tag{3.42}$$

Notice in (3.42) we have abused notation by extending the domain of π_{xy} to any zonotope

$\mathcal{Z} = <c,\ G>$ in $\mathbb{R}^{9+n_p}$ as

$$\pi_{xy}(\mathcal{Z}) = \left\langle \begin{bmatrix} [c]_1 \\ [c]_2 \end{bmatrix},\ \begin{bmatrix} [G]_{1:} \\ [G]_{2:} \end{bmatrix} \right\rangle. \tag{3.43}$$

3.6.2 Slicing

The FRS computed in the previous subsection contain the behavior of the hybrid vehicle dynamics model for all initial conditions belonging to $\mathcal{Z}_0$ and $\mathcal{P}$. To use this set during online optimization, REFINE plugs in the predicted initial velocity of the vehicle dynamics at time t_0+t_{plan} and then optimizes over the space of trajectory parameters. Recall the hybrid vehicle model is assumed to have zero initial position condition during the computation of $\{\mathcal{R}_j\}_{j\in\mathcal{J}}$ by Assumption 22. This subsection describes how to plug in the initial velocity into the pre-computed FRS.

We start by describing the following useful property of the zonotopes $\mathcal{R}_j$ that make up the FRS, which follows from Lemma 22 in [43]:

Proposition 24. *Let $\{\mathcal{R}_j = <c_{\mathcal{R}_j},\ G_{\mathcal{R}_j}>\}_{j\in\mathcal{J}}$ be the set of zonotopes computed by CORA under the hybrid vehicle dynamics model beginning from $\mathcal{Z}_0^{aug}$. Then for any $j \in \mathcal{J}$, $G_{\mathcal{R}_j} = [g_{\mathcal{R}_j,1}, g_{\mathcal{R}_j,2}, \ldots, g_{\mathcal{R}_j,\ell_j}]$ has only one generator, $g_{\mathcal{R}_j,b_k}$, that has a nonzero element in the k-th dimension for each $k \in \{7,\ldots,(9+n_p)\}$. In particular, $b_k \neq b_{k'}$ for $k \neq k'$.*

We refer to the generators with a nonzero element in the k-th dimension for each $k \in \{7,\ldots,(9+n_p)\}$ as a *sliceable generator* of $\mathcal{R}_j$ in the k-th dimension. In other words, for each $\mathcal{R}_j = <c_{\mathcal{R}_j},\ G_{\mathcal{R}_j}>$, there are exactly $3+n_p$ nonzero elements in the last $3+n_p$ rows of $G_{\mathcal{R}_j}$, and none of these nonzero elements appear in the same row or column. By construction $\mathcal{Z}_0^{\text{aug}}$ has exactly $3+n_p$ generators, which are each sliceable. Using Proposition 24, one can conclude that $\mathcal{R}_j$ has no less than $3+n_p$ generators (i.e., $\ell \geq 3+n_p$).

Proposition 24 is useful because it allows us to take a known $z_0^{\text{vel}} \in \mathcal{Z}_0^{\text{vel}}$ and $p \in \mathcal{P}$ and plug them into the computed $\{\mathcal{R}_j\}_{j\in\mathcal{J}}$ to generate a *slice* of the conservative approximation of the FRS that includes the evolution of the hybrid vehicle dynamics model beginning from z_0^{vel} under trajectory parameter p. In particular, one can plug the initial velocity into the sliceable generators as described in the following definition:

Definition 25. *Let $\{\mathcal{R}_j = <c_{\mathcal{R}_j},\ G_{\mathcal{R}_j}>\}_{j\in\mathcal{J}}$ be the set of zonotopes computed by CORA under the hybrid vehicle dynamics model beginning from $\mathcal{Z}_0^{aug}$ where*

$$G_{\mathcal{R}_j} = [g_{\mathcal{R}_j,1}, g_{\mathcal{R}_j,2}, \ldots, g_{\mathcal{R}_j,\ell_j}]. \tag{3.44}$$

Without loss of generality, assume that the sliceable generators of each $\mathcal{R}_j$ are the first

$3 + n_p$ *columns of* $G_{\mathcal{R}_j}$. *In addition, without loss of generality assume that the sliceable generators are ordered so that the dimension in which the non-zero element appears is increasing. The* slicing operator $\texttt{slice} : P(\mathbb{R}^{9+n_p}) \times \mathcal{Z}_0^{vel} \times \mathcal{P} \to P(\mathbb{R}^{9+n_p})$ *is defined as*

$$\texttt{slice}(\mathcal{R}_j, z_0^{vel}, p) = {<}c^{slc},\ [g_{\mathcal{R}_j,(4+n_p)}, \ldots, g_{\mathcal{R}_j,\ell_j}]{>} \tag{3.45}$$

where

$$c^{slc} = c_{\mathcal{R}_j} + \sum_{k=7}^{9} \frac{[z_0^{vel}]_{(k-6)} - [c_{\mathcal{R}_j}]_k}{[g_{\mathcal{R}_j,(k-6)}]_k} g_{\mathcal{R}_j,(k-6)} + \sum_{k=10}^{9+n_p} \frac{[p]_{(k-9)} - [c_{\mathcal{R}_j}]_k}{[g_{\mathcal{R}_j,(k-6)}]_k} g_{\mathcal{R}_j,(k-6)}. \tag{3.46}$$

Note, that in the interest of avoiding introducing novel notation, we have abused notation and assumed that the domain of $\texttt{slice}$ is $P(\mathbb{R}^{9+n_p})$ rather than the space of zonotopes in $P(\mathbb{R}^{9+n_p})$. However, throughout this chapter we only plug in zonotopes belonging to $P(\mathbb{R}^{9+n_p})$ into the first argument of $\texttt{slice}$. Using this definition, one can show the following useful property:

Theorem 26. *Let* $\{\mathcal{R}_j\}_{j\in\mathcal{J}}$ *be the set of zonotopes computed by CORA under the hybrid vehicle dynamics model beginning from* $\mathcal{Z}_0^{aug}$ *and satisfy the statement of Definition 25. Then for any* $j \in \mathcal{J}$, $z_0 = [0, 0, 0, (z_0^{vel})^\top]^\top \in \mathcal{Z}_0$, *and* $p \in \mathcal{P}$, $\texttt{slice}(\mathcal{R}_j, z_0^{vel}, p) \subset \mathcal{R}_j$. *In addition, suppose* z^{aug} *is a solution to* HS *with initial condition* z_0 *and control parameter* p. *Then for each* $j \in \mathcal{J}$ *and* $t \in T_j$

$$z^{aug}(t) \in \texttt{slice}(\mathcal{R}_j, z_0^{vel}, p). \tag{3.47}$$

Proof. Because z_0^{vel} and p have zero dynamics in HS, the last $3 + n_p$ dimensions in $\mathcal{R}_j$ are identical to $\mathcal{Z}_0^{\text{vel}} \times \mathcal{P}$ for all $j \in \mathcal{J}$. A direct result of Proposition 24 and Definition 25 is $\mathcal{Z}_0^{\text{vel}} \times \mathcal{P} = {<}c'_j,\ G'_j{>}$ where $c'_j = \left[[c_{\mathcal{R}_j}]_7, [c_{\mathcal{R}_j}]_8, \ldots, [c_{\mathcal{R}_j}]_{(9+n_p)}\right]^\top$ and $G'_j = \texttt{diag}\left(\left[[g_{\mathcal{R}_j,1}]_7, [g_{\mathcal{R}_j,2}]_8, \ldots, [g_{\mathcal{R}_{j,(3+n_p)}}]_{(9+n_p)}\right]\right)$ for any $j \in \mathcal{J}$.

Because $z_0^{\text{vel}} \in \mathcal{Z}_0^{\text{vel}}$ and $p \in \mathcal{P}$, then $\frac{[z_0^{\text{vel}}]_{(k-6)} - [c_{\mathcal{R}_j}]_k}{[g_{\mathcal{R}_j,(k-6)}]_k} \in [-1, 1]$ for all $k \in \{7, 8, 9\}$, and $\frac{[p]_{(k-9)} - [c_{\mathcal{R}_j}]_k}{[g_{\mathcal{R}_j,(k-6)}]_k} \in [-1, 1]$ for all $k \in \{10, 11, \ldots, (9 + n_p)\}$ by Definition 8. $\texttt{slice}(\mathcal{R}, z_0^{\text{vel}}, p)$ is generated by specifying the coefficients of the first $3 + n_p$ generators in $\mathcal{R}_j$ via (3.46), thus $\texttt{slice}(\mathcal{R}_j, z_0^{\text{vel}}, p) \subset \mathcal{R}_j$.

If a solution of HS has initial velocity z_0^{vel} and control parameter p, then the last $3 + n_p$ dimensions in z^{aug} are fixed at $[(z_0^{\text{vel}})^\top, p^\top]^\top$ for all $t \in T_j$ because of (3.39). $\mathcal{R}_j$ is generated from CORA, so $z^{\text{aug}}(t) \in \mathcal{R}_j$ for all $t \in T_j$ by Theorem 23, which proves the result. □

3.6.3 Accounting for the Vehicle Footprint in the FRS

The conservative representation of the FRS generated by CORA only accounts for the ego vehicle's center of mass because HS treats the ego vehicle as a point mass. To ensure not-at-fault behavior while planning using REFINE, one must account for the footprint of the ego vehicle, $\mathcal{O}^{\text{ego}}$, as in Definition 17.

To do this, define a projection operator $\pi_h : \{R_j\}_{j\in\mathcal{J}} \to P(\mathbb{R})$ as

$$\pi_h(R_j) \mapsto <[c_{R_j}]_3,\ [G_{R_j}]_{3:}> \tag{3.48}$$

where $R_j = <c_{R_j},\ G_{R_j}>$ is a zonotope computed by CORA as described in Section 3.6.1. Then by definition $\pi_h(\mathcal{R}_j)$ is a zonotope and it conservatively approximates of the ego vehicle's heading during T_j. Moreover, because $\pi_h(\mathcal{R}_j)$ is a 1-dimensional zonotope, it can be rewritten as a 1-dimensional box $\texttt{int}(h^{\text{mid}} - h^{\text{rad}}, h^{\text{mid}} + h^{\text{rad}})$ where $h^{\text{mid}} = [c_{\mathcal{R}_j}]_3$ and $h^{\text{rad}} = \texttt{sum}(|[G_{\mathcal{R}_j}]_{3:}|)$. We can then use π_h to define a map to account for vehicle footprint within the FRS:

Definition 27. *Let $\mathcal{R}_j$ be the zonotope computed by CORA under the hybrid vehicle dynamics model beginning from $\mathcal{Z}_0^{aug}$ for arbitrary $j \in \mathcal{J}$, and denote $\pi_h(\mathcal{R}_j)$ as $\texttt{int}(h^{mid} - h^{rad}, h^{mid} + h^{rad})$. Let $\mathcal{S} \subset \mathcal{W}$ be a 2-dimensional box centered at the origin with length $\sqrt{L^2 + W^2}$ and width $L|\sin(h^{rad})| + W|\cos(h^{rad})|$. Define the* rotation map $\texttt{rot} : P(\mathbb{R}) \to P(\mathcal{W})$ *as*

$$\texttt{rot}\big(\pi_h(\mathcal{R}_j)\big) = \begin{bmatrix} \cos(h^{mid}) & -\sin(h^{mid}) \\ \sin(h^{mid}) & \cos(h^{mid}) \end{bmatrix} \mathcal{S}. \tag{3.49}$$

Note that in the interest of simplicity, we have abused notation and assumed that the argument to $\texttt{rot}$ is any subset of $\mathbb{R}$. In fact, it must always be a 1-dimensional box. In addition note that $\texttt{rot}\big(\pi_h(\mathcal{R}_j)\big)$ is a zonotope because the 2-dimenional box $\mathcal{S}$ is equivalent to a 2-dimensional zonotope and it is multiplied by a matrix via (3.3).

By applyin geometry, one can verify that by definition $\mathcal{S}$ bounds the area that $\mathcal{O}^{\text{ego}} = \texttt{int}([-0.5L, -0.5W]^\intercal, [0.5L, 0.5W]^\intercal)$ travels through while rotating within $[-h^{\text{rad}}, h^{\text{rad}}]$. As a result, $\texttt{rot}\big(\pi_h(\mathcal{R}_j)\big)$ over-approximates the area over which $\mathcal{O}^{\text{ego}}$ sweeps according to $\pi_h(\mathcal{R}_j)$ as shown in Fig. 3.4. Because $\mathcal{S}$ can be represented as a zonotope with 2 generators, one can denote $\texttt{rot}(\pi_h(\mathcal{R}_j))$ as $<c_{\texttt{rot}},\ G_{\texttt{rot}}> \subset \mathbb{R}^2$ where $G_{\texttt{rot}} \in \mathbb{R}^{2\times 2}$. Notice $\texttt{rot}(\pi_h(\mathcal{R}_j))$ in (3.49) is a set in $\mathcal{W}$ rather than the higher dimensional space where $\mathcal{R}_j$ exists. We extend $\texttt{rot}(\pi_h(\mathcal{R}_j))$ to $\mathbb{R}^{9+n_p}$ as

$$\texttt{ROT}(\pi_h(\mathcal{R}_j)) := \left\langle \begin{bmatrix} c_{\texttt{rot}} \\ 0_{(7+n_p)\times 1} \end{bmatrix},\ \begin{bmatrix} G_{\texttt{rot}} \\ 0_{(7+n_p)\times 2} \end{bmatrix} \right\rangle. \tag{3.50}$$

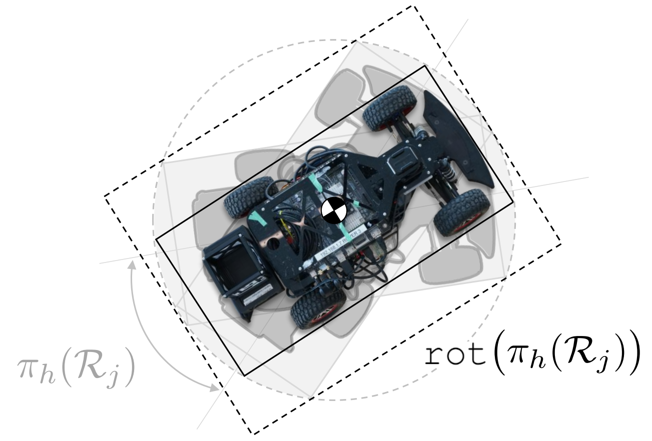


Figure 3.4: Rotation of the ego vehicle and its footprint within range $\pi_h(\mathcal{R}_j)$. The ego vehicle with heading equals to the mean value of $\pi_h(\mathcal{R}_j)$ is bounded by the box with solid black boundaries. The range of rotated heading is indicated by the grey arc. The area the ego vehicle's footprint sweeps is colored in grey, and is bounded by box $\texttt{rot}\big(\pi_h(\mathcal{R}_j)\big)$ with dashed black boundaries.

Using this definition, one can extend the FRS to account for the vehicle footprint as in the following lemma:

Lemma 28. *Let $\{\mathcal{R}_j\}_{j\in\mathcal{J}}$ be the set of zonotopes computed by CORA under the hybrid vehicle dynamics model beginning from $\mathcal{Z}_0^{aug}$. Let z^{aug} be a solution to HS with initial velocity z_0^{vel} and control parameter p and let $\xi : P(\mathbb{R}^{9+n_p}) \times \mathcal{Z}_0^{vel} \times \mathcal{P} \to P(\mathcal{W})$ be defined as*

$$\xi(\mathcal{R}_j, z_0^{vel}, p) = \pi_{xy}\Big(\texttt{slice}\big(\mathcal{R}_j \oplus ROT(\pi_h(\mathcal{R}_j)), z_0^{vel}, p\big)\Big). \tag{3.51}$$

Then $\xi(\mathcal{R}_j, z_0^{vel}, p)$ is a zonotope and for all $j \in \mathcal{J}$ and $t \in T_j$, the vehicle footprint oriented and centered according to $z^{aug}(t)$ is contained within $\xi(\mathcal{R}_j, z_0^{vel}, p)$.

Again note that in the interest of simplicity we have abused notation and assumed that the first argument to ξ is any subset of $\mathbb{R}^{9+n_p}$. This argument is always a zonotope in $\mathbb{R}^{9+n_p}$.

Before proving Lemma 28, we prove the following lemma first:

Lemma 29. *Let $\mathcal{R}_j$ be the zonotope computed by CORA under the hybrid vehicle dynamics model beginning from $\mathcal{Z}_0^{aug}$ for arbitrary $j \in \mathcal{J}$. Then for any $z_0^{vel} \in \mathcal{Z}_0^{vel}$ and $p \in \mathcal{P}$*

$$\texttt{slice}\big(\mathcal{R}_j \oplus ROT(\pi_h(\mathcal{R}_j)), z_0^{vel}, p\big) = ROT(\pi_h(\mathcal{R}_j)) \oplus \texttt{slice}(\mathcal{R}_j, z_0^{vel}, p). \tag{3.52}$$

Proof. Because $\texttt{ROT}(\pi_h(\mathcal{R}_j))$ is independent of z_0^{vel} and p by definition, $\mathcal{R}_j$ shares the same sliceable generators as $\mathcal{R}_j \oplus \texttt{ROT}(\pi_h(\mathcal{R}_j))$. The slice operator only affects sliceable generators, thus (3.52) holds. □

Now we prove Lemma 28:

Proof. By definition $\texttt{slice}(\mathcal{R}_j, z_0^{\text{vel}}, p)$ and $\texttt{ROT}(\pi_h(\mathcal{R}_j))$ are both zonotopes, thus per (3.52) $\texttt{slice}\big(\mathcal{R}_j \oplus \texttt{ROT}(\pi_h(\mathcal{R}_j)), z_0^{\text{vel}}, p\big)$ is a zonotope . For simplicity let $<c'', G''>$ denote $\texttt{slice}\big(\mathcal{R}_j \oplus \texttt{ROT}(\pi_h(\mathcal{R}_j)), z_0^{\text{vel}}, p\big)$, then $\xi(\mathcal{R}_j, z_0^{\text{vel}}, p)$ is a zonotope because

$$\pi_{xy}\big(<c'', G''>\big) = \left\langle \begin{bmatrix} [c'']_1 \\ [c'']_2 \end{bmatrix}, \begin{bmatrix} [G'']_{1:} \\ [G'']_{2:} \end{bmatrix} \right\rangle. \tag{3.53}$$

Note $\pi_{xy}\big(\texttt{ROT}(\pi_h(\mathcal{R}_j))\big) = \texttt{rot}(\pi_h(\mathcal{R}_j))$, and by using the definition of π_{xy} one can check that $\pi_{xy}(\mathcal{A}_1 \oplus \mathcal{A}_2) = \pi_{xy}(\mathcal{A}_1) \oplus \pi_{xy}(\mathcal{A}_2)$ for any zonotopes $\mathcal{A}_1, \mathcal{A}_2 \subset \mathbb{R}^{9+n_p}$. Then by Lemma 29,

$$\xi(\mathcal{R}_j, z_0^{\text{vel}}, p) = \pi_{xy}\big(\texttt{slice}(\mathcal{R}_j, z_0^{\text{vel}}, p)\big) \oplus \texttt{rot}(\pi_h(\mathcal{R}_j)). \tag{3.54}$$

By Theorem 26 for any $t \in T_j$ and $j \in \mathcal{J}$, $z^{\text{aug}}(t) \in \texttt{slice}(\mathcal{R}_j, z_0^{\text{vel}}, p) \subset \mathcal{R}_j$, then $h(t) \in \pi_h(\mathcal{R}_j)$. Because $\texttt{rot}(\pi_h(\mathcal{R}_j))$ by construction outer approximates the area over which $\mathcal{O}^{\text{ego}}$ sweeps according to all possible heading of the ego vehicle during T_j, then $\xi(\mathcal{R}_j, z_0^{\text{vel}}, p)$ contains the vehicle footprint oriented according to $\pi_h(\mathcal{R}_j)$ and centered at $\pi_{xy}(z^{\text{aug}}(t))$ during T_j. $\square$

Remark 30. *Other than $\xi(\mathcal{R}_j, z_0^{vel}, p)$ one can have a tighter embedding of the vehicle footprint in the FRS as*

$$\xi'(\mathcal{R}_j, z_0^{vel}, p) \coloneqq \pi_{xy}\big(\mathit{slice}(\mathcal{R}_j, z_0^{vel}, p)\big) \oplus \mathit{rot}\big(\pi_h(\mathit{slice}(\mathcal{R}_j, z_0^{vel}, p))\big). \tag{3.55}$$

In other words, instead of rotating the vehicle footprint according to all possible headings in $\pi_h(\mathcal{R}_j)$ as $\xi(\mathcal{R}_j, z_0^{vel}, p)$ does, $\xi'(\mathcal{R}_j, z_0^{vel}, p)$ only rotates the vehicle footprint according to vehicle's heading with respect to particular z_0^{vel} and p. However the rotational piece in $\xi'(\mathcal{R}_j, z_0^{vel}, p)$ requires values of z_0^{vel} and p that are only available online, while $\xi(\mathcal{R}_j, z_0^{vel}, p)$ as in (3.51) *allows us to compute $\mathit{ROT}(\pi_h(\mathcal{R}_j))$ completely offline. Thus for computational efficiency in real-time, we use $\xi(\mathcal{R}_j, z_0^{vel}, p)$ to account for the ego vehicle's footprint. Furthermore, ξ results in nearly linear constraints on obstacle avoidance as shown in Section 3.7.3.*

3.7 Online Planning

This section begins by taking nonzero initial position condition into account and formulating the optimization for online planning in REFINE to search for a safety guaranteed control policy in real time. It then explains how to represent each of the constraints of the online optimization problem in a differentiable fashion, and concludes by describing the performance of the online planning loop.

Before continuing we make an assumption regarding predictions of surrounding obstacles. Because prediction is not the primary emphasis of this work, we assume that the future position of any sensed obstacle within the sensor horizon during $[t_0, t_0 + t_{\text{plan}} + t_{\text{f}}]$ is conservatively known at time t_0:

Assumption 31. *There exists a map $\vartheta : \mathcal{J} \times \mathcal{I} \to P(\mathcal{W})$ such that $\vartheta(j,i)$ is a zonotope and*

$$\cup_{t \in T_j} \mathcal{O}_i(t) \cap \mathcal{B}\left((x(t_0), y(t_0)), S\right) \subseteq \vartheta(j,i). \tag{3.56}$$

3.7.1 Nonzero Initial Position

Recall that the FRS computed in Section 3.6 is computed offline while assuming that the initial position of the ego vehicle is zero (i.e., Assumption 22). The zonotope collection $\{\mathcal{R}_j\}_{j \in \mathcal{J}}$ can be understood as a local representation of the FRS in the local frame. This local frame is oriented at the ego vehicle's location $[x_0, y_0]^\top \in \mathbb{R}^2$ with its x-axis aligned according to the ego vehicle's heading $h_0 \in \mathbb{R}$, where $z_0^{\text{pos}} = [x_0, y_0, h_0]^\top$ gives the ego vehicle's position $[x(t), y(t), h(t)]^\top$ at time $t = 0$ in the world frame. Similarly, $\xi(\mathcal{R}_j, z_0^{\text{vel}}, p)$ is a local representation of the area that the ego vehicle may occupy during T_j in the same local frame.

Because obstacles are defined in the world frame, to generate not-at-fault trajectories, one has to either transfer $\xi(\mathcal{R}_j, z_0^{\text{vel}}, p)$ from the local frame to the world frame, or transfer the obstacle position $\vartheta(j,i)$ from the world frame to the local frame using a 2D rigid body transformation. This work utilizes the second option and transforms $\vartheta(j,i)$ into the local frame as

$$\vartheta^{\text{loc}}(j, i, z_0^{\text{pos}}) = \begin{bmatrix} \cos(h_0) & \sin(h_0) \\ -\sin(h_0) & \cos(h_0) \end{bmatrix} \left(\vartheta(j,i) - \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \right). \tag{3.57}$$

3.7.2 Online Optimization

Given the predicted initial condition of the vehicle at $t = 0$ as $z_0 = [(z_0^{\text{pos}})^\top, (z_0^{\text{vel}})^\top]^\top \in \mathbb{R}^3 \times \mathcal{Z}_0^{\text{vel}}$, REFINE computes a not-at-fault trajectory by solving the following optimization problem at each planning iteration:

$$\begin{aligned} \min_{p\in\mathcal{P}} \quad & \texttt{cost}(z_0, p) && (\texttt{Opt}) \\ \text{s.t.} \quad & \xi(\mathcal{R}_j, z_0^{\text{vel}}, p) \cap \vartheta^{\text{loc}}(j, i, z_0^{\text{pos}}) = \varnothing, && \forall j \in \mathcal{J}, \forall i \in \mathcal{I} \end{aligned}$$

where $\texttt{cost} : \mathbb{R}^3 \times \mathcal{Z}_0^{\text{vel}} \times \mathcal{P} \to \mathbb{R}$ is a user-specified cost function and ξ is defined as in Lemma 28. Note that the constraint in (`Opt`) is satisfied if for a particular trajectory parameter p, there is no intersection between any obstacle and the reachable set of the ego vehicle with its footprint considered during any time interval while following p.

3.7.3 Representing the Constraint and its Gradient in (Opt)

The following theorem, whose proof can be found in Appendix C, describes how to represent the set intersection constraint in (`Opt`) and how to compute its derivative with respect to $p \in \mathcal{P}$:

Theorem 32. *There exists matrices A and B and a vector b such that $\xi(\mathcal{R}_j, z_0^{vel}, p) \cap \vartheta^{loc}(j, i, z_0^{pos}) = \varnothing$ if and only if $\max(BA \cdot p - b) > 0$. In addition, the subgradient of $\max(BA \cdot p - b)$ with respect to p is $\max_{k\in\hat{K}}[BA]_{k:}$, where $\hat{K} = \{k \mid [BA \cdot p - b]_k = \max(BA \cdot p - b)\}$.*

Formulas for the matrices A and B and vector b in the previous theorem can be found in (C.4), (C.7), and (C.8), respectively.

3.7.4 Online Operation

Algorithm 1 summarizes the online operations of REFINE. In each planning iteration, the ego vehicle executes the feasible control parameter that is computed in the previous planning iteration (Line 4). Meanwhile, `SenseObstacles` senses and predicts obstacles as in Assumption 31 (Line 5) in local frame decided by z_0^{pos}. (`Opt`) is then solved to compute a control parameter p^* using z_0 and $\{\vartheta^{\text{loc}}(j, i, z_0^{\text{pos}})\}_{(j,i)\in\mathcal{J}\times\mathcal{I}}$ (Line 6). If (`Opt`) fails to find a feasible solution within t_{plan}, the contingency braking maneuver whose safety is verified in the last planning iteration is executed, and REFINE is terminated (Line 7). In the case when (`Opt`) is able to find a feasible p^*, `StatePrediction` predicts the state value at $t = t_{\text{m}}$ based on z_0 and p^* as in Assumption 13 (Lines 8 and 9). If the predicted velocity value does not belong to $\mathcal{Z}_0^{\text{vel}}$, then its corresponding FRS is not available and the planning has to stop while executing a braking maneuver (Line 10). Otherwise we reset time to 0 (Line 11) and start the next planning iteration. Note Lines 4 and 7 are assumed to execute instantaneously, but in practice the time spent for these steps can be subtracted from t_{plan} to ensure real-time performance. By iteratively applying Definition 15, Lemmas 21 and 28, Assumption 31 and (3.57), the following theorem holds:

Theorem 33. *Suppose the ego vehicle can sense and predict surrounding obstacles as in Assumption 31, and starts with a not-at-fault control parameter $p_0 \in \mathcal{P}$. Then by performing planning and control as in Algorithm 1, the ego vehicle is not-at-fault for all time.*

Algorithm 1 REFINE Online Planning

```
Require: $p_0 \in \mathcal{P}$ and $z_0 = [(z_0^{\text{pos}})^\top, (z_0^{\text{vel}})^\top]^\top \in \mathbb{R}^3 \times \mathcal{Z}_0^{\text{vel}}$
    Initialize: $p^* = p_0$, $t = 0$
    Loop: // Line 3 executes at the same time as Line 4-8
        Execute $p^*$ during $[0, t_{\text{m}})$
        $\{\vartheta^{\text{loc}}(j, i, z_0^{\text{pos}})\}_{(j,i) \in \mathcal{J} \times \mathcal{I}} \leftarrow$ SenseObstacles()
        Try $p^* \leftarrow$ OnlineOpt$(z_0, \{\vartheta^{\text{loc}}(j, i, z_0^{\text{pos}})\}_{(j,i) \in \mathcal{J} \times \mathcal{I}})$    // within $t_{plan}$ seconds
        Catch execute $p^*$ during $[t_{\text{m}}, t_{\text{f}}]$, then break
        $(z_0^{\text{pos}}, z_0^{\text{vel}}) \leftarrow$ StatePrediction$(z_0, p^*, t_{\text{m}})$
        $z_0 \leftarrow [(z_0^{\text{pos}})^\top, (z_0^{\text{vel}})^\top]^\top$
        If $(z_0^{\text{vel}} \notin \mathcal{Z}_0^{\text{vel}})$, execute $p^*$ during $[t_{\text{m}}, t_{\text{f}}]$ and break
        Reset $t$ to 0
    End
```

3.8 Extensions

This section describes how to extend various components of REFINE. This section begins by describing how to apply CORA to compute tight, conservative approximations of the FRS. Next, it illustrates how to verify the satisfaction of Assumption 10. The section concludes by describing how to apply REFINE to AWD and RWD vehicles.

3.8.1 Subdivision of Initial Set and Families of Desired Trajectories

In practice, CORA may generate overly conservative representations for the FRS if the initial condition set is large. To address this challenge, one can instead partition $\mathcal{Z}_0$ and $\mathcal{P}$ and compute a FRS beginning from each element in this partition. Note one could then still apply REFINE as in Algorithm 1. However in Line 5 must solve multiple optimizations of the form (Opt) in parallel. Each of these optimizations optimizes over a unique partition element that contains initial condition z_0, then p^* is set to be the feasible control parameter that achieves the minimum cost function value among these optimizations. Similarly note if one had multiple classes of desired trajectories (e.g. lane change, longitudinal speed changes, etc.) that were each parameterized in distinct ways, then one could extend REFINE just as in the instance of having a partition of the initial condition set. In this way one could apply REFINE to optimize over multiple families of desired trajectories to generate not-at-fault behavior. Note, that the planning horizon t_f is constant within each element of the partition, but can vary between different elements in the partition.

3.8.2 Satisfaction of Assumption 10

Throughout our analysis thus far, we assume that the slip ratios and slip angles stay within the linear regime as described in Assumption 10. This subsection describes how to ensure that Assumption 10 is satisfied by performing an offline verification on the computed reachable sets.

Recall that in an FWD vehicle model, $F_\text{xr}(t) = 0$ for all t as in Remark 9. By plugging (3.12) in (3.21), one can derive:

$$\lambda_\text{f}(t) = \frac{l}{g l_\text{r} \bar{\mu}}\big(- K_u u(t) + K_u u^\text{des}(t,p) + \dot{u}^\text{des}(t,p) - v(t)r(t) + \tau_u(t,p)\big). \tag{3.58}$$

Similarly by plugging (3.13) in (3.27) one can derive:

$$\begin{aligned} \alpha_\text{f}(t) = &- \frac{I_\text{zz}K_r}{l_\text{f}\bar{c}_{\alpha\text{f}}}\big(r(t) - r^\text{des}(t,p)\big) - \frac{I_\text{zz}K_h}{l_\text{f}\bar{c}_{\alpha\text{f}}}\big(h(t) - h^\text{des}(t,p)\big) + \\ &+ \frac{I_\text{zz}}{l_\text{f}\bar{c}_{\alpha\text{f}}}\dot{r}^\text{des}(t,p) + \frac{l_\text{r}}{l_\text{f}\bar{c}_{\alpha\text{f}}}F_\text{yr}(t) + \frac{I_\text{zz}}{l_\text{f}\bar{c}_{\alpha\text{f}}}\tau_r(t,p). \end{aligned} \tag{3.59}$$

If the slip ratio and slip angle computed in (3.58) and (3.59) satisfy Assumption 10, they achieve the expected tire forces as introduced in Section 3.5.1.

By Definition 8 any $\mathcal{R}_j = <c_{\mathcal{R}_j},\ G_{\mathcal{R}_j}>$ that is computed by CORA under the hybrid vehicle dynamics model from a partition element in Section 3.8.1, can be bounded by a multi-dimensional box $\texttt{int}(c_{\mathcal{R}_j} - |G_{\mathcal{R}_j}|\cdot\mathbf{1}, c_{\mathcal{R}_j} + |G_{\mathcal{R}_j}|\cdot\mathbf{1})$ where $\mathbf{1}$ is a column vector of ones. This multi-dimensional box gives interval ranges of all elements in z^aug during T_j, which allows us to conservatively estimate $\{|\alpha_\text{r}(t)|\}_{t\in T_j}$, $\{\mathcal{F}_\text{yr}(t)\}_{t\in T_j}$ and $\{|\lambda_\text{f}(t)|\}_{t\in T_j}$ via (3.9), (3.13) and (3.58) respectively using Interval Arithmetic [42]. The approximation of $\{\mathcal{F}_\text{yr}(t)\}_{t\in T_j}$ makes it possible to over-approximate $\{|\alpha_\text{f}(t)|\}_{t\in T_j}$ via (3.59).

Note in (3.58) and (3.59) integral terms are embedded in $\tau_u(t,p)$ and $\tau_r(t,p)$ as described in (3.22) and (3.28). Because it is nontrivial to perform Interval Arithmetic over integrals, we extend z^{aug} to $z^{\text{aug+}}$ by appending three more auxiliary states $\varepsilon_u(t) := \int_{t_0}^{t} \|u(s) - u^{\text{des}}(s,p)\|^2 ds$, $\varepsilon_r(t) := \int_{t_0}^{t} \|r(s) - r^{\text{des}}(s,p)\|^2 ds$ and $\varepsilon_h(t) := \int_{t_0}^{t} \|h(s) - h^{\text{des}}(s,p)\|^2 ds$. Notice

$$\begin{bmatrix} \dot{\varepsilon}_u(t) \\ \dot{\varepsilon}_r(t) \\ \dot{\varepsilon}_h(t) \end{bmatrix} = \begin{bmatrix} \|u(t) - u^{\text{des}}(t,p)\|^2 \\ \|r(t) - r^{\text{des}}(t,p)\|^2 \\ \|h(t) - h^{\text{des}}(t,p)\|^2 \end{bmatrix}, \tag{3.60}$$

then we can compute a higher-dimensional FRS of $z^{\text{aug+}}$ during $[0, t_{\text{f}}]$ through the same process as described in Section 3.6. This higher-dimensional FRS makes over-approximations of $\{\varepsilon_u(t)\}_{t\in\mathcal{T}_j}$, $\{\varepsilon_r(t)\}_{t\in\mathcal{T}_j}$ and $\{\varepsilon_h(t)\}_{t\in\mathcal{T}_j}$ available for computation in (3.58) and (3.59).

If the supremum of $\{|\lambda_{\text{f}}(t)|\}_{t\in T_j}$ exceeds λ^{cri} or any supremum of $\{|\alpha_{\text{f}}(t)|\}_{t\in T_j}$ and $\{|\alpha_{\text{r}}(t)|\}_{t\in T_j}$ exceeds α^{cri}, then the corresponding partition section of $\mathcal{Z}_0 \times \mathcal{P}$ may result in a system trajectory that violates Assumption 10. Therefore to ensure not-at-fault, we only run optimization over partition elements whose FRS outer-approximations satisfy Assumption 10. Finally we emphasize that such verification of Assumption 10 over each partition element that is described in Section 3.8.1 can be done offline.

3.8.3 Generalization to All-Wheel-Drive and Rear-Wheel-Drive

This subsection describes how REFINE can be extended to AWD and RWD vehicles. AWD vehicles share the same dynamics as (3.14) in Section 3.3 with one exception. In an AWD vehicle, only the lateral rear tire force is estimated and all the other three tire forces are controlled by using wheel speed and steering angle. In particular, computations related to the lateral tire forces as (3.27) and (3.59) are identical to the FWD case . However, both the front and rear tires contribute nonzero longitudinal forces, and they can be specified by solving the following system of linear equations:

$$\begin{aligned} l_{\text{f}} F_{\text{xf}}(t) &= l_{\text{r}} F_{\text{xr}}(t) \\ F_{\text{xf}}(t) + F_{\text{xr}}(t) &= -mK_u u(t) + mK_u u^{\text{des}}(t,p) + \\ &\quad + m\dot{u}^{\text{des}}(t,p) - mv(t)r(t) + m\tau_u(t,p) \end{aligned} \tag{3.61}$$

Longitudinal tire forces $F_{\text{xf}}(t)$ and $F_{\text{xr}}(t)$ computed from (3.61) can then be used to compute wheel speed $\omega_{\text{f}}(t) = \omega_{\text{r}}(t)$ as in (3.36). In this formulation, (3.58) also needs to be modified to

$$\lambda_{\text{f}}(t) = \lambda_{\text{r}}(t) = \frac{1}{g\bar{\mu}}\big(- K_u u(t) + K_u u^{\text{des}}(t,p) + \dot{u}^{\text{des}}(t,p) - v(t)r(t) + \tau_u(t,p)\big) \tag{3.62}$$

to verify Assumption 10 along the longitudinal direction. Compared to FWD, in RWD the longitudinal front tire force is 0 and the longitudinal rear tire force is controlled. Thus one can generalize to RWD by switching all related computations on $F_{\text{xf}}(t)$ and $F_{\text{xr}}(t)$ from the FWD case.

3.9 Experiments

This section describes the implementation and evaluation of REFINE in simulation using a FWD, full-size vehicle model and on hardware using an AWD, $\frac{1}{10}$th size race car model. Readers can find a link to the software implementation[2] and videos[3] online.

3.9.1 Desired Trajectories

As detailed in Section 3.5.1, the proposed controller relies on desired trajectories of vehicle longitudinal speed and yaw rate satisfying Definition 14. As detailed in Section 3.5.1, the proposed controller relies on desired trajectories of vehicle longitudinal speed and yaw rate satisfying Definition 14. To test the performance of the proposed controller and planning framework, we selected 3 families of desired trajectories that are observed during daily driving. Each desired trajectory is the concatenation of a driving maneuver and a contingency braking maneuver. The driving maneuver is either a *speed change*, *direction change*, or *lane change* (i.e. each option corresponds to one of the 3 families of desired trajectories). Moreover, each desired trajectory is parameterized by $p = [p_u, p_y]^\top \in \mathcal{P} \subset \mathbb{R}^2$ where p_u denotes desired longitudinal speed, and p_y decides desired lateral displacement. Recall the duration of a driving maneuver is t_{m}, which can vary according to the maneuver type, and t_{stop} is the time from when the reference trajectory of longitudinal speed remains 0.

Assuming that the ego vehicle has initial longitudinal speed $u_0 \in \mathbb{R}$ at time 0, the desired trajectory for longitudinal speed is the same for each of the 3 families of desired trajectories:

$$u^{\text{des}}(t,p) = \begin{cases} u_0 + \frac{p_u - u_0}{t_{\text{m}}} t, & \text{if } 0 < t < t_{\text{m}} \\ u^{\text{brake}}(t,p), & \text{if } t \geq t_{\text{m}} \end{cases} \tag{3.63}$$

[2]https://github.com/roahmlab/REFINE

[3]https://drive.google.com/drive/folders/1bXl07gTnaA3rJBl7J05SL0tsfIJEDfKy?usp=sharing, https://drive.google.com/drive/folders/1FvGHuqIRQpDS5xWRgB30h7exmGTjRyel?usp=sharing

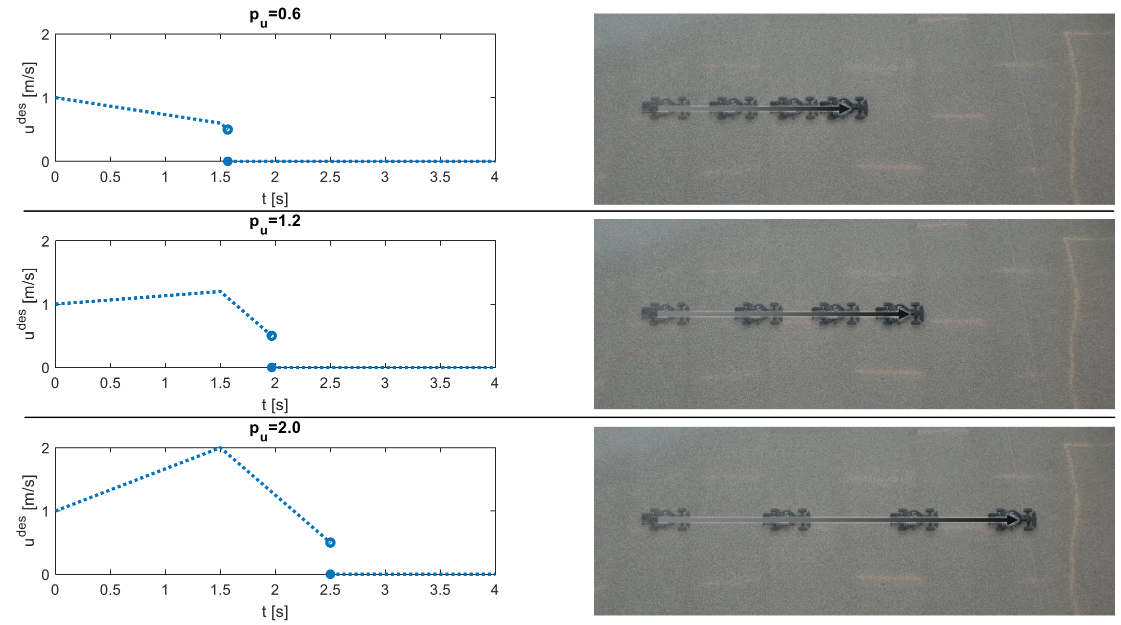


Figure 3.5: Examples of $u^{\text{des}}(t, p)$ with $u_0 = 1.0$ [m/s], $u^{\text{cri}} = 0.5$[m/s], $t_{\text{m}} = 1.5$[s], $a^{\text{dec}} = -1.5$[m/s^2], and p_u taking values of 0.6, 1.2 and 2.0 from top to bottom. Note zero lateral controls are commanded among all 3 examples.

where

$$u^{\text{brake}}(t, p) = \begin{cases} p_u + (t - t_{\text{m}})a^{\text{dec}}, & \text{if } p_u > u^{\text{cri}} \text{ and } t_{\text{m}} \le t < t_{\text{m}} + \frac{u^{\text{cri}} - p_u}{a^{\text{dec}}} \\ 0, & \text{if } p_u > u^{\text{cri}} \text{ and } t \ge t_{\text{m}} + \frac{u^{\text{cri}} - p_u}{a^{\text{dec}}} \\ 0, & \text{if } p_u \le u^{\text{cri}} \text{ and } t \ge t_{\text{m}} \end{cases} \tag{3.64}$$

with some deceleration $a^{\text{dec}} < 0$. Note by Definition 14, t_{stop} can be specified as

$$t_{\text{stop}} = \begin{cases} t_{\text{m}} + \frac{u^{\text{cri}} - p_u}{a^{\text{dec}}}, & \text{if } p_u > u^{\text{cri}} \\ t_{\text{m}}, & \text{if } p_u \le u^{\text{cri}}. \end{cases} \tag{3.65}$$

As shown in Figure 3.5, desired longitudinal speed approaches p_u linearly from u_0 before braking begins at time t_{m}, then decreases to u^{cri} with deceleration a^{dec} and immediately drops down to 0 at time t_{stop}. Moreover, one can verify that the chosen $u^{\text{des}}(t, p)$ in (3.63) satisfies the assumptions on desired longitudinal speed in Lemma 21.

Assuming the ego vehicle has initial heading $h_0 \in [-\pi, \pi]$ at time 0, the desired heading trajectory varies among the different trajectory families. Specifically, for the trajectory

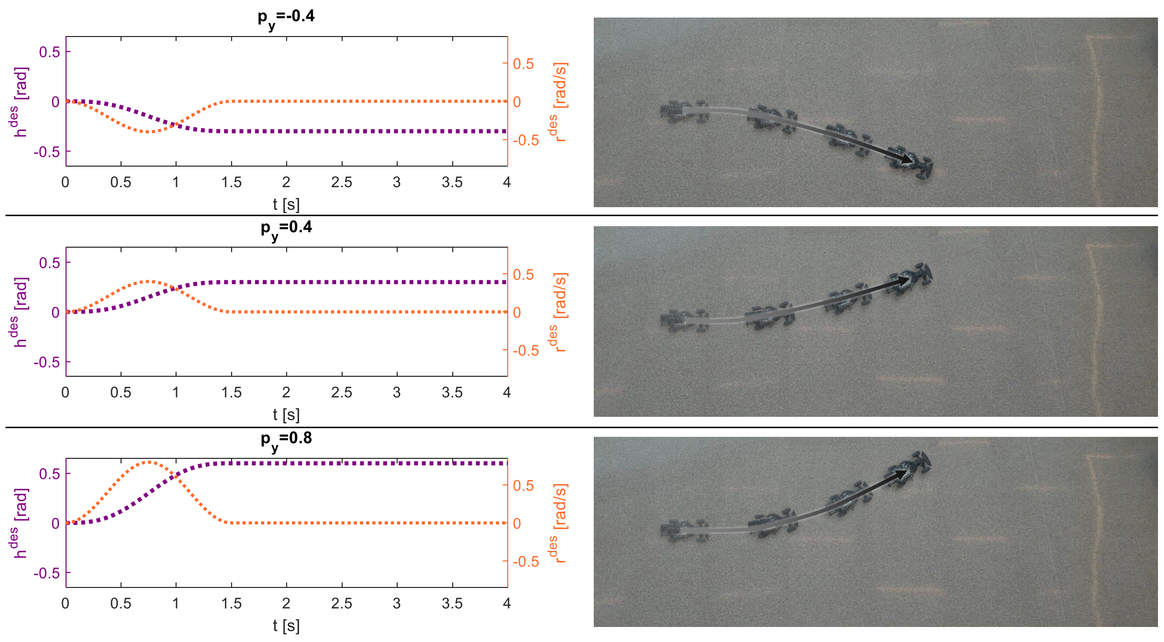


Figure 3.6: Examples of $h^{\text{des}}(t,p)$ and $r^{\text{des}}(t,p)$ to achieve direction changes with $u_0 = 1.0$[m/s], $t_{\text{m}} = 1.5$[s], $h_1^{\text{des}} = \frac{20}{27}$, $h_2^{\text{des}} = \frac{27}{10}$, and p_y taking values of -0.4, 0.4 and 0.8 from top to bottom. Note p_u is set as u_0 to maintain the vehicle longitudinal speed before t_{m} among all 3 examples.

family associated with speed change:

$$h^{\text{des}}(t,p) = h_0, \;\; \forall t \geq 0. \tag{3.66}$$

Desired heading trajectory for the trajectory family associated with direction change:

$$h^{\text{des}}(t,p) = \begin{cases} h_0 + \frac{p_y t}{2} - \frac{p_y t_{\text{m}}}{4\pi}\sin\left(\frac{2\pi t}{t_{\text{m}}}\right), & \text{if } 0 \leq t < t_{\text{m}} \\ h_0 + \frac{p_y t_{\text{m}}}{2}, & \text{if } t \geq t_{\text{m}} \end{cases} \tag{3.67}$$

and for the trajectory family associated with lane change:

$$h^{\text{des}}(t,p) = \begin{cases} h_0 + h_1^{\text{des}} p_y \cdot \mathrm{e}^{-h_2^{\text{des}}(t-0.5t_{\text{m}})^2}, & \text{if } 0 \leq t < t_{\text{m}} \\ h_0, & \text{if } t \geq t_{\text{m}} \end{cases} \tag{3.68}$$

where e is Euler's number, and h_1^{des} and h_2^{des} are user-specified auxiliary constants that adjust the desired heading amplitude. As shown in Figure 3.6 and 3.7, $h^{\text{des}}(t,p)$ remains constant for all $t \geq t_{\text{m}}$ among all families of desired trajectories. By Definition 14, desired trajectory of yaw rate is set as $r^{\text{des}}(t,p) = \frac{d}{dt}h^{\text{des}}(t,p)$ among all trajectory families.

In this work, t_{m} for the speed change and direction change trajectory families are set

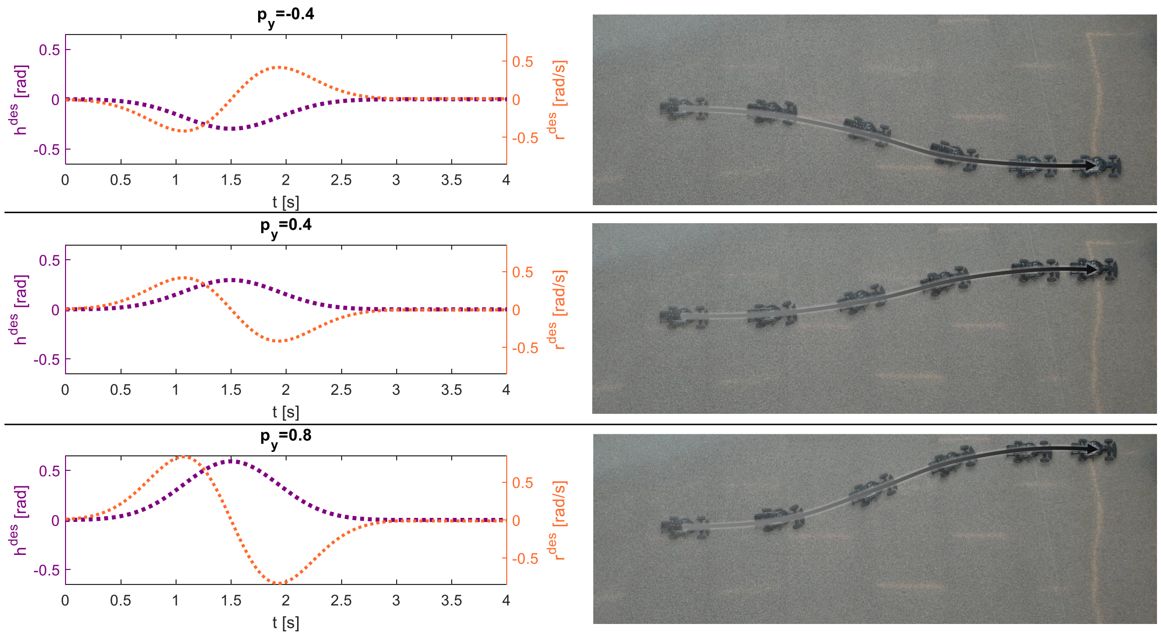


Figure 3.7: Examples of $h^{\text{des}}(t,p)$ and $r^{\text{des}}(t,p)$ to achieve lane changes with $u_0 = 1.0$[m/s], $t_{\text{m}} = 3.0$[s], $h_1^{\text{des}} = \frac{20}{27}$, $h_2^{\text{des}} = \frac{27}{10}$, and p_y taking values of -0.4, 0.4 and 0.8 from top to bottom. Note p_u is set as u_0 to maintain the vehicle longitudinal speed before t_{m} among all 3 examples.

equal to one another. t_{m} for the lane change trajectory family is twice what it is for the direction change and speed change trajectory families. This is because a lane change can be treated as a concatenation of two direction changes. Because we do not know which desired trajectory ensures not-at-fault *a* priori, during each planning iteration, to guarantee real-time performance, t_{plan} should be no greater than the smallest duration of a driving maneuver, i.e. speed change or direction change.

3.9.2 Simulation on a FWD Model

This subsection describes the evaluation of REFINE in simulation. In particular, this section describes the simulation environment, how we implement REFINE, the methods we compare it to, and the results of the evaluation.

3.9.2.1 Simulation Environment

We evaluate the performance on 1000 randomly generated 3-lane highway scenarios in which the same full-size, FWD vehicle as the ego vehicle is expected to autonomously navigate through dynamic traffic for 1[km] from a fixed initial condition. All lanes of all highway scenario share the same lane width as 3.7[m]. Each highway scenario contains

Vehicle Parameter (Simulation)		REFINE Controller Parameter (Simulation)	
m	1575[kg]	K_u	4.0
l_f	1.13[m]	K_r	2.0
l_r	1.67[m]	K_h	5.0
I_zz	3273[kg·m^2]	$\kappa_{1,u}$	1.3
r_w	0.33[m]	$\kappa_{2,u}$	0.7
λ^cric	0.15	$\phi_{1,u}$	1.3
α^cric	0.1[rad]	$\phi_{2,u}$	0.7
$\bar{\mu}$	10	$\kappa_{1,r}$	0.5
$\bar{c}_{\alpha\text{f}}$	1.72e5[N/rad]	$\kappa_{2,r}$	1.0
$\bar{c}_{\alpha\text{r}}$	2.90e5[N/rad]	$\phi_{1,r}$	4.0
u^cri	5[m/s]	$\phi_{2,r}$	1.0
L	4.8[m]	M_u	0.25
W	2.2[m]	b_u^pro	0.05
		b_u^off	0
		M_r	0.01

Table 3.1: Vehicle specification for simulation.

up to 24 moving vehicles and up to 5 static vehicles that start from random locations and are all treated as obstacles to the ego vehicle. Moreover, each moving obstacle maintains its randomly generated highway lane and initial speed up to 25[m/s] for all time. Because each highway scenario is randomly generated, there is no guarantee that the ego vehicle has a path to navigate itself from the start to the goal. Such cases allow us to verify if the tested methods can still keep the ego vehicle safe even in infeasible scenarios. Parameters of the ego vehicle are listed in Table 3.1.

During each planning iteration, all evaluated methods use the same high level planner. This high level planner generates waypoints by first choosing the lane on which the nearest obstacle ahead has the largest distance from the ego vehicle. Subsequently it picks a waypoint that is ahead of the ego vehicle and stays along the center line of the chosen lane. The cost function in (`Opt`) or in any of the evaluated optimization-based motion planning algorithms is set to be the Euclidean distance between the waypoint generated by the high level planner and the predicted vehicle location based on initial state z_0 and decision variable p. All simulations are implemented and evaluated in MATLAB R2022a on a laptop with an Intel i7-9750H processor and 16GB of RAM.

3.9.2.2 REFINE Simulation Implementation

REFINE invokes C++ for the online optimization via IPOPT [110]. Parameters of REFINE's controller are chosen to satisfy the conditions in Lemma 21 and can be found in Table 3.1. REFINE tracks families of desired trajectories as described in Section 3.9.1 with $\mathcal{P} = \{(p_u, p_y) \in [5, 30] \times [-0.8, 0.8] \mid p_u = u_0 \text{ if } p_y \neq 0\}$, $a^{\text{dec}} = -5.0[\text{m/s}^2]$, $h_1^{\text{des}} = \frac{6\sqrt{2e}}{11}$ and $h_2^{\text{des}} = \frac{121}{144}$. The duration t_{m} of driving maneuvers for each trajectory family is 3[s] for speed change, 3[s] for direction change and 6[s] for lane change, therefore t_{plan} is set to be 3[s]. As discussed in Section 3.8.1, during offline computation, we evenly partition the first and second dimensions of $\mathcal{P}$ into intervals of lengths 0.5 and 0.4, respectively. For each partition element, t_{f} is assigned to be the maximum possible value of t_{brake} as computed in (B.19) in which t_{fstop} is by observation no greater than 0.1[s]. An outer-approximation of the FRS is computed for every partition element of $\mathcal{P}$ using CORA with Δt as 0.015[s], 0.010[s], 0.005[s] and 0.001[s]. Note, that we choose these different values of Δt to highlight how this choice affects the performance of REFINE.

3.9.2.3 Other Implemented Methods

We compare REFINE against several state of the art trajectory planning methods: a baseline zonotope reachable set method [69], a Sum-of-Squares-based RTD (SOS-RTD) method [52], and an NMPC method using GPOPS-II [81].

The first trajectory planning method that we implement is a baseline zonotope based reachability method that selects a finite number of possible trajectories rather than a continuum of possible trajectories as REFINE does. This baseline method is similar to the classic funnel library approach to motion planning [66] in that it chooses a finite number of possible trajectories to track. The baseline method computes zonotope reachable sets using CORA with $\Delta t = 0.010$[s] over a sparse discrete control parameter space $\mathcal{P}^{\text{sparse}} := \{(p_u, p_y) \in \{5, 5.1, 5.2, \ldots, 30\} \times \{0, 0.4\} \mid p_u = u_0 \text{ if } p_y \neq 0\}$ and a dense discrete control parameter space $\mathcal{P}^{\text{dense}} := \{(p_u, p_y) \in \{5, 5.1, 5.2, \ldots, 30\} \times \{0, 0.04, 0.08, \ldots, 0.8\} \mid p_u = u_0 \text{ if } p_y \neq 0\}$. We use $\mathcal{P}^{\text{sparse}}$ and $\mathcal{P}^{\text{dense}}$ to illustrate the challenges associated with applying this baseline method in terms of computation time, memory consumption, and the ability to robustly travel through complex simulation environments. During each planning iteration, the baseline method searches through the discrete control parameter space until a feasible solution is found such that the corresponding zonotope reachable sets have no intersection with any obstacles over the planning horizon. The search procedure over this discrete control space is biased to select the same trajectory parameter that worked in the prior planning iteration or to search first from trajectory parameter that are close to one that worked in the previous planning iteration.

The SOS-RTD plans a controller that also tracks families of trajectories to achieve speed change, direction change and lane change maneuvers with braking maneuvers as described in Section 3.9.1. SOS-RTD offline approximates the FRS by solving a series of polynomial optimizations using Sum-of-Squares so that the FRS can be over-approximated as a union of superlevel sets of polynomials over successive time intervals of duration 0.1[s] [52]. Computed polynomial FRS are further expanded to account for footprints of other vehicles offline in order to avoid buffering each obstacle with discrete points online [106]. During online optimization, SOS-RTD plans every 3[s] and uses the same cost function as REFINE does, but checks collision against obstacles by enforcing that no obstacle has its center stay inside the FRS approximation during any time interval.

The NMPC method does not perform offline reachability analysis. Instead, it directly computes the control inputs that are applicable for t_{m} seconds by solving an optimal control problem. This optimal control problem is solved using GPOPS-II in a receding horizon fashion. The NMPC method conservatively ensures collision-free trajectories by covering the footprints of the ego vehicle and all obstacles with two overlapping balls, and requiring that no ball of the ego vehicle intersects with any ball of any obstacle. Notice during each online planning iteration, the NMPC method does not need pre-defined desired trajectories for solving control inputs. Moreover, it does not require the planned control inputs to stop the vehicle by the end of planned horizon as the other three methods do.

3.9.2.4 Evaluation Criteria

We evaluate each implemented trajectory planning method in several ways as summarized in Table 3.2. First, we report the percentage of times that each planning method either came safely to a stop (in a not-at-fault manner), crashed, or successfully navigated through the scenario. Note a scenario is terminated when one of those three conditions is satisfied. Second, we report the average travel speed during all scenarios. Third, we report the average and maximum planning time over all scenarios. Finally, we report on the size of the pre-computed reachable set.

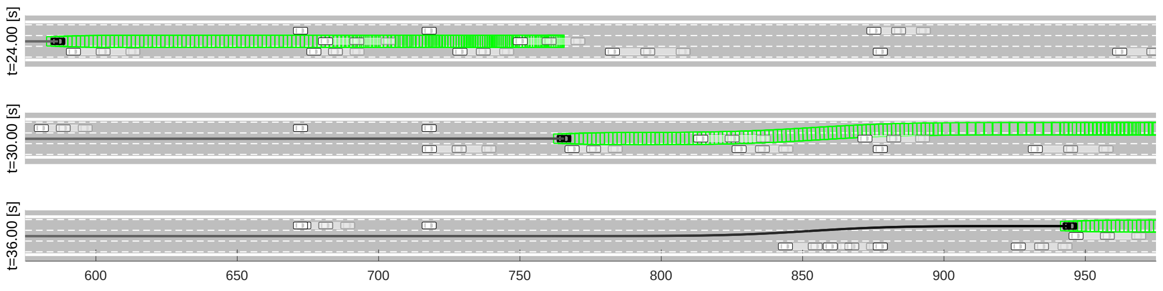
3.9.2.5 Results

REFINE achieves the highest success rate among all evaluated methods and has no crashes. The success rate of REFINE converges to 84% as the value of Δt decreases because the FRS approximation becomes tighter with denser time discretization. However as the time discretization becomes finer, memory consumption grows larger because more zonotopes are used to over-approximate FRS. Furthermore, due to the increasing number of zonotope reachable sets, the solving time also increases and begins to exceed the allotted planning time. According to our simulation, we see that $\Delta t = 0.010$[s] results in high enough successful rate while maintaining a planning time no greater than 3[s].

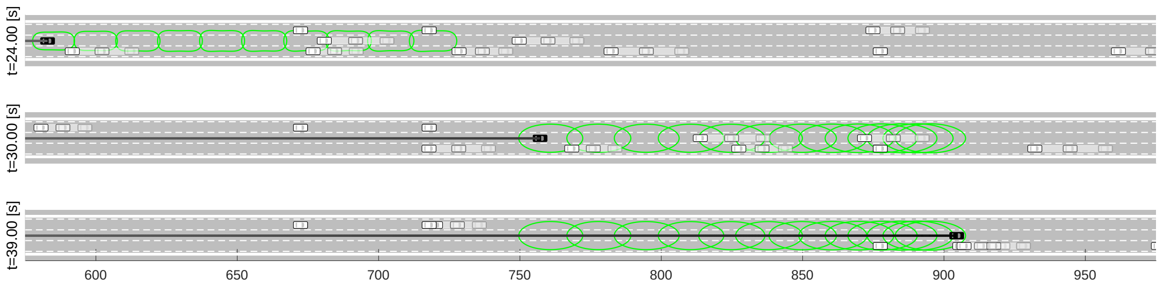
The baseline method with $\mathcal{P}^{\text{sparse}}$ shares almost the same memory consumption as REFINE with $\Delta t = 0.005$[s], but results in a much lower successful rate and smaller average travel speed. When the baseline method runs over $\mathcal{P}^{\text{dense}}$, its success rate is increased, but still smaller than that of REFINE. More troublingly, its memory consumption increases to 9.1 GB. Neither evaluated baseline is able to finish online planning within 3[s]. Compared to REFINE, SOS-RTD completes online planning faster and can also guarantee vehicle safety with a similar average travel speed. However SOS-RTD needs a memory of 2.4 GB to store its polynomial reachable sets, and its success rate is only 64% because the polynomial reachable sets are more conservative than zonotope reachable sets.

When the NMPC method is utilized for motion planning, the ego vehicle achieves a similar success rate as SOS-RTD, but crashes occur 29% of the time. Note the NMPC method achieves a higher average travel speed of the ego vehicle when compared to the other three methods. More aggressive operation can allow the ego vehicle drive closer to obstacles, but can make subsequent obstacle avoidance difficult. The NMPC method uses 40.8906[s] on average to compute a solution, which makes real-time path planning untenable.

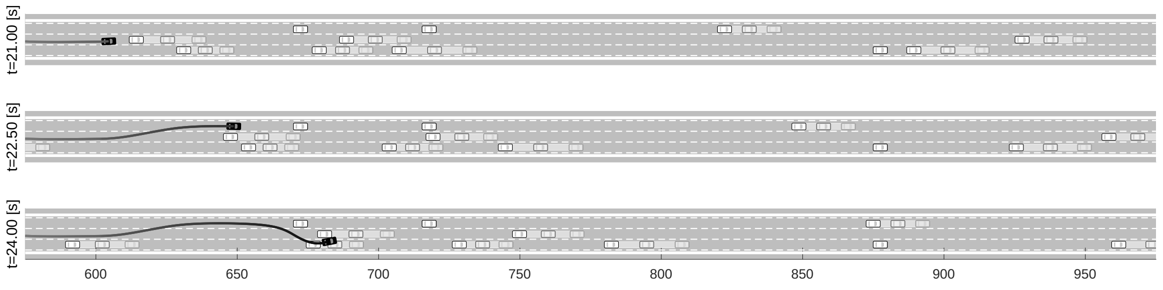
Figure 3.8 illustrates the performance of the three methods in the same scene at three different time instances. In Figure 3.8(a), because REFINE gives a tight approximation of the ego vehicle's FRS using zonotopes, the ego vehicle is able to first bypass static vehicles in the top lane from $t = 24$[s] to $t = 30$[s], then switch to the top lane and bypass vehicles in the middle lane from $t = 30$[s] to $t = 36$[s]. In Figure 3.8(b) SOS-RTD is used for planning. In this case the ego vehicle bypasses the static vehicles in the top lane from $t = 24$[s] to $t = 30$[s]. However because online planing becomes infeasible due to the conservatism of polynomial reachable sets, the ego vehicle executes the braking maneuver to stop itself $t = 30$[s] to $t = 36$[s]. In Figure 3.8(c) because NMPC is used for planning, the ego vehicle drives at a faster speed and arrives at 600[m] before the other evaluated methods. Because the NMPC method only enforces collision avoidance constraints at discrete time instances, the ego vehicle ends up with a crash at $t = 24$[s] though NMPC claims to find a feasible solution for the planning iteration at $t = 21$[s].



(a) REFINE utilized.



(b) SOS-RTD utilized.



(c) NMPC utilized.

Figure 3.8: An illustration of the performance of REFINE, SOS-RTD, and NMPC on the same simulated scenario. In this instance REFINE successfully navigates the ego vehicle through traffic (top three images), SOS-RTD stops the ego vehicle to avoid collision due to the conservatism of polynomial reachable sets (middle three images), and NMPC crashes the ego vehicle even though its online optimization claims that it has found a feasible solution (bottom three images). In each set of images, the ego vehicle and its trajectory are colored in black. Zonotope reachable sets for REFINE and polynomial reachable sets for SOS-RTD are colored in green. Other vehicles are obstacles and are depicted in white. If an obstacle is moving, then it is plotted at 3 time instances t, $t+0.5$ and $t+1$ with increasing transparency. Static vehicles are only plotted at time t.

3.9.3 Real World Experiments

REFINE was also implemented in C++17 and tested in the real world using a $\frac{1}{10}$th All-Wheel-Drive car-like robot, Rover, based on a Traxxax RC platform. The Rover is equipped with a front-mounted Hokuyo UST-10LX 2D lidar that has a sensing range of 10[m] and a field of view of 270 The Rover is equipped with a VectorNav VN-100 IMU unit which publishes data at 800Hz. Sensor drivers, state estimator, obstacle detection, and the proposed controller are run on an NVIDIA TX-1 on-board computer. A standby laptop with Intel i7-9750H processor and 32GB of RAM is used for localization, mapping, and solving (`Opt`) in over multiple partitions of $\mathcal{P}$. The rover and the standby laptop communicate over wifi using ROS [97].

Desired trajectories on the Rover are parameterized with $\mathcal{P} = \{(p_u, p_y) \in [0.05, 2.05] \times [-1.396, 1.396] \mid p_u = u_0 \text{ if } p_y \neq 0\}$, $a^{\text{dec}} = -1.5$[m/sec^2], $h_1^{\text{des}} = \frac{20}{27}$ and $h_2^{\text{des}} = \frac{27}{10}$ as described in Section 3.9.1. The duration t_{m} of driving maneuvers for each trajectory family is set to 1.5[s] for speed change, 1.5[s] for direction change, and 3[s] for lane change, thus planning time for real world experiments is set as $t_{\text{plan}} = 1.5$[s]. The parameter space $\mathcal{P}$ is evenly partitioned along its first and second dimensions into small intervals of lengths 0.25 and 0.349, respectively. For each partition element, t_{f} is set equal to the maximum possible value of t_{brake} as computed in (B.19) in which t_{fstop} is by observation no greater than 0.1[s]. The FRS of the Rover for every partition element of $\mathcal{P}$ is overapproximated using CORA with $\Delta t = 0.01$[s]. During online planning, a waypoint is selected in real time using Dijkstra's algorithm [23], and the cost function of (`Opt`) is set in the same way as we do in simulation as described in Section 3.9.2. The robot model, environment sensing, and state estimation play key roles in real world experiments. In the rest of this subsection, we first explain how the Rover performs localization, mapping, and obstacle detection and how we perform system identification of the tire models. We then describe how to bound the modeling error in (3.14) and summarize the real world experiments.

3.9.3.1 Environment Sensing

To sense the environment, we perform simultaneous localization and mapping (SLAM) using Cartographer [117] at a rate of 200Hz with lidar scans and IMU readings as its sensor inputs. Cartographer is a lidar based 2D SLAM algorithm that consists of a local subsystem, which builds locally consistent and successive submaps, and a global sub-system, which runs in background to achieve loop closure. Lidar scans are also used for obstacle detection at a rate of 10Hz using the method illustrated in [87], which detects an object in the environment by multiple line segments. To account for estimation error as discussed in [87], we inflate the detected line segments and convert them into zonotopes for online planning. Both Cartographer and obstacle detection are validated compared against data collected by a motion capture system, and estimation errors of both both algorithms are at most 10 [cm].

3.9.3.2 System Identification of Tire Models

The goal of system identification is to specify necessary parameters that describe dynamics of the Rover. Because parameters like mass, length, and moment of inertia can be directly measured, we focus on specifying tire force related parameters including λ^{cri}, α^{cri}, $\bar{\mu}$, $\bar{c}_{\alpha\text{f}}$ and $\bar{c}_{\alpha\text{r}}$, and explain how we generate the computational error Δ_u, Δ_v, Δ_r in (3.14) as well as their bounding parameters b_u^{pro}, b_u^{off}, M_u, M_v, and M_r in the next subsection. Note this system identification is done by using a motion capture system; however, when REFINE is applied, the motion capture system is not used.

Recall actual tire models in (3.6), (3.7), (3.10) and (3.11) become saturated at large slip ratios and slip angles. However, during experiments, the Rover is always expected to operate in linear regimes of tires by Assumption 10. Thus, to figure out the tire force-related parameters of the Rover, we need to identify the critical slip ratio and critical slip angle at which tire force saturation begins, then fit linear tire models within the linear regimes.

To identify the parameters related to longitudinal tire forces, the Rover executed a series of speed change maneuvers in a motion capture system to estimate u, v, and r. $\dot{u}$ is estimated using the onboard IMU. Recall the ideal dynamics of longitudinal speed as in (3.4). By plugging in the speed information from the motion capture system, we generate the longitudinal tire force $F_x(t) := F_{x\text{f}}(t) + F_{x\text{r}}(t)$ that achieves the observed velocity trajectory. Because the Rover is AWD, both the front and the rear tires have the same tire speed and thus the same slip ratio, i.e. $\lambda_\text{f} = \lambda_\text{r}$. Adding the two equations in (3.12) results in $F_x(t) = mg\bar{\mu}\lambda_\text{i}(t)$ where the subscript 'i' can be replaced by either 'f' for front tire or 'r' for rear tire. Using the information from the encoder of driving motor and $u(t)$, slip ratios of both tires can be computed via (3.5). As shown in Figure 3.9, the longitudinal tire force saturates when the slip ratio becomes bigger than 0.45. Thus we set $\lambda^{\text{cri}} = 0.45$ and fit $\bar{\mu}$ from $F_x(t) = mg\bar{\mu}\lambda_\text{i}(t)$ by performing least squares over collected data that satisfies $|\lambda_\text{i}(t)| \leq \lambda^{\text{cri}}$ at any time.

To identify the parameters related to lateral tire forces, we follow a similar procedure and let the Rover execute a series of direction change maneuvers with various longitudinal speeds. Ground truth u, v, and r are again estimated using the motion capture system, and $\dot{v}$ and $\dot{r}$ are estimated using the onboard IMU for all time. Recall when $u(t) > u^{\text{cri}}$, the error-free dynamics of v and r are as in (3.4). One can then compute $F_{\text{yf}}(t)$ and $F_{\text{yr}}(t)$ by using the relevant components of (3.4) for $F_{\text{yf}}(t)$ and $F_{\text{yr}}(t)$. Using $u(t)$, $v(t)$, $r(t)$, and the steering motor input, one can compute slip angles for both tires via (3.8) and (3.9). As shown in Figure 3.10, the lateral tire force saturates when the slip angle becomes bigger than 0.165. Thus we set $\alpha^{\text{cri}} = 0.165$, and fit $\bar{c}_{\alpha\text{f}}$ and $\bar{c}_{\alpha\text{r}}$ in (3.13) by performing least squares over collected data that satisfies $|\alpha_\text{i}(t)| \leq \alpha^{\text{cri}}$ at any time.

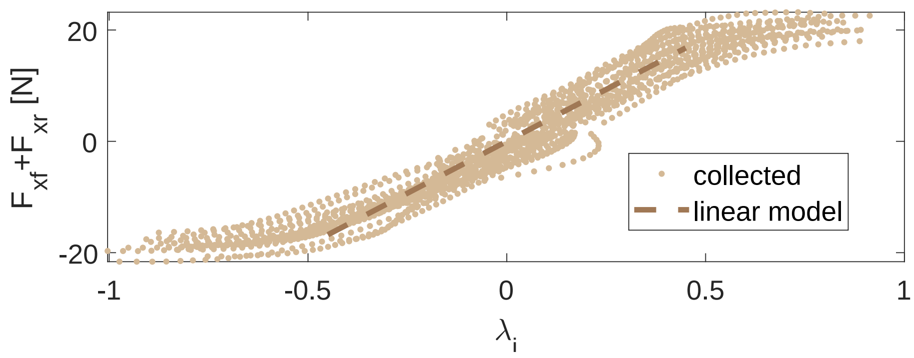


Figure 3.9: System Identification on longitudinal tire force. A linear model is fit using data collected within the linear regime [-0.45,0.45] of slip ratio.

3.9.3.3 State Estimation and System Identification on Computation Error of Vehicle Dynamics

The modeling error in the dynamics (3.14) arise from ignoring aerodynamic drag force and the inaccuracies of state estimation and the tire models. We use the data collected to fit the tire models to identify the modeling errors Δ_u, Δ_v, and Δ_r.

We compute the model errors as the difference between the *actual accelerations* collected by the IMU and the *estimation of applied accelerations* computed via (3.4) in which tire forces are calculated via (3.12) and (3.13). The estimation of applied accelerations is computed using the estimated system states via an Unscented Kalman Filter (UKF) [111], which treats SLAM results, IMU readings, and encoding information of wheel and steering motors as observed outputs of the Rover model. The robot dynamics that UKF uses to estimate the states is the error-free, high-speed dynamics (3.4) with linear tire models. Note the UKF state estimator is still applicable in the low-speed case except the estimation of v and r are ignored. To ensure Δ_u, Δ_v and Δ_r are square integrable, we set $\Delta_u(t) = \Delta_v(t) = \Delta_r(t) = 0$ for all $t \geq t_{\text{brake}}$ where t_{brake} is computed in Lemma 21. As shown in Figure 3.11 bounding parameters M_u, M_v, and M_r are selected to be the maximum value of $|\Delta_u(t)|$, $|\Delta_v(t)|$, and $|\Delta_r(t)|$ respectively over all time, and b_u^{pro} and b_u^{off} are generated by bounding $|\Delta_u(t)|$ from above when $u(t) \leq u^{\text{cri}}$.

Identified model parameters of the Rover and the REFINE's controller parameters to be used in real world experiments are listed in Table 3.3. An example of tracking performance of the proposed controller on the Rover with identified parameters is shown in Figure 3.12.

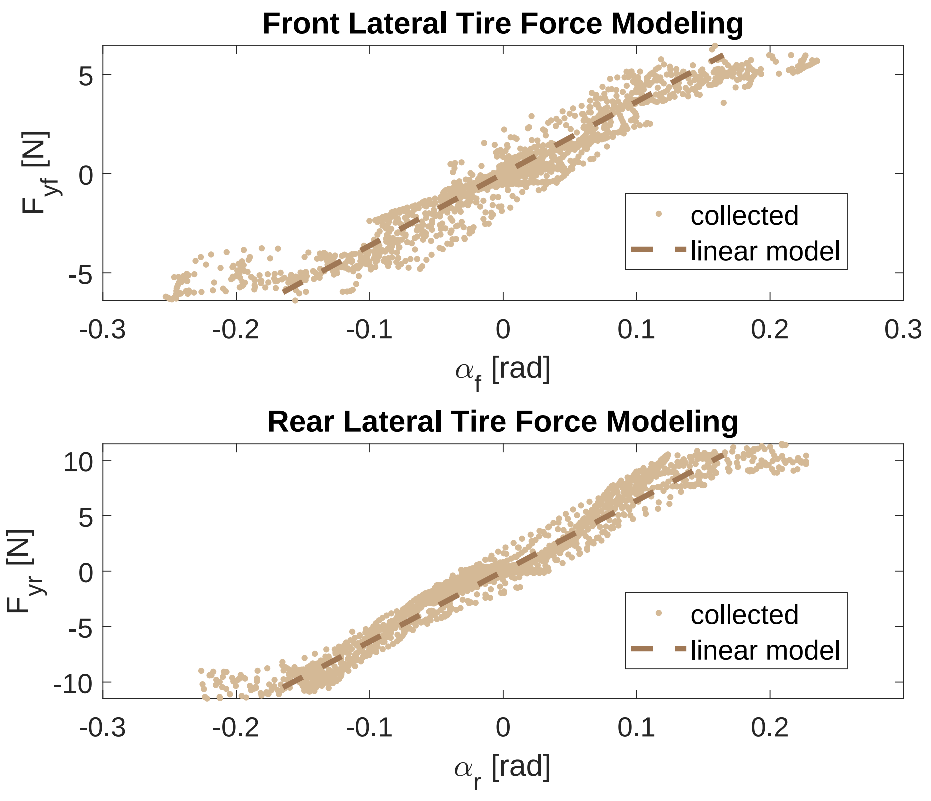


Figure 3.10: System identification on lateral tire forces. Linear models are fit using data collected within the linear regime [-0.165,0.165] of slip angles.

3.9.3.4 Demonstration

The Rover was tested indoors under the proposed controller and planning framework in 6 small trials and 1 loop trial[4]. In every small trail, up to 11 identical $0.3\times0.3\times0.3[\mathrm{m}]^3$ cardboard cubes were placed in the scene before the Rover began to navigate itself. The Rover was not given prior knowledge of the obstacles for each trial. Figure 3.13 illustrates the scene in the 6th small trial and illustrates REFINE's performance. The zonotope reachable sets over-approximate the trajectory of the Rover and never intersect with obstacles.

In the loop trial, the Rover was required to perform 3 loops, and each loop is about 100[m] in length. In the first loop of the loop trial, no cardboard cube was placed in the loop, while in the last two loops the cardboard cubes were randomly thrown at least 5[m] ahead of the running Rover to test its maneuverability and safety. During the loop trial, the Rover occasionally stoped because a randomly thrown cardboard cube might be close to a waypoint or the end of an executing maneuver. In such cases, because the Rover was able to eventually locate obstacles more accurately when it was stopped, the Rover began

[4]https://drive.google.com/drive/folders/1FvGHuqIRQpDS5xWRgB30h7exmGTjRyel?usp=sharing

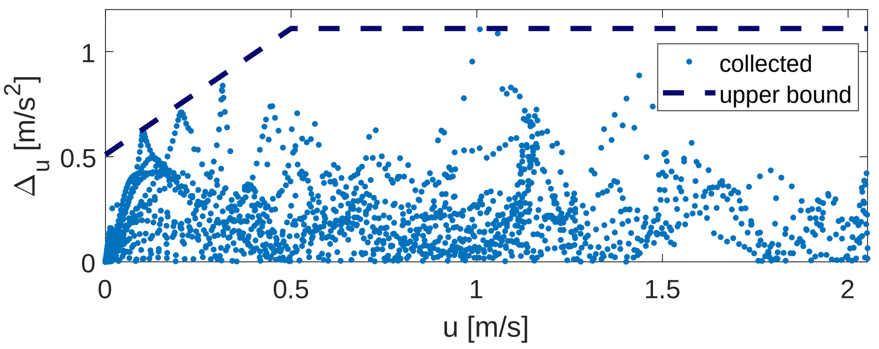


Figure 3.11: An illustration of the modeling error along the dynamics of u. Collected $\Delta_u(t)$ is bounded by $M_u = 1.11$ for all time. Whenever $u(t) \leq u^{\text{cri}} = 0.5$, $\Delta_u(t)$ is bounded by $b_u^{\text{pro}} u(t) + b_u^{\text{off}}$ with $b_u^{\text{pro}} = 1.2$ and $b_u^{\text{off}} = 0.51$.

a new planning iteration immediately after stopping and passed the cube when a feasible plan with safety guaranteed was found.

For all 7 real-world testing trials, the Rover either safely finishes the given task, or it stops itself before running into an obstacle if no clear path is found. The Rover is able to finish all computation of a planning iteration within 0.4021[s] on average and 0.6545[s] in maximum, which are both smaller than $t_{\text{plan}} = 1.5$[s], thus real-time performance is achieved.

3.10 Conclusion

This work presents a controller-oriented trajectory design framework REFINE using zonotope reachable sets. A robust controller is designed to partially linearize the full-order vehicle dynamics with modeling error by performing feedback linearization on a subset of vehicle states. The proposed controller can be generalized to FWD, AWD and RWD vehicle models. Zonotope-based reachability analysis is performed on the closed-loop vehicle dynamics for FRS computation, and achieves less conservative FRS approximation than that of the traditional reachability-based approaches. Mathematical derivation of constraints on collision-free using zonotope reachable sets is provided for online planning. Tests on a full-size vehicle model in simulation and a 1/10th race car robot in real hardware experiments show that the proposed method is able to safely navigate the vehicle through random environments in real time and outperforms all evaluated state of the art methods. Unfortunately as a deterministic approach, REFINE accounts for perception uncertainty by requiring the robot to avoid all possible area that could cause a collision, thus a downside of

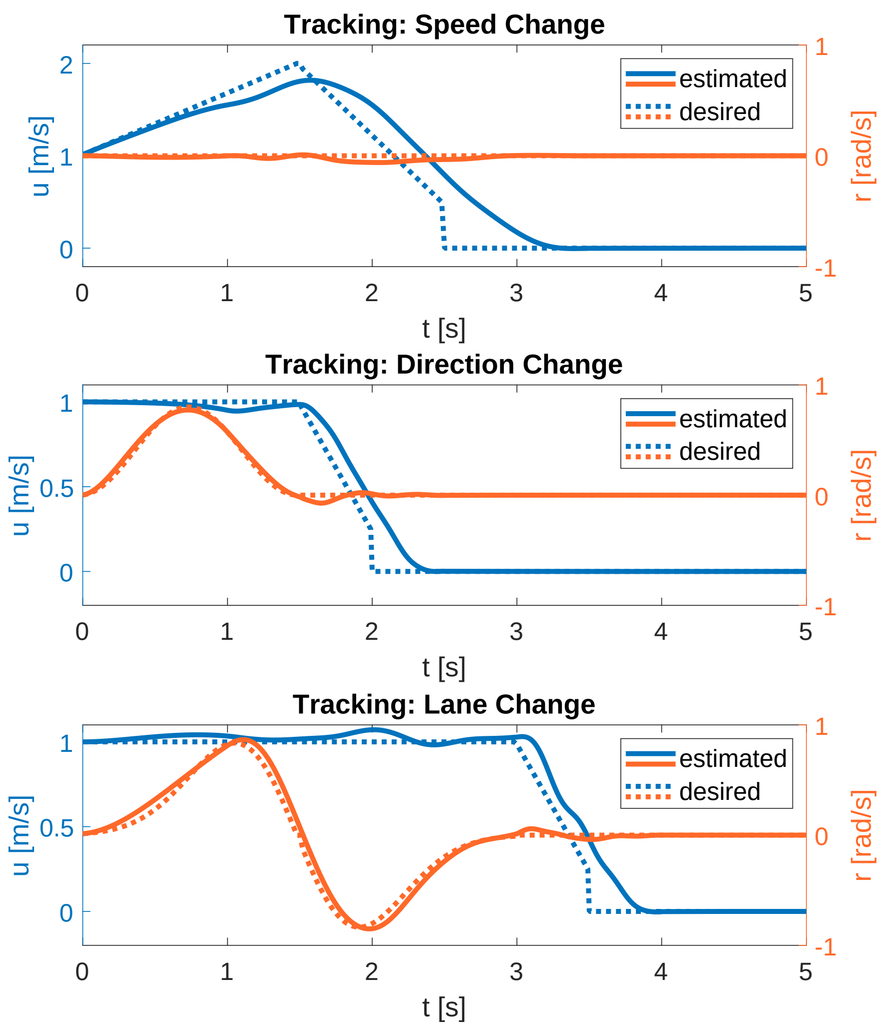
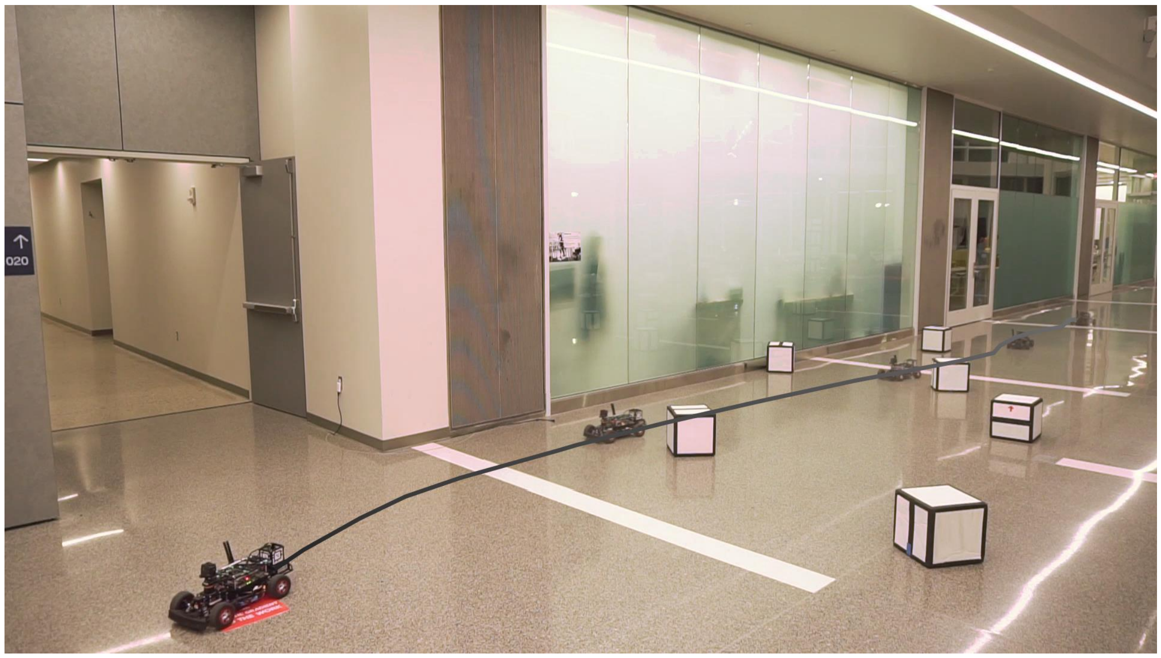
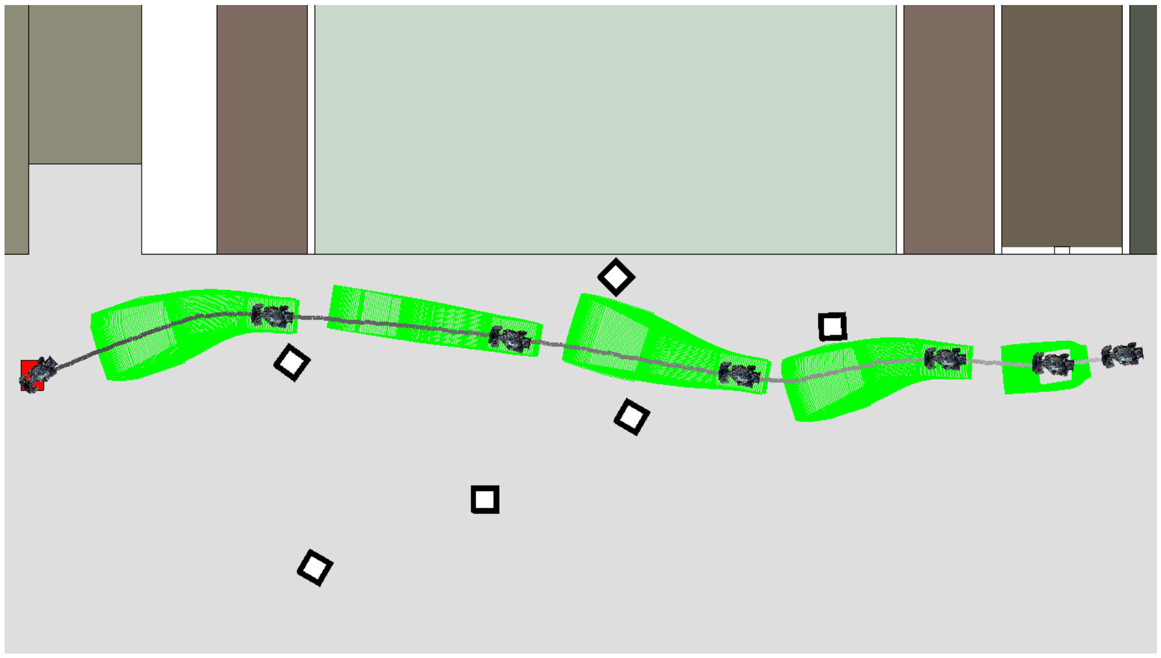


Figure 3.12: An illustration of the tracking performance of the Rover with the proposed controller and identified parameters in Table 3.3.

the planning performance. In the next chapter, we extend the framework of REFINE within uncertain environment and achieve path planning with risk of collision being bounded.



(a)



(b)

Figure 3.13: An illustration of the performance of REFINE during the 6th real-world trial. The rover was able to navigate itself to the goal in red through randomly thrown white cardboard cubes as shown in (a). Online planning using zonotope reachable sets is illustrated in (b) in which trajectory of the Rover is shown from gray to black along time, goal is shown in red, and the zonotope reachable sets at different planning iterations are colored in green.

Method	Safely Stop	Crash	Success	Avg. Travel Speed	Online Planning Runtime (Average, Maximum)	Memory
Baseline (sparse, Δt = 0.010)	38%	**0%**	62%	22.3572[m/s]	(2.03[s], 4.15[s])	980 MB
Baseline (dense, Δt = 0.010)	30%	**0%**	70%	23.6327[m/s]	(12.42[s], 27.74[s])	9.1 GB
SOS-RTD	36%	**0%**	64%	24.8049[m/s]	(**0.05[s]**, 1.58[s])	2.4 GB
NMPC	3%	29%	68%	27.3963[m/s]	(40.89[s], 534.82[s])	N/A
REFINE (Δt = 0.015)	27%	**0%**	73%	23.2452[m/s]	(0.34[s], **0.95[s]**)	**488 MB**
REFINE (Δt = 0.010)	17%	**0%**	83%	24.8311[m/s]	(0.52[s], 1.57[s])	703 MB
REFINE (Δt = 0.005)	16%	**0%**	**84%**	24.8761[m/s]	(1.28[s], 4.35[s])	997 MB
REFINE (Δt = 0.001)	16%	**0%**	**84%**	24.8953[m/s]	(6.48[s], 10.78[s])	6.4 GB

Table 3.2: Summary of performance of various tested techniques on the same 1000 simulation environments.

Rover Parameter (Real World)		REFINE Controller Parameter (Real World)	
m	4.96[kg]	K_u	4.0
l_f	0.203[m]	K_r	8.0
l_r	0.107[m]	K_h	5.0
I_zz	0.11[kg·m^2]	$\kappa_{1,u}$	1.2
r_w	0.055[m]	$\kappa_{2,u}$	0.8
λ^cric	0.45	$\phi_{1,u}$	1.2
α^cric	0.165[rad]	$\phi_{2,u}$	0.8
$\bar{\mu}$	0.77	$\kappa_{1,r}$	0.9
$\bar{c}_{\alpha\mathrm{f}}$	36.24[N/rad]	$\kappa_{2,r}$	0.6
$\bar{c}_{\alpha\mathrm{r}}$	63.52[N/rad]	$\phi_{1,r}$	0.9
u^cri	0.5[m/s]	$\phi_{2,r}$	0.6
L	0.52[m]	M_u	1.11
W	0.28[m]	b_u^pro	1.2
		b_u^off	0.51
		M_r	1.03

Table 3.3: Specification of the Rover.

CHAPTER 4

Real-time Risk-aware Reachability-based Trajectory Design

4.1 Introduction

For robots to safely operate in unstructured environments, especially where humans are present, they must be able to sense their environments and develop plans to dynamically react to changes in those environments and avoid obstacles. Safe navigation through these uncertain environments in the presence of dynamic obstacles is a critical challenge. It is difficult to accurately predict an obstacle's movement due to imperfect knowledge of the obstacle motions as well as the presence of noise in sensors. For robots to operate robustly, this uncertainty must be accounted for while generating motion plans. Various approaches to account for this uncertainty have been proposed in the literature, but these methods either (1) have difficulty being evaluated in real-time, (2) make strong assumptions on the class of probability distributions used to model the uncertainty, or (3) generate motion plans that are overly conservative and thereby limit most motion. This chapter develops an algorithm called **Risk-RTD** for **real-time risk-aware reachability-based trajectory design** while addressing each of the aforementioned challenges.

Depending on how accurate the provided environment knowledge is, deterministic and stochastic planning strategies have been developed in the field. With locations and predictions of obstacles accurately given, deterministic approaches including works in previous chapters and related literature therein can provide robot safety by either enforcing that no collision happens between the robot and any obstacle at multiple discrete time instances [25, 27], or ensuring that the forward reachable set has no intersection with any obstacle during time intervals [52, 56]. In real application, observing and predicting the motion of a moving obstacle exact can be impractical due to sensor noise or the lack of knowledge on obstacle motion. In the case when locations and predictions of obstacles can be conservatively bounded, aforementioned approaches are still applicable because they can require the robot to avoid larger area for safety. However it is possible that such conservative estimations of obstacles' locations make online planning infeasible, thus a loss of performance on maneuverability [91].

To balance the tradeoff between safety and maneuverability in uncertain environments, stochastic methods embed probabilistic information about uncertain environments in order to achieve risk-aware motion planning. [19, 86, 112, 121] achieve motion planning with probabilistic explorations of environments by solving an optimization problem in which chance constraints are enforced to limit the probability of collision from above. However because in most scenarios the probability of collision could be hardly evaluated due to the arbitrariness of probabilistic environmental observation, chance-constrained methods have to approximate the probability of collision while maintaining real-time performance. Sampling-based methods, [12, 47] for example, apply numerical integration techniques to compute the probability of collision. In this context the chance constraint is expressed as an indicator function on the set of variables that violate the collision constraint. Random samples are drawn from the uncertain region and the weighted sum of these samples is taken to approximate the integral. These methods are simple to implement but can be computationally expensive and have slow convergence rates. Moreover, they are difficult to utilize in gradient-based optimization algorithms due to the non-smooth nature of the indicator function used as the integrand. Moment-Based methods [18, 112, 113] upper bound the probability of collision using the moments of the probability distribution, which can be computed efficiently. However because multiple probability distribution can share the same moments up to some finite order [98], online planning may result in avoiding unconsidered probabilistic distributions. Chance Constrained Parallel Bernstein Algorithm (CCPBA) [105] as a branch-and-bound method uses reachability-based methods in conjunction with a branch-and-bound style algorithm to compute tight bounds for the probability of collision. In order to compute these probabilities they make strong assumptions about knowing a-priori a closed-form cumulative distribution function that returns the probability of intersection with their reachable set. Unfortunately such closed-form cumulative distribution may not exist in real application.

Compared to chance constraints, Conditional Value-at-Risk (CVaR) constraints can also be utilized to incorporate probabilistic estimation of the environment for decision making due to their coherency and convexity [89, 100]. For example in the application of motion planning, CVaR-based method in [38] regulates safety by limiting the expectation of the distance to safe region within certain worst-case quantile of its distribution. However one may have to approximate such CVaR constraints via sampling to avoid computationally expensive multi-dimensional integration. Moreover, because CVaR-constrained optimiza-

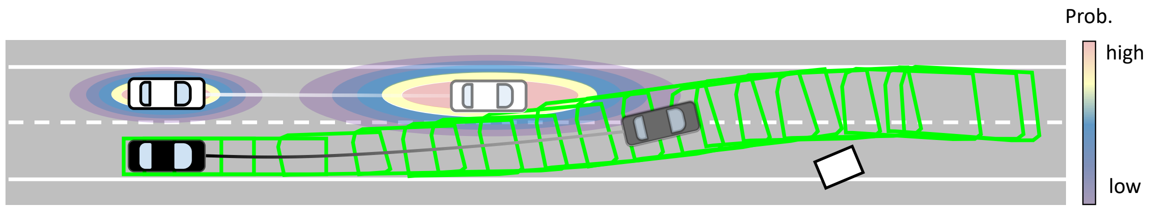


Figure 4.1: Illustration of the proposed Risk-RTD framework in an uncertain dynamic environment for autonomous driving in which the ego vehicle in black tries to avoid a static obstacle and a moving vehicle shown in white. Transparencies of the ego vehicle and the moving vehicle increase along time. Given probabilistic descriptions of location and prediction of the moving vehicle, Risk-RTD limits the risk of collision between the ego vehicle and the moving vehicle from above by enforcing an upper bound on the over-approximation of the probabilistic integration over offline-computed zonotope reachable set during each time interval.

tions are typically formulated as mixed integer convex problem, achieving real-time planning can be challenging. Safety performance of CVaR-constrained motion planning can be improved by using Entropic Value-at-Risk(EVaR) risk measure [24], but EVaR shares similar issues as CVaR.

In this work we propose a real-time Risk-aware Reachability-based Trajectory Design (Risk-RTD) framework for path planning in dynamic environments shown in Figure 4.1. As the probabilistic extension of the original RTD method [52], Risk-RTD shares the same offline reachability analysis with REFINE to benefit from the tightness of zonotope reachable sets, but enforces robot safety by bounding the probability of any collision from above as chance constraints at all time during online planning. Contributions of this work are three folds. First, we provide a closed-form over-approximation on the probability of a collision with mild assumption on the probabilistic distribution of obstacle estimation. Second, we illustrate the analytical derivative of the probability over-approximation with respect to control parameters, which parametrize desired trajectory for the robot to track and are the decision variables of online optimization. Lastly, computations of both probability over-approximation and its derivative are parallelizable so that real-time motion planning can be achieved.

Note to simplify exposition, this chapter explains the idea of Risk-RTD with an autonomous driving application, but the proposed idea can be generalized to other robot platforms with zonotope reachable sets in uncertain environments. The remainder of this chapter is organized as follows. Section 4.2 provides necessary preliminaries on obstacle uncertainty, risk-aware vehicle safety, and chance-constrained online planning. Section 4.3 explains how Risk-RTD matches frames among zonotope reachable sets and obstacle probability descriptions. Because the probability of collision in the chance constraint of online planning can be hard to evaluate in general, Section 4.4 conservatively relaxes the chance constraint in a differentiable closed-form, which is more tractable to enforce during online planning. Section 4.5 describes a methodology to select risk threshold dynamically. Experiments of the proposed framework are presented in Section 4.6, and Section 4.7 concludes the chapter.

4.2 Preliminaries

In this work we adopt all notations, vehicle dynamics and control from the last chapter, and for convenience we denote `Prob(E)` the probability of occurrence of some event `E`. Recall in REFINE the i-th obstacle $\mathcal{O}_i(t)$ during the j-th time interval T_j is assumed to be over-approximated by zonotope $\vartheta(j,i) \subset \mathcal{W}$. However such assumption can be empirically difficult to achieve because in real application a sensor used for obstacle detection could have measurement error and an obstacle detection algorithm may result in a probabilistic description of an obstacle [119,122]. One can certainly make $\vartheta(j,i)$ large enough to bound the obstacle location based on its probabilistic description to achieve a confidence level of almost 1, but this may result in conservativeness that could affect the maneuverability of the ego vehicle. To incorporate the uncertainty in obstacle sensing into the motion planning framework, in this work we re-define vehicle safety and relax the zonotope description of an obstacle from the probability perspective.

4.2.1 Obstacle Uncertainty

For notional simplicity, for arbitrary time t and $i \in \mathcal{I}$, denote $c_i^{\text{obs}}(t) \in \mathcal{W}$ the center of $\mathcal{O}_i(t)$. We then make the following assumption about the probabilistic estimation of an obstacle's location.

Assumption 34. *For any $(i,j) \in \mathcal{I} \times \mathcal{J}$, the probability distribution of $w_{i,j}^{obs} := c_i^{obs}((j-0.5)\Delta t)$ satisfies a probability density function $q(\cdot \mid i,j) : \mathcal{W} \to [0,+\infty)$. The probability density function q is further assumed twice-differentiable.*

Note Assumption 34 is the only assumption we make on the probabilistic observation of any obstacle, thus the proposed framework is general to any probability distribution as long as the corresponding density function is twice-differentiable. To account for the obstacle's potential location over T_j, we make the following assumption:

Assumption 35. *There exists a 2-row matrix G^{obs} such that $\cup_{t \in T_j} \mathcal{O}_i(t)$ stays inside a zonotope $<w_{i,j}^{obs}, G^{obs}> \subset \mathcal{W}$ for all $(i,j) \in \mathcal{I} \times \mathcal{J}$.*

According to Assumptions 34 and 35, $<w_{i,j}^{\text{obs}},\ G^{\text{obs}}> = w_{i,j}^{\text{obs}} + <0,\ G^{\text{obs}}>$ has an uncertain center $w_{i,j}^{\text{obs}}$ with probability density $q(w_{i,j}^{\text{obs}} \mid i, j)$, and has invariant shape and size with respect to i and j due to the constant generator matrix G^{obs}. Such probability density functions q can be generated by, for example, performing variants of Kalman Filter on the i-th obstacle given its dynamics or detecting the i-th obstacle with a Bayesian confidence framework [28] and a neural network [31] during T_j. The generator matrix G^{obs} can be generated according to the vehicle footprint and maximum speed ν^{obs} of all obstacles.

4.2.2 Risk-Aware Vehicle Safety

For arbitrary time t, given state $z(t)$ of the ego vehicle with the closed-loop dynamics (3.39) that starts from some initial condition $z_0 \in \mathbb{R}^6$ and applies a control input parametrized by $p \in \mathcal{P}$, the ego vehicle's footprint at time t can be represented as

$$\mathcal{E}(t, z_0, p) := \texttt{rotate}(h(t)) \cdot \mathcal{O}^{\text{ego}} + [x(t), y(t)]^\top \tag{4.1}$$

where $\texttt{rotate}(a)$ gives the 2-dimensional rotation matrix $\begin{bmatrix} \cos(a) & -\sin(a) \\ \sin(a) & \cos(a) \end{bmatrix}$ for an arbitrary value $a \in \mathbb{R}$. Recall that $\mathcal{O}^{\text{ego}}$ is a zonotope as defined in Definition 17, then $\mathcal{E}(t, z_0, p)$ is also a zonotope by (3.3). Then we can redefine vehicle's safety in this work from the probability perspective by bounding the probability of the ego vehicle running into any obstacles.

Definition 36. *Let the ego vehicle start from initial condition $z_0 \in \mathcal{Z}_0$ with control parameter $p \in \mathcal{P}$. Given some* allowable risk threshold $\epsilon \in [0, 1]$, *the ego vehicle is* not-at-fault, *with a chance of collision of at most ϵ, if it is stopped, or if*

$$\texttt{Prob}\Big(\cup_{t \in [0, t_f]} \big(\mathcal{E}(t, z_0, p) \cap \big(\cup_{i \in \mathcal{I}} \mathcal{O}_i(t) \big) \big) \Big) \leq \epsilon \tag{4.2}$$

while it is moving during time interval $[0, t_f]$.

In the case when $\epsilon = 0$, inequality (4.2) results in

$$\cup_{t \in [0, t_f]} \Big(\mathcal{E}\big(z(t), z_0, p\big) \cap \big(\cup_{i \in \mathcal{I}} \mathcal{O}_i(t) \big) \Big) = \varnothing, \tag{4.3}$$

which happens to be the constraint of no intersection in REFINE.

4.2.3 Online Optimization

To construct a motion plan that ensures the ego vehicle is not-at-fault with a chance of collision at most ϵ during online planning, one could solve the following chance constrained optimization problem:

$$\begin{aligned} \min_{p \in \mathcal{P}} \quad & \texttt{cost}(z_0, p) && (\texttt{Opt-C}) \\ \text{s.t.} \quad & \texttt{Prob}\Bigg(\cup_{t \in [0, t_\text{f}]} \Big(\mathcal{E}\big(t, z_0, p\big) \cap \big(\cup_{i \in \mathcal{I}} \mathcal{O}_i(t) \big) \Big) \Bigg) \leq \epsilon \end{aligned}$$

where $\texttt{cost} : \mathcal{Z}_0 \times \mathcal{P} \to \mathbb{R}$ is a user-specified cost function, and the constraint is a chance constraint that ensures the vehicle is not-at-fault with a maximum chance of collision of ϵ during the planning horizon as stated in Definition 36.

To achieve real-time motion planning, ($\texttt{Opt-C}$) must be solved within t_plan seconds. Unfortunately, efficiently evaluating the chance constraint in ($\texttt{Opt-C}$) in a closed-form can be challenging in real applications because of two reasons. Firstly, the exact information of the ego vehicle's location $\mathcal{E}(t, z_0, p)$ at any time is usually not accessible due to the nonlinearity and hybrid nature of the vehicle dynamics. Secondly, the probability distribution that describes the probabilistic observation of an obstacle as assumed in Assumption 34 can be arbitrary. Therefore to achieve real-time performance, Risk-RTD seeks a closed-form approximation of the chance of collision for evaluation efficiency. In addition, such approximation needs to be an over-approximation to ensure that Risk-RTD does not underestimate the true risk of collision or generate plans that violate the not-at-fault condition from Definition 36. Moreover, it is preferable that this approximation is differentiable, as providing the gradient of the constraint can speed up the solving procedure of online optimization. We describe how we generate an approximation that satisfies these requirements in the following sections.

4.3 Offline Reachability Analysis

Due to the nonlinearity and hybrid nature of the vehicle dynamics as in (3.39), it is challenging to compute trajectory of the ego vehicle exactly for evaluating the chance constraint in ($\texttt{Opt-C}$). Therefore Risk-RTD adopts from REFINE the same offline reachability analysis in which a collection of zonotopes $\{\mathcal{R}_j\}_{j \in \mathcal{J}}$ is generated to outer-approximate the FRS as described in Theorem 23. Notice $\{\mathcal{R}_j\}_{j \in \mathcal{J}}$ generated in Theorem 23 is a local representation of the ego vehicle's behavior during the planning horizon in ego vehicle's body frame because it assumes $x(0) = y(0) = h(0) = 0$. To enforce vehicle safety condition online, instead of transferring obstacle observation from the world frame into ego vehicle's body frame as REFINE does, Risk-RTD accounts for nonzero initial position and heading conditions during zonotope slicing to match obstacle observation in the world frame through the following lemma and theorem.

Lemma 37. *Given arbitrary* $[x_0, y_0, h_0]^\top \in \mathbb{R}^3$, $z_0^{vel} \in \mathbb{R}^3$ *and* $p \in \mathcal{P}$, *let* z *and* $\tilde{z}$ *be solutions to* (3.39) *with initial conditions* $z(0) = [x_0, y_0, h_0, (z_0^{vel})^\top]^\top$ *and* $\tilde{z}(0) = [0, 0, 0, (z_0^{vel})^\top]^\top$ *respectively, then*

$$\pi_{xy}(z(t)) = \texttt{rotate}(h_0) \cdot \pi_{xy}(\tilde{z}(t)) + [x_0, y_0]^\top. \tag{4.4}$$

Proof. Notice that $z(0)$ and $\tilde{z}(0)$ shares the same initial velocities, and that the dynamics of $[u, v, r]^\top$ is invariant to the initial condition of $[x, y, h]^\top$. Therefore the last 3 dimensions of $z(t)$ and $\tilde{z}(t)$ coincides for all t. Because $\dot{h}(t) = r(t)$, then

$$[z(t)]_3 = [\tilde{z}(t)]_3 + h_0. \tag{4.5}$$

Then the claim follows from the fact that

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \end{bmatrix} = \begin{bmatrix} \cos(h(t)) & -\sin(h(t)) \\ \sin(h(t)) & \cos(h(t)) \end{bmatrix} \begin{bmatrix} u(t) \\ v(t) \end{bmatrix}. \tag{4.6}$$

□

Theorem 38. *Let* $\{\mathcal{R}_j\}_{j \in \mathcal{J}}$ *be the collection of zonotopes in Theorem 23, and let* z *be the solution to* (3.39) *with initial condition* $z(0) = z_0 \in \mathbb{R}^3 \times \mathcal{Z}_0^{vel}$ *and control parameter* $p \in \mathcal{P}$. *Then there exists a map* $\bar{\xi} : P(\mathbb{R}^{9+n_p}) \times \mathcal{Z}_0 \times \mathcal{P} \to P(\mathcal{W})$ *such that*

1. *for any* $j \in \mathcal{J}$, $\bar{\xi}(\mathcal{R}_j, z_0, p)$ *contains footprint of the ego vehicle during* T_j, *i.e.,* $\cup_{t \in T_j} \mathcal{E}(t, z_0, p) \subseteq \bar{\xi}(\mathcal{R}_j, z_0, p)$, *and*
2. $\bar{\xi}(\mathcal{R}_j, z_0, p) \subset \mathcal{W}$ *is a zonotope of the form* $<c_{\bar{\xi},j}(z_0) + A_{\bar{\xi},j} \cdot p,\ G_{\bar{\xi},j}>$ *with some vector* $c_{\bar{\xi},j}(z_0) \in \mathbb{R}^2$, *some matrix* $A_{\bar{\xi},j} \in \mathbb{R}^{2 \times n_p}$ *and some 2-row matrix* $G_{\bar{\xi},j}$.

Proof. For any $z(0) = z_0 = [x_0, y_0, h_0, (z_0^{\text{vel}})^\top]^\top \in \mathbb{R}^3 \times \mathcal{Z}_0^{\text{vel}}$, let $\tilde{z}$ be the solution to (3.39) with initial condition $\tilde{z}(0) = [0, 0, 0, (z_0^{\text{vel}})^\top]^\top$ with control parameter p. Then by Lemma 28, there exists a zonotope $\xi(\mathcal{R}_j, z_0^{\text{vel}}, p) \subset \mathcal{W}$ such that for any $j \in \mathcal{J}$ and $t \in T_j$, the vehicle footprint oriented and centered according to $\tilde{z}(t)$ is contained within $\xi(\mathcal{R}_j, z_0^{\text{vel}}, p)$. Let

$$\bar{\xi}(\mathcal{R}_j, z_0, p) \coloneqq \texttt{rotate}(h_0) \cdot \xi(\mathcal{R}_j, z_0^{\text{vel}}, p) + [x_0, y_0]^\top, \tag{4.7}$$

then $\bar{\xi}(\mathcal{R}_j, z_0, p)$ contains the ego vehicle's footprint according to $z(t)$ during T_j based on Lemma 37.

In addition, $\xi(\mathcal{R}_j, z_0^{\text{vel}}, p)$ is a zonotope and can be represented as $<c_{\xi,j}(z_0^{\text{vel}}) + A_{\xi,j} \cdot p,\ G_{\xi,j}>$ with some $c_{\xi,j}(z_0^{\text{vel}}) \in \mathcal{W}$, some $A_{\xi,j} \in \mathbb{R}^{2\times n_p}$ and some 2-row real matrix $G_{\xi,j}$ as derived in Lemma 48 in Appendix C. Thus $\bar{\xi}(\mathcal{R}_j, z_0, p)$ is a zonotope of the form $<c_{\bar{\xi},j}(z_0) + A_{\bar{\xi},j} \cdot p,\ G_{\bar{\xi},j}>$ where

$$\begin{cases} c_{\bar{\xi},j}(z_0) = \texttt{rot}(h_0) \cdot c_{\xi,j}(z_0^{\text{vel}}) + [x_0, y_0]^\top \\ A_{\bar{\xi},j} = \texttt{rot}(h_0) \cdot A_{\xi,j} \\ G_{\bar{\xi},j} = \texttt{rot}(h_0) \cdot G_{\xi,j} \end{cases} . \tag{4.8}$$

□

4.4 An Implementable Alternative to (`Opt-C`)

This section describes an implementable alternative to (`Opt-C`) that can be solved rapidly. This is done by describing how to relax the chance constraint in (`Opt-C`) in a conservative fashion.

4.4.1 Chance Constraint Relaxation

The chance constraint in (`Opt-C`) is relaxed conservatively as illustrated in Figure 4.2. First, the chance of collision during $[0, t_\text{f}]$ is over approximated using the integration of the probability density function (PDF) assumed in Assumption 34. Then we relax the domain of integration into a collection of right-angled triangles, and relax the PDF as a quadratic polynomial. Finally a closed-form formulation of the relaxed integration is provided.

4.4.1.1 PDF Integration

Recall the ego vehicle's performance during the planning horizon is over-approximated by $\bar{\xi}(\mathcal{R}_j, z_0, p)$ over all time interval T_j's, then the chance constraint in (`Opt-C`) is relaxed in the following theorem.

Theorem 39. *Suppose the ego vehicle starts with initial condition $z_0 \in \mathcal{Z}_0$ and control parameter $p \in \mathcal{P}$. Let $\{q(\cdot \mid i, j)\}_{i\in\mathcal{I}, j\in\mathcal{J}}$ be assumed in Assumption 34, let matrix G^{obs} be assumed in Assumption 35, and let map $\bar{\xi}$ be described in Theorem 38. Then for arbitrary*

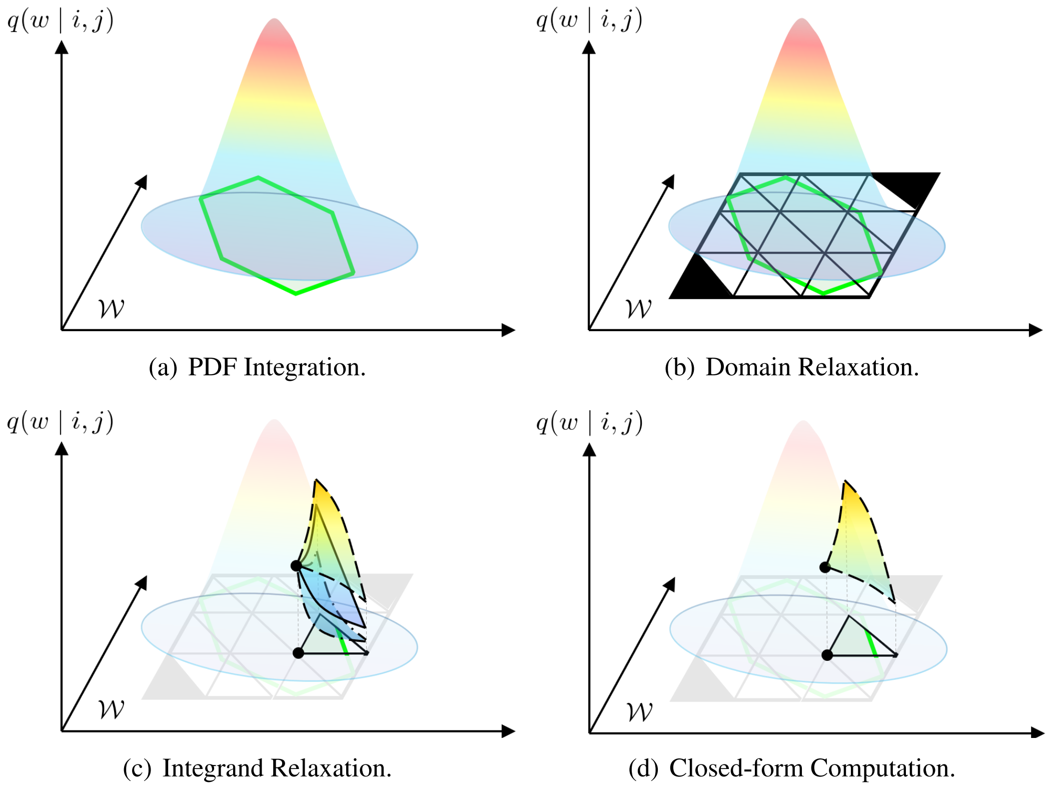


Figure 4.2: An illustration of chance constraint relaxation. Given arbitrary $(i,j) \in \mathcal{I}\times\mathcal{J}$, in (a) the probability of collision between the ego vehicle and the i-th obstacle during time interval T_j is relaxed as the integration of probability density function $q(\cdot \mid i,j)$ over zonotope $\bar{\xi}(\mathcal{R}_j, z_0, p) \oplus <0,\ G^{\text{obs}}> \subset \mathcal{W}$ shown in green. In (b), $\bar{\xi}(\mathcal{R}_j, z_0, p) \oplus <0,\ G^{\text{obs}}>$ is over-approximated by a collection of right-angled triangles colored in white and black depending on if a triangle has nontrivial intersection with $\bar{\xi}(\mathcal{R}_j, z_0, p) \oplus <0,\ G^{\text{obs}}>$ or not. In (c), $q(\cdot \mid i,j)$ is approximated from below and above over each right-angled triangle using Interval Arithmetic. And in (d), the integration of the over-approximation of $q(\cdot \mid i,j)$ over each right-angled triangle is computed in closed-form.

$\epsilon \in [0,1]$,

$$\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} \int_{\bar{\xi}(\mathcal{R}_j, z_0, p)\oplus<0,\ G^{obs}>} q(w \mid i,j)\ dw \leq \epsilon \tag{4.9}$$

sufficiently implies

$$\texttt{Prob}\Bigg(\cup_{t\in[0,t_f]} \Big(\mathcal{E}\big(t, z_0, p\big) \cap \big(\cup_{i\in\mathcal{I}} \mathcal{O}_i^{obs}(t)\big)\Big)\Bigg) \leq \epsilon. \tag{4.10}$$

Proof. Notice that

$$\begin{aligned}
&\cup_{t\in[0,t_f]} \Big(\mathcal{E}\big(t, z_0, p\big) \cap \big(\cup_{i\in\mathcal{I}}\, \mathcal{O}_i^{\text{est}}(t)\big)\Big) \\
&= \cup_{j\in\mathcal{J}} \cup_{t\in T_j} \Big(\mathcal{E}\big(t, z_0, p\big) \cap \big(\cup_{i\in\mathcal{I}}\, \mathcal{O}_i^{\text{est}}(t)\big)\Big) && (4.11)\\
&= \cup_{j\in\mathcal{J}} \cup_{i\in\mathcal{I}} \cup_{t\in T_j} \Big(\mathcal{E}\big(t, z_0, p\big) \cap \mathcal{O}_i^{\text{est}}(t)\Big) && (4.12)\\
&\subseteq \cup_{j\in\mathcal{J}} \cup_{i\in\mathcal{I}} \big(\cup_{t\in T_j} \mathcal{E}(t, z_0, p)\big) \cap \big(\cup_{t\in T_j} \mathcal{O}_i^{\text{est}}(t)\big). && (4.13)
\end{aligned}$$

For simplicity denote $\mathtt{E}_1$ the event of $\cup_{t\in[0,t_f]}\Big(\mathcal{E}\big(t, z_0, p\big) \cap \big(\cup_{i\in\mathcal{I}}\, \mathcal{O}_i^{\text{est}}(t)\big)\Big) \neq \emptyset$, and denote $\mathtt{E}_2$ the event of $\cup_{j\in\mathcal{J}} \cup_{i\in\mathcal{I}} \big(\cup_{t\in T_j} \mathcal{E}(t, z_0, p)\big) \cap \big(\cup_{t\in T_j} \mathcal{O}_i^{\text{est}}(t)\big) \neq \emptyset$. Because $\mathtt{E}_1$ being true ensures $\mathtt{E}_2$ being true based on (4.13), then by monotonicity of probability measure,

$$\mathtt{Prob}(\mathtt{E}_1) \leq \mathtt{Prob}(\mathtt{E}_2). \tag{4.14}$$

Therefore $\mathtt{Prob}(\mathtt{E}_2) \leq \epsilon$ implies $\mathtt{Prob}(\mathtt{E}_1) \leq \epsilon$. Notice by Boole's inequality [72],

$$\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} \mathtt{Prob}\Big(\big(\cup_{t\in T_j} \mathcal{E}(t, z_0, p)\big) \cap \big(\cup_{t\in T_j} \mathcal{O}_i^{\text{est}}(t)\big) \neq \emptyset\Big) \leq \epsilon. \tag{4.15}$$

is a sufficient condition of $\mathtt{Prob}(\mathtt{E}_2) \leq \epsilon$, thus it suffices to show that (4.9) sufficiently implies (4.15).

By Theorem 38, $\cup_{t\in T_j}\mathcal{E}\big(t, z_0, p\big) \subset \bar{\xi}(\mathcal{R}_j, z_0, p)$ for arbitrary $j \in \mathcal{J}$, thus

$$\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} \int_{\cup_{t\in T_j}\mathcal{E}\big(t,z_0,p\big)\oplus<0,\ G^{\text{obs}}>} q(w \mid i, j)\ dw \leq \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} \int_{\bar{\xi}(\mathcal{R}_j,z_0,p)\oplus<0,\ G^{\text{obs}}>} q(w \mid i, j)\ dw \leq \epsilon. \tag{4.16}$$

Notice by Assumption 34, $q(w_{i,j}^{\text{obs}} \mid i, j)$ describes the probability density of $w_{i,j}^{\text{obs}}$ during T_j, thus

$$\int_{\cup_{t\in T_j}\mathcal{E}(t,z_0,p)\oplus<0,\ G^{\text{obs}}>} q(w \mid i, j)\ dw = \mathtt{Prob}\big(w_{i,j}^{\text{obs}} \in \cup_{t\in T_j}\mathcal{E}(t, z_0, p) \oplus <0,\ G^{\text{obs}}>\big). \tag{4.17}$$

Then as a result of [37, Lem. 5.1],

$$\int_{\cup_{t\in T_j}\mathcal{E}(t,z_0,p)\oplus<0,\ G^{\text{obs}}>} q(w \mid i, j)\ dw = \mathtt{Prob}\Big(\big(\cup_{t\in T_j} \mathcal{E}(t, z_0, p)\big) \cap <w_{i,j}^{\text{obs}},\ G^{\text{obs}}> \neq \emptyset\Big). \tag{4.18}$$

For simplicity denote $\texttt{E}_3$ the event that $\big(\cup_{t\in T_j}\mathcal{E}(t,z_0,p)\big)\cap <w_{i,j}^{\text{obs}},\ G^{\text{obs}}> \neq \emptyset$, and denote $\texttt{E}_4$ the event that $\big(\cup_{t\in T_j}\mathcal{E}(t,z_0,p)\big)\cap\big(\cup_{t\in T_j}\mathcal{O}_i^{\text{est}}(t)\big)\neq\emptyset$. Because $\texttt{E}_4$ being true sufficiently ensures $\texttt{E}_3$ being true due to $\cup_{t\in T_j}\mathcal{O}_i^{\text{est}}(t)\subset <w_{i,j}^{\text{obs}},\ G^{\text{obs}}>$ by Assumption 35, then by monotonicity of probability measure

$$\texttt{Prob}(\texttt{E}_4)\le\texttt{Prob}(\texttt{E}_3). \tag{4.19}$$

Therefore (4.15) holds based on (4.16), (4.18) and (4.19). □

As a sufficient condition of the chance constraint in ($\texttt{Opt-C}$), (4.9) in Theorem 39 relaxes the original collision probability as accumulated probability density integration over all elements in $\mathcal{I}\times\mathcal{J}$, where the domain of integration $\bar{\xi}(\mathcal{R}_j,z_0,p)\oplus <0,\ G^{\text{obs}}>$ is an over-approximation of the ego vehicle's footprint buffered by swept volume of any obstacle during time interval T_j. Next we focus on approximating the probability density integration with arbitrary $(i,j)\in\mathcal{I}\times\mathcal{J}$ by relaxing the domain and integrand of integration $\int_{\bar{\xi}(\mathcal{R}_j,z_0,p)\oplus<0,\ G^{\text{obs}}>} q(w\mid i,j)\,dw$.

4.4.1.2 Domain Relaxation

Recall by Theorem 38 that $\bar{\xi}(\mathcal{R}_j,z_0,p)$ can be rewritten as zonotope $<c_{\bar{\xi},j}(z_0)+A_{\bar{\xi},j}\cdot p,\ G_{\bar{\xi},j}> = <c_{\bar{\xi},j}(z_0),\ G_{\bar{\xi},j}>+A_{\bar{\xi},j}\cdot p$. Then to relax the domain of integration $\bar{\xi}(\mathcal{R}_j,z_0,p)\oplus <0,\ G^{\text{obs}}>$, we start by constructing a k-by-k grid that covers zonotope $\bar{\xi}(\mathcal{R}_j,z_0,0)\oplus <0,\ G^{\text{obs}}> = <c_{\bar{\xi},j}(z_0),\ [G_{\bar{\xi},j},G^{\text{obs}}]>$ where k is some user-specified positive integer. Each cell in the grid shares the same size and is indexed by its row and column index in the grid. Each cell in the grid is further divided into two right triangles that are indexed by 1 or -1 depending on if the lower or upper regions of the block is covered respectively. For notational convenience, denote $\mathcal{T}_{j,k_1,k_2,k_3}(z_0)\subset\mathcal{W}$ the right triangle indexed by $k_3\in\{-1,1\}$ in the block on the k_1-th row and k_2-th column of the grid that covers $<c_{\bar{\xi},j}(z_0),\ [G_{\bar{\xi},j},G^{\text{obs}}]>$. Define

$$\begin{aligned}\overline{\mathcal{T}}_j(z_0) := \big\{\mathcal{T}_{j,k_1,k_2,k_3}(z_0)\ |k_1,k_2\in\{1,2,\dots,k\},|k_3|=1,\\ \mathcal{T}_{j,k_1,k_2,k_3}(z_0)\cap <c_{\bar{\xi},j}(z_0),\ [G_{\bar{\xi},j},G^{\text{obs}}]>\neq\emptyset\big\}.\end{aligned} \tag{4.20}$$

that collects all possible $\mathcal{T}_{j,k_1,k_2,k_3}(z_0)$ intersecting with $<c_{\bar{\xi},j}(z_0),\ [G_{\bar{\xi},j},G^{\text{obs}}]>$, thus

$$<c_{\bar{\xi},j}(z_0),\ [G_{\bar{\xi},j},G^{\text{obs}}]>\subset\big(\cup_{\mathcal{T}\in\overline{\mathcal{T}}_j(z_0)}\mathcal{T}\big). \tag{4.21}$$

Notice $\bar{\xi}(\mathcal{R}_j, z_0, p) = \bar{\xi}(\mathcal{R}_j, z_0, 0) + A_{\bar{\xi},j} \cdot p$, therefore

$$\bar{\xi}(\mathcal{R}_j, z_0, p) \oplus <0,\ G^{\mathrm{obs}}> \subset (\cup_{\mathcal{T} \in \overline{\mathcal{T}}_j(z_0)} \mathcal{T}) + A_{\bar{\xi},j} \cdot p \tag{4.22}$$

and

$$\int_{\bar{\xi}(\mathcal{R}_j, z_0, p) \oplus <0,\ G^{\mathrm{obs}}>} q(w \mid i, j)\, dw \leq \sum_{\mathcal{T} \in \overline{\mathcal{T}}_j(z_0)} \int_{\mathcal{T} + A_{\bar{\xi},j} p} q(w \mid i, j)\, dw. \tag{4.23}$$

Remark 40. *Note that for any $\mathcal{T} \in \overline{\mathcal{T}}_j(z_0)$, $\mathcal{T}$ depends on z_0 and j. However we drop its dependency on z_0 and j in the remainder of this manuscript for notational simplicity. Certainly as the value of k increases, one could generate a tighter cover of $\bar{\xi}(\mathcal{R}_j, z_0, p) \oplus$ $<0,\ G^{obs}>$, but the computational consumption also increases as a trade-off due to the increasing cardinality of $\overline{\mathcal{T}}_j(z_0)$.*

4.4.1.3 Integrand Relaxation

Given arbitrary $p \in \mathcal{P}$ and $\mathcal{T} \in \overline{\mathcal{T}}_j(z_0)$, denote $w_{\mathcal{T}}^{\mathrm{ctr}}$ the vertex of the right angle in $\mathcal{T}$, then $w_{\mathcal{T}}^{\mathrm{ctr}} + A_{\bar{\xi},j}p$ gives the vertex of the right angle in $\mathcal{T} + A_{\bar{\xi},j}p$. To relax the integrand q, we start by computing the 2nd-order Taylor Expansion of $q(\cdot \mid i, j)$ centered at $w_{\mathcal{T}}^{\mathrm{ctr}} + A_{\bar{\xi},j}p$ and applying Mean Value Theorem (MVT) [90, Theorem 4.1] to eliminate higher order terms in the Taylor Expansion as:

$$\begin{aligned} q(w \mid i, j) =& q(w_{\mathcal{T}}^{\mathrm{ctr}} + A_{\bar{\xi},j}p \mid i, j) + \frac{\partial q}{\partial w}(w_{\mathcal{T}}^{\mathrm{ctr}} + A_{\bar{\xi},j}p \mid i, j) \cdot (w - w_{\mathcal{T}}^{\mathrm{ctr}} - A_{\bar{\xi},j}p) + \\ &+ \frac{1}{2}(w - w_{\mathcal{T}}^{\mathrm{ctr}} - A_{\bar{\xi},j}p)^\top \cdot \mathtt{Hess}_q(w' \mid i, j) \cdot (w - w_{\mathcal{T}}^{\mathrm{ctr}} - A_{\bar{\xi},j}p) \end{aligned} \tag{4.24}$$

where $w' \in \mathcal{T} + A_{\bar{\xi},j}p$ is some point on the line segment joining points $w_{\mathcal{T}}^{\mathrm{ctr}} + A_{\bar{\xi},j}p$ and w in $\mathcal{T} + A_{\bar{\xi},j}p$, and $\mathtt{Hess}_q$ gives the Hessian of q. Because MVT does not provide w' in a closed-form, we then drop the dependency of w' in (4.24) by bounding the Hessian of q as a matrix $H_{\mathcal{T}} \in \mathbb{R}^{2\times 2}$ where $H_{\mathcal{T}}$ is generated by taking element-wise supremum of $\mathtt{Hess}_q$ over $\mathcal{T} \oplus A_{\bar{\xi},j}\mathcal{P}$ using Interval Arithmetic [42]. Note by construction of $\mathcal{T}$ and $w_{\mathcal{T}}^{\mathrm{ctr}}$, it is guaranteed that either $w \geq w_{\mathcal{T}}^{\mathrm{ctr}} + A_{\bar{\xi},j}p$ for all $w \in \mathcal{T} + A_{\bar{\xi},j}p$ or $w \leq w_{\mathcal{T}}^{\mathrm{ctr}} + A_{\bar{\xi},j}p$ for all

$w \in \mathcal{T} + A_{\bar{\xi},j}p$, thus by definition of $H_{\mathcal{T}}$ the following inequality holds for all $w \in \mathcal{T} + A_{\bar{\xi},j}p$:

$$\begin{aligned} q(w \mid i,j) \le\ & q(w^{\text{ctr}}_{\mathcal{T}} + A_{\bar{\xi},j}p \mid i,j) + \frac{\partial q}{\partial w}(w^{\text{ctr}}_{\mathcal{T}} + A_{\bar{\xi},j}p \mid i,j) \cdot (w - w^{\text{ctr}}_{\mathcal{T}} - A_{\bar{\xi},j}p) + \\ & + \frac{1}{2}(w - w^{\text{ctr}}_{\mathcal{T}} - A_{\bar{\xi},j}p)^\top \cdot H_{\mathcal{T}} \cdot (w - w^{\text{ctr}}_{\mathcal{T}} - A_{\bar{\xi},j}p) \\ =:\ & q_{\mathcal{T}}(w, p \mid i,j). \end{aligned} \tag{4.25}$$

Note the inequality in (4.25) flips if $H_{\mathcal{T}}$ is generated by taking element-wise infimum of $\texttt{Hess}_q$, thus a under-estimation of q is constructed. However because our goal is to ensure vehicle safety with a chance of collision bounded from above, an over-estimation of q is preferred.

Remark 41. *To achieve a tighter upper bound of q on $\mathcal{T} + A_{\bar{\xi},j}p$, one can replace $H_{\mathcal{T}}$ in* (4.25) *by some matrix $H_{\mathcal{T}}(p) \in \mathbb{R}^{2\times 2}$, which is computed by taking element-wise supremum of $\texttt{Hess}_q$ over $\mathcal{T} + A_{\bar{\xi},j}p$ instead of $\mathcal{T} \oplus A_{\bar{\xi},j}\mathcal{P}$ using Interval Arithmetic. However recall that we aim for over-approximating $\int_{\bar{\xi}(\mathcal{R}_j, z_0, p)\oplus <0,\, G^{obs}>} q(w \mid i,j)\, dw$ in closed-form while maintain differentiability, when q gets complicated, taking the derivative of $H_{\mathcal{T}}(p)$ with respect to p becomes intractable easily due to the involvement of Interval Arithmetic. On the other hand, derivative of $H_{\mathcal{T}}$ with respect to p is simply 0 because by construction $H_{\mathcal{T}}$ is invariant over $\mathcal{P}$.*

4.4.1.4 Closed-form Computation

As a result of (4.23) and (4.25), the following inequality holds:

$$\int_{\bar{\xi}(\mathcal{R}_j, z_0, p)\oplus <0,\, G^{\text{obs}}>} q(w \mid i,j)\, dw \le \sum_{\mathcal{T} \in \overline{\mathcal{T}}_j(z_0)} \int_{\mathcal{T} + A_{\bar{\xi},j}p} q_{\mathcal{T}}(w, p \mid i,j)\, dw. \tag{4.26}$$

Notice that $q_{\mathcal{T}}(w, p \mid i,j)$ defined in (4.25) is indeed a quadratic polynomial of w, and $\mathcal{T} + A_{\bar{\xi},j}p$ is a simplex in $\mathbb{R}^2$. One can then compute $\int_{\mathcal{T}+A_{\bar{\xi},j}p} q_{\mathcal{T}}(w, p \mid i,j)\, dw$ in closed-form in the following theorem, and the derivative of $\int_{\mathcal{T}+A_{\bar{\xi},j}p} q_{\mathcal{T}}(w, p \mid i,j)\, dw$ with respect to p is provided in Section 4.4.3.

Theorem 42. *For any $i \in \mathcal{I}$, $j \in \mathcal{J}$, $z_0 \in \mathcal{Z}_0$, $p \in \mathcal{P}$ and $\mathcal{T} \in \overline{\mathcal{T}}_j(z_0)$ where $\overline{\mathcal{T}}_j(z_0)$ is defined as in* (4.20)*, let $A_{\bar{\xi},j} \in \mathbb{R}^{2\times n_p}$ be defined as in Theorem 38, let $q_{\mathcal{T}}$ be defined as in* (4.25)*, and let $\texttt{idx}_{k_3}(\mathcal{T}) := k_3 \in \{-1, 1\}$ denote the last index of $\mathcal{T}_{j,k_1,k_2,k_3}(z_0)$ as in* (4.20)*. Assume that positive numbers l_1 and l_2 give the lengths of horizontal and vertical right angle sides* *of $\mathcal{T}$ respectively, then*

$$\int_{\mathcal{T}+A_{\bar{\xi},j}p} q_{\mathcal{T}}(w,p \mid i,j)\, dw = \frac{1}{2\det(A_{\mathcal{T}})}\big(f_{\mathcal{T},0}(p \mid i,j) + f_{\mathcal{T},1}(p \mid i,j) + f_{\mathcal{T},2}(p \mid i,j)\big) \tag{4.27}$$

with

$$A_{\mathcal{T}} = \begin{bmatrix} \mathtt{idx}_{k_3}(\mathcal{T})/l_1 & 0 \\ 0 & \mathtt{idx}_{k_3}(\mathcal{T})/l_2 \end{bmatrix}, \tag{4.28}$$

$$\hat{H}_{\mathcal{T}} = A_{\mathcal{T}}^{-\intercal} H_{\mathcal{T}} A_{\mathcal{T}}^{-1}, \tag{4.29}$$

$$f_{\mathcal{T},0}(p \mid i,j) = q(w_{\mathcal{T}}^{ctr} + A_{\bar{\xi},j}p \mid i,j), \tag{4.30}$$

$$f_{\mathcal{T},1}(p \mid i,j) = \frac{\partial q}{\partial w}(w_{\mathcal{T}}^{ctr} + A_{\bar{\xi},j}p \mid i,j) \cdot A_{\mathcal{T}}^{-1} \cdot \begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \end{bmatrix}, \tag{4.31}$$

$$f_{\mathcal{T},2}(p \mid i,j) = \begin{bmatrix} \frac{1}{4\sqrt{3}} & \frac{1}{4\sqrt{3}} \end{bmatrix} \left(\hat{H}_{\mathcal{T}} \odot \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \right) \begin{bmatrix} \frac{1}{2\sqrt{3}} \\ \frac{1}{2\sqrt{3}} \end{bmatrix}, \tag{4.32}$$

where $\odot$ denotes the element-wise multiplication.

Proof. The claim follows from [58, Theorem 1.1] and the fact that $A_{\mathcal{T}}\big((\mathcal{T} + A_{\bar{\xi},j}p) - (w_{\mathcal{T}}^{\text{ctr}} + A_{\bar{\xi},j}p)\big)$ equals the canonical simplex $\Delta := \{(x,y) \in \mathbb{R}^2 \mid x + y \le 1, x \ge 0, y \ge 0\}$. □

4.4.2 Enhanced Online Optimization

Computation in Section 4.4.1 provides a more tractable way of enforcing vehicle safety compared to the original chance constraint in (`Opt-C`), thus we instead solve the following optimization during online planning in Risk-RTD.

$$\begin{aligned} \min_{p \in \mathcal{P}} \quad & \mathtt{cost}(z_0, p) \qquad\qquad (\mathtt{Opt\text{-}CE}) \\ \text{s.t.} \quad & \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} \sum_{\mathcal{T} \in \overline{\mathcal{T}}_j(z_0)} \int_{\mathcal{T}+A_{\bar{\xi},j}p} q_{\mathcal{T}}(w,p \mid i,j)\, dw \le \epsilon \end{aligned}$$

(`Opt-CE`) is an enhanced version of (`Opt-C`) because the chance constraint in (`Opt-CE`) sufficiently implies the chance constraint in (`Opt-C`) by Theorem 39, (4.21) and (4.25). Thus the following theorem holds based on Definition 36.

Theorem 43. *If the ego vehicle applies any feasible solution, $p^* \in \mathcal{P}$, of (`Opt-CE`) beginning from z_0 at $t = 0$, then the ego vehicle is not-at-fault with a chance of collision at most ϵ during $[0, t_f]$.*

4.4.3 Constraint Gradient

Because (`Opt-CE`) is expected to be solved in real time, providing derivatives of constraints could be helpful for speeding up the solving procedure of online optimization. To compute the gradient of chance constraint in (`Opt-CE`), it suffices to compute the derivatives of $f_{\mathcal{T},0}(p \mid i,j)$, $f_{\mathcal{T},1}(p \mid i,j)$ and $f_{\mathcal{T},2}(p \mid i,j)$ in Theorem 42 with respect to p. Note q is twice-differentiable as in Assumption 34, then

$$\frac{\partial f_{\mathcal{T},0}}{\partial p}(p \mid i,j) = \frac{\partial q}{\partial w}(w_{\mathcal{T}}^{\text{ctr}} + A_{\bar{\xi},j} p \mid i,j) \cdot A_{\bar{\xi},j}, \tag{4.33}$$

$$\frac{\partial f_{\mathcal{T},1}}{\partial p}(p \mid i,j) = \begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \end{bmatrix}^{\mathsf{T}} A_{\mathcal{T}}^{-\mathsf{T}} \mathtt{Hess}_q(w_{\mathcal{T}}^{\text{ctr}} + A_{\bar{\xi},j} p \mid i,j) \cdot A_{\bar{\xi},j}, \tag{4.34}$$

$$\frac{\partial f_{\mathcal{T},2}}{\partial p}(p \mid i,j) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}. \tag{4.35}$$

4.4.4 Parallelization

Notice the computation of $\int_{\mathcal{T}+A_{\bar{\xi},j}p} q_{\mathcal{T}}(w,p \mid i,j)\ dw$ can be achieved in parallel for all $(i,j,\mathcal{T}) \in \mathcal{I} \times \mathcal{J} \times \overline{\mathcal{T}}_j(z_0)$. Thus we summarize the parallel computation of the chance constraint in (`Opt-CE`) in Algorithm 2. $\overline{\mathcal{T}}_j(z_0)$ is generated in line 2 for all $j \in \mathcal{J}$. Within the parallel for loop from line 3 to 9, $w_{\mathcal{T}}^{\text{ctr}}$ and $A_{\mathcal{T}}$ are first generated in line 4, the over-approximation of the probability integration is computed from line 5 to 6, and gradient of the over-approximation is computed from line 7 to 8. Finally the chance constraint and its gradient are evaluated in line 10 and 11.

Algorithm 2 Chance Constraint Parallelization

```
Require: $z_0, p', \{f(\cdot \mid i,j)\}_{i\in\mathcal{I}, j\in\mathcal{J}}, \{\mathcal{R}_j\}_{j\in\mathcal{J}}$
    Generate $\{\overline{\mathcal{T}}_j(z_0)\}_{j\in\mathcal{J}}$ as in (4.20) using $\{\mathcal{R}_j\}_{j\in\mathcal{J}}$
    Parfor $(i,j,\mathcal{T}) \in \mathcal{I} \times \mathcal{J} \times \overline{\mathcal{T}}_j(z_0)$ do
        Compute $w_{\mathcal{T}}^{\text{ctr}}$ and $A_{\mathcal{T}}$
        Compute $f_{\mathcal{T},0}, f_{\mathcal{T},1}, f_{\mathcal{T},2}$ as in (4.30), (4.31), (4.32)
        $c(p',i,j,\mathcal{T}) \leftarrow \frac{1}{2\det(A_{\mathcal{T}})}(f_{\mathcal{T},0} + f_{\mathcal{T},1} + f_{\mathcal{T},2})$
        Compute $\frac{\partial f_{\mathcal{T},0}}{\partial p}, \frac{\partial f_{\mathcal{T},1}}{\partial p}, \frac{\partial f_{\mathcal{T},2}}{\partial p}$ as in (4.33), (4.34), (4.35)
        $dc(p',i,j,\mathcal{T}) \leftarrow \frac{1}{2\det(A_{\mathcal{T}})}\left(\frac{\partial f_{\mathcal{T},0}}{\partial p} + \frac{\partial f_{\mathcal{T},1}}{\partial p} + \frac{\partial f_{\mathcal{T},2}}{\partial p}\right)$
    End parfor
    con$\leftarrow \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} \sum_{\mathcal{T}\in\overline{\mathcal{T}}_j(z_0)} c(p',i,j,\mathcal{T})$
    dcon$\leftarrow \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} \sum_{\mathcal{T}\in\overline{\mathcal{T}}_j(z_0)} dc(p',i,j,\mathcal{T})$
```

4.5 Extension on Risk Threshold Selection

So far we have assumed the risk threshold ϵ is some constant in $[0,1]$. However it is unclear how the risk threshold ϵ should be selected. Most works evaluate the ego vehicle performance at different thresholds and arbitrarily select a threshold that they deem fit and keep it constant for the entire time during the operation of a robot [65, 105, 113]. In this section we present a novel methodology for systematically varying the risk threshold based on how likely it is for the passengers of the vehicle to be injured should a crash occur during the motion plan. For the rest of this section, we shall refer to this likelihood of injury should a crash occur as the risk of injury. This methodology allows the ego vehicle to only be conservative when it has to ensure passenger safety (i.e., when the risk of injury is high), and be less conservative in situations where the passenger are less likely to be injured should a collision occur (i.e., when the risk of injury is low).

This methodology hinges on being able to compute this risk of injury at the beginning on each motion plan. To compute the risk of injury, we make the following assumption, whose satisfaction can be satisfied by, for example, using lidar data in conjunction with an extended Kalman filter [1, 5].

Assumption 44. *There exists some upper and lower bounds of the longitudinal velocity of the i-th obstacle given by $\overline{u_i}$ and $\underline{u_i}$ such that $u_i(t) \in [\underline{u_i}, \overline{u_i}]$, for all $t \in [0, t_f]$.*

Given the estimated velocity bounds of sensed obstacles, Risk-RTD then computes the relative velocity of the ego vehicle with respect to each obstacle. It is important to note that the ego vehicle velocity may vary over the time interval $[0, t_{\mathrm{m}}]$ based on the planned action. As such, for each obstacle and ego vehicle pair, the relative velocity will vary over the time interval $[0, t_{\mathrm{m}}]$. Thus, to ensure that we do not underestimate the risk of injury, for each obstacle and ego vehicle pair we must compute the supremum of this range of relative velocities. Such supremum is computable because velocity of any obstacle is bounded as assumed in Assumption 44 and velocity of ego vehicle is bounded by tracking error u^{small} as derived the proof of Lemma 21 in Appendix B. Let u_i^{rel} for any $i \in \mathcal{I}$ be the supremum of the estimated relative velocity of the ego vehicle with respect to the i-th obstacle during $[0, t_{\mathrm{m}}]$. We then compute the estimated risk of serious injuries (MAIS3+) [32] to passengers should a collision occur with the i-th obstacle as $\alpha^{inj}(u_i^{rel})$ where $\alpha^{inj} : \mathbb{R} \to [0,1]$ gives the binary logistic model with parameters adopted from [13, Table A]. For each obstacle, Risk-RTD then decides how conservatively this obstacle should be treated by either 1) enforcing a near zero chance of collision with the obstacle if the risk of serious injury is greater than some user-defined threshold α^{crit}, or 2) selecting risk levels from human crash data if it can afford to treat this obstacle less conservatively because the risk of serious injury is less than α^{crit}.

In the first case when $\alpha^{inj}(u_i^{rel}) \geq \alpha^{crit}$, precisely enforcing 0 chance of collision could be too conservative because the probabilistic description of an obstacle may have non-zero probability density for all $w \in \mathcal{W}$. Therefore we instead relax the chance of collision to some user-specified number $\eta > 0$ sufficiently close to 0, and make the following assumption.

Assumption 45. *For each $(i,j) \in \mathcal{J} \times \mathcal{I}$, there exists a 2-row matrix $G_{i,j}^{ovr}(\eta)$ such that*

$$1 - \int_{<\mathbb{E}_q(w|i,j),\ G_{i,j}^{ovr}(\eta)>} q(w \mid i,j) dw \leq \eta \tag{4.36}$$

where $\mathbb{E}_q(w \mid i,j)$ is the expectation of the location of the i-th obstacle over T_j satisfying q.

Based on Assumption 45, zonotope $<\mathbb{E}_q(w \mid i,j),\ G_{i,j}^{\text{ovr}}(\eta)>$ captures almost all probability mass of q given $(i,j) \in \mathcal{I} \times \mathcal{J}$. Therefore $\bar{\xi}(\mathcal{R}_j, z_0, p) \cap <\mathbb{E}_q(w \mid i,j),\ G_{i,j}^{\text{ovr}}(\eta)> = \varnothing$ implies that $\texttt{Prob}\Big(\big(\cup_{t \in T_j} \mathcal{E}(t, z_0, p)\big) \cap \big(\cup_{t \in T_j} \mathcal{O}_i(t)\big) \neq \varnothing\Big) \leq \eta$. For notational simplicity, define $\mathcal{I}^{\eta\text{-risk}} := \{i \in \mathcal{I} \mid \alpha^{inj}(u_i^{rel}) \geq \alpha^{crit}\}$ and $\mathcal{I}^{\text{risk}} := \{i \in \mathcal{I} \mid \alpha^{inj}(u_i^{rel}) < \alpha^{crit}\}$.

For the i-th obstacle with $i \in \mathcal{I}^{\text{risk}}$ we apply our chance constraint from (Opt-CE) and set the risk threshold to match human risk levels. We estimate human risk levels based on the percentage of licensed drivers that are involved in crashes per year using crash statistics from the National Highway Traffic Safety Agency (NHTSA) [29]. We leverage the crash data reported at various driving speeds to get a set of human risk data that varies as a function of traveling velocity. The following polynomial $\epsilon^{\text{dyn}} : \mathbb{R} \to \mathbb{R}$ was used to fit these data with an R-squared value of 0.965 in order to obtain a closed-form approximation of these human risk levels:

$$\begin{aligned} \epsilon^{\text{dyn}}(u(t)) =& 1.53 \times 10^{-8} u^4(t) - 1.84 \times 10^{-6} u^3(t) + \\ &+ 5.49 \times 10^{-5} u^2(t) + 1.1 \times 10^{-4} u(t) + 2.23 \times 10^{-5} \end{aligned} \tag{4.37}$$

where $u(t)$ is in miles per hour (mph). ϵ^{dyn} is a continuous function that gives the risk levels human drivers use as as function of their traveling velocity. Recall that $[0, t_{\text{m}}]$ is the time interval that the driving maneuver is executed in. Depending on the driving maneuver, $u(t)$ will vary over $t \in [0, t_{\text{m}}]$. To ensure that we do not exceed human risk levels over the execution of the trajectory we select a lowest human risk threshold, i.e. $\inf_{t \in [0,t_{\text{m}}]} \epsilon^{\text{dyn}}(u(t))$, over our range of velocities during the trajectory. The following theorem shows that $\inf_{t \in [0,t_{\text{m}}]} \epsilon^{\text{dyn}}(u(t))$ is piece-wise differentiable, which can be helpful to increase the solving procedure of online optimization as presented in the next subsection.

Theorem 46. *Let u be the longitudinal velocity of the solution to* (3.39) *from initial condition $z_0 \in \mathcal{Z}_0$ with control parameter $p \in \mathcal{P}$. There exists some piece-wise differentiable function $\epsilon : \mathcal{Z}_0 \times \mathcal{P} \to [0, +\infty)$ such that $\epsilon(z_0, p) = \inf_{t\in[0,t_m]} \epsilon^{dyn}(u(t))$.*

Proof. See Appendix D. □

Note that in Theorem 46 we have abused notation and have allowed ϵ to vary as a function of z_0 and p. This varying threshold allows us to dynamically select the risk threshold based on the risk of injury to the passenger beginning of each planning iteration, instead of arbitrarily setting a constant ϵ value as done in the earlier chance constraint in (`Opt-CE`). With this varying threshold we instead solve the following optimization:

$$\begin{aligned}
&\min_{p\in\mathcal{P}} \texttt{cost}(z_0, p) && (\texttt{Opt-CV})\\
&\text{s.t. } \xi\big(\mathcal{R}_j, z_0, p\big) \cap {<}\mathbb{E}_q(w \mid i, j),\ G^{\text{ovr}}_{i,j}(\eta){>} \neq \varnothing, \forall i \in \mathcal{I}^{\eta\text{-risk}}, j \in \mathcal{J}\\
&\qquad \sum_{i\in\mathcal{I}^{\text{risk}}} \sum_{j\in\mathcal{J}} \sum_{\mathcal{T}\in\overline{\mathcal{T}}_j(z_0)} \int_{\mathcal{T}+A_j p} q_{\mathcal{T}}(w, p \mid i, j)\, dw \le \epsilon(z_0, p)
\end{aligned}$$

where $\texttt{cost} : \mathcal{Z}_0 \times \mathcal{P} \to \mathbb{R}$ is a user-specified cost function. The first constraint conservatively bounds the uncertain region with a zonotope and ensures that any solution found by Risk-RTD has bounded chance of colliding with all $\mathcal{O}_i$ such that $i \in \mathcal{I}^{\eta\text{-risk}}$ as assumed in Assumption 45. The second constraint is similar to the chance constraint introduced in (`Opt-CE`), except that $\epsilon(z_0, p)$ has replaced ϵ and it only accounts for all $\mathcal{O}_i$ such that $i \in \mathcal{I}^{\text{risk}}$. The not-at-fault behavior of the vehicle is given by the following theorem.

Theorem 47. *If the ego vehicle applies any feasible solution, $p^* \in \mathcal{P}$, of (`Opt-CV`) beginning from z_0 at $t = 0$, then the ego vehicle is not-at-fault with a chance of collision at most $\epsilon(z_0, p^*) + |\mathcal{I}^{\eta\text{-risk}}||\mathcal{J}|\eta$ during $[0, t_f]$.*

Proof. Through the same reasoning as in the proof of Theorem 39, it suffices to show that

$$\sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} \texttt{Prob}\Big(\big(\cup_{t\in T_j} \mathcal{E}(t, z_0, p^*)\big) \cap \big(\cup_{t\in T_j} \mathcal{O}_i(t)\big) \neq \varnothing\Big) \le \epsilon(z_0, p^*) + |\mathcal{I}^{\eta\text{-risk}}||\mathcal{J}|\eta. \tag{4.38}$$

In addition, as an analog to (4.16) we have

$$\begin{aligned}
&\sum_{i\in\mathcal{I}^{\text{risk}}} \sum_{j\in\mathcal{J}} \int_{\cup_{t\in T_j}\mathcal{E}\big(t,z_0,p\big)\oplus<0,\ G^{\text{obs}}>} q(w \mid i, j)\, dw \le\\
&\qquad\qquad \le \sum_{i\in\mathcal{I}^{\text{risk}}} \sum_{j\in\mathcal{J}} \int_{\bar{\xi}(\mathcal{R}_j, z_0, p)\oplus<0,\ G^{\text{obs}}>} q(w \mid i, j)\, dw \le \epsilon(z_0, p^*).
\end{aligned} \tag{4.39}$$

Notice that the set intersection constraints in (`Opt-CV`) ensure

$$\texttt{Prob}\Big(\big(\cup_{t\in T_j}\mathcal{E}(t,z_0,p^*)\big)\cap\big(\cup_{t\in T_j}\mathcal{O}_i^{\text{est}}(t)\big)\neq\varnothing\Big)\leq\eta \tag{4.40}$$

for all $i\in\mathcal{I}^{\eta\text{-risk}}$ by Assumption 45. Then based on (4.18), (4.19) and the fact that $\mathcal{I}=\mathcal{I}^{\eta\text{-risk}}\cup\mathcal{I}^{\text{risk}}$, (4.38) follows by iteratively adding (4.40) to (4.39) for all $i\in\mathcal{I}^{\eta\text{-risk}}$ and for all $j\in\mathcal{J}$. □

4.6 Experiments

This section describes multiple experiments to demonstrate the performance of Risk-RTD. We first verify tightness and computational efficiency of the proposed closed-form over-approximation of risk of collision. Next we test the proposed Risk-RTD framework in simulation with randomly generated scenarios. All experiments are conducted in MATLAB R2021b on a laptop with an 8 Cores Intel i9-10980HK CPU, an Nvidia GeForce RTX 3080 GPU, and 32GB RAM. Involved parallel implementation for experiments is achieved using CUDA 11.0. Among all experiments, we adopt vehicle and control parameters, families of desired trajectories, initial condition and control parameter spaces, and the collection of zonotope reachable sets $\{\mathcal{R}_j\}_{j\in\mathcal{J}}$ generated with $\Delta t=0.01$ [sec] from REFINE in Chapter 3.

4.6.1 Evaluation on Tightness and Computational Efficiency

For effective motion planning, we expect a tight over-approximation of the risk of collision. To evaluate the tightness of proposed over-approximation, we compare the proposed method of chance integration approximation against Monte-Carlo integration [17, Chapter 4], Chance-Constrained Parallel Bernstein Algorithm (CCPBA) [105, Chapter 6] and Cantalli's inequality [113, eq. (17)] through 3039 randomly generated tests. In each test, (j,i,z_0,p) is randomly chosen from $\mathcal{J}\times\mathcal{I}\times\mathcal{Z}_0\times\mathcal{P}$ where $\mathcal{I}=\{1\}$, and q is set to be the probability density function of a 2D Gaussian distribution with randomly generated mean and variance. Assume the only obstacle is a dynamic vehicle that has zero heading and shares the same size of footprints with the ego vehicle, then G^{obs} is chosen such that $<0,\ G^{\text{obs}}>=\mathcal{O}^{\text{ego}}\oplus\texttt{int}([0,0]^\intercal,[\nu^{\text{obs}}\cdot\Delta t,0]^\intercal)$ where $\nu^{\text{obs}}=25$ [mps]. Note such choice of G^{obs} is used in all the following experiments as well.

We treat Monte-Carlo integration of $q(\cdot\mid j,i)$ over $\bar{\xi}(\mathcal{R}_j,z_0,p)\oplus<0,\ G^{\text{obs}}>$ as the ground truth of chance integration $\int_{\bar{\xi}(\mathcal{R}_j,z_0,p)\oplus<0,\ G^{\text{obs}}>}q(w\mid j,i)\ dw$. Such Monte-Carlo

Method	Avg. Absolute Error	Max. Absolute Error	Avg. Runtime [ms]	Max. RunTime [ms]
Cantelli	0.4366	0.5533	0.0140	0.0663
CCPBA	0.1881	0.2149	1.1012	3.5820
Risk-RTD	0.0073	0.0523	0.4372	2.3150

Table 4.1: Integration error of Gaussiandistribution compared to Monte-Carlo integration as the ground truth.

integration is performed by randomly sampling the world space 10^6 times according to q and counting the number of samples that fall inside $\bar{\xi}(\mathcal{R}_j, z_0, p) \oplus <0,\ G^{\text{obs}}>$. With the proposed method being utilized, integration area $\bar{\xi}(\mathcal{R}_j, z_0, p) \oplus <0,\ G^{\text{obs}}>$ is covered by a 24-by-24 grid for the generation of $\overline{\mathcal{T}}_j(z_0)$. Because CCPBA can only integrate over a polynomial set, to utilize CCPBA we need to approximate $\bar{\xi}(\mathcal{R}_j, z_0, p) \oplus <0,\ G^{\text{obs}}>$ as the superlevel set of some degree-4 polynomial function $\rho : \mathcal{W} \to \mathbb{R}$ solved by the following polynomial optimization.

$$\begin{aligned} \min_{\rho} \quad & \int_{\mathcal{W}} \rho(w)\, dw && \texttt{(Poly-Opt)} \\ & \rho(w) - 1 \geq 0, \quad \forall w \in \bar{\xi}(\mathcal{R}_j, z_0, p) \oplus <0,\ G^{\text{obs}}> \\ & \rho(w) \geq 0, \quad \forall w \in \mathcal{W} \end{aligned}$$

By construction, the optimal solution ρ of `(Poly-Opt)` is an over-approximation of the indicator function of $\bar{\xi}(\mathcal{R}_j, z_0, p) \oplus <0,\ G^{\text{obs}}>$. Note ρ is constructed conservatively because polynomial reachable sets are in general more conservative than zonotope reachable sets as observed in Chapter 3.

As shown in Figure 4.3, compared to CCPBA, the proposed method provides much tighter over-approximation of the ground truth generated by Monte-Carlo integration in all 3039 testing scenarios. This is because CCPBA has to integrate over a larger area than the proposed method. In most scenarios, Cantelli's inequality gives even more conservative over-approximation of the risk of collision compared to CCPBA. Statistic of computation error is illustrated in Table 4.1 along with the associated average and maximum times to approximate the probability integration. Among all tests, Cantelli achieves the lowest computation time of the approximation and CCPBA needs the most time to approximate a chance integration in general.

To test generality of the proposed method, we reuse aforementioned 3039 random tests except that this time q is set to be the probability density of either a randomly generated bivariate beta distribution [78] or a multimodel distribution as a mixture of two randomly generated 2D Gaussian distributions. Ground truth of the probability density integration

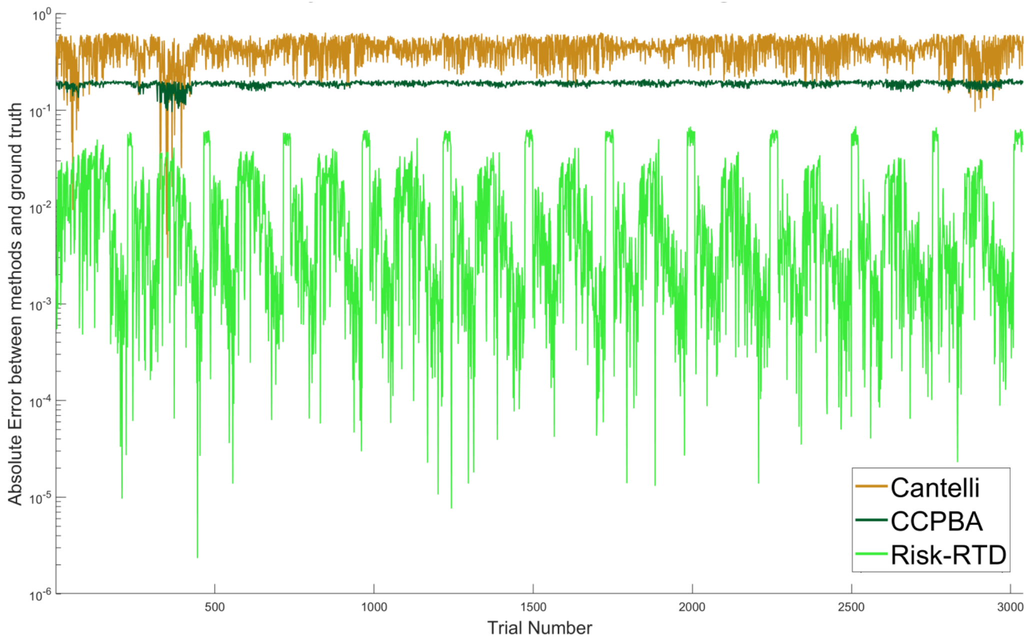


Figure 4.3: This shows the testing results for the risk of collision estimation error between the each of the comparison methods and the ground truth risk of collision generated via Monte-Carlo integration. Risk-RTD is able to get a significantly lower risk estimation error in most scenarios.

Exponential Family	Avg. Absolute Error	Max. Absolute Error
Gaussian	0.0073	0.0523
Bivariate beta	0.0055	0.0172
Multimodal	0.0138	0.0435

Table 4.2: Integration error of multiple exponential families using the proposed method.

is also generated using Monte-Carlo integration. Because CCPBA is unable to handle a distribution that is not normal, we only test using Risk-RTD. As shown in Table 4.2, the proposed method is able to consistently provide tight over-approximation of the chance integration when q describes a 2D beta distribution or a multimodel distribution.

4.6.2 Simulation

We test the performance of Risk-RTD under a number of 3-lane highway driving scenarios with lane width as 3.7 [m]. In each scenario, a full-size FWD vehicle with $u(0) = 20$ [m/s] and parameters as shown in Table 3.1 is treated as the ego vehicle. The ego vehicle is controlled with the robust feedback linearization controller proposed in Chapter 3, and is tasked to autonomously navigate itself through the traffic. The probability density function q is assumed to describe a Gaussian distribution so that CCPBA can be applied.

4.6.2.1 Single Planning Iteration

In this experiment we compare the performance of Risk-RTD to that of CCPBA, and Cantelli MPC in simulation over 10 randomly generated scenarios. Each scenario contains 1 static obstacle and between 4 to 9 dynamic obstacles, where the number of dynamic obstacles is randomly selected for each scenario. The initial speed of all dynamic obstacles are also randomly sampled from between 15 [m/s] to 25 [m/s] in each scenario. The ego vehicle is randomly initialized in a lane, and is commanded to navigate itself to a given waypoint in a different lane within a single planning iteration with $t_{\text{plan}} = 3$ [sec] and limited risk of collision as 5%. The waypoint is provided in a way that the ego vehicle is expected to reach by a lane change maneuver within 6 [sec] without any collision. Each scenario is simulated for 200 trials, so these result in a total of 2000 simulation cases.

Among the 200 trials of the same scenario, for arbitrary $(j,i) \in \mathcal{J} \times \mathcal{I}$, the i-th obstacle is always initialized with the same states but follows different trajectories that satisfies the same probability density function $q(\cdot \mid j,i)$ during T_j. In other words, these trajectories are selected such that the locations of the i-th obstacle at the middle time of T_j are randomly sampled from $q(\cdot \mid j,i)$ for each trial, where $q(\cdot \mid j,i)$ is kept constant for all trials of the same scenario for any $(j,i) \in \mathcal{J} \times \mathcal{I}$. In particular, for each $(j,i) \in \mathcal{J} \times \mathcal{I}$, q is generated such that $\mathcal{E}_q(w \mid j,i)$ results in a trajectory on the center of a lane with constant speed as j increases within $\mathcal{J}$. And the standard deviation $\sigma_{j,i}$ of $q(\cdot \mid j,i)$ is chosen such that the $3\sigma_{j,i}$-region of this Gaussian distribution covers the area of width $3.7 - W$ and length $L + u_i(0) \cdot \Delta_t$. Note the choice of width $3.7 - W$ ensures that the footprint of the obstacle stays inside the lane with a probability more than 99.7%.

Table 4.3 presents the statistic of simulation results of Risk-RTD, CCPBA and Cantelli MPC for the experiment on single planning iteration. In this experiment, a success is defined as being able to find a feasible lane change maneuver. Risk-RTD is able to successfully execute the lane change maneuver 81.8% among all simulation cases, and among the other 18.2% of simulation cases it either decides to stay in lane, or is unable to find a solution and executes its fail-safe stopping maneuver. In addition, Risk-RTD does not have any crashes in these 2000 simulation cases. CCPBA is able to successfully execute the lane change maneuver 55.2% among all simulation cases, and it decides to either stay in lane or come to a safe stop 44.8% of simulation cases. CCPBA also does not have any crashes. Cantelli MPC is only able to successfully execute the lane change maneuver 9.1% among all simulation cases. It either stays in lane or comes to a safe stop among 55.5% of all simulation cases, and it crashes in 35.4% of all simulation cases. Risk-RTD is able to achieve a higher success rate than CCPBA and Cantelli MPC due to the fact that it is able to more closely approximate the actual risk of collision as indicated in Section 4.6.1.

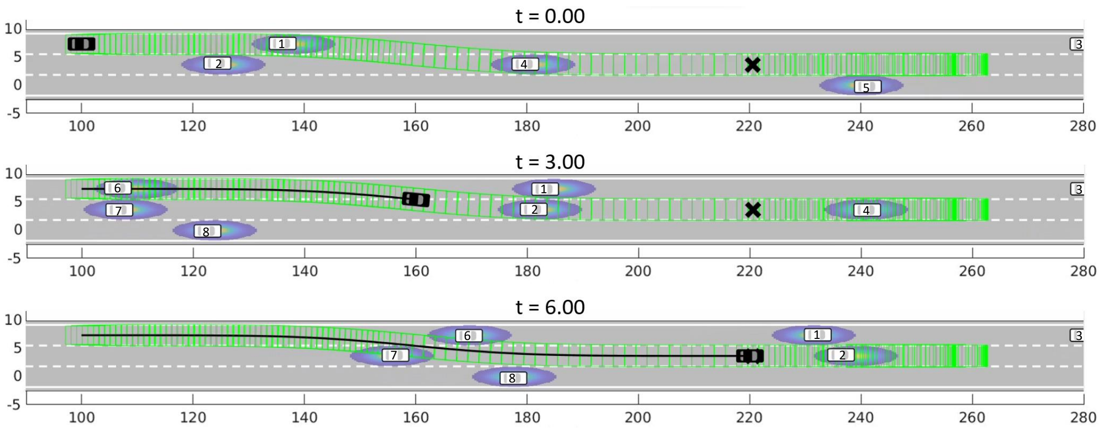
Method	Success [%]	Crash [%]	Other Action [%]	Online Planning Runtime (Avg., Max.) [sec]
Risk-RTD	81.8	0.0	18.2	(0.812, 0.907)
CCPBA	55.2	0.0	44.8	(0.291, 0.396)
Cantelli MPC	9.1	35.4	55.5	(0.643, 0.658)

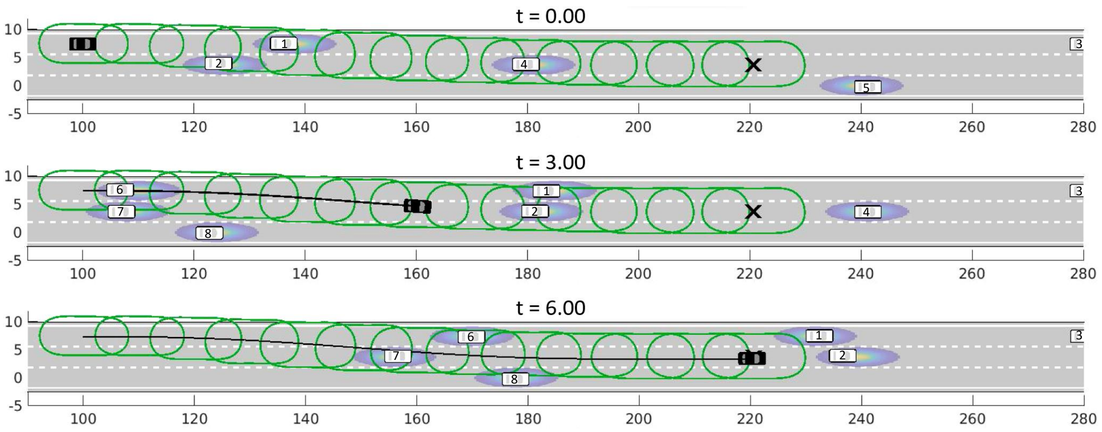
Table 4.3: Single planning iteration results using Risk-RTD, CCPBA and Cantelli MPC. "Other Action" encompasses the trials where each method does not complete the lane change manuever, but instead executes a speed change maneuver, a direction change maneuver, a safe stop manuever, or just decides to keep driving in lane.

This allows Risk-RTD to generate plans that navigate the ego vehicle through more difficult scenarios, while conservative over-approximation of the risk of collision is still possible to yield plans that are infeasible. Cantelli MPC has a relatively substantial number of crashes, because unlike Risk-RTD and CCPBA, Cantelli MPC does not have a failsafe manuever for when it cannot find a solution. As a result when Cantelli MPC cannot find a solution, it just maintains the ego vehicle's velocity within the lane while keeping searching, thus a crash may occur if there is an obstacle in front of the ego vehicle. Figure 4.4 provides an example of tested simulation case in which the chance of collision is limited by 5% and the ego vehicle is able to reach the provided waypoint through a lane change maneuver without a collision using Risk-RTD and CCPBA. However because Cantelli fails to find a feasible plan, the ego vehicle ends up with a crash.

Note both Risk-RTD and CCPBA had 0 crashes for these 200 trials due to the fact that they still over-approximate the risk of collision. In the plans that they are able to execute, the actual probability of crashing evaluated via Monte-Carlo integration is usually around 0.1%, which is much smaller than 5%, so it is possible to see no crash within just 2000 trials. A detailed analysis of conservatism of the proposed probability integration is provided in Section 4.6.3.

To test how different values of risk threshold affects the performance of the proposed method, the same 2000 simulation cases are tested again with Risk-RTD with the risk of collision limited by 0.5%, 5%, 10% and $\epsilon(z_0, p)$ presented in Theorem 46. Among all 2000 simulation cases, $\epsilon(z_0, p)$ results in staying between 0.01 and 0.02. Table 4.4 presents the statistic of simulation results using Risk-RTD under various risk thresholds. Clearly as the risk threshold increases, Risk-RTD gets easier to find a feasible solution to fulfil the task, thus the success rate increases as well. Figure 4.5 provides an example of tested simulation case in which Risk-RTD is used to navigate the ego vehicle towards the provided waypoint. With the risk threshold being 0.005, Risk-RTD is unable to find a feasible lane change maneuver, thus executes a speed change maneuver to follow the traffic. With the


(a) Risk-RTD utilized.


(b) CCPBA utilized.

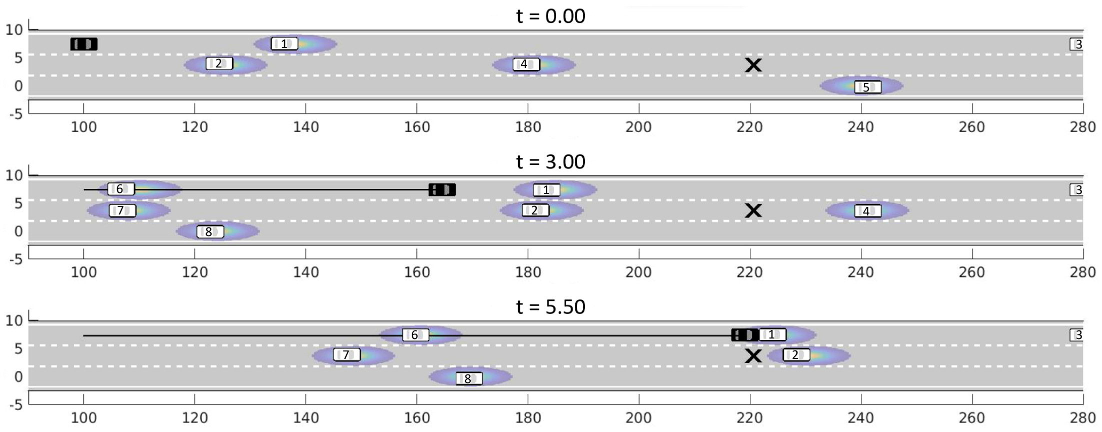

(c) Cantelli MPC utilized.

Figure 4.4: Example of a simulated scenario in which Risk-RTD and CCPBA are able to navigate the ego vehicle (black) to the provided waypoint (black cross) through a lane change maneuver solved by one planning iteration, while Cantelli MPC results in a crash. Forward reachable sets are shown in green. Obstacles are shown in white and are marked by their indices to make them trackable among different time instances.

Risk Threshold	**Success [%]**	**Crash [%]**	**Other Action [%]**
0.005	54.6	0.0	45.4
0.05	81.8	0.0	18.2
0.10	90.6	0.0	9.4
$\epsilon(z_0, p) \in [0.01, 0.02]$	76.8	0.0	33.2

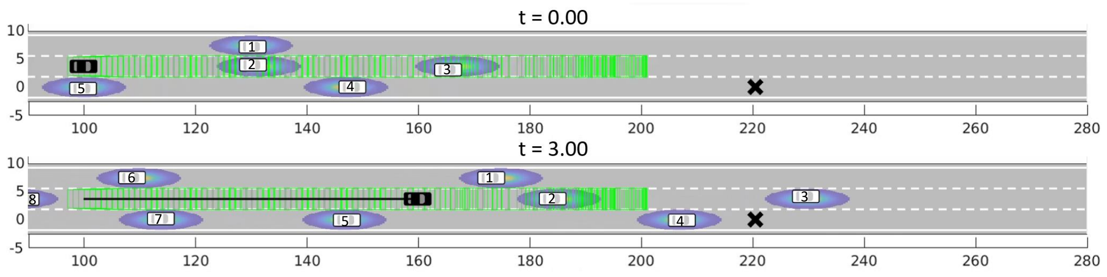
Table 4.4: Single planning iteration results using Risk-RTD under various risk thresholds. "Other Action" encompasses the trials where each method does not complete the lane change manuever, but instead executes a speed change maneuver, a direction change maneuver, or a safe stop manuever.

risk threshold being 0.05 and $\epsilon(z_0, p)$, Risk-RTD is able to find a feasible lane change maneuver that only results in a lateral displacement of half of the lane width. With the risk threshold being 0.10, Risk-RTD is able to find a feasible lane change maneuver that results in a lateral displacement of three quarters of the lane width.

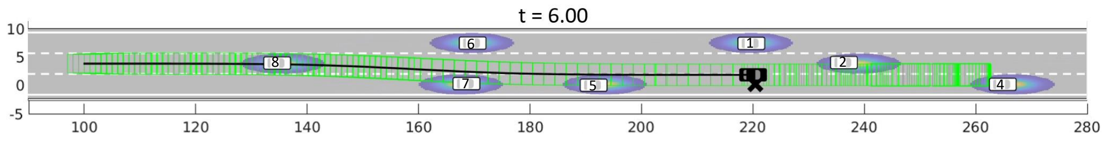
4.6.2.2 Multiple Planning Iterations

We compare Risk-RTD against CCPBA and Cantelli MPC over a 1000 randomly generated 3-lane highway scenarios in simulation with risk threshold set to be a constant value 0.05. In each simulation scenario the ego vehicle is expected to navigate through dynamic traffic for 1000[m] from a given initial position with $t_{\text{plan}} = 3$[sec]. Each highway scenario contains 3 static obstacles and a number of moving vehicles as dynamic obstacles, where the number of moving vehicles in each scenario randomly varies between 5 and 25. Initializations of each dynamic obstacle and its corresponding probility density function q with each $j \in \mathcal{J}$ are performed in the same way as explained in Section 4.6.2.1. The ego vehicle is initialized in the center of the first (bottom) lane with zero heading, zero lateral velocity, zero yaw rate, and initial longitudinal speed as 20 [m/s].

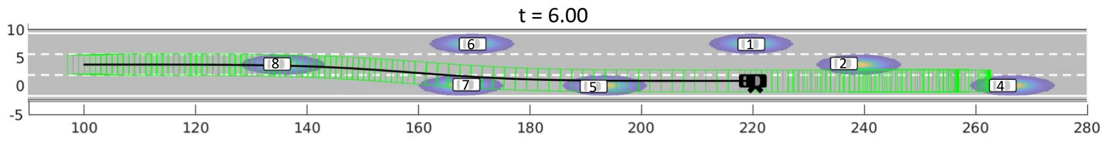
Similar as in Section 4.6.2.1, each scenario is simulated for 10 trials among which each dynamic obstacle follows different trajectories that satisfy the same series of probability density functions. The uncertain observation of each dynamic obstacle is described in the same way as in the single planning iteration highway experiment. Table 4.5 presents the statistic of simulation results using Risk-RTD, CCPBA and Cantelli MPC among all the 10000 simulation cases. In this experiment, a success is defined as being able to successfully navigate through the entire 1000 [m] highway and reach the goal. Risk-RTD is able to successfully navigate through the highway 78.9% of the time, comes to a safe stop partially through the highway 20.7% of the time and crashes 0.4% of the time. CCPBA is able to successfully through the highway 19.4% of the time, comes to a safe stop 79.6% of the time and crashes 1.0% of the time. Cantelli MPC is only able to successfully through the



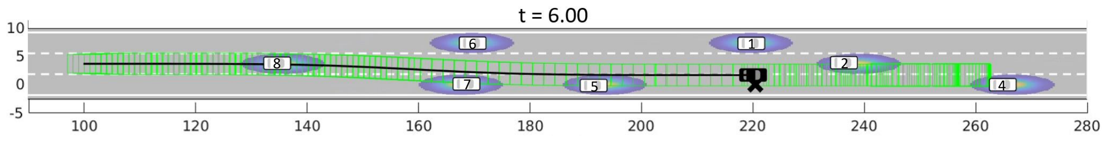
(a) Risk threshold as 0.005.



(b) Risk threshold as 0.050.



(c) Risk threshold as 0.100.



(d) Risk threshold as $\epsilon(z_0, p)$.

Figure 4.5: Example of a simulated scenario in which Risk-RTD is utilized under various risk thresholds. Ego vehicle and its trajectory are shown in black, forward reachable sets are shown in green, and the provided waypoint is shown as the black cross. Obstacles are shown in white and are marked by their indices to make them trackable among different time instances.

highway 17.5% of the time, crashed 82.5% of the time, and never comes to a safe stop due to the lack of a braking maneuver. Compared to the single planning iteration experiments, Risk-RTD has a small drop in success rate. This is expected as over a highway scenario of multiple planning iterations, once Risk-RTD has executed an action, it may start the next planning iteration from an initial condition where it is difficult to find a solution that satisfies the constraint. This is also the case for CCPBA and Cantelli MPC.

Risk-RTD is also compared against deterministic approaches REFINE and SOS-RTD to see how chance constraint improves the aggressiveness of the driving behavior. As shown in Table 4.5, success rates of REFINE and SOS-RTD are significantly lower than that of Risk-RTD, because deterministic approaches are more conservative and try to avoid a much larger area to achieve fully safety. However, both deterministic approaches results

Method	Success [%]	Crash[%]	Safely Stop [%]	Avg. Speed [m/s]
Risk-RTD	78.9	0.4	20.7	18.1328
CCPBA	19.4	1.0	79.6	15.8395
Cantelli MPC	17.5	82.5	0.0	16.6322
REFINE	61.6	0	38.4	16.5927
SOS-RTD	14.6	0	85.4	15.3302

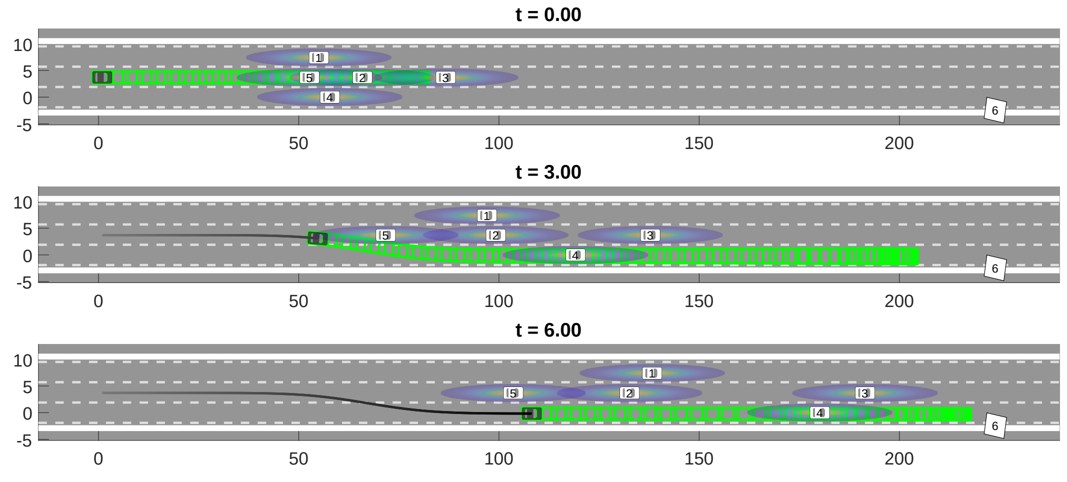
Table 4.5: Statistic of simulation results using Risk-RTD, CCPBA, Cantelli MPC, REFINE and SOS-RTD.

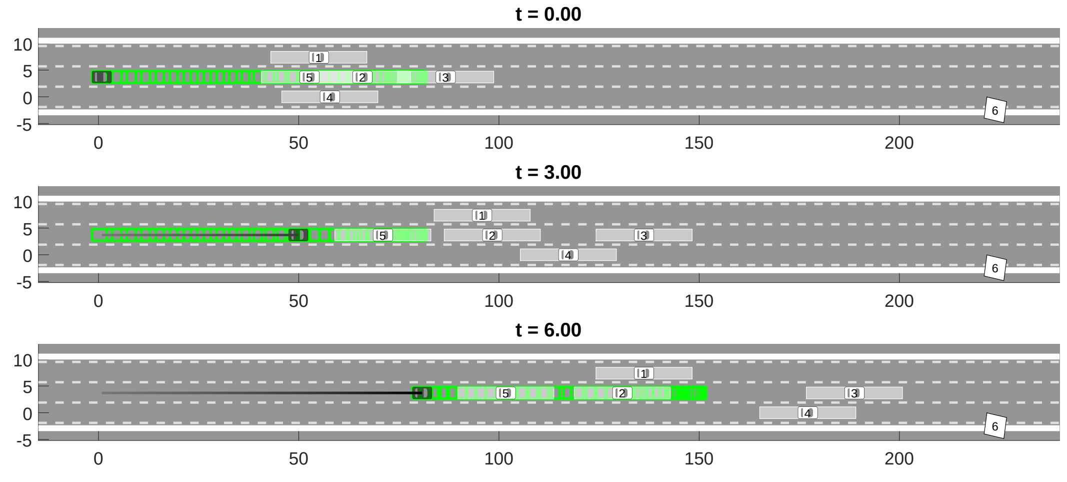
in zero crash rates. Figure 4.6 provides an example of tested simulation scenarios with Risk-RTD, REFINE and SOS-RTD. It can be seen that Risk-RTD allows the ego vehicle to travel aggressively and try to surpass moving obstacles in the middle lane. However with REFINE being applied, the ego vehicle fails to find a feasible plan at the second planning iteration thus execute a stopping maneuver. And later on as its speed is decreased till $t = 6$[s], the ego vehicle finds a feasible plan again and starts following the moving vehicle ahead. On the other hand, because the traffic is dense at time $t = 0$[s] and the polynomial reachable sets are more conservative than zonotope reachable sets, the ego vehicle tries to vary its speed but fails and ends up with a stop.

4.6.3 Analysis on Tightness of Probability Integration

Recall among all simulation cases in Section 4.6.2.1, Risk-RTD results in 0 crash rate even the risk threshold is set to $0.10 = 10\%$. This suggests the proposed over-approximation of risk of collision during the planning horizon, $\sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}^{\text{risk}}}\sum_{\mathcal{T}\in\overline{\mathcal{T}}_j(z_0)}\int_{\mathcal{T}+A_{\bar{\xi},j}p} q_{\mathcal{T}}(w,p \mid j,i)\; dw$, is still too conservative compared to the actual risk of collision $\texttt{Prob}\Big(\cup_{t\in[0,t_\text{f}]}\Big(\mathcal{E}\big(z(t),z_0,p\big)\cap\big(\cup_{i\in\mathcal{I}}\mathcal{O}_i(t)\big)\Big)\Big)$. This subsection discusses three reasons that contribute notable conservatism to the proposed over-approximation of risk of collision.

The first reason that introduces conservatism is the usage of Boole's inequality in Theorem 39. Boole's inequality is used to account for the interdependence among time intervals $\{T_j\}_{j\in\mathcal{J}}$, and it is known that Boole's inequality is overly conservative in general. In addition, Boole's inequality becomes more conservative if one increases the number of probabilities to be accumulated. To test how much conservatism Boole's inequality introduces, we consider the same planning iterations that are used in Section 4.6.2.1, and we use Monte-Carlo integration to compute the chance integration so that the conservatism introduced by the proposed over-approximation in Risk-RTD is avoided. In particular given arbitrary $i \in \mathcal{I}$, Boole's inequality over-approximates the risk of collision with the i-th


(a) Risk-RTD utilized.


(b) REFINE utilized.

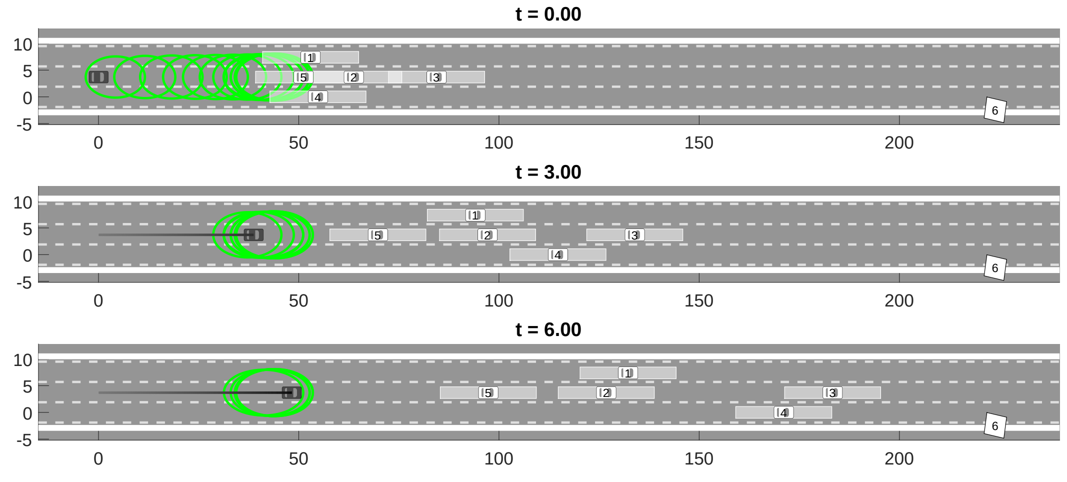

(c) SOS-RTD utilized.

Figure 4.6: Example of a simulated scenario in which Risk-RTD successes to achieve a lane change, but REFINE and SOS-RTD cannot. Probability densities of all obstacles are visualized in (a), and areas that are used to approximate all probability densities are visualized as transparent white boxes (b) and (c).

obstacle during the planning horizon:

$$\texttt{prob}^{\texttt{Boole}}(i, z_0, p) := \sum_{j \in \mathcal{J}} \int_{\bar{\xi}(\mathcal{R}_j, z_0, p) \oplus <0,\ G^{\text{obs}}>} q(w \mid j, i)\, dw. \tag{4.41}$$

If we conservatively consider the risk of collision during different time intervals independently, then the risk of collision furing the planning horizon can be over-approximated as

$$\texttt{prob}^{\texttt{ind}}(i, z_0, p) := 1 - \prod_{j \in \mathcal{J}} \left(1 - \int_{\bar{\xi}(\mathcal{R}_j, z_0, p) \oplus <0,\ G^{\text{obs}}>} q(w \mid j, i)\, dw\right) \tag{4.42}$$

Among all tested planning iterations, Monte-Carlo integration shows that the probability $\texttt{prob}^{\texttt{Boole}}(i, z_0, p)$ is relatively 10.99% higher than $\texttt{prob}^{\texttt{ind}}(i, z_0, p)$ on average, and the maximum relative error is 12.94%. This means Boole's inequality introduces at least 10.99% relative error on average.

The second reason that introduces conservatism is mentioned in Remark 41. Recall that we relax the probability density function q in Section by evaluating $\texttt{Hess}_q$ over $\mathcal{T} + A_{\bar{\xi},j}\mathcal{P}$ in order to make the resulted matrix $H_{\mathcal{T}}$ invariant over $\mathcal{P}$. Notice $\cup_{\mathcal{T} \in \overline{\mathcal{T}}_j(z_0)} \mathcal{T} + A_{\xi,j}\mathcal{P}$ covers $\cup_{p \in \mathcal{P}} \bar{\xi}(\mathcal{R}_j, z_0, p) \oplus <0,\ G^{\text{obs}}>$, thus the proposed chance integration approximation evaluates the Hessian over a much larger area instead of the actual space of integration, $\bar{\xi}(\mathcal{R}_j, z_0, p) \oplus <0,\ G^{\text{obs}}>$, that we really care about. As suggested in Remark 41, one can apply the idea of 'slice before Interval Arithmetic' to bound the Hessian of q by evaluating it over $\mathcal{T} + A_{\bar{\xi},j}p$ with the exact value of control parameter p using Interval Arithmetic, so that a tighter over-approximation of the risk of collision can be generated. This idea is intuited by the fact that Interval Arithmetic generates tighter result as the evaluated interval space gets smaller. By testing all scenarios in Section 4.6.1, compare to applying the idea of 'slice before Interval Arithmetic', evaluating $\texttt{Hess}_q$ over $\mathcal{T} + A_{\bar{\xi},j}\mathcal{P}$ results in the final chance constraint over-approximation a relative error of 1.34% on average and 20.28% at maximal. However by evaluating $\texttt{Hess}_q$ over $\mathcal{T} + A_{\bar{\xi},j}p$ using Interval Arithmetic, the gradient of the chance constraint over-approximation easily becomes intractable, thus a decrease in real-time performance.

The last reason that introduces conservatism comes from the way we generate triangle covering, but it is also related to Interval Arithmetic. With the grid size k being fixed, generated triangles for chance integration approximation becomes larger if they are used to cover a bigger zonotope, thus the evaluated Hessian by Interval Arithmetic gets looser, and so does the final chance integration approximation. Given arbitrary $\mathcal{T} \in \overline{\mathcal{T}}_j(z_0)$ with arbitrary $j \in \mathcal{J}$ and initial condition z_0, we point out that it is how the probability density varies within $\mathcal{T} + A_{\bar{\xi},j}\mathcal{P}$ that affects the conservatism of Interval Arithmetic. For nota-

Range of $\frac{\texttt{Area}_{\mathcal{T}}}{\texttt{Area}_q}$	Avg. Relative Error [%]	Max. Relative Error [%]
[0.1,0.5)	1032.6	2174.3
[0.08, 0.1)	153.54	264.25
[0.05, 0.08)	83.37	125.74
[0.01, 0.05)	37.88	53.42
[0.001, 0.01)	13.39	30.41
(0, 0.001)	1.58	14.73

Table 4.6: Tightness of the proposed over-approximation of risk of collision with varying $\texttt{Area}_{\mathcal{T}}/\texttt{Area}_q$. Relative error is computed by comparing against the result of Monte-Carlo integration as the ground truth of chance integration.

tional simplicity, denote $\texttt{Area}_{\mathcal{T}}$ the area of triangle $\mathcal{T}$ and $\texttt{Area}_q$ the area of the region whose element results in a nontrivial probability density according to the density function q. In this experiment we use all 3039 random tests utilized in Section 4.6.1 to evaluate the performance of different values of $\texttt{Area}_{\mathcal{T}}/\texttt{Area}_q$, and have the resulted chance integration approximations compared against Monte-Carlo integration as the ground truth. Notice the value of $\texttt{Area}_{\mathcal{T}}/\texttt{Area}_q$ can be changed by varying the value of k. By regenerating $\mathcal{T} \in \overline{\mathcal{T}}_j(z_0)$ with respect to different values of $\texttt{Area}_{\mathcal{T}}/\texttt{Area}_q$, we see that the proposed over-approximation of risk of collision gets significantly improved when the area ratio $\texttt{Area}_{\mathcal{T}}/\texttt{Area}_q$ decreases as illustrated in Table 4.6. It happens that all tests in Section 4.6.2.1 and Section 4.6.2.2 are evaluated with the area ratio $\texttt{Area}_{\mathcal{T}}/\texttt{Area}_q$ no smaller than 0.05, and this explains why the actual crash rates among our tests are significantly smaller than the actual risk thresholds that we enforce.

4.7 Conclusion

In this work we propose a real-time Risk-aware Reachability-based Trajectory Design (Risk-RTD) framework for path planning in dynamic environments. Risk-RTD shares the same offline reachability analysis with REFINE using zonotopes, but online enforces robot safety by limiting the probability of any collision from above as chance constraints. In particular, we provide a closed-form over-approximation on the probability of a collision with mild assumption on the probabilistic distribution of obstacle estimation, and illustrate the analytical derivative of the probability over-approximation with respect to the decision variables of online optimization. Experiments show that the proposed computation method consistently provides tight upper bound of probability integration, and real-time performance is achieved through the parallelized evaluation of the proposed chance integration. Compared with state-of-arts methods, Risk-RTD allows the ego vehicle to travel through crowd traffic more aggressively with higher success rate in the application of autonomous driving. However, the proposed over-approximation probability integration is still considerably conservative compared to the actual risk of collision as discussed in the last subsection. Hence potential improving directions are provided in the next chapter.

CHAPTER 5

Conclusion and Future Directions

This thesis makes contributions to generalize the original Reachability-based Trajectory Design (RTD) framework and enable less conservatism in real-time motion planning both deterministically and probabilistically. Recall the original RTD framework suffers from two shortcomings. First, because the original RTD framework performs reachability analysis by solving polynomial optimization programs whose performances are sensitive to system dimensionality, RTD introduces undue conservatism due to the representation of the full order dynamics as a reduced order model. Second, more conservatism is introduced when RTD deals with uncertain environment sensing by requiring the ego robot to avoid any possible unsafe area. This chapter provides a summary of the contributions and potential future research directions.

5.1 Summary of Contributions

In Chapter 2, we extend the RTD framework to bipedal robots for flat ground walking. We use the idea of template and anchor to approximate the full-order robot model using a simplified model with the error between the two models being conservatively bounded. In Chapter 3, we focus on autonomous driving scenarios and present a robust, partial feedback linearization controller that allows for tight reachability analysis on the full-order model dynamics using zonotopes for real-time motion planning in deterministic environments. In Chapter 4, we address the case when there is uncertainty in the estimated locations of surrounding obstacles by proposing a real-time risk-aware motion planning framework. This framework enforces safety amidst the uncertainty by limiting the risk of collision as a chance constraint. A differentiable over-approximation of the risk of collision is provided in closed-form in order to make online planning tractable.

5.2 Future Directions

Reachability Analysis on Bipedal Robots. Instead of performing reachability analysis on the full-order robot dynamics as REFINE does in Chapter 3, the motion planning framework on bipedal robots proposed in Chapter 2 uses the simplified model (SBM) from which conservatism is introduced. Constructing the FRS using the full-order dynamics of bipedal robots could be extremely challenging due to the hybrid nature and high dimensionality of bipedal robots, i.e. the degrees of freedom of Cassie and Digit are 20 and 30 respectively. One potential way to utilize the full-order robot model for tighter reachability analysis is by using the Error Reachable Set (ERS), which captures all possible tracking error of the closed-loop system dynamics. Then the FRS can be over-approximated by the Minkowski sum of ERS and the reference trajectory that the controller tries to track. However to achieve provably small tracking error, larger control inputs are possibly required and can potentially violate torque limits of robot motors. Therefore a robust controller of bipedal robots that achieves tight tracking performance even with the presence of the reset map for stance foot switching might be of interest to design.

Faster Online Motion Planning using Zonotope Reachable Sets. Although real-time planning is achieved by REFINE proposed in Chapter 3 as shown in Table 3.2, in real applications, we would desire motions plans to be generated even faster so that the robot can act as soon as possible to the new sensing information that is rapidly collected at runtime. Notice in (`Opt`) each zonotope reachable set results in one intersection constraint and the FRS is over-approximated by hundreds of zonotopes. Therefore to solve (`Opt`) , hundreds of intersection constraints need to be evaluated for each obstacle, and this is the major reason why REFINE may spend more than 1 [sec] for a planning iteration. There are two potential ways to speed up the solving procedure of online planning for REFINE. First, one can decrease the number of zonotope reachable sets by grouping multiplproprioceptivee zonotopes into one while maintaining sliceability without introducing too much conservatism. Note this is a post-processing step of offline reachability analysis, so that we can avoid using a larger value of Δt which could potentially cause divergence during zonotope reachable set generation. Second, notice that all intersection constraints in (`Opt`) result in almost-linear inequalities of the same form as shown in Theorem 32. Then one can instead enforce the minimum value of the left hand sides of these inequalities among all $(j, i) \in \mathcal{J} \times \mathcal{I}$ being positive so that (`Opt`) ends up with only one constraint evaluation during each of its solving iteration.

Tighter Approximation on Risk of Collision. Improving the tightness of the probability integration proposed in Risk-RTD can potentially increase the maneuverability for autonomous driving. As explained in Section 4.6.3, Interval Arithmetic results in looser approximation of the risk of collision when the triangles we use to cover the buffered zonotope reachable sets are relatively large compared to the area that has nontrivial probability density. One can certainly increase the number of triangles to achieve a finer covering, but this requires more calculations and can affect real-time performance. Because a low tolerance of collision risk is preferred for real application, usually most triangle results in 0 probability masses, thus a waste of computational resources. To maintain real-time performance while increasing the quality of the approximation on risk of collision, one can keep the value of grid size k and only cover the intersection between buffered zonotope reachable set and the area that has nontrivial probability density. In this way, because the same amount of triangles are used to cover a subset of a buffered zonotope reachable set, the sizes of triangles become smaller so that the approximation of risk of collision suffers less from the conservatism due to the application of Interval Arithmetic.

Planning with Uncertainty. The trajectory planning framework in Chapter 4 handles exteroceptive uncertainty from environment sensing. However possible proprioceptive uncertainty should also be taken into account for motion planning including imperfect system identification during the mechanical design of a robot [104]. For example in the scenario of legged robot walking, a commonly made assumption is that the stance foot is relatively static to the contact ground, which requires the contact ground to provide enough friction [36]. However because the friction coefficient could vary when the robot walks over different kinds of ground, this assumption may be violated so that the dynamics is not accurate any more. Another example could be a manipulator tasked to move packages with unknown weights. In this situation, the dynamics of the manipulator would change depending on the mass of the package and whether the robot is holding a package or not. This would introduce uncertainty into the manipulator's dynamics. In the case when these kinds of uncertainty could be provided probabilistically [26], the proposed idea of over-approximation the probability integration in Risk-RTD can be generalized to handle the uncertainties under task-specific modifications.

APPENDIX A

Derivation of SBM Dynamics

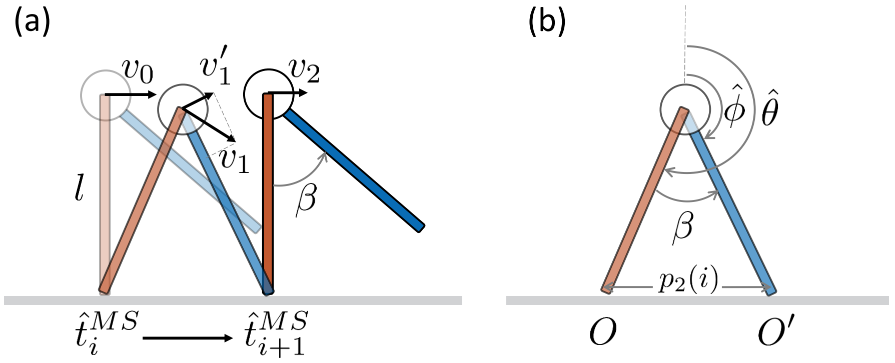


Figure A.1: (a) SBM walking from the i-th mid-stance to the $(i+1)$-st mid-stance. (b) SBM at the touch-down moment.

Consider a *Simplified Biped Model* (SBM) adopted from [116], illustrated in Fig. A.1. The model consists of a point-mass (also called *hip*, shown as black circle) and two mass-less legs each with constant length l. The stance leg is colored in red, and the swing leg is colored in blue. The stance leg angle with respect to the upright direction is denoted as $\hat{\theta}$, and the swing leg angle with respect to the upright direction is denoted as $\hat{\phi}$. In the scope of this work we consider the walking motion of SBM starting from the mid-stance position $\hat{\theta} = \pi$ with positive hip velocity $v_0 > 0$. During single stance phase, the stance leg rotates around the pivot point O, and the swing leg swings forward instantaneously to form an angle β relative to the stance leg. Notice $\beta = \hat{\theta} - \hat{\phi}$, and the value of β stays constant during the stance phase. Given any control parameter $p_2(i)$ that represents the step length during the i-th step, such β can be obtained using the Law of Cosines (Fig. A.1(b)):

$$\beta = \arccos\left(\frac{2l^2 - p_2(i)^2}{2l^2}\right). \tag{A.1}$$

As the single stance phase continues, the touch-down event, described by the guard condition $\hat{\theta} + \hat{\phi} = 2\pi$, will eventually be triggered, and an instantaneous stance phase takes place as shown in Fig. A.1(b). Subsequent to the double stance phase, an impact with the ground happens with a coefficient of restitution of 0. That is, the axial component of v_1 resets to zero after the impact, but the lateral component v_1' remains unchanged. The stance leg is then pivoted at a new point O' and the system keeps evolving forward.

We denote a hybrid execution of the SBM as a pair $(\hat{\mathcal{I}}, \hat{a})$ where $\hat{\mathcal{I}} = \{\hat{I}_i\}_{i=0}^N$ is a hybrid time set with $\hat{I}_i := [\hat{\tau}_i^+, \hat{\tau}_{i+1}^-]$ and $\hat{a} = \{\hat{a}_i(\cdot)\}_{i=0}^N$ is a finite sequence of solutions to the SBM's equations of motion. In the scope of this work we require $\hat{\theta} \in [\pi/2, 3\pi/2]$, and only consider the motion of SBM in the duration of $\hat{I}_i \cup \hat{I}_{i+1}$.

Now we derive the functions $f_{\hat{y}_1}$, $f_{\hat{y}_2}$, $f_{\hat{y}_3}$, and $f_{\hat{y}_4}$ in our manuscript by writing down $\hat{y}_1(i+1)$, $\hat{y}_2(i)$, $\hat{y}_3(i)$, $\hat{y}_4(i)$ explicitly. Ideally we expect SBM to walk from mid-stance to mid-stance as shown in Fig. A.1 (a). Assume SBM arrives at the i-th mid-stance at t_i^{MS} with positive hip velocity v_0 and positive stance leg angular velocity $\dot{\hat{\theta}}(t_i^{MS})$. As SBM moves forward, denote v_1 as the hip velocity when touch-down happens, and v_1' as the projection of v_1 to the direction that is perpendicular to the swing leg. Eventually, SBM should reach the $(i+1)$-th mid-stance at t_{i+1}^{MS} with hip velocity v_2. Notice that v_1, v_1' and v_2 all remain to be computed. From [116, (3)] we know

$$\ddot{\hat{\theta}}(t) = \frac{g \sin(\hat{\theta}(t))}{l} \tag{A.2}$$

and thus $\dot{\hat{\theta}}(t) > 0$ for all $t \in [\hat{t}_i^{MS}, \tau_{i+1}^-]$.

We want to point out an important observation at the touch-down moment as shown in Fig. A.1(b). Given any $p_2(i)$, the angle between stance and swing leg, $\beta := \hat{\theta} - \hat{\phi}$ computed in (A.1) is independent of the states $\hat{\theta}$ and $\hat{\phi}$. Moreover, for all $p_2(i) \in [0.15, 0.7]$ considered in this work, $0 < \beta \leq \pi/3$.

By conservation of energy, we have:

$$0.5(l \cdot \dot{\hat{\theta}}(t_i^{MS}))^2 + g(l - l\cos(\beta/2)) = 0.5(v_1)^2, \tag{A.3}$$

$$v_1' = v_1 \cdot \cos\beta, \tag{A.4}$$

$$0.5(v_1')^2 - g(l - l\cos(\beta/2)) = 0.5(v_2)^2, \tag{A.5}$$

where the unknowns are marked in red. Notice from (A.1) that $0 < \beta \leq \pi/3$ for all $p_2(i) \in [0.15, 0.7]$ considered in this work, therefore $\cos\beta > 0$. The solution to the system of equations (A.3)-(A.5) may or may not exists, depending on whether SBM eventually reaches the $(i+1)$-st mid-stance. These two cases are discussed separately as follows.

1. If $0.5(v_1')^2 - g(l - l\cos(\beta/2)) \geq 0$, the reader can solve for a positive v_2 from (A.3)-

(A.5), and thus

$$\hat{y}_1(i+1) = v_2/l. \tag{A.6}$$

where v_2 is a function of $\hat{y}_1(i) = \dot{\hat{\theta}}(t_i^{MS})$. Furthermore, we have

$$\hat{y}_2(i) = \pi - \beta/2. \tag{A.7}$$

Notice that $\hat{\theta}(t) > \pi$, $\dot{\hat{\theta}}(t) > 0$ for all $t \in [\hat{t}_i^{MS}, \hat{\tau}_{i+1}^-]$, and $\hat{\theta}(t) < \pi$, $\dot{\hat{\theta}}(t) > 0$ for all $t \in [\hat{\tau}_{i+1}^-, \hat{t}_{i+1}^{MS}]$. We then have

$$\hat{y}_3(i) = \hat{\theta}(\hat{\tau}_{i+1}^-) = \hat{y}_2(i) = \pi - \beta/2, \tag{A.8}$$

$$\hat{y}_4(i) = \hat{\theta}(\hat{\tau}_{i+1}^-) = \pi + \beta/2. \tag{A.9}$$

2. If $0.5(v_1')^2 - g(l - l\cos(\beta/2)) < 0$, then $\dot{\hat{\theta}}(t)$ becomes 0 at some time $\hat{t}_{i+1}^0$ before the $(i+1)$-st mid-stance is reached, and SBM may fall backward as in Fig. A.2.

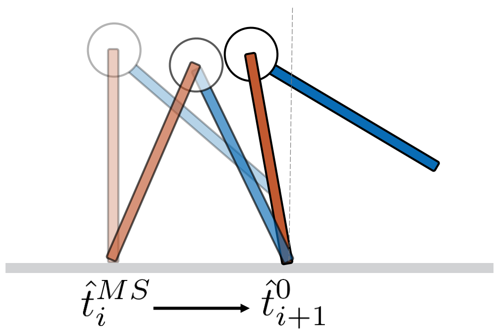


Figure A.2: SBM fails to reach the $(i+1)$-th mid-stance.

By on conservation of energy, we have

$$0.5(v_1')^2 - g(l \cdot \cos(\hat{\theta}(\hat{t}_{i+1}^0)) - l\cos(\beta/2)) = 0, \tag{A.10}$$

where the unknown are again marked in red. Using (A.3), (A.4), and (A.10), one can compute that

$$\hat{y}_1(i+1) = -\sqrt{2gl\big(1 - \cos(\hat{\theta}(\hat{t}_{i+1}^0))\big)}/l. \tag{A.11}$$

Again we have $\hat{y}_2(i) = \pi - \beta/2$. Since $\hat{\theta}(t) < \pi$ for all $t \geq \tau_{i+1}^+$, $t_i^{MS} = +\infty$, then we have $\hat{y}_3(i) = -\infty$ and $\hat{y}_4(i) = +\infty$.

Notice that β is actually a function of $p_2(i)$, one can then check that $\hat{y}_1(i+1)$ is essentially a function of $\hat{y}_1(i)$ and $p_2(i)$. Furthermore, since $\hat{y}_2(i) = \pi - \beta/2$, $\hat{y}_2(i)$ is then only a function of $p_2(i)$. Since the values of $y_3(i)$ and $y_4(i)$ depend on the positivity of $0.5(v_1')^2 - g(l - l\cos(\beta/2))$, then we can obtain the expressions for $y_3(i)$ and $y_4(i)$ both as functions of $\hat{y}_1(i)$ and $p_2(i)$.

APPENDIX B

Proof of Lemma 21

Proof. This proof defines a Lyapunov function candidate and uses it to analyze the tracking error of the ego vehicle's longitudinal speed before time t_{stop}. Then it describes how u evolves after time t_{stop} in different scenarios depending on the value of $u(t_{\text{stop}})$. Finally it describes how to set the time t_{brake} to guarantee $u(t) = 0$ for all $t \geq t_{\text{brake}}$. For convenience, let $u^{\text{small}} := \frac{M_u}{\kappa_{1,u}M_u+\phi_{1,u}}$, then by assumption of the theorem $u^{\text{small}} \in (0.15, u^{\text{cri}}]$. This proof suppresses the dependence on p in $u^{\text{des}}(t,p)$, $\tau_u(t,p)$, $\kappa_u(t,p)$, $\phi_u(t,p)$ and $e_u(t,p)$.

Note by (3.25) and rearranging (3.26),

$$\dot{e}_u(t) = -K_u e_u(t) + \tau_u(t) + \Delta_u(t). \tag{B.1}$$

Recall u^{des} is piecewise continuously differentiable by Definition 14, so are e_u and τ_u. Without loss of generality we denote $\{t_1, t_2, \dots, t_{k_{\max}}\}$ a finite subdivision of $[0, t_{\text{stop}})$ with $t_1 = 0$ and $t_{k_{\max}} = t_{\text{stop}}$ such that u^{des} is continuously differentiable over time interval $[t_k, t_{k+1})$ for all $k \in \{1, 2, \dots, k_{\max} - 1\}$. Define $V(t) := \frac{1}{2}e_u^2(t)$ as a Lyapunov function candidate for $e_u(t)$, then for arbitrary $k \in \{1, 2, \dots, k_{\max} - 1\}$ and $t \in [t_k, t_{k+1})$, one can check that $V(t)$ is always non-negative and $V(t) = 0$ only if $e_u(t) = 0$. Then

$$\begin{aligned}
\dot{V}(t) &= e_u(t)\dot{e}_u(t) &\text{(B.2)}\\
&= -K_u e_u^2(t) + e_u(t)\tau_u(t) + e_u(t)\Delta_u(t) &\text{(B.3)}\\
&= -K_u e_u^2(t) - (\kappa_u(t)M_u + \phi_u(t))e_u^2(t) + e_u(t)\Delta_u(t) &\text{(B.4)}
\end{aligned}$$

in which the second equality comes from (B.1) and the third equality comes from (3.22). Because the integral terms in (3.23) and (3.24) are both non-negative, $\kappa_u(t) \geq \kappa_{1,u}$ and $\phi_u(t) \geq \phi_{1,u}$ hold. Then

$$\dot{V}(t) \leq -K_u e_u^2(t) - (\kappa_{1,u}M_u + \phi_{1,u})|e_u(t)|^2 + |e_u(t)||\Delta_u(t)|. \tag{B.5}$$

By factoring out $|e_u(t)|$ in the last two terms in (B.5):

$$\dot{V}(t) \leq -K_u e_u^2(t) < 0 \tag{B.6}$$

holds when $|e_u(t)| > 0$ and $|e_u(t)| \geq \frac{|\Delta_u(t)|}{\kappa_{1,u} M_u + \phi_{1,u}}$. Note $|e_u(t)| \geq u^{\text{small}}$ conservatively implies $|e_u(t)| \geq \frac{|\Delta_u(t)|}{\kappa_{1,u} M_u + \phi_{1,u}}$ given $|\Delta_u(t)| \leq M_u$ for all time by Assumption 11. Then when $|e_u(t)| \geq u^{\text{small}} > 0$ we have (B.6) hold, or equivalently $V(t)$ decreases. Therefore if $|e_u(t_k)| \geq u^{\text{small}}$, $|e_u(t)|$ monotonically decreases during time interval $[t_k, t_{k+1})$ as long as $|e_u(t)|$ does not reach at the boundary of closed ball $\mathcal{B}(0, u^{\text{small}})$. Moreover, if $|e_u(t')|$ hits the boundary of $\mathcal{B}(0, u^{\text{small}})$ at some time $t' \in [t_k, t_{k+1})$, $e_u(t)$ is prohibited from leaving the ball for all $t \in [t', t_{k+1})$ because $\dot{V}(t)$ is strictly negative when $|e_u(t)| = u^{\text{small}}$. Similarly $|e_u(t_k)| \leq u^{\text{small}}$ implies $|e_u(t)| \leq u^{\text{small}}$ for all $t \in [t_k, t_{k+1})$.

We now analyze the behavior of $e_u(t)$ for all $t \in [0, t_{\text{stop}})$. By assumption $u^{\text{des}}(0) = u(0)$, then $|e_u(0)| = 0 < u^{\text{small}}$ and thus $|e_u(t)| \leq u^{\text{small}}$ for all $t \in [t_1, t_2)$. Because both $u(t)$ and $u^{\text{des}}(t)$ are continuous during $[0, t_{\text{stop}})$, so is $e_u(t_2)$. Thus $|e_u(t_2)| \leq u^{\text{small}}$. By iteratively applying the same reasoning, one can show that $|e_u(t)| \leq u^{\text{small}}$ for all $t \in [t_k, t_{k+1})$ and for all $k \in \{1, 2, \ldots, k_{\text{max}} - 1\}$, therefore $|e_u(t)| \leq u^{\text{small}}$ for all $t \in [0, t_{\text{stop}})$. Furthermore, because $u^{\text{des}}(t)$ converges to u^{cri} as t converges to t_{stop} from below, $u(t_{\text{stop}}) \in [u^{\text{cri}} - u^{\text{small}}, u^{\text{cri}} + u^{\text{small}}]$. Note $u(t_{\text{stop}}) \geq 0$ because $u^{\text{small}} \leq u^{\text{cri}}$.

Next we analyze how longitudinal speed of the ego vehicle evolves after time t_{stop}. Using $V(t) = \frac{1}{2} e_u^2(t)$, we point out that (B.5) remains valid for all $t \geq t_{\text{stop}}$, and (B.6) also holds when $|e_u(t)| \geq u^{\text{small}}$ with $t \geq t_{\text{stop}}$. Recall $u(t) = e_u(t)$ for all $t \geq t_{\text{stop}}$ given $u^{\text{des}}(t) = 0$ for all $t \geq t_{\text{stop}}$, then for simplicity, the remainder of this proof replaces every $e_u(t)$ by $u(t)$ in (B.5), (B.6) and $V(t)$. Because $u(0) > 0$ and u is continuous with respect to time, the longitudinal speed of the ego vehicle cannot decrease from a positive value to a negative value without passing 0. However when $u(t) = 0$, $\Delta_u(t) = 0$ by Assumption 12, thus $\dot{u}(t) = 0$ by (3.26) given $u^{\text{des}}(t) = 0$ for all $t \geq t_{\text{stop}}$. In other words, once u arrives at 0, it remains at 0 forever. For the ease of expression, from now on we assume $t \geq t_{\text{stop}}$ and $u(t) \geq 0$ for all $t \geq t_{\text{stop}}$. Recall $u(t_{\text{stop}}) \in [u^{\text{cri}} - u^{\text{small}}, u^{\text{cri}} + u^{\text{small}}]$ and $u^{\text{cri}} - u^{\text{small}} \in [0, u^{\text{cri}} - 0.15)$. We now discuss how u evolves after time t_{stop} by considering three scenarios, and giving an upper bound of the time at when u reaches 0 for each scenario.

Case 1 - When $u(t_{\text{stop}}) \leq 0.15$: Because the longitudinal speed stays at 0 once it becomes 0, by Assumption 20 the ego vehicle reaches to a full stop no later than $t_{\text{fstop}} + t_{\text{stop}}$.

Case 2 - When $0.15 < u(t_{\text{stop}}) \leq u^{\text{small}}$: By Assumption 12, upper bound of $\dot{V}(t)$ can be further relaxed from (B.5) to

$$\dot{V}(t) \leq -K_u u^2(t) - (\kappa_{1,u} M_u + \phi_{1,u} - b_u^{\text{pro}}) u^2(t) + b_u^{\text{off}} u(t). \tag{B.7}$$

Moreover, by completing the square among the last two terms in (B.7), one can derive

$$\dot{V}(t) \leq -K_u u^2(t) + \frac{(b_u^{\text{off}})^2}{4(\kappa_{1,u} M_u + \phi_{1,u} - b_u^{\text{pro}})}. \tag{B.8}$$

Notice $\frac{(b_u^{\text{off}})^2}{4(\kappa_{1,u} M_u + \phi_{1,u} - b_u^{\text{pro}})} < 0.15^2 K_u$ by assumption, thus

$$\dot{V}(t) < -K_u (u^2(t) - 0.15^2). \tag{B.9}$$

This means as long as $u(t) \in [0.15, u^{\text{cri}}]$ with $t \geq t_{\text{stop}}$, we obtain $\dot{V}(t) < 0$, or equivalently $V(t) = \frac{1}{2} u^2(t)$ decreases monotonically. Recall $u(t_{\text{stop}}) \leq u^{\text{small}} \leq u^{\text{cri}}$, then the longitudinal speed decreases monotonically from $u(t_{\text{stop}})$ to 0.15 as time increases from t_{stop}. Suppose u becomes 0.15 at time $t'_{\text{brake}} \geq t_{\text{stop}}$, then $u(t) \leq 0.15$ for all $t \geq t'_{\text{brake}}$ because of the fact that $\dot{V}(t)$ is strictly negative when $u(t) = 0.15$.

Define $q_u := \frac{(b_u^{\text{off}})^2}{4(\kappa_{1,u} M_u + \phi_{1,u} - b_u^{\text{pro}})}$, then when $u(t) \in [0.15, u(t_{\text{stop}})]$, (B.8) can be relaxed to

$$\dot{V}(t) \leq -K_u \cdot 0.15^2 + q_u. \tag{B.10}$$

Integrate both sides of (B.10) from time t_{stop} to t'_{brake} results in

$$t'_{\text{brake}} \leq \frac{u(t_{\text{stop}})^2 - 0.15^2}{2 \cdot 0.15^2 K_u - 2q_u} + t_{\text{stop}}. \tag{B.11}$$

Because $u(t_{\text{stop}}) \leq u^{\text{small}}$,

$$t'_{\text{brake}} \leq \frac{(u^{\text{small}})^2 - 0.15^2}{2 \cdot 0.15^2 K_u - 2q_u} + t_{\text{stop}}. \tag{B.12}$$

Then u becomes 0 no later than time $t_{\text{fstop}} + \sup(t'_{\text{brake}})$ based on Assumption 20, where $\sup(t'_{\text{brake}})$ as the upper bound of t'_{brake} reads

$$\sup(t'_{\text{brake}}) = \frac{(u^{\text{small}})^2 - 0.15^2}{2 \cdot 0.15^2 K_u - 2q_u} + t_{\text{stop}}. \tag{B.13}$$

Case 3 - When $u^{\text{small}} < u(t_{\text{stop}}) \leq u^{\text{cri}} + u^{\text{small}}$: Recall (B.6) holds given $|e_u(t)| = u(t) \geq u^{\text{small}}$, then

$$\dot{V}(t) \leq -K_u e_u^2(t) \leq -K_u (u^{\text{small}})^2, \tag{B.14}$$

and we have the longitudinal speed monotonically decreasing from $u(t_{\text{stop}})$ at time t_{stop} until it reaches at u^{small} at some time $t_{\text{small}} \geq t_{\text{stop}}$. Integrating the left hand side and right hand side of (B.14) from t_{stop} to t_{small} gives

$$\frac{1}{2}(u^{\text{small}})^2 - \frac{1}{2}u(t_{\text{stop}})^2 \leq -K_u(u^{\text{small}})^2(t_{\text{small}} - t_{\text{stop}}). \tag{B.15}$$

Because $u(t_{\text{stop}}) \leq u^{\text{cri}} + u^{\text{small}}$, (B.15) results in

$$t_{\text{small}} \leq \frac{(u^{\text{cri}} + u^{\text{small}})^2 - (u^{\text{small}})^2}{2K_u(u^{\text{small}})^2} + t_{\text{stop}}. \tag{B.16}$$

Once the longitudinal speed decreases to u^{small}, we can then follow the same reasoning as in the second scenario for seeking an upper bound of some time t''_{brake} that is no smaller than t_{small} and gives $u(t''_{\text{brake}}) = 0.15$. However, this time we need to integrate both sides of (B.10) from time t_{small} to t''_{brake}. As a result,

$$t''_{\text{brake}} \leq \frac{(u^{\text{small}})^2 - 0.15^2}{2 \cdot 0.15^2 K_u - 2q_u} + t_{\text{small}}. \tag{B.17}$$

Then u becomes 0 no later than time $t_{\text{fstop}} + \sup(t''_{\text{brake}})$ based on Assumption 20, where $\sup(t''_{\text{brake}})$ as the upper bound of t''_{brake} reads

$$\sup(t''_{\text{brake}}) = \frac{(u^{\text{small}})^2 - 0.15^2}{2 \cdot 0.15^2 K_u - 2q_u} + \frac{(u^{\text{cri}} + u^{\text{small}})^2 - (u^{\text{small}})^2}{2K_u(u^{\text{small}})^2} + t_{\text{stop}}. \tag{B.18}$$

Now that we have the upper bound for u across these three scenarios, recall that once u arrives at 0, it remains at 0 afterwards, and notice $\sup(t''_{\text{brake}}) > \sup(t'_{\text{brake}}) > t_{\text{stop}}$. Considering all three scenarios discussed above, setting t_{brake} as the maximum value among $t_{\text{fstop}} + t_{\text{stop}}$, $t_{\text{fstop}} + \sup(t'_{\text{brake}})$ and $t_{\text{fstop}} + \sup(t''_{\text{brake}})$, i.e.,

$$t_{\text{brake}} = t_{\text{fstop}} + \frac{(u^{\text{small}})^2 - 0.15^2}{2 \cdot 0.15^2 K_u - 2q_u} + \frac{(u^{\text{cri}} + u^{\text{small}})^2 - (u^{\text{small}})^2}{2K_u(u^{\text{small}})^2} + t_{\text{stop}} \tag{B.19}$$

guarantees that $u(t) = 0$ for all $t \geq t_{\text{brake}}$. $\square$

APPENDIX C

Proof of Theorem 32

We first present a pair of lemmas.

Lemma 48. *Let* $\mathcal{R}_j = {<}c_{\mathcal{R}_j},\ [g_{\mathcal{R}_j,1}, g_{\mathcal{R}_j,2}, \dots, g_{\mathcal{R}_j,\ell_j}]{>}$ *be the zonotope computed by CORA under the hybrid vehicle dynamics model* HS *beginning from* $\mathcal{Z}_0^{aug}$ *for arbitrary* $j \in \mathcal{J}$*, and let* $\texttt{rot}(\pi_h(\mathcal{R}_j)) = {<}c_{\texttt{rot}},\ G_{\texttt{rot}}{>}$ *be defined as* (3.49)*. Then for arbitrary* $z_0^{vel} \in \mathcal{Z}_0^{vel}$ *and* $p \in \mathcal{P}$*, there exist* $c_\xi \in \mathcal{W}$*,* $A \in \mathbb{R}^{2\times n_p}$ *and a real matrix* G_ξ *with two rows such that* $\xi(\mathcal{R}_j, z_0^{vel}, p) = {<}c_\xi + A\cdot p,\ G_\xi{>}$*.*

Proof. Recall c^{slc} is defined as in (3.46), then

$$\begin{aligned}\xi(\mathcal{R}_j, z_0^{\text{vel}}, p) &= \pi_{xy}\big({<}c^{\text{slc}},\ [g_{\mathcal{R}_j,(3+n_p+1)}, \dots, g_{\mathcal{R}_j,\ell_j}]{>}\big) \oplus \texttt{rot}(\pi_h(\mathcal{R}_j)) \\ &= {<}\pi_{xy}(c^{\text{slc}}) + c_{\texttt{rot}},\ [\pi_{xy}(g_{\mathcal{R}_j,(4+n_p)}), \dots, \pi_{xy}(g_{\mathcal{R}_j,\ell_j}), G_{\texttt{rot}}]{>}.\end{aligned} \tag{C.1}$$

where the first equality comes from using (3.54) and (3.45) and the last equality comes from denoting $\texttt{rot}(\pi_h(\mathcal{R}_j))$ as ${<}c_{\texttt{rot}},\ G_{\texttt{rot}}{>}$ and performing Minkowski addition on two zonotopes. c^{slc} can be rewritten as

$$c^{\text{slc}} = c_{\mathcal{R}_j} + \sum_{k=7}^{9} \frac{[z_0^{\text{vel}}]_{(k-6)} - [c_{\mathcal{R}_j}]_k}{[g_{\mathcal{R}_j,(k-6)}]_k} g_{\mathcal{R}_j,(k-6)} - \sum_{k=10}^{9+n_p} \frac{[c_{\mathcal{R}}]_k}{[g_{\mathcal{R},(k-6)}]_k} g_{\mathcal{R},(k-6)} + A' \cdot p \tag{C.2}$$

with $A' = \left[\frac{1}{[g_{\mathcal{R}_j,4}]_{10}} g_{\mathcal{R}_j,4}, \dots, \frac{1}{[g_{\mathcal{R}_j,(3+n_p)}]_{(9+n_p)}} g_{\mathcal{R}_j,(3+n_p)}\right]$. Therefore by performing algebra one can find that $\xi(\mathcal{R}_j, z_0^{\text{vel}}, p) = {<}c_\xi + A\cdot p,\ G_\xi{>}$ where

$$c_\xi = c_{\texttt{rot}} + \sum_{k=7}^{9} \frac{[z_0^{\text{vel}}]_{k-6} - [c_{\mathcal{R}_j}]_k}{[g_{\mathcal{R}_j,(k-6)}]_k} \pi_{xy}(g_{\mathcal{R}_j,(k-6)}) - \sum_{k=10}^{9+n_p} \frac{[c_{\mathcal{R}}]_k}{[g_{\mathcal{R},(k-6)}]_k} \pi_{xy}(g_{\mathcal{R},(k-6)}) + \pi_{xy}(c_{\mathcal{R}_j}), \tag{C.3}$$

$$A = \left[\frac{1}{[g_{\mathcal{R}_j,4}]_{10}} \pi_{xy}(g_{\mathcal{R}_j,4}), \frac{1}{[g_{\mathcal{R}_j,5}]_{11}} \pi_{xy}(g_{\mathcal{R}_j,5}), \dots, \frac{1}{[g_{\mathcal{R}_j,(3+n_p)}]_{(9+n_p)}} \pi_{xy}(g_{\mathcal{R}_j,(3+n_p)}) \right], \tag{C.4}$$

and

$$G_\xi = [\pi_{xy}(g_{\mathcal{R}_j,(6+n_p+1)}), \dots, \pi_{xy}(g_{\mathcal{R}_j,\ell_j}), G_{\texttt{rot}}]. \tag{C.5}$$

□

Note $\vartheta^{\text{loc}}(j, i, z_0^{\text{pos}})$ is a zonotope by construction in (3.57) because $\vartheta(j, i)$ is assumed to be a zonotope. The following lemma follows from [37, Lem. 5.1] and allows us to represent the intersection constraint in (Opt) .

Lemma 49. *Let $\xi(\mathcal{R}_j, z_0^{vel}, p) = <c_\xi + A \cdot p,\ G_\xi>$ be computed as in Lemma 48, and let $\vartheta^{loc}(j, i, z_0^{pos}) =< c_\vartheta, G_\vartheta >$ be computed from* (3.57)*. Then $\xi(\mathcal{R}_j, z_0^{vel}, p) \cap \vartheta^{loc}(j, i, z_0^{pos}) \neq \emptyset$ if and only if $A \cdot p \in <c_\vartheta - c_\xi,\ [G_\vartheta, G_\xi]>$.*

Now we can finally state the proof of Theorem 32:

Proof. Let zonotope $\xi(\mathcal{R}_j, z_0^{\text{vel}}, p) = <c_\xi + A \cdot p,\ G_\xi>$ be computed as in Lemma 48, and let $\vartheta^{\text{loc}}(j, i, z_0^{\text{pos}}) =< c_\vartheta, G_\vartheta >$ be computed from (3.57). Because all zonotopes are convex polytopes [37], zonotope $<c_\vartheta - c_\xi,\ [G_\vartheta, G_\xi]> \subset \mathcal{W} \subseteq \mathbb{R}^2$ can be transferred into a half-space representation $\mathcal{A} := \{a \in \mathcal{W} \mid B \cdot a - b \leq 0\}$ for some matrix B and vector b. To find such B and b, we denote $c = c_\vartheta - c_\xi \in \mathbb{R}^2$ and $G = [G_\vartheta, G_\xi] \in \mathbb{R}^{2\times\ell}$ with some positive integer ℓ, and denote $B^- = \begin{bmatrix} -[G]_{2:} \\ [G]_{1:} \end{bmatrix} \in \mathbb{R}^{2\times\ell}$. Define

$$B^+ := \left[\frac{[B^-]_{:1}}{\|[B^-]_{:1}\|}, \frac{[B^-]_{:2}}{\|[B^-]_{:2}\|}, \dots, \frac{[B^-]_{:\ell}}{\|[B^-]_{:\ell}\|} \right]^\top \in \mathbb{R}^{\ell\times 2}. \tag{C.6}$$

Then as a result of [6, Thm 2.1], $<c,\ G> = \{a \in \mathcal{W} \mid B \cdot a - b \leq 0\}$ with

$$B = \begin{bmatrix} B^+ \\ -B^+ \end{bmatrix} \in \mathbb{R}^{2\ell\times 2}, \tag{C.7}$$

$$b = \begin{bmatrix} B^+ \cdot c + |B^+ \cdot G| \cdot \mathbf{1} \\ -B^+ \cdot c + |B^+ \cdot G| \cdot \mathbf{1} \end{bmatrix} \in \mathbb{R}^{2\ell} \tag{C.8}$$

where $\mathbf{1} \in \mathbb{R}^\ell$ is the column vector of ones. By Lemma 49, $\xi(\mathcal{R}_j(d), z_0^{\text{vel}}, p) \cap \vartheta^{\text{loc}}(j, i, z_0^{\text{pos}})$ is empty if and only if $A \cdot p \notin <c_\vartheta - c_\xi,\ [G_\vartheta, G_\xi]>$, or in other words $A \cdot p \notin \mathcal{A}$. Notice $A \cdot p \notin \mathcal{A}$ if and only if $\max(B \cdot A \cdot p - b) > 0$.

The subgradient claim follows from [83, Theorem 5.4.5]. □

APPENDIX D

Proof of Theorem 46

Proof. Based on the reasoning of Lemma 21, the tracking error on longitudinal speed using the proposed robust partial feedback linearization controller is bounded above by u^{small}, i.e., $u^{\text{des}}(z_0, p) - u^{\text{small}} \leq u(t) \leq u^{\text{des}}(z_0, p) + u^{\text{small}}$, for all $t \in [0, t_{\text{m}}]$ and arbitrary $p \in \mathcal{P}$. Because we have a closed form representation of ϵ^{dyn} as in (4.37), we know its monoticity at every point. In particular, for $u(t) \in [0, 31.5)$ [mph] it is monotonically increasing, for $u(t) \in (31.5, 59.6)$ [mph] it is monotonically decreasing, and for $u(t) \in (59.6, 67.1]$ [mph] it is monotonically increasing.

For notational simplicity, denote $a(z_0, p) := \epsilon^{\text{dyn}}\big(\min_{t\in[0,t_{\text{m}}]} u^{des}(t, z_0, p) - u_{small}\big)$ and $b(z_0, p) := \epsilon^{\text{dyn}}\big(\max_{t\in[0,t_{\text{m}}]} u^{des}(t, z_0, p) + u_{small}\big)$. Let $\underline{u} = \min_{t\in[0,t_{\text{m}}]} u(t)$ and let $\overline{u} = \max_{t\in[0,t_{\text{m}}]} u(t)$. Assuming the ego vehicle can only reach a maximum velocity of 67.1 [mph] = 30.0 [m/s], then $\epsilon(z_0, p) = \inf_{t\in[0,t_{\text{m}}]} \left(\epsilon^{\text{dyn}}(u(t))\right)$ is given as:

$$\epsilon(z_0, p) = \begin{cases} a(z_0, p), & \text{if } a(z_0, p) \leq b(z_0, p) \ and \ 59.6 \notin [\underline{u}, \overline{u}] \\ b(z_0, p), & \text{if } a(z_0, p) > b(z_0, p) \ and \ 59.6 \notin [\underline{u}, \overline{u}] \\ \epsilon^{\text{dyn}}(59.6), & \text{otherwise} \end{cases} . \tag{D.1}$$

Notice that ϵ^{dyn} is a polynomial and thus differentiable. Denote

$$\frac{\partial \epsilon^{\text{dyn}}_{min}}{\partial u}(z_0, p) := \frac{\partial \epsilon^{\text{dyn}}}{\partial u}\big(\min_{t\in[0,t_{\text{m}}]} u^{des}(t, z_0, p) - u_{small}\big), \tag{D.2}$$

$$\frac{\partial \epsilon^{\text{dyn}}_{max}}{\partial u}(z_0, p) := \frac{\partial \epsilon^{\text{dyn}}}{\partial u}\big(\max_{t\in[0,t_{\text{m}}]} u^{des}(t, z_0, p) + u_{small}\big). \tag{D.3}$$

Then

$$\frac{\partial \epsilon}{\partial p}(z_0,p) = \begin{cases} \frac{\partial \epsilon_{min}^{\text{dyn}}}{\partial u}(z_0,p) \cdot \frac{\partial u^{\text{des}}}{\partial p}(t,z_0,p), \text{ if } a(z_0,p) \le b(z_0,p) \text{ and } 59.6 \notin [\underline{u},\overline{u}] \\ \frac{\partial \epsilon_{max}^{\text{dyn}}}{\partial u}(z_0,p) \cdot \frac{\partial u^{\text{des}}}{\partial p}(t,z_0,p), \text{ if } a(z_0,p) > b(z_0,p) \text{ and } 59.6 \notin [\underline{u},\overline{u}] \\ 0, \text{ otherwise} \end{cases} . \tag{D.4}$$

□

BIBLIOGRAPHY

[1] Real-time ego-motion estimation using lidar and a vehicle model based extended kalman filter. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 431–438, 2014.

[2] M Abduljabbar, Nader Meskin, and Christos G Cassandras. Control barrier function-based lateral control of autonomous vehicle for roundabout crossing. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 859–864. IEEE, 2021.

[3] Amir Ali Ahmadi and Anirudha Majumdar. Dsos and sdsos optimization: Lp and socp-based alternatives to sum of squares optimization. In *2014 48th annual conference on information sciences and systems (CISS)*, pages 1–5. IEEE, 2014.

[4] Anil Alan, Andrew J Taylor, Chaozhe R He, Aaron D Ames, and Gabor Orosz. Control barrier functions and input-to-state safety with application to automated vehicles. *arXiv preprint arXiv:2206.03568*, 2022.

[5] Jorge Almeida and Vitor Manuel Santos. Real time egomotion of a nonholonomic vehicle using lidar measurements. *Journal of Field Robotics*, 30(1):129–141, 2013.

[6] Matthias Althoff. *Reachability analysis and its application to the safety assessment of autonomous cars*. PhD thesis, Technische Universität München, 2010.

[7] Matthias Althoff. An introduction to cora 2015. In *Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems*, 2015.

[8] Matthias Althoff and John M Dolan. Online verification of automated road vehicles using reachability analysis. *IEEE Transactions on Robotics*, 30(4):903–918, 2014.

[9] Aaron D Ames, Kevin Galloway, Koushil Sreenath, and Jessy W Grizzle. Rapidly exponentially stabilizing control lyapunov functions and hybrid zero dynamics. *IEEE Transactions on Automatic Control*, 59(4):876–891, 2014.

[10] Aaron D Ames, Jessy W Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs with application to adaptive cruise control. In *53rd IEEE Conference on Decision and Control*, pages 6271–6278. IEEE, 2014.

[11] Aaron D Ames, Paulo Tabuada, Austin Jones, Wen-Loong Ma, Matthias Rungger, Bastian Schürmann, Shishir Kolathaya, and Jessy W Grizzle. First steps toward

formal controller synthesis for bipedal robots with experimental implementation. *Nonlinear Analysis: Hybrid Systems*, 25:155–173, 2017.

[12] Søren Asmussen and Peter W Glynn. *Stochastic simulation: algorithms and analysis*, volume 57. Springer, 2007.

[13] George Bahouth, Jill Graygo, Kennerly Digges, Carl Schulman, and Peter Baur. The benefits and tradeoffs for varied high-severity injury risk thresholds for advanced automatic crash notification systems. *Traffic injury prevention*, 15(sup1):S134–S140, 2014.

[14] James Balkwill. *Performance vehicle dynamics: engineering and applications*. Butterworth-Heinemann, 2017.

[15] Somil Bansal, Mo Chen, Sylvia Herbert, and Claire J Tomlin. Hamilton-jacobi reachability: A brief overview and recent advances. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 2242–2253. IEEE, 2017.

[16] Stephen Boyd, Laurent El Ghaoui, Eric Feron, and Venkataramanan Balakrishnan. *Linear matrix inequalities in system and control theory*, volume 15. Siam, 1994.

[17] Russel E Caflisch. Monte carlo and quasi-monte carlo methods. *Acta numerica*, 7:1–49, 1998.

[18] Francesco Paolo Cantelli. Sui confini della probabilita. In *Atti del Congresso Internazionale dei Matematici: Bologna del 3 al 10 de settembre di 1928*, pages 47–60, 1929.

[19] Manuel Castillo-Lopez, Philippe Ludivig, Seyed Amin Sajadi-Alamdari, Jose Luis Sanchez-Lopez, Miguel A Olivares-Mendez, and Holger Voos. A real-time approach for chance-constrained motion planning with dynamic obstacles. *IEEE Robotics and Automation Letters*, 5(2):3620–3625, 2020.

[20] Christine Chevallereau, Gabriel Abba, Yannick Aoustin, Franck Plestan, Eric Westervelt, Carlos Canudas De Wit, and Jessy Grizzle. Rabbit: A testbed for advanced control theory. 2003.

[21] Jason J Choi, Donggun Lee, Koushil Sreenath, Claire J Tomlin, and Sylvia L Herbert. Robust control barrier–value functions for safety-critical control. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 6814–6821. IEEE, 2021.

[22] S Dieter, M Hiller, and R Baradini. Vehicle dynamics: Modeling and simulation, 2018.

[23] Edsger W Dijkstra et al. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[24] Anushri Dixit, Mohamadreza Ahmadi, and Joel W Burdick. Risk-sensitive motion planning using entropic value-at-risk. In *2021 European Control Conference (ECC)*, pages 1726–1732. IEEE, 2021.

[25] Mohamed Elbanhawi and Milan Simic. Sampling-based robot motion planning: A review. *IEEE Access*, 2:56–77, 2014.

[26] Parker Ewen, Adam Li, Yuxin Chen, Steven Hong, and Ram Vasudevan. These maps are made for walking: Real-time terrain property estimation for mobile robots. *IEEE Robotics and Automation Letters*, 2022.

[27] Paolo Falcone, Francesco Borrelli, J Asgari, HE Tseng, and Davor Hrovat. Low complexity mpc schemes for integrated vehicle dynamics control problems. In *9th international symposium on advanced vehicle control (AVEC)*, 2008.

[28] Jaime F Fisac, Andrea Bajcsy, Sylvia L Herbert, David Fridovich-Keil, Steven Wang, Claire J Tomlin, and Anca D Dragan. Probabilistically safe robot planning with confidence-based human predictions. *arXiv preprint arXiv:1806.00109*, 2018.

[29] National Center for Statistics and Analysis. 2019 traffic safety facts: A compilation of motor vehicle crash data (annual report). *National Highway Traffic Safety Agency*, 2021.

[30] Robert J Full and Daniel E Koditschek. Templates and anchors: neuromechanical hypotheses of legged locomotion on land. *Journal of experimental biology*, 202(23):3325–3332, 1999.

[31] Lu Gan, Youngji Kim, Jessy W Grizzle, Jeffrey M Walls, Ayoung Kim, Ryan M Eustice, and Maani Ghaffari. Multitask learning for scalable and dense multilayer bayesian map inference. *IEEE Transactions on Robotics*, 2022.

[32] Thomas A. Gennarelli and Elaine Wodzin. Ais 2005: A contemporary injury scale. *Injury*, 37(12):1083–1091, 2006. Special Issue: Trauma Outcomes.

[33] Thomas D Gillespie. Fundamentals of vehicle dynamics. Technical report, SAE Technical Paper, 1992.

[34] Antoine Girard. Reachability of uncertain linear systems using zonotopes. In *International Workshop on Hybrid Systems: Computation and Control*, pages 291–305. Springer, 2005.

[35] Andrea Giusti and Matthias Althoff. Ultimate robust performance control of rigid robot manipulators using interval arithmetic. In *2016 American Control Conference (ACC)*, pages 2995–3001. IEEE, 2016.

[36] Yukai Gong, Ross Hartley, Xingye Da, Ayonga Hereid, Omar Harib, Jiunn-Kai Huang, and Jessy Grizzle. Feedback control of a cassie bipedal robot: Walking, standing, and riding a segway. In *2019 American Control Conference (ACC)*, pages 4559–4566. IEEE, 2019.

[37] Leonidas J Guibas, An Thanh Nguyen, and Li Zhang. Zonotopes as bounding volumes. In *SODA*, volume 3, pages 803–812, 2003.

[38] Astghik Hakobyan, Gyeong Chan Kim, and Insoon Yang. Risk-aware motion planning and control using cvar-constrained optimization. *IEEE Robotics and Automation letters*, 4(4):3924–3931, 2019.

[39] Didier Henrion and Milan Korda. Convex computation of the region of attraction of polynomial control systems. *IEEE Transactions on Automatic Control*, 59(2):297–312, 2013.

[40] Sylvia L Herbert, Mo Chen, SooJean Han, Somil Bansal, Jaime F Fisac, and Claire J Tomlin. Fastrack: a modular framework for fast and guaranteed safe motion planning. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 1517–1522. IEEE, 2017.

[41] Ayonga Hereid and Aaron D Ames. Frost: Fast robot optimization and simulation toolkit. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 719–726. IEEE, 2017.

[42] Timothy Hickey, Qun Ju, and Maarten H Van Emden. Interval arithmetic: From principles to implementation. *Journal of the ACM (JACM)*, 48(5):1038–1068, 2001.

[43] Patrick Holmes, Shreyas Kousik, Bohao Zhang, Daphna Raz, Corina Barbalata, Matthew Johnson-Roberson, and Ram Vasudevan. Reachable sets for safe, real-time manipulator trajectory design. In *Robotics: Science and Systems*, 2020.

[44] Thomas M Howard and Alonzo Kelly. Optimal rough terrain trajectory generation for wheeled mobile robots. *The International Journal of Robotics Research*, 26(2):141–166, 2007.

[45] S. C. Hsu, X. Xu, and A. D. Ames. Control barrier function based quadratic programs with application to bipedal robotic walking. In *2015 American Control Conference (ACC)*, pages 4542–4548, July 2015.

[46] Lucas Janson, Edward Schmerling, Ashley Clark, and Marco Pavone. Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions. *The International Journal of Robotics Research*, 34(7):883–921, 2015.

[47] Lucas Janson, Edward Schmerling, and Marco Pavone. Monte carlo motion planning for robot trajectory optimization under uncertainty. In *Robotics Research*, pages 343–361. Springer, 2018.

[48] Reza N Jazar. *Vehicle dynamics: theory and application*. Springer, 2008.

[49] Tae-Yun Kim, Samuel Jung, and Wan-Suk Yoo. Advanced slip ratio for ensuring numerical stability of low-speed driving simulation: Part ii—lateral slip ratio. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of automobile engineering*, 233(11):2903–2911, 2019.

[50] Twan Koolen, Michael Posa, and Russ Tedrake. Balance control using center of mass height variation: limitations imposed by unilateral contact. In *Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on*, pages 8–15. IEEE, 2016.

[51] Shreyas Kousik, Patrick Holmes, and Ramanarayan Vasudevan. Safe, aggressive quadrotor flight via reachability-based trajectory design. In *Dynamic Systems and Control Conference*, volume 59162. American Society of Mechanical Engineers, 2019.

[52] Shreyas Kousik, Sean Vaskov, Fan Bu, Matthew Johnson-Roberson, and Ram Vasudevan. Bridging the gap between safety and real-time performance in receding-horizon trajectory design for mobile robots. *The International Journal of Robotics Research*, 39(12):1419–1469, 2020.

[53] Shreyas Kousik, Sean Vaskov, Matthew Johnson-Roberson, and Ram Vasudevan. Safe trajectory synthesis for autonomous driving in unforeseen environments. In *ASME 2017 Dynamic Systems and Control Conference*, pages V001T44A005–V001T44A005. American Society of Mechanical Engineers, 2017.

[54] Xiaolong Kuang, Bissan Ghaddar, Joe Naoum-Sawaya, and Luis F Zuluaga. Alternative sdp and socp approximations for polynomial optimization. *EURO Journal on Computational Optimization*, 7(2):153–175, 2019.

[55] Arthur D Kuo. Choosing your steps carefully. *IEEE Robotics & Automation Magazine*, 14(2):18–29, 2007.

[56] Alexander B Kurzhanski and Pravin Varaiya. Ellipsoidal techniques for reachability analysis: internal approximation. *Systems & control letters*, 41(3):201–211, 2000.

[57] Yoshiaki Kuwata, Justin Teo, Gaston Fiore, Sertac Karaman, Emilio Frazzoli, and Jonathan P How. Real-time motion planning with applications to autonomous urban driving. *IEEE Transactions on control systems technology*, 17(5):1105–1118, 2009.

[58] Jean B Lasserre. Simple formula for integration of polynomials on a simplex. *BIT Numerical Mathematics*, 61(2):523–533, 2021.

[59] Steven M LaValle. *Planning algorithms*. Cambridge university press, 2006.

[60] Steven M LaValle and James J Kuffner Jr. Randomized kinodynamic planning. *The international journal of robotics research*, 20(5):378–400, 2001.

[61] Jinsun Liu, Yifei Shao, Lucas Lymburner, Hansen Qin, Vishrut Kaushik, Lena Trang, Ruiyang Wang, Vladimir Ivanovic, H Eric Tseng, and Ram Vasudevan. Refine: Reachability-based trajectory design using robust feedback linearization and zonotopes. *arXiv preprint arXiv:2211.11997*, 2022.

[62] Jinsun Liu, Pengcheng Zhao, Zhenyu Gan, Matthew Johnson-Roberson, and Ram Vasudevan. Leveraging the template and anchor framework for safe, online robotic gait design. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020.

[63] Jan Lunze and Françoise Lamnabhi-Lagarrigue. *Handbook of hybrid systems control: theory, tools, applications*. Cambridge University Press, 2009.

[64] John Lygeros, Shankar Sastry, and Claire Tomlin. Hybrid systems: Foundations, advanced topics and applications. *under copyright to be published by Springer Verlag*, 2012.

[65] Daniel Lyons, Jan-P. Calliess, and Uwe D. Hanebeck. Chance constrained model predictive control for multi-agent systems with coupling constraints. pages 1223–1230, 2012.

[66] Anirudha Majumdar and Russ Tedrake. Funnel libraries for real-time robust feedback motion planning. *The International Journal of Robotics Research*, 36(8):947–982, 2017.

[67] Anirudha Majumdar, Ram Vasudevan, Mark M Tobenkin, and Russ Tedrake. Convex optimization of nonlinear feedback controllers via occupation measures. *The International Journal of Robotics Research*, page 0278364914528059, 2014.

[68] Sandeep Kumar Malu and Jharna Majumdar. Kinematics, localization and control of differential drive mobile robot. *Global Journal of Research In Engineering*, 2014.

[69] Stefanie Manzinger, Christian Pek, and Matthias Althoff. Using reachable sets for trajectory planning of automated vehicles. *IEEE Transactions on Intelligent Vehicles*, 6(2):232–248, 2020.

[70] Shankar Mohan, Jinsun Liu, and Ram Vasudevan. Synthesizing the optimal luenberger-type observer for nonlinear systems. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 3658–3663. IEEE, 2017.

[71] Mohamad Shafiee Motahar, Sushant Veer, and Ioannis Poulakakis. Composing limit cycles for motion planning of 3d bipedal walkers. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 6368–6374. IEEE, 2016.

[72] Arkadi Nemirovski and Alexander Shapiro. Convex approximations of chance constrained programs. *SIAM Journal on Optimization*, 17(4):969–996, 2007.

[73] Q. Nguyen, A. Hereid, J. W. Grizzle, A. D. Ames, and K. Sreenath. 3d dynamic walking on stepping stones with control barrier functions. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 827–834, Dec 2016.

[74] Quan Nguyen, Ayonga Hereid, Jessy W Grizzle, Aaron D Ames, and Koushil Sreenath. 3d dynamic walking on stepping stones with control barrier functions. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 827–834. IEEE, 2016.

[75] Quan Nguyen and Koushil Sreenath. Optimal robust control for bipedal robots through control lyapunov function based quadratic programs. In *Robotics: Science and Systems*, 2015.

[76] Quan Nguyen and Koushil Sreenath. Exponential control barrier functions for enforcing high relative-degree safety-critical constraints. In *American Control Conference (ACC), 2016*, pages 322–328. IEEE, 2016.

[77] Quan Nguyen and Koushil Sreenath. Optimal robust time-varying safety-critical control with application to dynamic walking on moving stepping stones. In *Dynamic Systems and Control Conference*, volume 50701. American Society of Mechanical Engineers, 2016.

[78] Ingram Olkin and Thomas A Trikalinos. Constructions for a bivariate beta distribution. *Statistics & Probability Letters*, 96:54–60, 2015.

[79] Dávid Papp and Sercan Yildiz. Sum-of-squares optimization without semidefinite programming. *SIAM Journal on Optimization*, 29(1):822–851, 2019.

[80] Pablo A Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, 2000.

[81] Michael A Patterson and Anil V Rao. Gpops-ii: A matlab software for solving multiple-phase optimal control problems using hp-adaptive gaussian quadrature collocation methods and sparse nonlinear programming. *ACM Transactions on Mathematical Software (TOMS)*, 41(1):1–37, 2014.

[82] Christian Pek, Stefanie Manzinger, Markus Koschi, and Matthias Althoff. Using online verification to prevent autonomous vehicles from causing accidents. *Nature Machine Intelligence*, 2(9):518–528, 2020.

[83] Elijah Polak. *Optimization: algorithms and consistent approximations*, volume 124. Springer Science & Business Media, 2012.

[84] Michael Posa, Twan Koolen, and Russ Tedrake. Balancing and step recovery capturability via sums-of-squares optimization. In *2017 Robotics: Science and Systems Conference*, 2017.

[85] Stephen Prajna and Ali Jadbabaie. Safety verification of hybrid systems using barrier certificates. In *International Workshop on Hybrid Systems: Computation and Control*, pages 477–492. Springer, 2004.

[86] András Prékopa. *Stochastic programming*, volume 324. Springer Science & Business Media, 2013.

[87] Mateusz Przybyła. Detection and tracking of 2d geometric obstacles from lrf data. In *2017 11th International Workshop on Robot Motion and Control (RoMoCo)*, pages 135–141. IEEE, 2017.

[88] Reinhold Remmert. *Theory of complex functions*, volume 122. Springer Science & Business Media, 1991.

[89] R Tyrrell Rockafellar, Stanislav Uryasev, et al. Optimization of conditional value-at-risk. *Journal of Risk*, 2:21–42, 2000.

[90] Prasanna Sahoo and Thomas Riedel. *Mean value theorems and functional equations*. World Scientific, 1998.

[91] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. On a formal model of safe and scalable self-driving cars. *arXiv preprint arXiv:1708.06374*, 2017.

[92] Yifei Simon Shao, Chao Chen, Shreyas Kousik, and Ram Vasudevan. Reachability-based trajectory safeguard (rts): A safe and fast reinforcement learning safety layer for continuous control. *IEEE Robotics and Automation Letters*, 6(2):3663–3670, 2021.

[93] V. Shia, R. Vasudevan, R. Bajcsy, and R. Tedrake. Convex computation of the reachable set for controlled polynomial hybrid systems. In *53rd IEEE Conference on Decision and Control*, pages 1499–1506, Dec 2014.

[94] Nils Smit-Anseeuw, C David Remy, and Ram Vasudevan. Walking with confidence: Safety regulation for full order biped models. *IEEE Robotics and Automation Letters*, 4(4):4177–4184, 2019.

[95] S.W. Smith, H. Yin, and M. Arcak. Continuous abstraction of nonlinear systems using sum-of-squares programming. In *58th Conference on Decision and Control*, 2019.

[96] Eric Squires, Pietro Pierpaoli, and Magnus Egerstedt. Constructive barrier certificates with applications to fixed-wing aircraft collision avoidance. In *2018 IEEE Conference on Control Technology and Applications (CCTA)*, pages 1656–1661. IEEE, 2018.

[97] Stanford Artificial Intelligence Laboratory et al. Robotic operating system.

[98] Jordan Stoyanov. Stieltjes classes for moment-indeterminate probability distributions. *Journal of Applied Probability*, 41(A):281–294, 2004.

[99] Matteo Tacchi, Carmen Cardozo, Didier Henrion, and Jean Lasserre. Approximating regions of attraction of a sparse polynomial differential system. *IFAC-PapersOnLine*, 53(2):3266–3271, 2020.

[100] Aviv Tamar, Yinlam Chow, Mohammad Ghavamzadeh, and Shie Mannor. Sequential decision making with coherent risk. *IEEE Transactions on Automatic Control*, 62(7):3323–3338, 2016.

[101] J. Z. Tang, A. M. Boudali, and I. R. Manchester. Invariant funnels for underactuated dynamic walking robots: New phase variable and experimental validation. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3497–3504, May 2017.

[102] A Galip Ulsoy, Huei Peng, and Melih Çakmakci. *Automotive control systems*. Cambridge University Press, 2012.

[103] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, MN Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, et al. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, 25(8):425–466, 2008.

[104] Karthik Urs, Challen Enninful Adu, Elliott J Rouse, and Talia Y Moore. Design and characterization of 3d printed, open-source actuators for legged locomotion. *arXiv preprint arXiv:2202.12395*, 2022.

[105] Sean Vaskov. *Fast and Safe Trajectory Optimization for Autonomous Mobile Robots Using Reachability Analysis*. PhD thesis, 2021.

[106] Sean Vaskov, Shreyas Kousik, Hannah Larson, Fan Bu, James Ward, Stewart Worrall, Matthew Johnson-Roberson, and Ram Vasudevan. Towards provably not-at-fault control of autonomous robots in arbitrary dynamic environments. *arXiv preprint arXiv:1902.02851*, 2019.

[107] Sean Vaskov, Hannah Larson, Shreyas Kousik, Matthew Johnson-Roberson, and Ram Vasudevan. Not-at-fault driving in traffic: A reachability-based approach. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 2785–2790. IEEE, 2019.

[108] Sushant Veer and Ioannis Poulakakis. Safe adaptive switching among dynamical movement primitives: Application to 3d limit-cycle walkers. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3719–3725. IEEE, 2019.

[109] Miomir Vukobratović and J Stepanenko. On the stability of anthropomorphic systems. *Mathematical biosciences*, 15(1-2):1–37, 1972.

[110] Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

[111] E.A. Wan and R. Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, pages 153–158, 2000.

[112] Allen Wang, Xin Huang, Ashkan Jasour, and Brian C Williams. Fast risk assessment for autonomous vehicles using learned models of agent futures. *Positions*, 2:10, 2020.

[113] Allen Wang, Ashkan Jasour, and Brian C. Williams. Non-gaussian chance-constrained trajectory planning for autonomous vehicles under agent uncertainty. *IEEE Robotics and Automation Letters*, 5(4):6041–6048, 2020.

[114] Eric R Westervelt, Christine Chevallereau, Jun Ho Choi, Benjamin Morris, and Jessy W Grizzle. *Feedback control of dynamic bipedal robot locomotion*. CRC press, 2007.

[115] Pierre-Brice Wieber. On the stability of walking systems. In *Proceedings of the international workshop on humanoid and human friendly robotics*, 2002.

[116] Derek L Wight, Eric G Kubica, and David W Wang. Introduction of the foot placement estimator: A dynamic measure of balance for bipedal robotics. *Journal of computational and nonlinear dynamics*, 3(1):011009, 2008.

[117] Holger Rapp Daniel Andor Wolfgang Hess, Damon Kohler. Real-time loop closure in 2d lidar slam. In *Robotics and Automation (ICRA)*, page 1271–1278, 2016.

[118] John Wurts, Jeffrey L Stein, and Tulga Ersal. Collision imminent steering using nonlinear model predictive control. In *2018 Annual American Control Conference (ACC)*, pages 4772–4777. IEEE, 2018.

[119] Yu Yao, Ella Atkins, Matthew Johnson-Roberson, Ram Vasudevan, and Xiaoxiao Du. Bitrap: Bi-directional pedestrian trajectory prediction with multi-modal goal estimation. *IEEE Robotics and Automation Letters*, 6(2):1463–1470, 2021.

[120] Hakan Yazarel and George J Pappas. Geometric programming relaxations for linear system reachability. In *Proceedings of the 2004 American control conference*, volume 1, pages 553–559. IEEE, 2004.

[121] Zhang-Cai Yin, Hui Liu, Zhi-Jun Zhang, Zhang-Hao-Nan Jin, San-Juan Li, Jia-Qiang Xiao, et al. Probabilistic model of random encounter in obstacle space. *ISPRS International Journal of Geo-Information*, 8(1):32, 2019.

[122] Ming-Yuan Yu, Ram Vasudevan, and Matthew Johnson-Roberson. Occlusion-aware risk assessment for autonomous driving in urban environments. *IEEE Robotics and Automation Letters*, 4(2):2235–2241, 2019.

[123] Ming-Yuan Yu, Ram Vasudevan, and Matthew Johnson-Roberson. Risk assessment and planning with bidirectional reachability for autonomous driving. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5363–5369. IEEE, 2020.

[124] Ji Zhang and Sanjiv Singh. Loam: Lidar odometry and mapping in real-time. In *Robotics: Science and Systems*, volume 2, pages 1–9. Berkeley, CA, 2014.

[125] Pengcheng Zhao. *Fast, Optimal, and Safe Motion Planning for Bipedal Robots*. PhD thesis, 2020.

[126] Pengcheng Zhao, Shankar Mohan, and Ram Vasudevan. Control synthesis for nonlinear optimal control via convex relaxations. In *2017 American Control Conference (ACC)*, pages 2654–2661. IEEE, 2017.

[127] Pengcheng Zhao, Shankar Mohan, and Ram Vasudevan. Optimal control for nonlinear hybrid systems via convex relaxations. *arXiv preprint arXiv:1702.04310*, 2017.