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

Jinsun Liu*, Yifei Shao*, Lucas Lymburner, Hansen Qin, Vishrut Kaushik,
Lena Trang, Ruiyang Wang, Vladimir Ivanovic, H. Eric Tseng, and Ram Vasudevan

***Abstract*—Performing real-time receding horizon motion planning for autonomous vehicles while providing safety guarantees remains difficult. This is because existing methods to accurately predict ego vehicle behavior under a chosen controller use online numerical integration that requires a fine time discretization and thereby adversely affects real-time performance. To address this limitation, several recent papers have proposed to apply offline reachability analysis to conservatively predict the behavior of the ego vehicle. Reachable sets can be constructed by utilizing a simplified model whose behavior is assumed *a priori* to conservatively bound the dynamics of a full-order model. However, it can be challenging to meticulously construct this conservative bound. This paper proposes a framework named REFINE[1] to overcome the limitations of these existing approaches. REFINE utilizes a parameterized robust controller that partially linearizes the vehicle dynamics even in the presence of modeling error. Zonotope-based reachability analysis is then performed on the closed-loop, full-order vehicle dynamics to offline compute the corresponding control-parameterized, over-approximate Forward Reachable Sets (FRS). Because reachability analysis is applied to the full-order model, the potential conservativeness introduced by using a simplified model is avoided. The pre-computed, control-parameterized FRS is then used online in an optimization framework to ensure safety. The proposed method is compared to several state-of-the-art methods during a simulation-based evaluation on a full-size vehicle model and is demonstrated on a $\frac{1}{10}$th race car robot in real hardware testing. In contrast to existing methods, REFINE is shown to enable the vehicle to safely navigate itself through complex environments.**

***Index Terms*—Motion and path planning, robot safety, reachability analysis, control, zonotopes.**

I. Introduction

Autonomous vehicles are expected to operate in unknown environments safely with limited sensing horizons. Because new sensor information is received while the 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 new environmental information. 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.

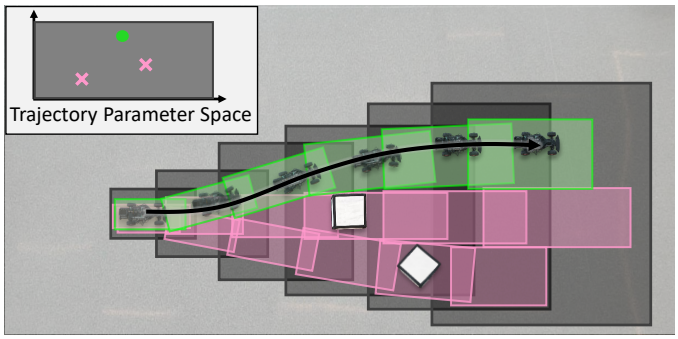


Fig. 1: REFINE first designs a robust controller to track parameterized reference trajectories by feedback linearizing a subset of vehicle states. Then REFINE performs offline reachability analysis using a closed-loop full-order vehicle dynamics to construct control-parameterized, zonotope reachable 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 trajectory parameter shown in green ensure a collision-free path while the other two trajectory parameters shown in pink lead to collisions with white obstacles.

We begin by summarizing related works on trajectory planning and discussing 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 actual vehicle and actuator dynamics are nonlinear, closed-form solutions of vehicle trajectories are incomputable, and approximations to the vehicle dynamics are typically 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 user-specified cost function [1], [2]. To model vehicle dynamics during real-time planning, sampling-based methods apply online numerical integration and buffer obstacles to compensate for numerical integration

Jinsun Liu, Lucas Lymburner, Lena Trang, Ruiyang Wang, and Ram Vasudevan are with the Department of Robotics, University of Michigan, Ann Arbor, MI 48109. {jinsunl, llyburn, ltrang, ruiyangw, ramv}@umich.edu.

Yifei Shao is with the Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104. yishao@seas.upenn.edu.

Hansen Qin is with Latitude AI. hqin@lat.ai.

Vishrut Kaushik is with Peer Robotics. vishrut@peerrobotics.ai.

Vladimir Ivanovic and Eric Tseng are with Ford Motor Company. {vivanovi, htseng}@ford.com.

This work is supported by the Ford Motor Company via the Ford-UM Alliance under award N022977.

*These two authors contributed equally to this work.

[1]https://roahmlab.github.io/REFINE_website/

error [3]–[5]. Ensuring that a numerically integrated trajectory is dynamically realizable and collision-free often requires applying a fine time discretization. This typically results in an undesirable trade-off among dynamical feasibility, collision avoidance, and real-time operation. Similarly, Nonlinear Model Predictive Control (NMPC) uses time discretization to generate an approximated solution to the vehicle dynamics that is embedded in an optimization program to compute a control input that is dynamically realizable while avoiding obstacles [6]–[9]. 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. These techniques rely on over-approximating the Forward Reachable Set (FRS) of a system. This FRS contains all possible behaviors of the vehicle dynamics over a fixed-time horizon. One of the classic papers that applies reachability analysis uses it to construct an algorithm to perform online verification of a trajectory [10]. This is done by constructing a zonotope-based, FRS representation that includes the behavior of the ego vehicle while it tracks this trajectory using a feedback controller. If this FRS does not intersect with objects in the surrounding environment, then it can be followed using the feedback controller in a collision-free manner. Note [10] is focused on online verification, thus it assumes the existence of a trajectory planner and does not describe how to construct a planner that outputs plans that can be followed safely. This online verification layer has been extended and applied to urban traffic situations to demonstrate its effectiveness in preventing accidents [11], [12].

Rather than just performing the online verification of a single trajectory, more recent techniques that rely on reachability analysis optimize over families of FRSes that are constrained to be collision-free. For instance, the funnel library method [13] precomputes a *finite* library of over-approximative FRSes for different maneuvers by applying Sums-of-Squares (SOS) Programming. During run-time, the algorithm searches over a discrete number of trajectories to identify one that can be tracked safely. As we show in the experimental study of this paper, precomputing a rich enough, finite 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 [14]. RTD 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 [15]–[17]. Unfortunately applying SOS optimization to compute the FRS of high dimensional systems can be challenging. As a result, RTD 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 trajectories for the offline reachability analysis. To overcome this issue, recent work has applied a Hamilton-Jacobi-Bellman based-approach [18] to pose the offline reachability analysis as a differential game between a full-order model and a simplified planning model [19]. 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 1 that builds on the reachability-based approach developed in [14] by using feedback linearization and zonotopes. This paper's contributions are three-fold. First, a novel parametrized partial feedback linearization controller to regulate the closed-loop vehicle dynamics that enables robust tracking performance even in the presence of modeling errors. Second, a zonotope-based technique that computes a control-parameterized, over-approximate reachable set, which describes the behavior of the closed-loop, full-order vehicle dynamics. Because this reachability analysis is performed on the full-order model, potential conservativeness introduced by using a simplified model is avoided. Third, 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. The proposed planning framework applies to Front-Wheel, Rear-Wheel, or All-Wheel-Drive vehicle models.

The rest of this manuscript is organized as follows: Section II describes necessary preliminaries, and section III describes Front-Wheel-Drive vehicle as a hybrid system. Section IV explains trajectory parameterization and vehicle safety in considered dynamic environments. Section V formulates the robust partial feedback linearization controller. Section VI describes REFINE at a high level and how to perform offline reachability analysis using zonotopes. Section VII formulates the online planning using an optimization program. Section VIII 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. Section IX discusses the applicability and limitations of the proposed framework, and Section X concludes the paper.

II. 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. A subscript is primarily used as an index or to describe a 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 $\mathcal{A} \oplus \mathcal{A}' = \{a + a' \mid a \in \mathcal{A},\ a' \in \mathcal{A}'\}$. 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 $\{\gamma \in \mathbb{R}^n \mid [\alpha]_i \leq [\gamma]_i \leq [\beta]_i,\ \forall i = 1, \ldots, n\}$. 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 respectively for any i, 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 paper:

Definition 1. *A* zonotope $\mathcal{Z}$ *is a subset of* $\mathbb{R}^n$ *defined as*

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

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* $G = [g_1, g_2, \ldots, g_\ell] \in \mathbb{R}^{n \times \ell}$.

Let $\mathbb{ZO}(\mathcal{A})$ denote the set of all zonotopes in some space $\mathcal{A}$. Note that an n-dimensional box is a zonotope because

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

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

$$A\mathcal{Z} = \left\{ x \in \mathbb{R}^n \mid x = Ac + \sum_{k=1}^{\ell} \beta_k A g_k,\ \beta_k \in [-1, 1] \right\}. \tag{3}$$

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

III. Hybrid Vehicle Model

This section introduces the vehicle model that is used throughout this manuscript while performing autonomous planning. Because traditional tire models become intractable when a vehicle travels at low speed, we model the vehicle as a hybrid system [20, Section 1.2] with a high-speed and a low-speed mode. This representation allows us to compute tight over-approximations to the behavior of the vehicle at all speeds as is described in Section VI. This section starts by introducing the vehicle states and tire models, then presents vehicle dynamics in different speed modes. It concludes by describing the instantaneous transitions between the two modes using a guard and reset map.

A. Vehicle States and Tire Models

The approach described in this paper can be applied to 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 in this manuscript. Extensions to AWD and RWD vehicles can be found online[2]. Given $\mathcal{W} \subset \mathbb{R}^2$ as the world space, we attach a body-fixed coordinate frame on the ground plane to the vehicle as shown in Fig. 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 $x(t) = [w_\text{x}(t), w_\text{y}(t), h(t), v_\text{x}(t), v_\text{y}(t), r(t)]^\top \in \mathbb{R}^6$ be the state value of the hybrid vehicle model at time t, where $w_\text{x}(t)$ and $w_\text{y}(t)$ give the position of vehicle's center of mass in the xy-plane of the world frame, $h(t)$ is the heading of the vehicle in the world frame, $v_\text{x}(t)$ and $v_\text{y}(t)$ 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. Note, negative longitudinal speed is not considered in this work.

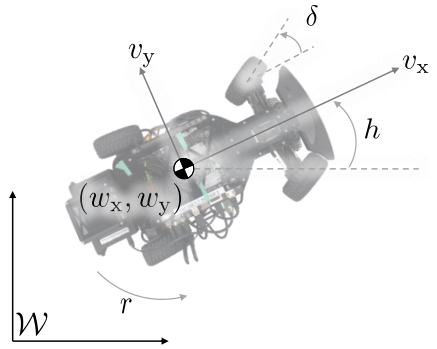


Fig. 2: Vehicle model with the world frame shown in black and body frame in gray.

To describe the tire forces along the longitudinal and lateral directions, we adapt tire models from [21, Chapter 4, Chapter 13]. We first define the *slip ratio* as

$$\lambda_\text{i}(t) = \begin{cases} \dfrac{r_\text{w}\omega_\text{i}(t) - v_\text{x}(t)}{v_\text{x}(t)}, & \text{during braking} \\ \dfrac{r_\text{w}\omega_\text{i}(t) - v_\text{x}(t)}{r_\text{w}\omega_\text{i}(t)}, & \text{during acceleration} \end{cases} \tag{4}$$

where the 'i' subscript can be replaced 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 $r_\text{w}\omega_\text{i}(t) - v_\text{x}(t) < 0$, and acceleration corresponds to whenever $r_\text{w}\omega_\text{i}(t) - v_\text{x}(t) \geq 0$. We then define *slip angles* of front and rear tires as

$$\alpha_\text{f}(t) = \delta(t) - \frac{v_\text{y}(t) + l_\text{f} r(t)}{v_\text{x}(t)}, \tag{5}$$

$$\alpha_\text{r}(t) = -\frac{v_\text{y}(t) - l_\text{r} r(t)}{v_\text{x}(t)}, \tag{6}$$

where l_f and l_r are the distances from center of mass to the front and back of the vehicle.

When the magnitudes of the slip ratio and slip angle are below critical values, λ^cri and α^cri, the longitudinal tire force, $F_\text{xi}(t)$, and lateral tire force, $F_\text{yi}(t)$, are linear functions of the

[2]https://roahmlab.github.io/REFINE_website/web_elements/REFINE_Supplimentary_generalization.pdf

slip ratio and slip angle, respectively. During online planning we are able to guarantee that tire forces operate in the linear regime as is described in supplementary material that can be found online [3]. Therefore, we make the following assumption:

Assumption 2. *The absolute values of 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 2 ensures that the longitudinal tire force can be described as

$$F_{\mathrm{xi}}(t) = \frac{mgl_{\mathrm{i}}}{l}\mu\lambda_{\mathrm{i}}(t), \tag{7}$$

and the lateral tire force can be described as

$$F_{\mathrm{yi}}(t) = c_{\alpha\mathrm{i}}\alpha_{\mathrm{i}}(t), \tag{8}$$

with constants $\mu, c_{\alpha\mathrm{i}} \in \mathbb{R}$, $l = l_{\mathrm{f}} + l_{\mathrm{r}}$, and g as the gravitational acceleration constant. Note μ and $c_{\alpha\mathrm{i}}$ are referred to as *surface-adhesion coefficient* and *cornering stiffness* respectively. Note that in FWD vehicles, the longitudinal rear wheel tire force has a much simpler expression:

Remark 3 ([22])**.** *In a FWD vehicle, $F_{xr}(t) = 0$ for all t.*

Note, as the vehicle speed approaches zero, the denominators of the slip ratio (4) and slip angles (5) and (6) approach zero, which makes applying the tire model, (7) and (8), untenable. Thus we describe the vehicle dynamics in two modes as a function of a critical longitudinal velocity $v_{\mathrm{x}}^{\mathrm{cri}}$: high-speed (i.e., $v_{\mathrm{x}}(t) > v_{\mathrm{x}}^{\mathrm{cri}}$) and low-speed (i.e., $v_{\mathrm{x}}(t) \leq v_{\mathrm{x}}^{\mathrm{cri}}$). Note that the critical velocity $v_{\mathrm{x}}^{\mathrm{cri}}$ can be computed [23, (5) and (18)].

B. High-Speed Mode

The vehicle dynamics in the high-speed mode, which is adapted from the standard bicycle model [22, Chapter 10.4], are described as:

$$\dot{x}(t) = \begin{bmatrix} v_{\mathrm{x}}(t)\cos h(t) - v_{\mathrm{y}}(t)\sin h(t) \\ v_{\mathrm{x}}(t)\sin h(t) + v_{\mathrm{y}}(t)\cos h(t) \\ r(t) \\ \frac{1}{m}\big(F_{\mathrm{xf}}(t) + F_{\mathrm{xr}}(t)\big) + v_{\mathrm{y}}(t)r(t) + \Delta_{v_{\mathrm{x}}}(t) \\ \frac{1}{m}\big(F_{\mathrm{yf}}(t) + F_{\mathrm{yr}}(t)\big) - v_{\mathrm{x}}(t)r(t) + \Delta_{v_{\mathrm{y}}}(t) \\ \frac{1}{I_{\mathrm{zz}}}\big(l_{\mathrm{f}}F_{\mathrm{yf}}(t) - l_{\mathrm{r}}F_{\mathrm{yr}}(t)\big) + \Delta_r(t) \end{bmatrix}. \tag{9}$$

where 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. Note that we have introduced time-varying affine modeling errors $\Delta_{v_{\mathrm{x}}}$, $\Delta_{v_{\mathrm{y}}}$ and Δ_r in (9) to account for aerodynamic-drag force [21, Section 4.2], inaccurate state estimation, and inaccurate tire force models that may arise in real applications. To ensure dynamics (9) is well-posed (i.e., its solution exists and is unique) and to aid in the development of our controller as described in Section V, we make the following assumption:

Assumption 4. *$\Delta_{v_{\mathrm{x}}}, \Delta_{v_{\mathrm{y}}}, \Delta_r$ are all square-integrable functions and are bounded (i.e., there exist real numbers $M_{v_{\mathrm{x}}}, M_{v_{\mathrm{y}}}, M_r \in [0, +\infty)$ such that $\|\Delta_{v_{\mathrm{x}}}(t)\|_\infty \leq M_{v_{\mathrm{x}}}$, $\|\Delta_{v_{\mathrm{y}}}(t)\|_\infty \leq M_{v_{\mathrm{y}}}$, $\|\Delta_r(t)\|_\infty \leq M_r$ for all t).*

[3]https://roahmlab.github.io/REFINE_website/web_elements/REFINE_Supplimentary_tire.pdf

In Section VIII-D1, we explain how we obtain $\Delta_u, \Delta_v, \Delta_r$ using real-world data and interested readers can read [24]–[26] for more systematic approaches on determining modeling errors. Note that the steering angle of the front wheel, δ, and the front tire rotational speed, ω_{f}, are the inputs that one is able to control in the real world. However, when we formulate our controller in Section V, 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} given F_{xf} and F_{yf}.

C. Low-Speed Mode

When $v_{\mathrm{x}}(t) \leq v_{\mathrm{x}}^{\mathrm{cri}}$, the vehicle dynamics in low-speed mode is modeled using a steady-state cornering model [27, Chapter 6], [28, Chapter 5], [29, Chapter 10], which is described using four states, $[w_{\mathrm{x}}(t), w_{\mathrm{y}}(t), h(t), v_{\mathrm{x}}(t)]^\top \in \mathbb{R}^4$, at time t. This model ignores transients on lateral velocity and yaw rate. Note that the dynamics of w_{x}, w_{y}, h and v_{x} are the same as in the high-speed mode; however, the steady-state corning model describes the yaw rate and lateral speed as

$$v_{\mathrm{y}}^{\mathrm{lo}}(t) = l_{\mathrm{r}}r^{\mathrm{lo}}(t) - \frac{ml_{\mathrm{f}}}{c_{\alpha\mathrm{r}}l}v_{\mathrm{x}}(t)^2 r^{\mathrm{lo}}(t) \tag{10}$$

$$r^{\mathrm{lo}}(t) = \frac{\delta(t)v_{\mathrm{x}}(t)}{l + C_{\mathrm{us}}v_{\mathrm{x}}(t)^2} \tag{11}$$

with understeer coefficient

$$C_{\mathrm{us}} = \frac{m}{l}\left(\frac{l_{\mathrm{r}}}{c_{\alpha\mathrm{f}}} - \frac{l_{\mathrm{f}}}{c_{\alpha\mathrm{r}}}\right). \tag{12}$$

Therefore in the low-speed mode, the dynamics of the first four states in x remain the same as in (9), but we assign v_{y} and r with *zero* dynamics. Note that such an assignment does not affect the evolution of the hybrid vehicle model because the actual lateral speed and yaw rate are directly computed via (10) and (11) in the low-speed mode.

Notice when $v_{\mathrm{x}}(t) = v_{\mathrm{y}}^{\mathrm{lo}}(t) = r^{\mathrm{lo}}(t) = 0$ and longitudinal tire forces are zero, $\dot{v}_{\mathrm{x}}(t)$ could still be nonzero due to a nonzero $\Delta_{v_{\mathrm{x}}}(t)$. To avoid this issue, we make a tighter assumption on $\Delta_{v_{\mathrm{x}}}(t)$ without violating Assumption 4:

Assumption 5. *For all t such that $v_x(t) \in [0, v_x^{cri}]$, $|\Delta_{v_x}(t)|$ is bounded from above by a linear function of $v_x(t)$ (i.e.,*

$$|\Delta_{v_x}(t)| \leq b_{v_x}^{pro} \cdot v_x(t) + b_{v_x}^{off}, \text{ if } v_x(t) \in [0, v_x^{cri}], \tag{13}$$

where $b_{v_x}^{pro}$ and $b_{v_x}^{off}$ are constants satisfying $b_{v_x}^{pro} \cdot v_x^{cri} + b_{v_x}^{off} \leq M_{v_x}$). In addition, $\Delta_{v_x}(t) = 0$ if $v_x(t) = 0$.

D. Guard and Reset Map

Transitions between high-speed and low-speed modes are described using the notion of *guard* and *reset map*. The guard determines when a transition occurs and is defined as $\{x(t) \in \mathbb{R}^6 \mid v_{\mathrm{x}}(t) = v_{\mathrm{x}}^{\mathrm{cri}}\}$. When a transition occurs at some time t, the reset map is an identity for the first four vehicle states. If $v_{\mathrm{x}}(t)$ approaches $v_{\mathrm{x}}^{\mathrm{cri}}$ from below (i.e., the system is transitioning from low to high-speed), then the lateral speed and yaw rate are set equal to $v_{\mathrm{y}}^{\mathrm{lo}}(t)$ and $r^{\mathrm{lo}}(t)$, respectively. If $v_{\mathrm{x}}(t)$ approaches $v_{\mathrm{x}}^{\mathrm{cri}}$ from above (i.e., the system is transitioning from high to low-speed), then one computes $v_{\mathrm{y}}^{\mathrm{lo}}(t)$ and $r^{\mathrm{lo}}(t)$ by using (10) and (11), respectively.

IV. Trajectory Design and Safety

REFINE is a receding-horizon trajectory planner that is able to guarantee the safety of any returned plan. To ensure real-time performance, REFINE optimizes over a parameterized set of desired trajectories. This parametrization is also used by the offline reachability analysis that is described in Section VI. This section describes the space of parameterized trajectories that are used online, defines safety during motion planning via the notion of not-at-fault behavior, and makes assumptions about the environment surrounding the ego-vehicle.

A. 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 6. *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. This trajectory is 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 7. *For each $p \in \mathcal{P}$, a* desired trajectory *consists of a function for the longitudinal speed, $v_x^{des}(\cdot, p) : [t_0 + t_{plan}, t_0 + t_{plan} + t_f] \to \mathbb{R}$, 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., $|v_x^{des}(t, p)|$, $|h^{des}(t, p)|$ 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 $v_x^{des}(t, p) = h^{des}(t, p) = r^{des}(t, p) = 0$ for all $t \geq t_{stop}$).*
3) *v_x^{des} and h^{des} are piecewise continuously differentiable [30, 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 failsafe contingency braking maneuver is always available and the latter two properties ensure that the tracking controller described in Section V is well-defined. Note, sometimes we abuse notation and evaluate a desired trajectory for $t > t_0 + t_{\mathrm{plan}} + t_{\mathrm{f}}$. In this instance, the value of the desired trajectory is equal to its value at t_{f}.

B. Not-At-Fault

In dynamic environments, avoiding collision may not always be possible (e.g., a parked car can be run into). Therefore, we instead develop a trajectory synthesis technique which ensures that the ego vehicle is not-at-fault [5], [10], [16]:

Definition 8. *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 leaves enough time for all surrounding vehicles to come safely to a stop as well, under the assumption that surrounding obstacles behave rationally (i.e., travel at some maximum speed, have a bounded reaction time, and begin to brake after that reaction time). We want to point out that stopping is merely one choice that can be applied in general as a fail-safe maneuver. In scenarios where stopping is not always safe, for example in a railroad crossing, one could extend Definition 8 and require that the final position of the vehicle be outside of some location. The remainder of the paper can be generalized to accommodate these variants of not-at-fault; however, we use the aforementioned definition in the interest of simplicity.

Remark 9. *Under Assumption 2, 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 8, the ego vehicle being stopped is equivalent to its longitudinal speed being* 0.

C. Environment and Sensing

To provide guarantees about vehicle behavior in a receding horizon planning framework and inspired by [16, Section 3], we define the ego vehicle's footprint as:

Definition 10. *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 11. *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 for all time. Assuming that the ego vehicle has a maximum speed ν^{ego} and all obstacles share the same maximum speed ν^{obs} for all time, we then make the following assumption on planning and sensing horizon.

Assumption 12. *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 12 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 [16, Theorem 15]. Note one could conservatively treat sensor occlusions as obstacles that travel at the maximum obstacle speed [31], [32]. However, planning with occlusions and uncertain behaviors of surrounding traffic participants is beyond the scope of this work, and interested readers may find how these are considered in [33].

V. Controller Design

REFINE uses a robust controller to follow the desired trajectories that are described in the last section. Recall that the actual control inputs to the vehicle dynamics model are the steering angle of the front wheel, δ, and the front tire rotational speed, ω_{f}. Section V-A describes how to select front tire forces to follow a desired trajectory and Section V-B describes how to compute a steering angle and tire rotational speed input from these computed front tire forces.

A. 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 vehicle dynamics in both high-speed and low-speed modes introduced in Section III 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 (i.e., feedback linearize the dynamics of a subset of the vehicle states). We also want the controller to be robust to account for modeling errors as described in Assumptions 4 and 5.

We start by introducing the controller on longitudinal speed whose dynamics appears in both high-speed and low-speed models. Recall $\|\Delta_{v_{\text{x}}}(t)\|_\infty \leq M_{v_{\text{x}}}$ in Assumption 4. Inspired by the controller developed in [34], we set the longitudinal front tire force to be

$$\begin{aligned} F_{\text{xf}}(t) = &-mK_{v_{\text{x}}}(v_{\text{x}}(t) - v_{\text{x}}^{\text{des}}(t,p)) + m\dot{v}_{\text{x}}^{\text{des}}(t,p) + \\ &-F_{\text{xr}}(t) - mv_{\text{y}}(t)r(t) + m\tau_{v_{\text{x}}}(t,p), \end{aligned} \tag{14}$$

where

$$\tau_{v_{\text{x}}}(t,p) = -\big(\kappa_{v_{\text{x}}}(t,p)M_{v_{\text{x}}} + \phi_{v_{\text{x}}}(t,p)\big)e_{v_{\text{x}}}(t,p), \tag{15}$$

$$\kappa_{v_{\text{x}}}(t,p) = \kappa_{1,v_{\text{x}}} + \kappa_{2,v_{\text{x}}}\int_{t_0}^{t} \|v_{\text{x}}(s) - v_{\text{x}}^{\text{des}}(s,p)\|^2 ds, \tag{16}$$

$$\phi_{v_{\text{x}}}(t,p) = \phi_{1,v_{\text{x}}} + \phi_{2,v_{\text{x}}}\int_{t_0}^{t} \|v_{\text{x}}(s) - v_{\text{x}}^{\text{des}}(s,p)\|^2 ds, \tag{17}$$

$$e_{v_{\text{x}}}(t,p) = v_{\text{x}}(t) - v_{\text{x}}^{\text{des}}(t,p), \tag{18}$$

with user-chosen constants $K_{v_{\text{x}}}, \kappa_{1,v_{\text{x}}}, \kappa_{2,v_{\text{x}}}, \phi_{1,v_{\text{x}}}, \phi_{2,v_{\text{x}}} \in \mathbb{R}_+$. Note in (14) we have suppressed the dependence on p in $F_{\text{xf}}(t)$ for notational convenience. Using (14), the closed-loop dynamics of v_{x} becomes:

$$\begin{aligned} \dot{v}_{\text{x}}(t) = &\tau_{v_{\text{x}}}(t,p) + \Delta_{v_{\text{x}}}(t) + \dot{v}_{\text{x}}^{\text{des}}(t,p) + \\ &-K_{v_{\text{x}}}\left(v_{\text{x}}(t) - v_{\text{x}}^{\text{des}}(t,p)\right). \end{aligned} \tag{19}$$

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

$$\begin{aligned} F_{\text{yf}}(t) = &-\frac{I_{\text{zz}}K_r}{l_{\text{f}}}\left(r(t) - r^{\text{des}}(t,p)\right) + \frac{I_{\text{zz}}}{l_{\text{f}}}\dot{r}^{\text{des}}(t,p) + \\ &-\frac{I_{\text{zz}}K_h}{l_{\text{f}}}\left(h(t) - h^{\text{des}}(t,p)\right) + \frac{l_{\text{r}}}{l_{\text{f}}}F_{\text{yr}}(t) + \frac{I_{\text{zz}}}{l_{\text{f}}}\tau_r(t,p), \end{aligned} \tag{20}$$

where

$$\tau_r(t,p) = -\big(\kappa_r(t,p)M_r + \phi_r(t,p)\big)e_r(t,p) \tag{21}$$

$$\kappa_r(t,p) = \kappa_{1,r} + \kappa_{2,r}\int_{t_0}^{t} \left\| \begin{bmatrix} r(s) \\ h(s) \end{bmatrix} - \begin{bmatrix} r^{\text{des}}(s,p) \\ h^{\text{des}}(s,p) \end{bmatrix} \right\|^2 ds \tag{22}$$

$$\phi_r(t,p) = \phi_{1,r} + \phi_{2,r}\int_{t_0}^{t} \left\| \begin{bmatrix} r(s) \\ h(s) \end{bmatrix} - \begin{bmatrix} r^{\text{des}}(s,p) \\ h^{\text{des}}(s,p) \end{bmatrix} \right\|^2 ds \tag{23}$$

$$e_r(t,p) = \begin{bmatrix} K_r & K_h \end{bmatrix} \begin{bmatrix} r(t) - r^{\text{des}}(t,p) \\ h(t) - h^{\text{des}}(t,p) \end{bmatrix} \tag{24}$$

with user-chosen constants $K_h, K_r, \kappa_{1,r}, \kappa_{2,r}, \phi_{1,r}, \phi_{2,r} \in \mathbb{R}_+$. Note in (20) we have again suppressed the dependence on p in $F_{\text{yf}}(t)$ for notational convenience. Using (20), the closed-loop dynamics of r becomes:

$$\begin{aligned} \dot{r}(t) = &\tau_r(t,p) + \Delta_r(t) + \dot{r}^{\text{des}}(t,p) + \\ &- K_r\big(r(t) - r^{\text{des}}(t,p)\big) + \\ &- K_h\big(h(t) - h^{\text{des}}(t,p)\big). \end{aligned} \tag{25}$$

Using (20), the closed-loop dynamics of v_{y} becomes:

$$\begin{aligned} \dot{v}_{\text{y}}(t) = \frac{1}{m}\Bigg(&\frac{l}{l_{\text{f}}}F_{\text{yr}}(t) + \frac{I_{\text{zz}}}{l_{\text{f}}}\Big(\tau_r(t,p) + \dot{r}^{\text{des}}(t,p) + \\ &-v_{\text{x}}(t)r(t) + \Delta_{v_{\text{y}}}(t) - K_r\big(r(t) - r^{\text{des}}(t,p)\big) + \\ &-K_h\big(h(t) - h^{\text{des}}(t,p)\big)\Big)\Bigg). \end{aligned} \tag{26}$$

In summary, the proposed controller linearizes the dynamics of the longitudinal speed and yaw. The proposed controller also affects the dynamics of v_{y}, but is unable to linearize it. As a result, the controller is only able to partially linearize the dynamics.

Because $v_{\text{x}}^{\text{des}}$, r^{des}, and h^{des} depend on trajectory parameter p, one can rewrite the closed-loop vehicle dynamics as

$$\dot{x}(t) = \begin{cases} f^{\text{hi}}(t, x(t), p), & \text{if } v_{\text{x}}(t) > v_{\text{x}}^{\text{cri}} \\ f^{\text{lo}}(t, x(t), p), & \text{if } v_{\text{x}}(t) \leq v_{\text{x}}^{\text{cri}} \end{cases} \tag{27}$$

where dynamics of w_{x}, w_{y} and h are stated as the first three dimensions in (9), closed-loop dynamics of v_{x} is described in (19), and closed-loop dynamics of v_{y} and r in the high-speed mode are presented in (26) and (25). Note that the lateral tire force could be defined to feedback linearize the dynamics on v_{y} instead of r, but the resulting closed-loop system may differ. Moreover, controlling the yaw rate may be

easier in real applications because r can be directly measured by an IMU. We point out that the proposed controller provides design flexibility in the sense that the user can manage the rate of convergence of the linearized states by selecting the gains within the controller.

We next illustrate that with the proposed controller, the vehicle eventually comes to a stop in finite time. To begin, note experimentally we observed that the vehicle quickly comes to a stop during braking once its longitudinal speed is no larger than 0.15[m/s]. Thus we make this assumption:

Assumption 13. *Suppose $v_x(t) = 0.15$ for some $t \geq t_{stop}$. Then under the control inputs* (14) *and* (20) *while tracking any desired trajectory as in Definition 7, 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 A:

Lemma 14. *Let $\mathcal{X}_0 \subset \mathbb{R}^6$ be a compact subset of initial conditions for the vehicle dynamic model at time $t_0 + t_{plan}$ and $\mathcal{P}$ be a compact set of trajectory parameters. Without loss of generality, assume $t_0 + t_{plan} = 0$. Let $\Delta_{v_x}(t)$ be bounded for all t as in Assumptions 4 and 5 with constants M_{v_x}, $b^{pro}_{v_x}$ and $b^{off}_{v_x}$. Let x be a solution to the hybrid vehicle model beginning from $x_0 \in \mathcal{X}_0$ under trajectory parameter $p \in \mathcal{P}$ while applying the control inputs* (14) *and* (20) *to track some desired trajectory satisfying Definition 7. Assume the desired longitudinal speed satisfies the following properties: $v^{des}_x(0, p) = v_x(0)$, $v^{des}_x(t, p)$ is only discontinuous at time t_{stop}, and $v^{des}_x(t, p)$ converges to v^{cri}_x as t converges to t_{stop} from below. If K_{v_x}, κ_{1,v_x}, ϕ_{1,v_x} are chosen such that $\frac{M_{v_x}}{\kappa_{1,v_x} M_{v_x} + \phi_{1,v_x}} \in (0.15, v^{cri}_x]$ and $\frac{(b^{off}_{v_x})^2}{4(\kappa_{1,v_x} M_{v_x} + \phi_{1,v_x} - b^{pro}_{v_x})} < 0.15^2 K_{v_x}$ hold, then for all $p \in \mathcal{P}$ and $x_0 \in \mathcal{X}_0$ satisfying $v_x(0) > 0$, there exists t_{brake} such that $v_x(t) = 0$ for all $t \geq t_{brake}$.*

Note, the proof of Lemma 14 includes an explicit formula for t_{brake} in (59). 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 8 and Remark 9.

B. 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 (14) and (20) is viable. Under Assumption 2, wheel speed and steering inputs can be directly computed in closed forms. Wheel speed to realize longitudinal front tire force (14) can be derived from (4) and (7) as

$$\omega_{\text{f}}(t) = \begin{cases} \left(\dfrac{l F_{\text{xf}}(t)}{\mu m g l_{\text{r}}} + 1 \right) \dfrac{v_{\text{x}}(t)}{r_{\text{w}}}, & \text{during braking} \\ \dfrac{v_{\text{x}}(t)}{\left(1 - \frac{l F_{\text{xf}}(t)}{\mu m g l_{\text{r}}} \right) r_{\text{w}}}, & \text{during acceleration.} \end{cases} \tag{28}$$

Similarly according to (5) and (8), steering input

$$\delta(t) = \frac{F_{\text{yf}}(t)}{c_{\alpha \text{f}}} + \frac{v_{\text{y}}(t) + l_{\text{f}} r(t)}{v_{\text{x}}(t)} \tag{29}$$

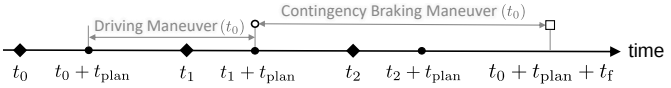


Fig. 3: An illustration of 3 successive planning/control iterations. t_{plan} seconds are allotted to compute a planned trajectory. Each plan is of duration t_{f} and consists of a driving maneuver of duration t_{m} and a contingency braking maneuver. Diamonds denote the time instances where planning 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.

achieves the lateral front tire force in (20) when $v_{\text{x}}(t) > v^{\text{cri}}_{\text{x}}$.

Notice lateral tire forces do not appear in the low-speed dynamics, but one can still control the lateral behavior of the ego vehicle because steering input $\delta(t)$ directly controls the yaw rate and affects the lateral velocity based on (10) and (11). Thus to achieve desired behavior on the lateral direction, one can set the steering input to be

$$\delta(t) = \frac{r^{\text{des}}(t)(l + C_{\text{us}} v_{\text{x}}(t)^2)}{v_{\text{x}}(t)}. \tag{30}$$

VI. Computing and Using the FRS

Before delving into how we construct FRS, we briefly describe how REFINE uses reachable sets. REFINE conservatively approximates a control-parameterized FRS of the full-order vehicle dynamics offline. Because the vehicle dynamics are partially linearized by the controller in Section V, REFINE is able to construct tight, over-approximatios of the FRS using zonotopes. During online planning, REFINE performs control synthesis in a receding horizon fashion by solving an optimization problem in each planning iteration, where the optimization problem computes a trajectory parameter to navigate the ego vehicle to a waypoint while operating in a not-at-fault manner.

As in Assumption 6, each planning iteration in REFINE is allotted $t_{\text{plan}} > 0$ to generate a plan. As depicted in Figure 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 7 brings the ego vehicle to a stop, we partition $[t_0 + t_{\text{plan}}, t_0 + t_{\text{plan}} + t_{\text{f}}]$ 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 14, 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 $[t_0 + t_{\text{plan}}, t_0 + t_{\text{plan}} + t_{\text{m}})$. 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, $t_{\text{plan}} \leq t_{\text{m}}$. To simplify notation, we reset time to 0 whenever a feasible control policy is about to be applied.

Note that this form of receding horizon planning that incorporates a contingency maneuver (e.g., bringing the ego vehicle

to a stop) has been proposed by several papers [5], [10], [16], [17]. However, in contrast to [10], our method performs real-time, provably not-at-fault trajectory planning; and in contrast to [17], our method uses a different representation of the forward reachable set of the vehicle that allows for a tighter overapproximation to the true behavior of the ego vehicle as we illustrate in Section VIII.

This section focuses on the offline reachability analysis, while the online planning is described in the next section. The rest of this section begins by describing how the initial velocity condition and trajectory parameter are appended to the vehicle state. Next, this section describes how to perform offline reachability analysis for the ego vehicle using zonotopes. It then presents a linear operation called *slicing* that allows one to generate a tighter FRS over-approximation for a user-specified initial velocity condition and trajectory parameter. This section concludes by illustrating how we account for the ego vehicle's footprint during reachability analysis.

A. Augmented State and Augmented Hybrid System

We begin by augmenting the initial velocity condition of the ego vehicle and trajectory parameter into the vehicle state x. As we show in Section VI-C, this allows us to select subsets of the reachable set of the system during online operation (i.e., *slice* the reachable set). Denote $x_0 = x(0) = [(x_0^{\text{pos}})^\top, (x_0^{\text{vel}})^\top]^\top \in \mathcal{X}_0 \subset \mathbb{R}^6$ the initial condition of the ego vehicle, consisting of $x_0^{\text{pos}} = [w_{\text{x},0}, w_{\text{y},0}, h_0]^\top \in \mathbb{R}^3$ and $x_0^{\text{vel}} = [v_{\text{x},0}, v_{\text{y},0}, r_0]^\top \in \mathbb{R}^3$. Then we augment the initial velocity condition x_0^{vel} of the vehicle model and trajectory parameter p into the vehicle state vector as $x^{\text{aug}}(t) = [x(t)^\top, (x_0^{\text{vel}})^\top, p^\top]^\top \in \mathbb{R}^9 \times \mathcal{P} \subset \mathbb{R}^{9+n_p}$, where the last $3+n_p$ states are time-invariant.

Consequently, the hybrid vehicle model introduced in Section III can be extended to an augmented hybrid system HS that is defined in the augmented vehicle state space. The state of HS is x^{aug}, whose dynamics during high-speed and low-speed modes can be written as

$$\dot{x}^{\text{aug}}(t) = \begin{cases} \begin{bmatrix} f^{\text{hi}}(t, x(t), p) \\ 0_{(3+n_p)\times 1} \end{bmatrix}, \text{ if } v_{\text{x}}(t) > v_{\text{x}}^{\text{cri}} \\ \\ \begin{bmatrix} f^{\text{lo}}(t, x(t), p) \\ 0_{(3+n_p)\times 1} \end{bmatrix}, \text{ if } v_{\text{x}}(t) \leq v_{\text{x}}^{\text{cri}}. \end{cases} \tag{31}$$

The guard of HS is defined as $\mathcal{G} = \{x^{\text{aug}}(t) \in \mathbb{R}^9 \times \mathcal{P} \mid v_{\text{x}}(t) = v_{\text{x}}^{\text{cri}}\}$. Once a transition happens, HS maintains the last $3 + n_p$ elements of $x^{\text{aug}}(t)$ and resets the first 6 elements of $x^{\text{aug}}(t)$ in the same way as described in Section III-D.

B. Offline FRS Computation

The *FRS* of the ego vehicle is defined as

$$\begin{aligned} \mathcal{F}([0, t_{\text{f}}]) = \Big\{ & (w_{\text{x}}(t), w_{\text{y}}(t)) \in \mathcal{W} \mid \exists x_0 \in \mathcal{X}_0, p \in \mathcal{P}, \\ & t \in [0, t_{\text{f}}] \text{ s.t. } w_{\text{x}}(t) \text{ and } w_{\text{y}}(t) \text{ are the} \\ & \text{first two components of } x(t) \text{ where } x \text{ is a} \\ & \text{solution to (27) with } x(0) = x_0 \text{ and} \\ & \text{trajectory parameter } p \Big\}. \end{aligned} \tag{32}$$

$\mathcal{F}([0, t_{\text{f}}])$ collects all possible behavior of the hybrid vehicle model described in Section III in the world space over time interval $[0, t_{\text{f}}]$ for all possible $p \in \mathcal{P}$ and initial condition $x_0 \in \mathcal{X}_0$. Computing $\mathcal{F}([0, t_{\text{f}}])$ precisely is intractable because the ego vehicle is modeled as a hybrid system with nonlinear dynamics. Thus we aim to compute an over-approximation of $\mathcal{F}([0, t_{\text{f}}])$ instead.

We begin by assuming that the initial position, x_0^{pos}, is at the origin for simplicity. Note, we address non-trivial x_0^{pos} by applying a coordinate transformation as is described in Section VII-A.

Assumption 15. *The initial condition space $\mathcal{X}_0 = \{0_{3\times 1}\} \times \mathcal{X}_0^{vel}$ where $\mathcal{X}_0^{vel} = int(\underline{x_0^{vel}}, \overline{x_0^{vel}}) \subset \mathbb{R}^3$ is a 3-dimensional box representing all possible initial velocity conditions x_0^{vel} of the ego vehicle.*

Because vehicles operate within a bounded range of speeds,the above assumption is trivial to satisfy. Notice that $\mathcal{X}_0^{\text{vel}}$ is a zonotope $<c_0^{\text{vel}},\ G_0^{\text{vel}}>$ where $c_0^{\text{vel}} = \frac{1}{2}(\underline{x_0^{\text{vel}}} + \overline{x_0^{\text{vel}}})$ and $G_0^{\text{vel}} = \frac{1}{2}\texttt{diag}(\overline{x_0^{\text{vel}}} - \underline{x_0^{\text{vel}}})$.

Recall that because $\mathcal{P}$ is a compact n_p-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 $x^{\text{aug}}(0)$ can be represented as a zonotope $\mathcal{X}_0^{\text{aug}} = <c_{x^{\text{aug}}},\ G_{x^{\text{aug}}}>$ where

$$c_{x^{\text{aug}}} = \begin{bmatrix} 0_{3\times 1} \\ c_0^{\text{vel}} \\ c_0^{\text{vel}} \\ c_p \end{bmatrix},\ G_{x^{\text{aug}}} = \begin{bmatrix} 0_{3\times 3} & 0_{3\times n_p} \\ G_0^{\text{vel}} & 0_{3\times n_p} \\ G_0^{\text{vel}} & 0_{3\times n_p} \\ 0_{n_p\times 3} & G_p \end{bmatrix}. \tag{33}$$

Observe that by construction each row of $G_{x^{\text{aug}}}$ has at most one 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 traced or augmented in x^{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*, where $T_j = [(j-1)\Delta_t, j\Delta_t]$ for each $j \in \mathcal{J} = \{1, 2, \cdots, t_{\text{f}}/\Delta_t\}$. Finally we use an open-source toolbox CORA [35], which takes the initial condition space $\mathcal{X}_0^{\text{aug}}$ and the augmented hybrid system HS introduced in Section VI-A, to over-approximate the FRS in (32) by a collection of zonotopes $\{\mathcal{R}_j\}_{j\in\mathcal{J}}$ over all time intervals where $\mathcal{R}_j \subset \mathbb{R}^{9+n_p}$. Notice that reachability analysis of a hybrid system requires the computation of $\mathcal{G} \cap \mathcal{R}_j$, which equals to $\mathcal{R}_j$ or $\emptyset$ depending on whether the projection of $\mathcal{R}_j$ onto its dimension of v_{x} contains $v_{\text{x}}^{\text{cri}}$ or not. As a direct application of Theorem 3.3, Proposition 3.7 and the derivation in Section 3.5.3 in [36], one can conclude that $x^{\text{aug}}(t) \in \mathcal{R}_j$ for all $j \in \mathcal{J}$ and $t \in T_j$ and

$$\mathcal{F}([0, t_{\text{f}}]) \subset \bigcup_{j\in\mathcal{J}} \pi_w(\mathcal{R}_j) \tag{34}$$

where $\pi_w : \mathbb{ZO}(\mathbb{R}^{9+n_p}) \to \mathbb{ZO}(\mathcal{W})$ denotes the projection operator that maps an arbitrary zonotope $\mathcal{R}_j = <c_{\mathcal{R}_j},\ G_{\mathcal{R}_j}>$ onto the world space $\mathcal{W}$ as

$$\pi_w(\mathcal{R}_j) = \left\langle \begin{bmatrix} [c_{\mathcal{R}_j}]_1 \\ [c_{\mathcal{R}_j}]_2 \end{bmatrix},\ \begin{bmatrix} [G_{\mathcal{R}_j}]_{1:} \\ [G_{\mathcal{R}_j}]_{2:} \end{bmatrix} \right\rangle. \tag{35}$$

Remark 16. *Note, that one does not need to apply CORA to compute the FRS. As long as one had a tool that computed the FRS as a set of zonoptes that satisfied* (34)*, the results in the remainder of the paper can be applied.*

C. Slicing

The FRS over-approximation computed in the previous subsection contains the behavior of the hybrid vehicle model for all initial conditions in $\mathcal{X}_0$ and trajectory parameters in $\mathcal{P}$. Therefore using this FRS to perform obstacle avoidance starting from a specific initial condition while following a specific trajectory parameter may be too conservative during online planning. 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 15. This subsection describes how REFINE selects subsets of the FRS by "slicing" in both the initial velocity of the vehicle and trajectory parameter into the FRS. This operation allows REFINE to compute tighter zonotope reachable sets that can then be used for collision checking during online planning.

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 [37]:

Proposition 17. *Let $\{\mathcal{R}_j = <c_{\mathcal{R}_j},\ G_{\mathcal{R}_j}>\}_{j\in\mathcal{J}}$ be the set of zonotopes that over-approximates the ego vehicle's FRS under the augmented hybrid system HS beginning from $\mathcal{X}_0^{aug}$. 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 $\mathcal{X}_0^{\text{aug}}$ has exactly $3+n_p$ generators, which are each sliceable. Using Proposition 17, one can conclude that $\mathcal{R}_j$ has no less than $3+n_p$ generators (i.e., $\ell \geq 3+n_p$). Notice the last $3+n_p$ dimensions of $\mathcal{R}_j$ correspond to the augmented initial velocity condition and trajectory parameter in x^{aug}.

Proposition 17 is useful because it allows us to take a known $x_0^{\text{vel}} \in \mathcal{X}_0^{\text{vel}}$ 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 x_0^{vel} under trajectory parameter p. In particular, one can plug the initial velocity and a trajectory parameter into the sliceable generators as described in the following definition:

Definition 18. *Let $\{\mathcal{R}_j = <c_{\mathcal{R}_j},\ G_{\mathcal{R}_j}>\}_{j\in\mathcal{J}}$ be the set of zonotopesthat over-approximates the ego vehicle's FRS under the augmented hybrid system HS beginning from $\mathcal{X}_0^{aug}$ where $G_{\mathcal{R}_j} = [g_{\mathcal{R}_j,1}, g_{\mathcal{R}_j,2}, \ldots, g_{\mathcal{R}_j,\ell_j}]$. 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 $\texttt{slice} : \mathbb{ZO}(\mathbb{R}^{9+n_p}) \times \mathcal{X}_0^{vel} \times \mathcal{P} \to \mathbb{ZO}(\mathbb{R}^{9+n_p})$ *is defined as*

$$\texttt{slice}(\mathcal{R}_j, x_0^{vel}, p) = <c^{slc},\ [g_{\mathcal{R}_j,(4+n_p)}, \ldots, g_{\mathcal{R}_j,\ell_j}]> \quad (36)$$

where

$$\begin{aligned} c^{slc} = c_{\mathcal{R}_j} + \sum_{k=7}^{9} \frac{[x_0^{vel}]_{(k-6)} - [c_{\mathcal{R}_j}]_k}{[g_{\mathcal{R}_j,(k-6)}]_k} g_{\mathcal{R}_j,(k-6)} + \\ + \sum_{k=10}^{9+n_p} \frac{[p]_{(k-9)} - [c_{\mathcal{R}_j}]_k}{[g_{\mathcal{R}_j,(k-6)}]_k} g_{\mathcal{R}_j,(k-6)}. \end{aligned} \quad (37)$$

To understand this definition, let's consider plugging in only the initial longitudinal speed $v_{\text{x},0}$ into an arbitrary zonotope reachable set $\mathcal{R}_j = <c_{\mathcal{R}_j},\ [g_{\mathcal{R}_j,1}, g_{\mathcal{R}_j,2}, \cdots, g_{\mathcal{R}_j,\ell_j}]>$. Recall $v_{\text{x},0}$ is the 7th dimension in x^{aug}, and assume $g_{\mathcal{R}_j,1}$ is the corresponding generator whose 7th dimension is nonzero. Notice $\mathcal{R}_j = <c_{\mathcal{R}_j},\ [g_{\mathcal{R}_j,2}, \cdots, g_{\mathcal{R}_j,\ell_j}]> \oplus \{\beta_1 \cdot g_{\mathcal{R}_j,1} \mid \beta_1 \in [-1,1]\}$, and $v_{\text{x},0}$ is constant due to its zero dynamics. Thus by Proposition 17,

$$v_{\text{x},0} = [c_{\mathcal{R}_j}]_7 + \beta_1 \cdot [g_{\mathcal{R}_j,1}]_7, \quad (38)$$

which means β_1 can be computed exactly. The exact value of β_1 allows us to construct a new zonotope

$$\mathcal{R}'_j := <c_{\mathcal{R}_j},\ [g_{\mathcal{R}_j,2}, \cdots, g_{\mathcal{R}_j,\ell_j}]> + \frac{v_{\text{x},0} - [c_{\mathcal{R}_j}]_7}{[g_{\mathcal{R}_j,1}]_7} g_{\mathcal{R}_j,1} \quad (39)$$

that over-approximates the vehicle behavior over time interval T_j if the ego vehicle starts with longitudinal speed $v_{\text{x},0}$ at $t = 0$. As illustrated in Figure 4, $\mathcal{R}'_j$ is a subset of $\mathcal{R}_j$. Recall we have in total $3+n_p$ augmented states that have zero dynamics in x^{aug}, thus one can iteratively repeat computations in (38) and (39) for all elements in x_0^{vel} and p, thus a *linear* operator can be constructed as in Definition 18.

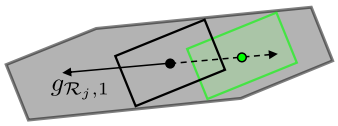


Fig. 4: An illustration of slicing. $\mathcal{R}_j = <c_{\mathcal{R}_j},\ [g_{\mathcal{R}_j,2}, \cdots, g_{\mathcal{R}_j,\ell_j}]> \oplus \{\beta_1 \cdot g_{\mathcal{R}_j,1} \mid \beta_1 \in [-1,1]\}$ is shown as the gray zonotope, where $c_{\mathcal{R}_j}$ is shown as the black dot, $<c_{\mathcal{R}_j},\ [g_{\mathcal{R}_j,2}, \cdots, g_{\mathcal{R}_j,\ell_j}]>$ is shown as the zonotope with black boundary, $g_{\mathcal{R}_j,1}$ and its negative are shown as black solid and dashed arrows, respectively. A sliced zonotope of $\mathcal{R}_j$ with $\beta_1 = -2/3$ is shown in green with its center depicted with a green dot.

Using the argument above, one can prove the following theorem:

Theorem 19. *Let $\{\mathcal{R}_j\}_{j\in\mathcal{J}}$ be the set of zonotopes that over-approximates the ego vehicle's FRS under the augmented hybrid system HS beginning from $\mathcal{X}_0^{aug}$ and satisfy the statement of Definition 18. Then for any $j \in \mathcal{J}$, $x_0 = [0,0,0,(x_0^{vel})^\top]^\top \in \mathcal{X}_0$, and $p \in \mathcal{P}$, $\texttt{slice}(\mathcal{R}_j, x_0^{vel}, p) \subset \mathcal{R}_j$. In addition, suppose x^{aug} is a solution to HS with initial condition x_0 and control parameter p. Then for each $j \in \mathcal{J}$ and $t \in T_j$,*

$$x^{aug}(t) \in \texttt{slice}(\mathcal{R}_j, x_0^{vel}, p). \quad (40)$$

Theorem 19 provides a tighter over-approximation of the vehicle behavior compared to $\mathcal{R}_j$ given the ego vehicle is operated with some known initial speed condition and trajectory parameter. As a result, this theorem improves the feasibility of online planning because we now only need to ensure not-at-fault during online planning for a particular value of initial speed and trajectory parameter via slicing instead of directly using $\mathcal{R}_j$ to ensure not-at-fault for all possible initial velocity conditions and trajectory parameters.

D. Accounting for the Vehicle Footprint in the FRS

The conservative representation of the FRS generated in Section VI-B 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 10.

Given any $R_j = <c_{R_j},\ G_{R_j}>$ computed in Section VI-B, define a projection operator $\pi_h : \mathbb{ZO}(\mathbb{R}^{9+n_p}) \to \mathbb{ZO}(\mathbb{R})$ as $\pi_h(R_j) \mapsto <[c_{R_j}]_3,\ [G_{R_j}]_{3:}>$. 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 20. *Let $\mathcal{R}_j$ be the zonotope that over-approximates the ego vehicle's FRS beginning from $\mathcal{X}_0^{aug}$ 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 $rot:\ \mathbb{ZO}(\mathbb{R}) \to\ \mathbb{ZO}(\mathcal{W})$ *as*

$$rot\big(\pi_h(\mathcal{R}_j)\big) := \begin{bmatrix} \cos(h^{mid}) & -\sin(h^{mid}) \\ \sin(h^{mid}) & \cos(h^{mid}) \end{bmatrix} \mathcal{S}. \tag{41}$$

Note that $\texttt{rot}\big(\pi_h(\mathcal{R}_j)\big)$ is a zonotope because the 2-dimensional box $\mathcal{S}$ is equivalent to a 2-dimensional zonotope and it is multiplied by a matrix via (3). By applying geometry, one can verify that by definition $\mathcal{S}$ bounds the area that $\mathcal{O}^{\text{ego}} = \texttt{int}([-0.5L, -0.5W]^\top, [0.5L, 0.5W]^\top)$ travels through while rotating within the range $[-h^{\text{rad}}, h^{\text{rad}}]$. 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. 5.

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 (41) 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

$$\texttt{ROT}(\pi_h(\mathcal{R}_j)) := \left\langle \begin{bmatrix} c_{\texttt{rot}} \\ 0_{(7+n_p)\times 1} \end{bmatrix},\ \begin{bmatrix} G_{\texttt{rot}} \\ 0_{(7+n_p)\times 2} \end{bmatrix} \right\rangle. \tag{42}$$

Using this definition, one can extend the FRS to account for the vehicle footprint as in the following lemma whose proof can be found in Appendix B:

Lemma 21. *Let $\{\mathcal{R}_j\}_{j\in\mathcal{J}}$ be the set of zonotopes that over-approximates the ego vehicle's FRS under the augmented hybrid system HS beginning from $\mathcal{X}_0^{aug}$. Let x^{aug} be a solution to HS with initial velocity x_0^{vel} and control parameter p and let ξ be defined as*

$$\xi(\mathcal{R}_j, x_0^{vel}, p) = \pi_w\Big(\texttt{slice}\big(\mathcal{R}_j \oplus \texttt{ROT}(\pi_h(\mathcal{R}_j)), x_0^{vel}, p\big)\Big). \tag{43}$$

Then $\xi(\mathcal{R}_j, x_0^{vel}, p)$ is a zonotope and for all $j \in \mathcal{J}$ and $t \in T_j$, the vehicle footprint oriented and centered according to $x^{aug}(t)$ is contained within zonotope $\xi(\mathcal{R}_j, x_0^{vel}, p)$.

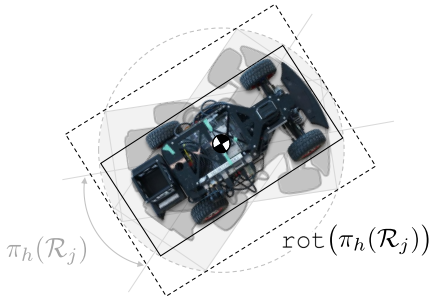


Fig. 5: Rotation of the ego vehicle and its footprint within range $\pi_h(\mathcal{R}_j)$. The ego vehicle with heading equals to the mean value of $\pi_h(\mathcal{R}_j)$ is bounded by the box with solid black boundaries. The range of rotated heading is indicated by the grey arc. The area the ego vehicle's footprint sweeps is colored in grey, and is bounded by box $\texttt{rot}\big(\pi_h(\mathcal{R}_j)\big)$ with dashed black boundaries.

VII. Online Planning

With the offline computed reachable sets as described in Section VI, REFINE solves an optimization problem online to construct not-at-fault behavior. This is done by enforcing that the intersection between any sliced zonotope reachable set and any obstacle is the empty set. This section begins by taking nonzero initial position conditions 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 the predictions of surrounding obstacles. Because our zonotope reachable sets describe vehicle behavior over time intervals, we assume that the predictions of the surrounding environment are constructed in a similar fashion. 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 22. *There exists a map $\vartheta : \mathcal{J} \times \mathcal{I} \to \mathbb{ZO}(\mathcal{W})$ such that $\vartheta(j, i)$ is a zonotope and*

$$\cup_{t\in T_j} \mathcal{O}_i(t) \cap \mathcal{B}\left((w_x(t_0), w_y(t_0)), S\right) \subseteq \vartheta(j, i), \tag{44}$$

where S is the sensing range, and $\mathcal{B}\left((w_x(t_0), w_y(t_0)), S\right)$ is the 2-dimensional closed ball with center $(w_x(t_0), w_y(t_0))$ and radius S under the Euclidean norm.

A. Nonzero Initial Position

Recall the FRS in Section VI is computed offline while assuming that the initial position of the ego vehicle is zero (i.e., Assumption 15). 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 $[w_{\mathrm{x},0}, w_{\mathrm{y},0}]^\top \in \mathcal{W}$ with its x-axis aligned according to the ego vehicle's heading $h_0 \in \mathbb{R}$, where $x_0^{\mathrm{pos}} = [w_{\mathrm{x},0}, w_{\mathrm{y},0}, h_0]^\top$ gives the ego vehicle's position $[w_{\mathrm{x}}(t), w_{\mathrm{y}}(t), h(t)]^\top$ at time $t = 0$ in the world frame. Similarly, $\xi(\mathcal{R}_j, x_0^{\mathrm{vel}}, p)$ 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, REFINE transfers the obstacle position $\vartheta(j,i)$ from the world frame to the local frame using a 2D rigid body transformation as

$$\vartheta^{\mathrm{loc}}(j,i,x_0^{\mathrm{pos}}) = \begin{bmatrix} \cos(h_0) & \sin(h_0) \\ -\sin(h_0) & \cos(h_0) \end{bmatrix} \left(\vartheta(j,i) - \begin{bmatrix} w_{\mathrm{x},0} \\ w_{\mathrm{y},0} \end{bmatrix} \right). \tag{45}$$

B. Online Optimization

Given the predicted initial condition of the vehicle at $t = 0$ as $x_0 = [(x_0^{\mathrm{pos}})^\top, (x_0^{\mathrm{vel}})^\top]^\top \in \mathbb{R}^3 \times \mathcal{X}_0^{\mathrm{vel}} \subset \mathbb{R}^6$, REFINE computes a not-at-fault trajectory by solving the following optimization problem at each planning iteration:

$$\begin{aligned} \min_{p\in\mathcal{P}} \quad & \texttt{cost}(z_0, p) && (\texttt{Opt}) \\ \text{s.t.} \quad & \xi(\mathcal{R}_j, x_0^{\mathrm{vel}}, p) \cap \vartheta^{\mathrm{loc}}(j,i,x_0^{\mathrm{pos}}) = \emptyset, && \forall j \in \mathcal{J}, \forall i \in \mathcal{I} \end{aligned}$$

where $\texttt{cost} : \mathbb{R}^6 \times \mathcal{P} \to \mathbb{R}$ is a user-specified cost function and ξ is defined as in Lemma 21. 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. Note the set intersection constraint in (Opt) is nonlinear as shown in the next subsection. In this work, we use the open-source interior point optimizer IPOPT [38] to solve (Opt). However, other nonlinear optimization solvers can also be used as alternatives.

C. 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 directional derivative with respect to $p \in \mathcal{P}$:

Theorem 23. *There exists matrices A and B and a vector b such that $\xi(\mathcal{R}_j, x_0^{vel}, p) \cap \vartheta^{loc}(j,i,x_0^{pos}) = \emptyset$ if and only if $\max(BA\cdot p - b) > 0$. In addition, the directional derivative of $\max(BA \cdot p - b)$ with respect to p along any vector $d \in \mathbb{R}^{n_p}$ is $\max_{k\in\hat{K}}([BA]_{k:} \cdot d)$ 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 (64), (66), and (67), respectively. Theorem 23 shows that each collision-free constraint in (Opt) is identical to an almost-linear inequality constraint whose feasible set may not be convex. One can also conclude from the proof of Theorem 23 that the evaluations of the set intersection constraint and its gradient have linear complexity with respect to the number of generators of $\mathcal{R}_j$.

Algorithm 1 REFINE Online Planning

Require: $p_0 \in \mathcal{P}$ and $x_0 = [(x_0^{\mathrm{pos}})^\top, (x_0^{\mathrm{vel}})^\top]^\top \in \mathbb{R}^3 \times \mathcal{X}_0^{\mathrm{vel}}$
1: **Initialize:** $p^* \leftarrow p_0$, $t \leftarrow 0$
2: **Loop:** *// Line 3 executes at the same time as Line 4-8*
3: **Execute** p^* during $[0, t_{\mathrm{m}})$
4: $\{\vartheta^{\mathrm{loc}}(j,i,x_0^{\mathrm{pos}})\}_{(j,i)\in\mathcal{J}\times\mathcal{I}} \leftarrow \texttt{SenseObstacles}()$
5: **Try** $p^* \leftarrow \texttt{OnlineOpt}(x_0, \{\vartheta^{\mathrm{loc}}(j,i,x_0^{\mathrm{pos}})\}_{(j,i)\in\mathcal{J}\times\mathcal{I}})$ *// within t_{plan} seconds*
6: **Catch** execute failsafe maneuver, then **break**
7: $(x_0^{\mathrm{pos}}, x_0^{\mathrm{vel}}) \leftarrow \texttt{StatePrediction}(x_0, p^*, t_{\mathrm{m}})$
8: $x_0 \leftarrow [(x_0^{\mathrm{pos}})^\top, (x_0^{\mathrm{vel}})^\top]^\top$
9: **If** $(x_0^{\mathrm{vel}} \notin \mathcal{X}_0^{\mathrm{vel}})$, execute failsafe maneuver and **break**
10: **Reset** t to 0
11: **End**

D. Online Operation

Algorithm 1 summarizes the online operations of REFINE. In each planning iteration, the ego vehicle executes the feasible trajectory parameter that is computed in the previous planning iteration (Line 3). Meanwhile, SenseObstacles senses and predicts obstacles as in Assumption 22 (Line 4) in local frame decided by x_0^{pos}. (Opt) is then solved to compute a trajectory parameter p^* using x_0 and $\{\vartheta^{\mathrm{loc}}(j,i,x_0^{\mathrm{pos}})\}_{(j,i)\in\mathcal{J}\times\mathcal{I}}$ (Line 5). If (Opt) fails to find a feasible solution within t_{plan} seconds, the contingency braking maneuver whose safety is verified in the last planning iteration is executed, and REFINE is terminated (Line 6). In the case when (Opt) is able to find a feasible p^*, StatePrediction predicts the state value at $t = t_{\mathrm{m}}$ based on x_0 and p^* as in Assumption 6 (Lines 7 and 8). If the predicted velocity value does not belong to $\mathcal{X}_0^{\mathrm{vel}}$, then its corresponding FRS is not available and the planning has to stop while executing a contingency braking maneuver (Line 9). Otherwise, we reset the time to 0 (Line 10) 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 8, Lemmas 14 and 21, Assumption 22 and (45), the following theorem holds:

Theorem 24. *Suppose the ego vehicle can sense and predict surrounding obstacles as in Assumption 22, and starts with a not-at-fault trajectory 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.*

VIII. 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[4] and videos[5] online.

A. Desired Trajectories

As detailed in Section V-A, the proposed controller relies on desired trajectories of vehicle longitudinal speed and yaw rate satisfying Definition 7. 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 a *speed change*, a *direction change*, or a *lane change*. Moreover, each desired trajectory is parameterized by $p = [p_{v_\mathrm{x}}, p_y]^\top \in \mathcal{P} \subset \mathbb{R}^2$ where p_{v_x} denotes desired longitudinal speed, and p_y decides desired lateral displacement.

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

$$v_\mathrm{x}^{\mathrm{des}}(t,p) = \begin{cases} v_{\mathrm{x},0} + \frac{p_{v_\mathrm{x}} - v_{\mathrm{x},0}}{t_\mathrm{m}} t, & \text{if } 0 < t < t_\mathrm{m} \\ v_\mathrm{x}^{\mathrm{brake}}(t,p), & \text{if } t \geq t_\mathrm{m} \end{cases} \tag{46}$$

where

$$v_\mathrm{x}^{\mathrm{brake}}(t,p) = \begin{cases} p_{v_\mathrm{x}} + (t - t_\mathrm{m}) a^{\mathrm{dec}}, \\ \quad \text{if } p_{v_\mathrm{x}} > v_\mathrm{x}^{\mathrm{cri}} \text{ and } t_\mathrm{m} \leq t < t_\mathrm{m} + \frac{v_\mathrm{x}^{\mathrm{cri}} - p_{v_\mathrm{x}}}{a^{\mathrm{dec}}} \\ 0, \ \text{if } p_{v_\mathrm{x}} > v_\mathrm{x}^{\mathrm{cri}} \text{ and } t \geq t_\mathrm{m} + \frac{v_\mathrm{x}^{\mathrm{cri}} - p_{v_\mathrm{x}}}{a^{\mathrm{dec}}} \\ 0, \ \text{if } p_{v_\mathrm{x}} \leq v_\mathrm{x}^{\mathrm{cri}} \text{ and } t \geq t_\mathrm{m} \end{cases} \tag{47}$$

with some deceleration $a^{\mathrm{dec}} < 0$. Note by Definition 7, t_{stop} can be specified as

$$t_{\mathrm{stop}} = \begin{cases} t_\mathrm{m} + \frac{v_\mathrm{x}^{\mathrm{cri}} - p_{v_\mathrm{x}}}{a^{\mathrm{dec}}}, & \text{if } p_{v_\mathrm{x}} > v_\mathrm{x}^{\mathrm{cri}} \\ t_\mathrm{m}, & \text{if } p_{v_\mathrm{x}} \leq v_\mathrm{x}^{\mathrm{cri}}. \end{cases} \tag{48}$$

The desired longitudinal speed approaches p_{v_x} linearly from $v_{\mathrm{x},0}$ before braking begins at time t_m, then decreases to $v_\mathrm{x}^{\mathrm{cri}}$ with deceleration a^{dec} and immediately drops down to 0 at time t_{stop}. Moreover, one can verify that the chosen $v_\mathrm{x}^{\mathrm{des}}(t,p)$ in (46) satisfies the assumptions on desired longitudinal speed in Lemma 14.

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 family associated with speed change:

$$h^{\mathrm{des}}(t,p) = h_0, \ \forall t \geq 0. \tag{49}$$

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

$$h^{\mathrm{des}}(t,p) = \begin{cases} h_0 + \frac{p_y t}{2} - \frac{p_y t_\mathrm{m}}{4\pi} \sin\left(\frac{2\pi t}{t_\mathrm{m}}\right), & \text{if } 0 \leq t < t_\mathrm{m} \\ h_0 + \frac{p_y t_\mathrm{m}}{2}, & \text{if } t \geq t_\mathrm{m} \end{cases} \tag{50}$$

and for the trajectory family associated with lane change:

$$h^{\mathrm{des}}(t,p) = \begin{cases} h_0 + h_1^{\mathrm{des}} p_y \cdot \mathrm{e}^{-h_2^{\mathrm{des}}(t - 0.5 t_\mathrm{m})^2}, \\ \qquad\qquad\qquad \text{if } 0 \leq t < t_\mathrm{m} \\ h_0, \qquad\qquad\qquad \text{if } t \geq t_\mathrm{m} \end{cases} \tag{51}$$

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. Illustrations of speed change and direction change maneuvers can be found in the software repository[6], and example lane change maneuvers are illustrated in Figure 6. By Definition 7, desired trajectory of yaw rate is set as $r^{\mathrm{des}}(t,p) = \frac{d}{dt} h^{\mathrm{des}}(t,p)$ among all trajectory families.

In this work, t_m for the speed change and direction change trajectory families are set 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.

B. Subdivision of Initial Set and Families of Trajectories

In practice, CORA may generate overly conservative representations for the FRS if the initial condition set is large. To address this challenge, we partition $\mathcal{X}_0$ and $\mathcal{P}$, and we over-approximate FRS beginning from each element in this partition. Note we could still apply REFINE as in Algorithm 1. However in Line 5, multiple optimizations of the form (`Opt`) must be solved in parallel. Each of these optimization problems optimizes over a unique partition element that contains initial condition x_0. p^* is set to be the feasible trajectory parameter that achieves the minimum cost function value among these optimizations. Similarly, because we have multiple families of desired trajectories that are each parameterized in distinct ways as described in Section VIII-A, we extend REFINE just as in the instance of having a partition of the initial condition set. In this way REFINE can be applied to optimize over multiple families of desired trajectories to generate not-at-fault behavior. Note that the planning horizon t_f can vary between different elements in the partition.

C. Simulation on an FWD Model

This subsection describes the evaluation on REFINE using a highway-like simulation. In particular, this subsection describes in order the simulation environment and task, REFINE implementation details, other benchmarked methods, and the evaluation results.

1) Simulation Environment and Task: We evaluate the performance on 1000 randomly generated 3-lane highway scenarios in which the same full-size, FWD vehicle is expected to autonomously navigate through dynamic traffic for 1[km] from a fixed initial condition. All lanes of all highway scenarios

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

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

[6]https://github.com/roahmlab/REFINE/blob/main/Rover_Robot_Implementation/README.md#1-desired-trajectories

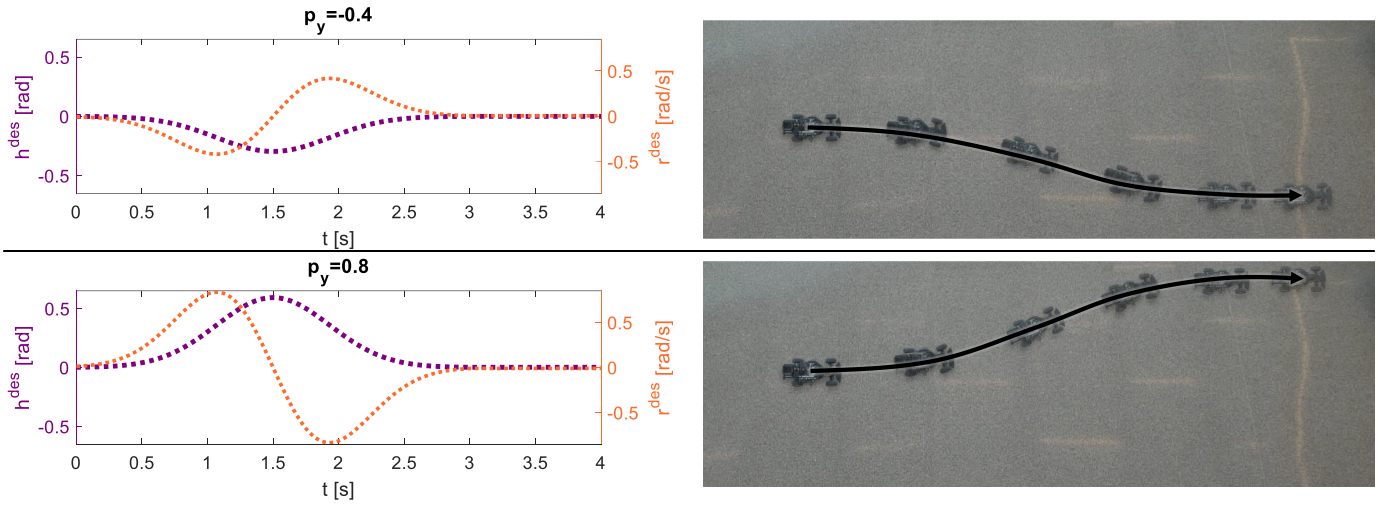


Fig. 6: Examples of $h^{\text{des}}(t,p)$ and $r^{\text{des}}(t,p)$ to achieve lane changes with $v_{\text{x},0} = 1.0$ [m/s], $t_{\text{m}} = 3.0$ [s], $h_1^{\text{des}} = \frac{20}{27}$, $h_2^{\text{des}} = \frac{27}{10}$, and p_y taking values of -0.4 and 0.8 from top to bottom. Note $p_{v_{\text{x}}}$ is set as $v_{\text{x},0}$ to maintain the vehicle longitudinal speed before t_{m} among both examples.

share the same lane width as 3.7[m]. Each highway scenario contains up to 24 moving vehicles and up to 5 static obstacles. They are all generated from random locations and treated as obstacles to the planning method. Moreover, each moving vehicle maintains its lane and initial speed up to 25[m/s] for all time. Because of the randomness, there is no guarantee that the ego vehicle has a path to navigate to the goal. Such cases allow us to verify whether the tested methods can keep the ego vehicle safe even in infeasible scenarios. Parameters of the ego vehicle can be found in the software implementation README[7].

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 x_0 and decision variable p. All simulations are implemented and evaluated in MATLAB R2022a and C++17 on a laptop with an Intel i7-9750H processor and 16GB of RAM.

2) REFINE Simulation Implementation: Parameters of REFINE's controller are chosen to satisfy the conditions in Lemma 14 and can be found in the software implementation README[8]. REFINE tracks families of desired trajectories as described in Section VIII-A with $\mathcal{P} = \{(p_{v_{\text{x}}}, p_y) \in [5,30] \times [-0.8, 0.8] \mid p_{v_{\text{x}}} = v_{\text{x},0} \text{ if } p_y \neq 0\}$, $a^{\text{dec}} = -5.0[\text{m/s}^2]$, $h_1^{\text{des}} = \frac{6\sqrt{2\text{e}}}{11}$ 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 VIII-B, during offline computation, we evenly partition the first and second dimensions of $\mathcal{P}$ into intervals based on the initial condition of longitudinal speed. For each partition element, t_{f} is assigned to be the maximum possible value of t_{brake} as computed in (59) in which t_{fstop} is by observation no greater than 0.1[s]. An over-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) Other Implemented Methods: We compare REFINE against several state-of-the-art trajectory planning methods: a baseline zonotope reachable set method, a Sum-of-Squares-based RTD (SOS-RTD) method, and an NMPC method.

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 an extension of the methods described in [10], [13]. Instead of planning over a continuous trajectory parameter space, the baseline method selects from among a discrete trajectory parameter space made up of a finite number of trajectory parameters. In particular, the baseline method computes a collection of zonotope reachable sets for each element in the discrete trajectory parameter space offline, and during online operation searches through the discrete trajectory parameter space until a feasible solution is found such that the corresponding zonotope reachable sets are verified to have no intersection with any obstacles over the planning horizon. The baseline method computes zonotope reachable sets using CORA with $\Delta t = 0.010$[s] over a sparse discrete trajectory parameter space $\mathcal{P}^{\text{sparse}} := \{(p_{v_{\text{x}}}, p_y) \in \{5, 5.1, 5.2, \ldots, 30\} \times \{0, 0.4\} \mid p_{v_{\text{x}}} = v_{\text{x},0} \text{ if } p_y \neq 0\}$ and a dense discrete trajectory parameter space $\mathcal{P}^{\text{dense}} := \{(p_{v_{\text{x}}}, p_y) \in \{5, 5.1, 5.2, \ldots, 30\} \times \{0, 0.04, 0.08, \ldots, 0.8\} \mid p_{v_{\text{x}}} = v_{\text{x},0} \text{ if } p_y \neq 0\}$. 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. Note during online planning, the search procedure over the provided discrete control space

[7]https://github.com/roahmlab/REFINE/blob/main/Full_Size_Vehicle_Simulation/README.md#vehicle-and-control-parameters

[8]https://github.com/roahmlab/REFINE/blob/main/Full_Size_Vehicle_Simulation/README.md#vehicle-and-control-parameters

is biased to select the same trajectory parameter that worked in the prior planning iteration or to search first from trajectory parameters that are close to one that worked in the previous planning iteration.

The SOS-RTD method [14] plans a controller that also tracks the same families of parametrized trajectories as REFINE uses to achieve speed change, direction change and lane change maneuvers with braking maneuvers as described in Section VIII-A. 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] [14]. 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 [17]. 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 3 seconds by solving an optimal control problem. This optimal control problem is solved using GPOPS-II [39] in a receding horizon fashion with a replanning rate of 1/3[Hz]. Note that we gave this NMPC method 1000 IPOPT iterations to find a solution and did not require that it generate a solution within 3 seconds. The NMPC method conservatively ensures collision-free trajectories by covering the footprints of the ego vehicle and all obstacles with two partially overlapping balls, and requiring that no ball of the ego vehicle intersects with any ball of any obstacle at discrete time instances over the planning horizon. 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 the planning horizon as the other three methods do.

4) Evaluation Criteria: We evaluate each implemented trajectory planning method in several ways as summarized in Table I. First, we report the ratio 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.

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. For the same reason, 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 a high enough success 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 success 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. Both baseline methods($\mathcal{P}^{\text{sparse}}$ and $\mathcal{P}^{\text{dense}}$) are unable 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 2.4 GB to store its polynomial reachable sets. Its success rate is only 64% because the polynomial reachable sets are more conservative than the zonotope reachable sets. In particular, we observe that zonotope reachable sets generated by REFINE are on average $9.46\%, 19.69\%$, and 20.84% tighter than polynomial reachable sets generated by SOS-RTD for speed change, direction change, and lane change maneuvers, respectively. We computed this numbers by evaluating the area of the sliced FRS at 100 random desired trajectory parameters for each of the three families.

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. NMPC method achieves a higher average travel speed of the ego vehicle when compared to the other three methods. More aggressive controls can allow the ego vehicle to drive closer to the obstacles at a higher speed, but can make subsequent obstacle avoidance difficult. The NMPC method uses 40.89[s] on average to compute a solution, which makes real-time path planning untenable. Finally, note that despite searching over a larger space of control inputs, NMPC has a lower success rate than REFINE.

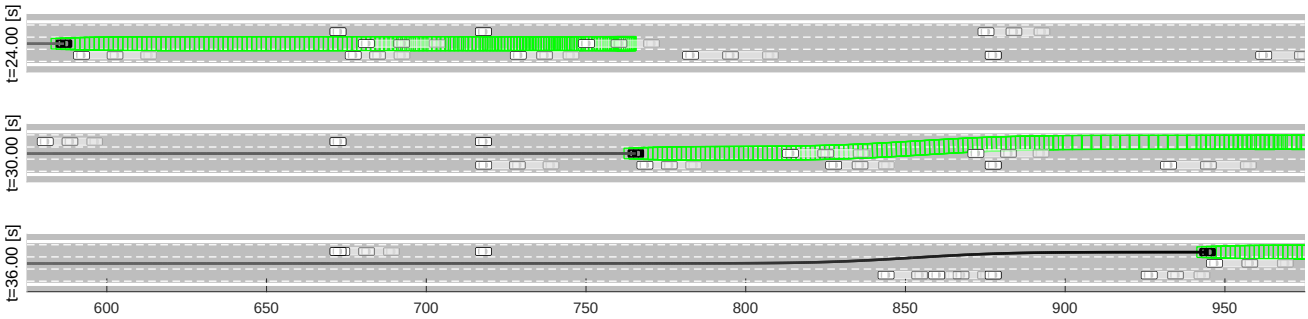
Figure 7 illustrates the performance of three methods in the same scene at three different time instances. In Figure 7a, because REFINE gives a tight approximation of the ego vehicle's FRS using zonotopes, the ego vehicle is able to first bypass static vehicles in the top lane from $t = 24$[s] to $t = 30$[s], then switch to the top lane and bypass vehicles in the middle lane from $t = 30$[s] to $t = 36$[s]. In Figure 7b 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 planning 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 7c because NMPC is used for planning, the ego vehicle drives at a faster speed and arrives at 600[m] before the other two 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].

D. Real World Experiments

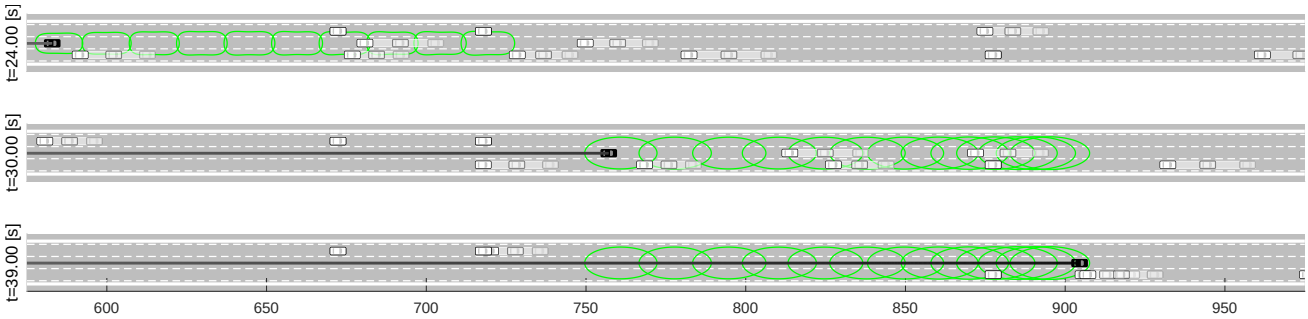
REFINE was also implemented in C++17 and tested in the real world using a $\frac{1}{10}$th All-Wheel-Drive car-like robot, Rover, based on a Traxxas RC platform. The Rover is equipped

Method	Safely Stop	Crash	Success	Average Travel Speed	Solving Time of Online Planning (Average, Maximum)	Memory
Baseline (sparse, $\Delta t = 0.010$)	38%	**0%**	62%	22.3572[m/s]	(2.03[s], 4.15[s])	980 MB
Baseline (dense, $\Delta t = 0.010$)	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 ($\Delta t = 0.015$)	27%	**0%**	73%	23.2452[m/s]	(0.34[s], **0.95[s]**)	**488 MB**
REFINE ($\Delta t = 0.010$)	17%	**0%**	82%	24.8311[m/s]	(0.52[s], 1.57[s])	703 MB
REFINE ($\Delta t = 0.005$)	16%	**0%**	**84%**	24.8761[m/s]	(1.28[s], 4.35[s])	997 MB
REFINE ($\Delta t = 0.001$)	16%	**0%**	**84%**	24.8953[m/s]	(6.48[s], 10.78[s])	6.4 GB

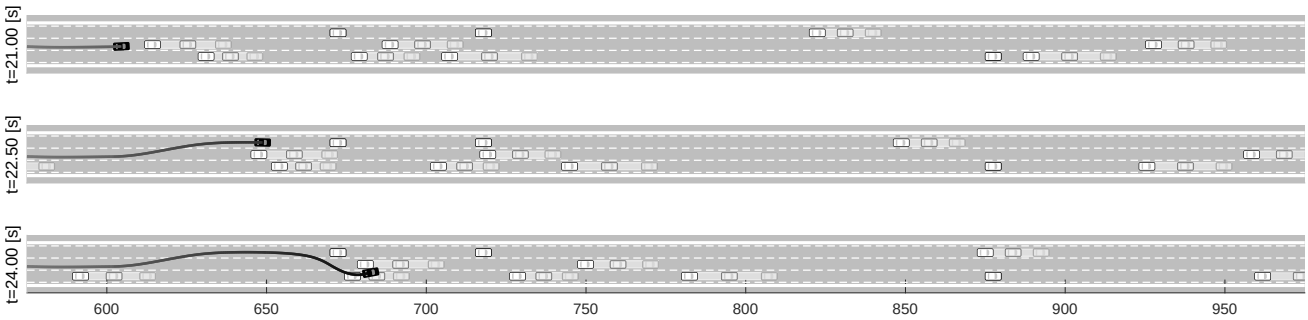
TABLE I: Summary of performance of various tested techniques on the same 1000 simulation environments.



(a) REFINE utilized.



(b) SOS-RTD utilized.



(c) NMPC utilized.

Fig. 7: 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.

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 onboard computer. A standby laptop with an Intel i7-9750H processor and 32GB of RAM is used for localization, mapping, and trajectory planning. The rover and the standby laptop communicate over wifi using ROS [40].

The desired trajectories on the Rover are parameterized with $\mathcal{P} = \{(p_{v_x}, p_y) \in [0.05, 2.05] \times [-1.396, 1.396] \mid p_{v_x} = v_{x,0} \text{ if } p_y \neq 0\}$, $a^{\text{dec}} = -1.5[\text{m/sec}^2]$, $h_1^{\text{des}} = \frac{20}{27}$ and $h_2^{\text{des}} = \frac{27}{10}$ as described in Section VIII-A. 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 based

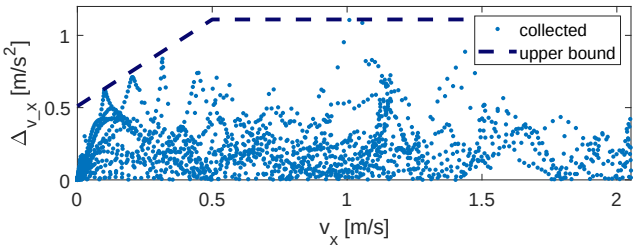


Fig. 8: An illustration of the modeling error along the dynamics of v_{x}. Collected $\Delta_{v_{\mathrm{x}}}(t)$ is bounded by $M_{v_{\mathrm{x}}} = 1.11$ for all time. Whenever $v_{\mathrm{x}}(t) \leq v_{\mathrm{x}}^{\mathrm{cri}} = 0.5$, $\Delta_{v_{\mathrm{x}}}(t)$ is bounded by $b_{v_{\mathrm{x}}}^{\mathrm{pro}} v_{\mathrm{x}}(t) + b_{v_{\mathrm{x}}}^{\mathrm{off}}$ with $b_{v_{\mathrm{x}}}^{\mathrm{pro}} = 1.2$ and $b_{v_{\mathrm{x}}}^{\mathrm{off}} = 0.51$.

on the initial condition of longitudinal speed. For each partition element, t_{f} is set equal to the maximum possible value of t_{brake} as computed in (59) 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 [41], and the cost function of (`Opt`) is set in the same way as we do in simulation.

The robot model, environment sensing, and state estimation play key roles in real-world experiments. In the rest of this subsection, we describe how to bound the modeling error in (9) and summarize the real-world experiments. Details regarding Rover model parameters, the controller parameters, how the Rover performs localization, mapping, and obstacle detection and how we perform system identification of the tire models can be found in our software implementation README[9].

1) State Estimation and System Identification on Modeling Error in Vehicle Dynamics: The modeling errors in the dynamics (9) 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 $\Delta_{v_{\mathrm{x}}}$, $\Delta_{v_{\mathrm{y}}}$, 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 (9) in which modeling errors are treated as zeros and tire forces are calculated via (7) and (8). The estimation of applied accelerations is computed using the estimated system states via an Unscented Kalman Filter (UKF) [42], 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 high-speed dynamics (9) with zero modeling errors. Note the UKF state estimator is still applicable in the low-speed case except the estimation of v and r are ignored. To ensure $\Delta_{v_{\mathrm{x}}}$, $\Delta_{v_{\mathrm{y}}}$ and Δ_r are square integrable, we set $\Delta_{v_{\mathrm{x}}}(t) = \Delta_{v_{\mathrm{y}}}(t) = \Delta_r(t) = 0$ for all $t \geq t_{\mathrm{brake}}$ where t_{brake} is computed in Lemma 14. As shown in Figure 8, bounding parameters $M_{v_{\mathrm{x}}}$, $M_{v_{\mathrm{y}}}$, and M_r are selected to be the maximum value of $|\Delta_{v_{\mathrm{x}}}(t)|$, $|\Delta_{v_{\mathrm{y}}}(t)|$, and $|\Delta_r(t)|$ respectively over all time, and $b_{v_{\mathrm{x}}}^{\mathrm{pro}}$ and $b_{v_{\mathrm{x}}}^{\mathrm{off}}$ are generated by bounding $|\Delta_{v_{\mathrm{x}}}(t)|$ from above when $v_{\mathrm{x}}(t) \leq v_{\mathrm{x}}^{\mathrm{cri}}$.

2) Demonstration: The Rover was tested indoors under the proposed controller and planning framework in 6 small trials and 1 loop trial[10]. In every small trail, up to 11 identical $0.3 \times 0.3 \times 0.3$[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 9 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 stopped 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 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 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 $t_{\mathrm{plan}} = 1.5$[s], thus real-time performance is achieved.

IX. Applicability and Limitations

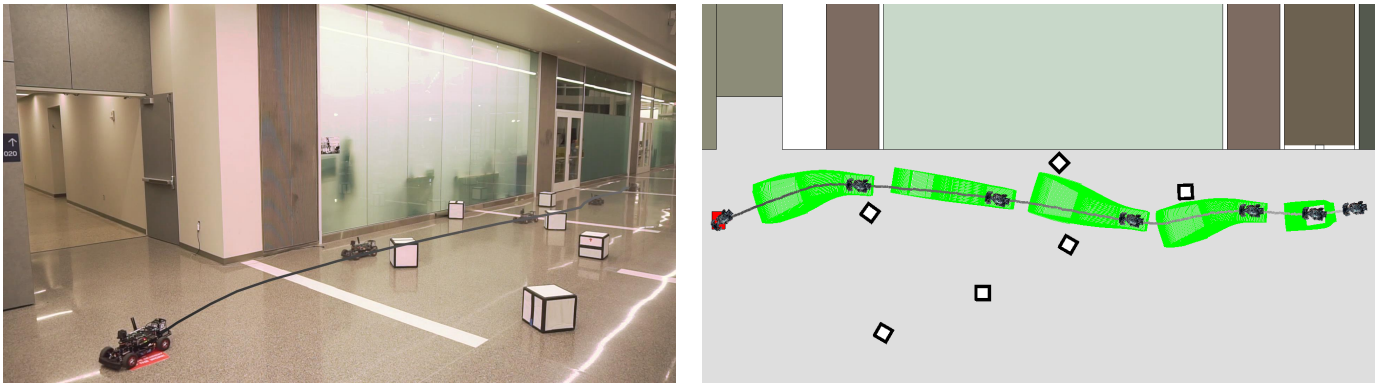
REFINE is applicable to FWD, RWD, and AWD vehicle models. Note that REFINE may not work well for desired trajectories that contains large jumps or oscillations because this may result in zonotope reachable sets that are too large to be useful. In addition, REFINE relies on deterministic information about the environment. In the case when positions and predictions of surrounding obstacles are described in a probabilistic fashion, the planning behavior of REFINE could be conservative because it would generate trajectory plans that avoid all possible locations of all surrounding obstacles even those with low probability.

X. Conclusion

This work presents REFINE, a trajectory design framework using zonotope reachable sets. A robust controller is designed to partially linearize the full-order vehicle dynamics with modeling error. 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. During online planning, this FRS is "sliced" to identify a trajectory that can be followed in a collision-free fashion. Tests on a full-size vehicle model in simulation and a 1/10th scale race car 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 safe planning methods.

[9]https://github.com/roahmlab/REFINE/blob/main/Rover_Robot_Implementation/README.md

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



(a) (b)

Fig. 9: 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.

REFERENCES

[1] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *The international journal of robotics research*, vol. 20, no. 5, pp. 378–400, 2001.

[2] L. Janson, E. Schmerling, A. Clark, and M. Pavone, "Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions," *The International journal of robotics research*, vol. 34, no. 7, pp. 883–921, 2015.

[3] M. Elbanhawi and M. Simic, "Sampling-based robot motion planning: A review," *Ieee access*, vol. 2, pp. 56–77, 2014.

[4] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[5] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How, "Real-time motion planning with applications to autonomous urban driving," *IEEE Transactions on control systems technology*, vol. 17, no. 5, pp. 1105–1118, 2009.

[6] T. M. Howard and A. Kelly, "Optimal rough terrain trajectory generation for wheeled mobile robots," *The International Journal of Robotics Research*, vol. 26, no. 2, pp. 141–166, 2007.

[7] P. Falcone, F. Borrelli, J. Asgari, H. Tseng, and D. Hrovat, "Low complexity mpc schemes for integrated vehicle dynamics control problems," in *9th international symposium on advanced vehicle control (AVEC)*, 2008.

[8] C. Urmson, J. Anhalt, D. Bagnell, *et al.*, "Autonomous driving in urban environments: Boss and the urban challenge," *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[9] J. Wurts, J. L. Stein, and T. Ersal, "Collision imminent steering using nonlinear model predictive control," in *2018 Annual American Control Conference (ACC)*, IEEE, 2018, pp. 4772–4777.

[10] M. Althoff and J. M. Dolan, "Online verification of automated road vehicles using reachability analysis," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 903–918, 2014.

[11] C. Pek, S. Manzinger, M. Koschi, and M. Althoff, "Using online verification to prevent autonomous vehicles from causing accidents," *Nature Machine Intelligence*, vol. 2, no. 9, pp. 518–528, 2020.

[12] S. Manzinger, C. Pek, and M. Althoff, "Using reachable sets for trajectory planning of automated vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 2, pp. 232–248, 2020.

[13] A. Majumdar and R. Tedrake, "Funnel libraries for real-time robust feedback motion planning," *The International Journal of Robotics Research*, vol. 36, no. 8, pp. 947–982, 2017.

[14] S. Kousik, S. Vaskov, F. Bu, M. Johnson-Roberson, and R. Vasudevan, "Bridging the gap between safety and real-time performance in receding-horizon trajectory design for mobile robots," *The International Journal of Robotics Research*, vol. 39, no. 12, pp. 1419–1469, 2020.

[15] S. Kousik, S. Vaskov, M. Johnson-Roberson, and R. Vasudevan, "Safe trajectory synthesis for autonomous driving in unforeseen environments," in *ASME 2017 Dynamic Systems and Control Conference*, American Society of Mechanical Engineers Digital Collection, 2017.

[16] S. Vaskov, H. Larson, S. Kousik, M. Johnson-Roberson, and R. Vasudevan, "Not-at-fault driving in traffic: A reachability-based approach," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, pp. 2785–2790.

[17] S. Vaskov, S. Kousik, H. Larson, *et al.*, "Towards provably not-at-fault control of autonomous robots in arbitrary dynamic environments," *arXiv preprint arXiv:1902.02851*, 2019.

[18] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, "Hamilton-jacobi reachability: A brief overview and recent advances," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, IEEE, 2017, pp. 2242–2253.

[19] S. L. Herbert, M. Chen, S. Han, S. Bansal, J. F. Fisac, and C. J. Tomlin, "Fastrack: A modular framework for fast and guaranteed safe motion planning," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, IEEE, 2017, pp. 1517–1522.

[20] J. Lunze and F. Lamnabhi-Lagarrigue, *Handbook of hybrid systems control: theory, tools, applications*. Cambridge University Press, 2009.

[21] A. G. Ulsoy, H. Peng, and M. Çakmakci, *Automotive control systems*. Cambridge University Press, 2012.

[22] R. N. Jazar, *Vehicle dynamics: theory and application*. Springer, 2008. [Online]. Available: https://link.springer.com/chapter/10.1007/978-0-387-74244-1_2.

[23] T.-Y. Kim, S. Jung, and W.-S. 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*, vol. 233, no. 11, pp. 2903–2911, 2019.

[24] H. Roehm, A. Rausch, and M. Althoff, "Reachset conformance and automatic model adaptation for hybrid systems," *Mathematics*, vol. 10, no. 19, p. 3567, 2022.

[25] E. Kutluay and H. Winner, "Assessment methodology for validation of vehicle dynamics simulations using double lane change maneuver," in *Proceedings of the 2012 Winter Simulation Conference (WSC)*, IEEE, 2012, pp. 1–12.

[26] J. D. Setiawan, M. Safarudin, and A. Singh, "Modeling, simulation and validation of 14 dof full vehicle model," in *International Conference on Instrumentation, Communication, Information Technology, and Biomedical Engineering 2009*, IEEE, 2009, pp. 1–6.

[27] T. D. Gillespie, "Fundamentals of vehicle dynamics," SAE Technical Paper, Tech. Rep., 1992.

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

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

[30] R. Remmert, *Theory of complex functions*. Springer Science & Business Media, 1991, vol. 122.

[31] M.-Y. Yu, R. Vasudevan, and M. Johnson-Roberson, "Occlusion-aware risk assessment for autonomous driving in urban environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 2235–2241, 2019.

[32] M.-Y. Yu, R. Vasudevan, and M. Johnson-Roberson, "Risk assessment and planning with bidirectional reachability for autonomous driving," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 5363–5369.

[33] M. Koschi and M. Althoff, "Set-based prediction of traffic participants considering occlusions and traffic rules," *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 2, pp. 249–265, 2020.

[34] A. Giusti and M. Althoff, "Ultimate robust performance control of rigid robot manipulators using interval arithmetic," in *2016 American Control Conference (ACC)*, IEEE, 2016, pp. 2995–3001.

[35] M. Althoff, "An introduction to cora 2015," in *Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems*, 2015.

[36] M. Althoff, "Reachability analysis and its application to the safety assessment of autonomous cars," Ph.D. dissertation, Technische Universität München, 2010.

[37] P. Holmes, S. Kousik, B. Zhang, *et al.*, "Reachable sets for safe, real-time manipulator trajectory design (version 1)," *arXiv preprint arXiv:2002.01591*, 2020, https://arxiv.org/abs/2002.01591v1.

[38] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[39] M. A. Patterson and A. 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)*, vol. 41, no. 1, pp. 1–37, 2014.

[40] Stanford Artificial Intelligence Laboratory et al., *Robotic operating system*, version ROS Melodic Morenia, May 23, 2018. [Online]. Available: https://www.ros.org.

[41] E. W. Dijkstra *et al.*, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[42] E. 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)*, 2000, pp. 153–158. DOI: 10.1109/ASSPCC.2000.882463.

[43] L. J. Guibas, A. T. Nguyen, and L. Zhang, "Zonotopes as bounding volumes.," in *SODA*, vol. 3, 2003, pp. 803–812.

[44] E. Polak, *Optimization: algorithms and consistent approximations*. Springer Science & Business Media, 2012, vol. 124.

APPENDIX A
PROOF OF LEMMA 14

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

Note by (18) and rearranging (19),

$$\dot{e}_{v_{\text{x}}}(t) = -K_{v_{\text{x}}} e_{v_{\text{x}}}(t) + \tau_{v_{\text{x}}}(t) + \Delta_{v_{\text{x}}}(t). \tag{52}$$

By Definition 7 $v_{\text{x}}^{\text{des}}$ is piecewise continuously differentiable, so are $e_{v_{\text{x}}}$ and $\tau_{v_{\text{x}}}$. Let $\{t_1, t_2, \ldots, t_{k_{\text{max}}}\}$ denote a finite subdivision of $[0, t_{\text{stop}})$ with $t_1 = 0$ and $t_{k_{\text{max}}} = t_{\text{stop}}$ such that $v_{\text{x}}^{\text{des}}$ is continuously differentiable over time interval $[t_k, t_{k+1})$ for all $k \in \{1, 2, \ldots, k_{\text{max}} - 1\}$. Define $V(t) := \frac{1}{2} e_{v_{\text{x}}}^2(t)$ as a candidate Lyapunov Function for $e_{v_{\text{x}}}(t)$, then for arbitrary $k \in \{1, 2, \ldots, k_{\text{max}} - 1\}$ and $t \in [t_k, t_{k+1})$, one can check that $V(t)$ is always non-negative and $V(t) = 0$ only if $e_{v_{\text{x}}}(t) = 0$. Then from (52) and (15), $\dot{V}(t) = -K_{v_{\text{x}}} e_{v_{\text{x}}}^2(t) - (\kappa_{v_{\text{x}}}(t) M_{v_{\text{x}}} + \phi_{v_{\text{x}}}(t)) e_{v_{\text{x}}}^2(t) + e_{v_{\text{x}}}(t) \Delta_{v_{\text{x}}}(t)$. Because the integral terms in (16) and (17) are both non-negative, $\kappa_{v_{\text{x}}}(t) \geq \kappa_{1,v_{\text{x}}}$ and $\phi_{v_{\text{x}}}(t) \geq \phi_{1,v_{\text{x}}}$ hold. Thus

$$\begin{aligned} \dot{V}(t) \leq -K_{v_{\text{x}}} e_{v_{\text{x}}}^2(t) - (\kappa_{1,v_{\text{x}}} M_{v_{\text{x}}} + \phi_{1,u}) |e_{v_{\text{x}}}(t)|^2 + \\ + |e_{v_{\text{x}}}(t)| |\Delta_{v_{\text{x}}}(t)|. \end{aligned} \tag{53}$$

By factoring out $|e_{v_{\text{x}}}(t)|$ in the last two terms in (53):

$$\dot{V}(t) \leq -K_{v_{\text{x}}} e_{v_{\text{x}}}^2(t) < 0 \tag{54}$$

holds when $|e_{v_{\text{x}}}(t)| > 0$ and $|e_{v_{\text{x}}}(t)| \geq \frac{|\Delta_{v_{\text{x}}}(t)|}{\kappa_{1,v_{\text{x}}} M_{v_{\text{x}}} + \phi_{1,v_{\text{x}}}}$. Note $|e_{v_{\text{x}}}(t)| \geq v_{\text{x}}^{\text{small}}$ conservatively implies $|e_{v_{\text{x}}}(t)| \geq \frac{|\Delta_{v_{\text{x}}}(t)|}{\kappa_{1,v_{\text{x}}} M_{v_{\text{x}}} + \phi_{1,v_{\text{x}}}}$ given $|\Delta_{v_{\text{x}}}(t)| \leq M_{v_{\text{x}}}$ for all time by Assumption 4. Then when $|e_{v_{\text{x}}}(t)| \geq v_{\text{x}}^{\text{small}} > 0$ we have (54) hold, or equivalently $V(t)$ decreases. Therefore if $|e_{v_{\text{x}}}(t_k)| \geq v_{\text{x}}^{\text{small}}$, $|e_{v_{\text{x}}}(t)|$ monotonically decreases during time interval $[t_k, t_{k+1})$ as long as $|e_{v_{\text{x}}}(t)|$ does not reach at the boundary of closed ball $\mathcal{B}(0, v_{\text{x}}^{\text{small}})$. Moreover, if $|e_{v_{\text{x}}}(t')|$ hits the boundary of $\mathcal{B}(0, v_{\text{x}}^{\text{small}})$ at some time $t' \in [t_k, t_{k+1})$, $e_{v_{\text{x}}}(t)$ is prohibited from leaving the ball for all $t \in [t', t_{k+1})$ because $\dot{V}(t)$ is strictly negative when $|e_{v_{\text{x}}}(t)| = v_{\text{x}}^{\text{small}}$. Similarly $|e_{v_{\text{x}}}(t_k)| \leq v_{\text{x}}^{\text{small}}$ implies $|e_{v_{\text{x}}}(t)| \leq v_{\text{x}}^{\text{small}}$ for all $t \in [t_k, t_{k+1})$.

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

Next we analyze how longitudinal speed of the ego vehicle evolves after time t_{stop}. Using $V(t) = \frac{1}{2} e_{v_{\text{x}}}^2(t)$, note (53) remains valid for all $t \geq t_{\text{stop}}$, and (54) also holds when $|e_{v_{\text{x}}}(t)| \geq v_{\text{x}}^{\text{small}}$ with $t \geq t_{\text{stop}}$. Recall $v_{\text{x}}(t) = e_{v_{\text{x}}}(t)$ for all $t \geq t_{\text{stop}}$ given $v_{\text{x}}^{\text{des}}(t) = 0$ for all $t \geq t_{\text{stop}}$, then for simplicity, the remainder of this proof replaces every $e_{v_{\text{x}}}(t)$ by $v_{\text{x}}(t)$ in (53), (54) and $V(t)$. Because $v_{\text{x}}(0) > 0$ and v_{x} 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 $v_{\text{x}}(t) = 0$, by Assumption 5 $\Delta_{v_{\text{x}}}(t) = 0$, thus $\dot{v}_{\text{x}}(t) = 0$ by (19) given $v_{\text{x}}^{\text{des}}(t) = 0$ for all $t \geq t_{\text{stop}}$.

From now on we assume $t \geq t_{\text{stop}}$ and $v_{\text{x}}(t) \geq 0$ for all $t \geq t_{\text{stop}}$. Recall $v_{\text{x}}(t_{\text{stop}}) \in [v_{\text{x}}^{\text{cri}} - v_{\text{x}}^{\text{small}}, v_{\text{x}}^{\text{cri}} + v_{\text{x}}^{\text{small}}]$ and $v_{\text{x}}^{\text{cri}} - v_{\text{x}}^{\text{small}} \in [0, v_{\text{x}}^{\text{cri}} - 0.15)$. 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 $v_{\text{x}}(t_{\text{stop}}) \leq 0.15$**:** Because the longitudinal speed stays at 0 once it becomes 0, by Assumption 13 the ego vehicle reaches a full stop no later than $t_{\text{fstop}} + t_{\text{stop}}$.

Case 2 - When $0.15 < v_{\text{x}}(t_{\text{stop}}) \leq v_{\text{x}}^{\text{small}}$**:** By Assumption 5, upper bound of $\dot{V}(t)$ can be further relaxed from (53) to $\dot{V}(t) \leq -K_{v_{\text{x}}} {v_{\text{x}}}^2(t) - (\kappa_{1,v_{\text{x}}} M_{v_{\text{x}}} + \phi_{1,v_{\text{x}}} - b_{v_{\text{x}}}^{\text{pro}}) {v_{\text{x}}}^2(t) + b_{v_{\text{x}}}^{\text{off}} v_{\text{x}}(t)$. Moreover, by completing the square among the last two terms in this equation, one can derive

$$\dot{V}(t) \leq -K_{v_{\text{x}}} {v_{\text{x}}}^2(t) + \frac{(b_{v_{\text{x}}}^{\text{off}})^2}{4(\kappa_{1,v_{\text{x}}} M_{v_{\text{x}}} + \phi_{1,v_{\text{x}}} - b_{v_{\text{x}}}^{\text{pro}})}. \tag{55}$$

Notice $\frac{(b_{v_{\text{x}}}^{\text{off}})^2}{4(\kappa_{1,v_{\text{x}}} M_{v_{\text{x}}} + \phi_{1,v_{\text{x}}} - b_{v_{\text{x}}}^{\text{pro}})} < 0.15^2 K_{v_{\text{x}}}$ by assumption, thus $\dot{V}(t) < -K_{v_{\text{x}}} ({v_{\text{x}}}^2(t) - 0.15^2)$. This means as long as $v_{\text{x}}(t) \in [0.15, v_{\text{x}}^{\text{cri}}]$ with $t \geq t_{\text{stop}}$, we obtain $\dot{V}(t) < 0$, or equivalently $V(t) = \frac{1}{2} {v_{\text{x}}}^2(t)$ decreases monotonically. Recall $v_{\text{x}}(t_{\text{stop}}) \leq v_{\text{x}}^{\text{small}} \leq v_{\text{x}}^{\text{cri}}$, then the longitudinal speed decreases monotonically from $v_{\text{x}}(t_{\text{stop}})$ to 0.15 as time increases from t_{stop}. Suppose u becomes 0.15 at time $t'_{\text{brake}} \geq t_{\text{stop}}$, then $v_{\text{x}}(t) \leq 0.15$ for all $t \geq t'_{\text{brake}}$ because of the fact that $\dot{V}(t)$ is strictly negative when $v_{\text{x}}(t) = 0.15$.

Define $q_{v_\mathrm{x}} := \frac{(b_{v_\mathrm{x}}^{\text{off}})^2}{4(\kappa_{1,v_\mathrm{x}} M_{v_\mathrm{x}} + \phi_{1,v_\mathrm{x}} - b_{v_\mathrm{x}}^{\text{pro}})}$, then when $v_\mathrm{x}(t) \in [0.15, v_\mathrm{x}(t_{\text{stop}})]$, (55) can be relaxed to

$$\dot{V}(t) \leq -K_{v_\mathrm{x}} \cdot 0.15^2 + q_{v_\mathrm{x}}. \tag{56}$$

Integrating (56) from time t_{stop} to t'_{brake} and noting $v_\mathrm{x}(t_{\text{stop}}) \leq v_\mathrm{x}^{\text{small}}$ results in $t'_{\text{brake}} \leq \frac{(v_\mathrm{x}^{\text{small}})^2 - 0.15^2}{2 \cdot 0.15^2 K_{v_\mathrm{x}} - 2q_{v_\mathrm{x}}} + t_{\text{stop}}$. Then v_x becomes 0 no later than time $t_{\text{fstop}} + \sup(t'_{\text{brake}})$.

Case 3 - When $v_\mathrm{x}^{\text{small}} < v_\mathrm{x}(t_{\text{stop}}) \leq v_\mathrm{x}^{\text{cri}} + v_\mathrm{x}^{\text{small}}$: Recall (54) holds given $|e_{v_\mathrm{x}}(t)| = v_\mathrm{x}(t) \geq v_\mathrm{x}^{\text{small}}$, then

$$\dot{V}(t) \leq -K_{v_\mathrm{x}} e_{v_\mathrm{x}}^2(t) \leq -K_{v_\mathrm{x}} (v_\mathrm{x}^{\text{small}})^2, \tag{57}$$

and we have the longitudinal speed monotonically decreasing from $v_\mathrm{x}(t_{\text{stop}})$ at time t_{stop} until it reaches at $v_\mathrm{x}^{\text{small}}$ at some time $t_{\text{small}} \geq t_{\text{stop}}$. Integrating (57) from t_{stop} to t_{small} gives

$$\frac{1}{2}(v_\mathrm{x}^{\text{small}})^2 - \frac{1}{2} v_\mathrm{x}(t_{\text{stop}})^2 \leq -K_{v_\mathrm{x}} (v_\mathrm{x}^{\text{small}})^2 (t_{\text{small}} - t_{\text{stop}}). \tag{58}$$

Because $v_\mathrm{x}(t_{\text{stop}}) \leq v_\mathrm{x}^{\text{cri}} + v_\mathrm{x}^{\text{small}}$, (58) results in $t_{\text{small}} \leq \frac{(v_\mathrm{x}^{\text{cri}} + v_\mathrm{x}^{\text{small}})^2 - (v_\mathrm{x}^{\text{small}})^2}{2K_{v_\mathrm{x}} (v_\mathrm{x}^{\text{small}})^2} + t_{\text{stop}}$.

Once the longitudinal speed decreases to $v_\mathrm{x}^{\text{small}}$, we can then apply the same reasoning as in the second scenario to construct an upper bound of some time t''_{brake} that is no smaller than t_{small} and gives $v_\mathrm{x}(t''_{\text{brake}}) = 0.15$. However, this time we need to integrate (56) from time t_{small} to t''_{brake}. As a result, $t''_{\text{brake}} \leq \frac{(v_\mathrm{x}^{\text{small}})^2 - 0.15^2}{2 \cdot 0.15^2 K_{v_\mathrm{x}} - 2q_{v_\mathrm{x}}} + t_{\text{small}}$. Then u becomes 0 no later than time $t_{\text{fstop}} + \sup(t''_{\text{brake}})$ based on Assumption 13.

Now that we have the upper bound for v_x across these three scenarios, recall that once u arrives at 0, it remains at 0 afterward, and notice $\sup(t''_{\text{brake}}) > \sup(t'_{\text{brake}}) > t_{\text{stop}}$. Considering all three scenarios, 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.,

$$\begin{aligned} t_{\text{brake}} = & t_{\text{fstop}} + \frac{(v_\mathrm{x}^{\text{small}})^2 - 0.15^2}{2 \cdot 0.15^2 K_{v_\mathrm{x}} - 2q_{v_\mathrm{x}}} + \\ & + \frac{(v_\mathrm{x}^{\text{cri}} + v_\mathrm{x}^{\text{small}})^2 - (v_\mathrm{x}^{\text{small}})^2}{2K_{v_\mathrm{x}} (v_\mathrm{x}^{\text{small}})^2} + t_{\text{stop}} \end{aligned} \tag{59}$$

guarantees that $v_\mathrm{x}(t) = 0$ for all $t \geq t_{\text{brake}}$. ∎

Appendix B
Proof of Lemma 21

Before proving Lemma 21, we prove the following lemma:

Lemma 25. *Let $\mathcal{R}_j$ be the zonotope computed under the augmented hybrid system HS beginning from $\mathcal{X}_0^{aug}$ for arbitrary $j \in \mathcal{J}$. Then for any $x_0^{vel} \in \mathcal{X}_0^{vel}$ and $p \in \mathcal{P}$*

$$\begin{aligned} \texttt{slice}\big(\mathcal{R}_j \oplus \texttt{ROT}(\pi_h(\mathcal{R}_j)), x_0^{vel}, p\big) = \texttt{ROT}(\pi_h(\mathcal{R}_j)) \oplus \\ \oplus \texttt{slice}(\mathcal{R}_j, x_0^{vel}, p). \end{aligned} \tag{60}$$

Proof: Because $\texttt{ROT}(\pi_h(\mathcal{R}_j))$ is independent of x_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 (60) holds. ∎

Now we prove Lemma 21:

Proof: By definition $\texttt{slice}(\mathcal{R}_j, x_0^{\text{vel}}, p)$ and $\texttt{ROT}(\pi_h(\mathcal{R}_j))$ are both zonotopes, thus $\texttt{slice}\big(\mathcal{R}_j \oplus \texttt{ROT}(\pi_h(\mathcal{R}_j)), x_0^{\text{vel}}, p\big)$ is a zonotope based on (60). For simplicity, denote $\texttt{slice}\big(\mathcal{R}_j \oplus \texttt{ROT}(\pi_h(\mathcal{R}_j)), x_0^{\text{vel}}, p\big)$ as $<c'',\ G''>$, then $\xi(\mathcal{R}_j, x_0^{\text{vel}}, p)$ is a zonotope because $\pi_w\big(<c'',\ G''>\big) = \left\langle \begin{bmatrix} [c'']_1 \\ [c'']_2 \end{bmatrix},\ \begin{bmatrix} [G'']_{1:} \\ [G'']_{2:} \end{bmatrix} \right\rangle$.

Note $\pi_w\big(\texttt{ROT}(\pi_h(\mathcal{R}_j))\big) = \texttt{rot}(\pi_h(\mathcal{R}_j))$, and by using the definition of π_w one can check that $\pi_w(\mathcal{A}_1 \oplus \mathcal{A}_2) = \pi_w(\mathcal{A}_1) \oplus \pi_w(\mathcal{A}_2)$ for any zonotopes $\mathcal{A}_1, \mathcal{A}_2 \subset \mathbb{R}^{9+n_p}$. By Lemma 25,

$$\xi(\mathcal{R}_j, x_0^{\text{vel}}, p) = \pi_w\big(\texttt{slice}(\mathcal{R}_j, x_0^{\text{vel}}, p)\big) \oplus \texttt{rot}(\pi_h(\mathcal{R}_j)). \tag{61}$$

By Theorem 19 for any $t \in T_j$ and $j \in \mathcal{J}$, $x^{\text{aug}}(t) \in \texttt{slice}(\mathcal{R}_j, x_0^{\text{vel}}, p) \subset \mathcal{R}_j$, 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 $\xi(\mathcal{R}_j, x_0^{\text{vel}}, p)$ contains the vehicle footprint oriented according to $\pi_h(\mathcal{R}_j)$ and centered at $\pi_w(x^{\text{aug}}(t))$ during T_j. ∎

Appendix C
Proof of Theorem 23

We first present a pair of lemmas. The first lemma simplifies the expression of $\xi(\mathcal{R}_j, x_0^{\text{vel}}, p)$.

Lemma 26. *Let $\mathcal{R}_j = <c_{\mathcal{R}_j},\ [g_{\mathcal{R}_j,1}, g_{\mathcal{R}_j,2}, \ldots, g_{\mathcal{R}_j,\ell_j}]>$ be the zonotope computed under the augmented hybrid system HS beginning from $\mathcal{X}_0^{aug}$ for arbitrary $j \in \mathcal{J}$, and let $\texttt{rot}(\pi_h(\mathcal{R}_j)) = <c_{\texttt{rot}},\ G_{\texttt{rot}}>$ be defined as* (41)*. Then for arbitrary $x_0^{vel} \in \mathcal{X}_0^{vel}$ 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 $\xi(\mathcal{R}_j, x_0^{vel}, p) = <c_\xi + A \cdot p,\ G_\xi>$.*

Proof: Recall c^{slc} is defined as in (37), then

$$\begin{aligned} \xi(\mathcal{R}_j, x_0^{\text{vel}}, p) = <\pi_w(c^{\text{slc}}) + c_{\texttt{rot}},\ [\pi_w(g_{\mathcal{R}_j,(4+n_p)}), \ldots \\ \ldots, \pi_w(g_{\mathcal{R}_j,\ell_j}), G_{\texttt{rot}}]>, \end{aligned} \tag{62}$$

which follows from using (61) and (36) and 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 written as

$$\begin{aligned} c^{\text{slc}} = c_{\mathcal{R}_j} + \sum_{k=7}^{9} \frac{[x_0^{\text{vel}}]_{(k-6)} - [c_{\mathcal{R}_j}]_k}{[g_{\mathcal{R}_j,(k-6)}]_k} g_{\mathcal{R}_j,(k-6)} + \\ - \sum_{k=10}^{9+n_p} \frac{[c_{\mathcal{R}}]_k}{[g_{\mathcal{R},(k-6)}]_k} g_{\mathcal{R},(k-6)} + A' \cdot p \end{aligned} \tag{63}$$

with $A' = \left[\frac{1}{[g_{\mathcal{R}_j,4}]_{10}} g_{\mathcal{R}_j,4}, \ldots, \frac{1}{[g_{\mathcal{R}_j,(3+n_p)}]_{(9+n_p)}} g_{\mathcal{R}_j,(3+n_p)} \right]$. Therefore by performing algebra one can find that $\xi(\mathcal{R}_j, x_0^{\text{vel}}, p) = <c_\xi + A \cdot p,\ G_\xi>$ with some c_ξ, G_ξ and

$$\begin{aligned} A = \Bigg[\frac{1}{[g_{\mathcal{R}_j,4}]_{10}} \pi_w(g_{\mathcal{R}_j,4}), \frac{1}{[g_{\mathcal{R}_j,5}]_{11}} \pi_w(g_{\mathcal{R}_j,5}), \ldots \\ \ldots, \frac{1}{[g_{\mathcal{R}_j,(3+n_p)}]_{(9+n_p)}} \pi_w(g_{\mathcal{R}_j,(3+n_p)}) \Bigg]. \end{aligned} \tag{64}$$

∎

Note $\vartheta^{\text{loc}}(j, i, x_0^{\text{pos}})$ is a zonotope by construction in (45) because $\vartheta(j, i)$ is a zonotope. The following lemma follows from [43, Lem. 5.1].

Lemma 27. *Let* $\xi(\mathcal{R}_j, x_0^{vel}, p) = <c_\xi + A \cdot p,\ G_\xi>$ *be computed as in Lemma 26, and let* $\vartheta^{loc}(j, i, x_0^{pos}) =< c_\vartheta, G_\vartheta >$ *be computed from Assumptions 22 and* (45)*. Then* $\xi(\mathcal{R}_j, x_0^{vel}, p) \cap \vartheta^{loc}(j, i, x_0^{pos}) \neq \emptyset$ *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 23:

Proof: Let $\xi(\mathcal{R}_j, x_0^{\text{vel}}, p) = <c_\xi + A \cdot p,\ G_\xi>$ as computed in Lemma 26, and let $\vartheta^{\text{loc}}(j, i, x_0^{\text{pos}}) =< c_\vartheta, G_\vartheta >$ be computed from Assumption 22 and (45). Because all zonotopes are convex polytopes [43], $<c_\vartheta - c_\xi,\ [G_\vartheta, G_\xi]> \subset \mathcal{W} \subseteq \mathbb{R}^2$ 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

$$B^+ := \left[\frac{[B^-]_{:1}}{\|[B^-]_{:1}\|}, \frac{[B^-]_{:2}}{\|[B^-]_{:2}\|}, \ldots, \frac{[B^-]_{:\ell}}{\|[B^-]_{:\ell}\|} \right]^\top \in \mathbb{R}^{\ell\times 2}. \quad (65)$$

Then as a result of [36, Thm 2.1], $<c,\ G> = \{a \in \mathcal{W} \mid B \cdot a - b \leq 0\}$ with

$$B = \begin{bmatrix} B^+ \\ -B^+ \end{bmatrix} \in \mathbb{R}^{2\ell\times 2}, \quad (66)$$

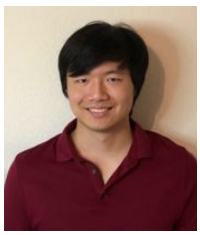
$$b = \begin{bmatrix} B^+ \cdot c + |B^+ \cdot G| \cdot \mathbf{1} \\ -B^+ \cdot c + |B^+ \cdot G| \cdot \mathbf{1} \end{bmatrix} \in \mathbb{R}^{2\ell} \quad (67)$$

where $\mathbf{1} \in \mathbb{R}^\ell$ is the column vector of ones. By Lemma 27, $\xi(\mathcal{R}_j, x_0^{\text{vel}}, p) \cap \vartheta^{\text{loc}}(j, i, x_0^{\text{pos}}) = \emptyset$ 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 [44, Theorem 5.4.5]. ■

Jinsun Liu received BE degrees in electrical engineering from the University of Minnesota and Beijing Jiaotong University both in 2015; an MS degree in electrical engineering and a PhD degree in robotics both from the University of Michigan in 2023. His research interests include robotics, control, motion planning, and optimization.

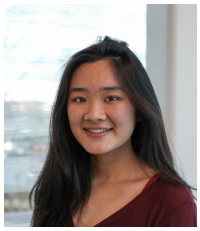
Yifei Simon Shao earned a BE degree in Mechanical Engineering from The Cooper Union in 2019. He then obtained MS dual degrees in Robotics and Mechanical Engineering from the University of Michigan in 2021. Currently, he is pursuing a PhD at the University of Pennsylvania. His research encompass the utilization of optimization tools and machine learning in motion planning.

Lucas Lymburner received a BSE degree in Computer Science from the University of Michigan in 2021. He is currently pursuing an MS degree in Robotics from the University of Michigan.



Hansen Qin received an MS degree in robotics from the University of Michigan. His background includes motion planning and control theory. Currently, he is working as a simulation engineer with Latitude AI on their L3 self-driving division.

Vishrut Kaushik received an MS degree in robotics from the University of Michigan in 2023. He currently works with Peer Robotics and is interested in solving perception problems for robots in dynamic environments.



Lena Trang is currently pursuing a Computer Engineering BSE and Robotics BSE dual degree at the University of Michigan. She is also on the Michigan Mars Rover Team where she was Robotic Arm Subteam Lead and a Mobility Subteam member. Her interests include deep space robotics and mechanical design.

Ruiyang Wang (Student Member, IEEE) received a BS degree in mechanical engineering from the University of Michigan in 2022. He is currently working toward an MS degree in robotics, focusing on multi-robot system coordination and control with the Department of Robotics, University of Michigan.

Vladimir Ivanovic received the Ph.D. degree in mechanical engineering from the University of Zagreb, Zagreb, Croatia, in 2010. He is currently a Technical Expert with the Controls and Automated Systems R&A Group, Research and Innovation Center, Ford Motor Company. Since joining Ford in 2011, he has been contributed to several technologies in the area of automatic transmissions, active drivelines, and active driver assist systems. His research interests include modeling and control of automotive systems.

H. Eric Tseng received the B.S. degree from the National Taiwan University in 1986, and the MS and PhD degrees in mechanical engineering from the University of California, Berkeley in 1991 and 1994, respectively. His technical achievements have been recognized seven times with Ford's highest technical award—the Henry Ford Technology Award, as well as by the American Automatic Control Council with the Control Engineering Practice Award in 2013. He has over 100 patents and over 120 publications. He is an NAE Member.

Ram Vasudevan is an associate professor in robotics and mechanical engineering at the University of Michigan. He received a BS, MS, and PhD degrees in electrical engineering and computer sciences from the University of California, Berkeley in 2006, 2009, and 2012, respectively. His work has received best paper awards at the IEEE Conference on Robotics and Automation, the ASME Dynamics Systems and Controls Conference, and IEEE OCEANS Conference.