Delft University of Technology

Delft Center for Systems and Control

Technical report 24-023

Efficient and safe learning-based control of piecewise affine systems using optimization-free safety filters*

K. He, S. Shi, T. van den Boom, and B. De Schutter

If you want to cite this report, please use the following reference instead:
K. He, S. Shi, T. van den Boom, and B. De Schutter, "Efficient and safe learning-based control of piecewise affine systems using optimization-free safety filters," *Proceedings of the 63rd IEEE Conference on Decision and Control*, Milan, Italy, pp. 5046–5053, Dec. 2024. doi:10.1109/CDC56724.2024.10886328

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.24.73 (secretary)
URL: https://www.dcsc.tudelft.nl

* This report can also be downloaded via https://pub.bartdeschutter.org/abs/24_023

Efficient and Safe Learning-based Control of Piecewise Affine Systems Using Optimization-Free Safety Filters*

Kanghui He[1], Shengling Shi[2], Ton van den Boom[1], and Bart De Schutter[1]

***Abstract*— Control of piecewise affine (PWA) systems under complex constraints faces challenges in guaranteeing both safety and online computational efficiency. Learning-based methods can rapidly generate control signals with good performance, but rarely provide safety guarantees. A safety filter is a modular method to improve safety for any controller. When applied to PWA systems, a traditional safety filter usually need to solve a mixed-integer convex program, which reduces the computational benefit of learning-based controllers. We propose a novel optimization-free safety filter designed to handle state constraints that involve a combination of polyhedra and ellipsoids. The proposed safety filter only utilizes algebraic and min-max operations to determine safe control inputs. This offers a notable advantage compared with traditional safety filters by allowing for significantly more efficient computation of control signals. The proposed safety filter can be integrated into various function approximators, such as neural networks, enabling safe learning throughout the learning process. Simulation results on a bicycle model with PWA approximation validate the proposed method regarding constraint satisfaction, CPU time, and the preservation of sub-optimality.**

I. Introduction

Backgrounds. There is a growing interest in controlling piecewise affine (PWA) systems due to their ability to represent hybrid models and to approximate nonlinear dynamics [1]. Control of PWA systems with constraints faces challenges in balancing multiple performance measures, such as safety, optimality, and online computational efficiency. In particular, hybrid model predictive control (MPC) can be applied to PWA systems, with mature results on sub-optimality and safety guarantees [2]. With a PWA prediction model, the MPC problem can be converted into a mixed-integer convex programming (MICP) problem. However, this comes at a large online computational price to solve the MICP problem, especially when there are complex state constraints such as a combination of polyhedral and ellipsoidal constraints. Explicit MPC, which finds the analytical solution of the hybrid MPC problem, also suffers from computational complexity issues [3]. In contrast, other methods that use a simple control structure, such as adaptive control [4] and piecewise linear feedback control [2], can be implemented rapidly but lack formal performance guarantees.

In recent years, various learning-based methods have been employed to design controllers. There are two main classes of learning-based methods: learning models and learning control policy parameters. In this paper, we assume the PWA model is given and thus focus on the second class of methods. A common issue of learning-based controllers under constraints is that they can lead to infeasible solutions. To deal with this issue, one can add a penalty to the learning objective [5] or design a safety filter [6]–[8]. The penalty method cannot always provide strict constraint satisfaction. In comparison, safety filters usually involve a constrained optimization problem to determine a safe control input. The main constraint of the problem is that the successor state should be within a controlled invariant set. Safety filters can be applied to any learning-based controllers but can introduce significant computational complexity. In this paper, our primary emphasis is on the safety filter approach, with a specific focus on mitigating its computational burden.

Related work. Simplifying hybrid MPC: To reduce the computational burden of hybrid MPC, research has been conducted to simplify or approximate the solution of the MICP problem. [9] considers warm start of MPC. [10] focuses on predicting the discrete solution of the MICP problem by machine learning and solving convex programs online. These methods provide safety and optimality certificates, but still require some online convex or mixed-integer optimization. The computational issue is thus not completely addressed.

Safety filters for PWA systems: For PWA systems encompassing linear or PWA constraints, a less conservative controlled invariant set is usually a union of polyhedra, making the optimization problem in the safety filter an MICP problem [6]. This means that the process of imposing safety comes at the expense of high computation times. Some papers have investigated how to enforce constraints without solving any optimization problems. For linear systems with linear constraints, leveraging their vertex representation can help to produce safe control actions [11]. [12] finds the explicit solution of an RL safety filter comprised of a quadratic program. In [13], an output-constrained neural network (NN) structure is proposed to approximate the solution of convex optimization problems. The optimization-free methods in [11]–[13], [13] require the convexity of both the model and constraints, and thus are not applicable to PWA systems.

Contributions. In this paper, we consider PWA systems with complex constraints consisting of a union of polyhedra and ellipsoids. We aim to impose safety for any reference

*This paper is part of a project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 101018826 - CLariNet).

[1]Kanghui He (k.he@tudelft.nl), Ton van den Boom (a.j.j.vandenBoom@tudelft.nl), and Bart De Schutter (b.deschutter@tudelft.nl) are with Delft Center for Systems and Control, Delft University of Technology, Delft, The Netherlands.

[2]Shengling Shi (slshi@mit.edu) is with the Department of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA.

controller with a low online computational price. We design a safety filter that computes the safe control input closest to the reference input along a predetermined direction. We obtain the analytical expression of the mapping from the reference input to the safe input. The computation of control inputs is thus based on some basic algebraic, min, and max operations. As a significant benefit, this usually in practice enables a much more efficient computation of control efforts compared to classical safety filters. The proposed safety filter requires a precomputed controlled invariant set consisting of a union of polyhedra and ellipsoids. To achieve this purpose, we transform the invariant set construction problem into a binary classification problem and propose a PWA and piecewise quadratic (PWQ) classification approach that can efficiently estimate a controlled invariant set with low conservativeness.

II. Preliminaries and problem formulation

Notations. The sets of non-negative integers, positive integers, non-negative integers less than or equal to a, and positive integers less than or equal to a are denoted by $\mathbb{N}$, $\mathbb{N}^+$, $\mathbb{N}_a$, and $\mathbb{N}_a^+$, respectively. The boundary of the set S is denoted by ∂S. The operator $\min\{\cdot,\cdot\}$ returns the smaller of two scalars, $\min(\cdot)$ outputs the minimum element of a vector, $\mathrm{step}(\cdot)$ is a unit step function.

A. Problem formulation

We consider a deterministic discrete-time PWA system

$$x_{t+1} = f_{\mathrm{PWA}}(x_t, u_t) = A_i x_t + B_i u_t + f_i, \quad \text{if } x_t \in \mathcal{C}_i, \ (1)$$

where $x_t \in \mathbb{R}^{n_x}$ and $u_t \in \mathbb{R}^{n_u}$ are the state and the input at time step t, $f_{\mathrm{PWA}}(\cdot,\cdot) : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ satisfies $f_{\mathrm{PWA}}(0,0) = 0$, $\{\mathcal{C}_i\}_{i=1}^s$ is a strict polyhedral partition [1, Definition 4.5] of the state space, s is the number of polyhedral regions, and the matrices A_i, B_i and the vectors f_i define the affine dynamics in the regions $\mathcal{C}_i$. The system is required to satisfy the state and input constraints $x_t \in X \subseteq \mathbb{R}^{n_x}$, $u_t \in U(x_t) \subseteq \mathbb{R}^{n_u}$ for all $t \in \mathbb{N}$. We assume that $X = \cup_{i=1}^{m_x} X_i$ where each X_i is a polyhedron or ellipsoid, and m_x is the number of these sets. We also assume that $U(x)$ is a projection of a polyhedron or ellipsoid in $\mathbb{R}^{n_x} \times \mathbb{R}^{n_u}$ onto $\mathbb{R}^{n_u}$. Note that $U(\cdot)$ accommodates the case when there is a state-dependent input constraint. Besides, we assume that f_{PWA} is known and there is no disturbance.

We will explore imposing safety on PWA systems in a computationally cheap manner, either by modifying a given unsafe controller or by directly learning a safe controller.

Main problem P1: Given a PWA system (1) and a reference controller $\pi_{\mathrm{r}}(\cdot) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_u}$, adapt inputs $\pi_{\mathrm{r}}(x)$ to $\pi^*(x)$ *online* in a computationally efficient way such that the trajectory of the closed-loop system with $\pi^*(\cdot)$ satisfies the constraints $x_t \in X$ and $\pi^*(x_t) \in U(x_t)$, $\forall t \in \mathbb{N}$.

Extended problem P2: Given a PWA system (1), synthesize *offline* a controller $\pi_\theta(\cdot) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_u}$ with the parameter θ such that the trajectory of the closed-loop system with $\pi_\theta(\cdot)$ satisfies the constraints $\forall t \in \mathbb{N}$, and simultaneously minimizes a predefined cost $J(\theta)$.

The reference controller is a baseline control strategy intended to optimize performance in certain aspects but does not provide safety guarantees. This includes, but is not limited to, controllers based on reinforcement learning.

B. Safety filter

Without considering the online computational complexity, problem P1 can be solved by constructing a safety filter [6]–[8]. In particular, suppose that a controlled invariant set $S \subseteq X$ [1, Definition 10.9] is available. In the online situation, when receiving $\pi_{\mathrm{r}}(x)$ and x, the safety filter solves the constrained optimization problem

$$\pi_{\mathrm{p}}(x) = \arg\min_{u \in U(x)} \ ||u - \pi_{\mathrm{r}}(x)||_2, \text{s.t. } f_{\mathrm{PWA}}(x,u) \in S, \ (2)$$

and applies $\pi_{\mathrm{p}}(x)$ to the system. The problem (2) thus implicitly determines the projected policy $\pi_{\mathrm{p}}(\cdot)$. The invariance of S guarantees the recursive feasibility of (2) for any $x_0 \in S$, and the relation $S \subseteq X$ ensures the safety of the closed-loop system with $\pi_{\mathrm{p}}(\cdot)$.

Because of (2), $\pi_{\mathrm{p}}(\cdot)$ is usually different from $\pi_{\mathrm{r}}(\cdot)$, and some other good performance of $\pi_{\mathrm{r}}(\cdot)$ could be sacrificed if $\pi_{\mathrm{p}}(\cdot)$ is applied. To reduce the conservatism of $\pi_{\mathrm{p}}(\cdot)$, we expect to use a large S in (2), ideally the maximal controlled invariant set. However, for PWA systems, especially those with complex state constraints such as $x \in X$ considered in this paper, a large S is usually a union of many polyhedra and ellipsoids, which has a very complex representation. As a result, *globally* solving (2) needs a mixed-integer encoding of the constraint $f_{\mathrm{PWA}}(x,u) \in S$. Therefore, the online computational issue is not completely solved by (2). To this end, we under-approximate the solution of (2) by finding an explicit map from $\pi_{\mathrm{r}}(\cdot)$ to a suboptimal solution of (2), without solving any mixed-integer program. This method will determine the safe control inputs through algebraic operations, and thus solve the computational issue.

To solve problem P2, existing literature [7] has made (2) an optimization layer when parameterizing the policy, and let (2) be included in the training loop of $\pi_\theta(\cdot)$. In particular, when training θ, this optimization layer is considered in the backward pass, such that the cost $J(\theta)$ is minimized for the projected policy. There are two advantages brought by this idea. First, safety can be ensured even during the learning process. Besides, some good performance of $\pi_\theta(\cdot)$ can be preserved. However, when performing the backward pass, it is difficult to compute the policy gradient $\nabla_\theta J$. To deal with this issue, our method for solving P1 allows for a closed-form representation of $\pi_\theta(\cdot)$ and thus makes $\pi_\theta(\cdot)$ end-to-end trainable. Consequently, problem P2 will be solved.

III. Optimization-free safety filter

Our main focus is on problem P1. We propose a safety filter that finds an analytical solution to a policy optimization problem. This solution can be directly applied to solve P2 by integrating the filter into the learning process of $\pi_\theta(\cdot)$.

We need a stronger assumption than the availability of the controlled invariant set S.

Assumption 1: A controlled invariant set $S \subseteq X$ and a safe policy $\pi_{\rm s}(\cdot) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_u}$ on S are available. In other words, S is positive invariant [1, Definition 10.7] for the autonomous system $x_{t+1} = f_{\rm PWA}(x_t, \pi_{\rm s}(x_t))$ under the constraints $x \in X$ and $\pi_{\rm s}(x) \in U(x)$.

Assumption A1 is not restrictive, because for many methods of synthesizing S such as the LMI-based method [2], polytopic trees [14], and learning control barrier functions [15], a safe policy $\pi_{\rm s}(\cdot)$ on S is obtained as a by-product. We assume that $S = \cup_{i=1}^{w} S_i$, where each S_i is a polyhedron or an ellipsoid, and $w \in \mathbb{N}^+$. For each polyhedral S_i, it can have the representation $S_i = \{x \in \mathbb{R}^{n_x} \mid H_i x \leq h_i\}$. For each ellipsoidal S_i, it has the representation $S_i = \{x \in \mathbb{R}^{n_x} \mid (x - e_i)^T E_i (x - e_i) \leq 1\}$. Here, $h_i \in \mathbb{R}^{n_{h_i}}$, $H_i \in \mathbb{R}^{n_{h_i} \times n_x}$, $E_i \in \mathbb{R}^{n_x \times n_x} \succ 0$, and $e_i \in \mathbb{R}^{n_x}$.

Inspired by [13], our method consists of a direction determination followed by a line search. In the online phase when the reference control signal $\pi_{\rm r}(x)$ is received, we find a safe control input $u \in U(x)$ along the line segment with the endpoints $\pi_{\rm s}(x)$ and $\pi_{\rm r}(x)$, such that $f_{\rm PWA}(x, u) \in S$ and the L_2 distance between u and $\pi_{\rm r}(x)$ is minimized. In other words, we consider the optimizer of the problem

$$\lambda^*(x) \triangleq \max_{\lambda \in [0,1]} \lambda \tag{3a}$$

$$\text{s.t. } u = \pi_{\rm s}(x) + \lambda(\pi_{\rm r}(x) - \pi_{\rm s}(x)), \tag{3b}$$

$$u \in U(x),\ f_{\rm PWA}(x, u) \in S, \tag{3c}$$

and apply the input

$$\pi^*(x) = \pi_{\rm s}(x) + \lambda^*(x)(\pi_{\rm r}(x) - \pi_{\rm s}(x)) \tag{4}$$

to the system.

The reasons why we consider polyhedral and ellipsoidal invariant sets are that (i) most existing methods [2], [6], [14] of synthesizing S for linear or PWA systems result in these types of sets, and that (ii) considering these types allows us to compute the optimizer $\lambda^*(\cdot)$ in a closed form. The methods of synthesizing S will be discussed in the later section.

Let $R(\cdot) : \mathbb{R}^{n_x} \to \mathbb{N}_s^+$ represent the function that maps x to the index i of $\mathcal{C}_i$ such that $x \in \mathcal{C}$. Since $\{\mathcal{C}_i\}_{i=1}^s$ is a strict polyhedral partition of the state space, $R(x)$ is well-defined. The constraint $f_{\rm PWA}(x, u) \in S$ in (3) is equivalent to $\exists i \in \mathbb{N}_w^+$ s.t. $A_{R(x)}x + B_{R(x)}u + f_{R(x)} \in S_i$. After substituting the expression (3b) of u into (3c), constraints (3b)-(3c) are equivalent to $\lambda \in \Pi_u(x)$ and $\lambda \in \cup_{i \in \mathbb{N}_w^+} \Pi_i(x)$, where $\Pi_u(x) = \{\lambda \in \mathbb{R} \mid \pi_{\rm s}(x) + \lambda(\pi_{\rm r}(x) - \pi_{\rm s}(x)) \in U(x)\}$, and $\Pi_i(x) = \{\lambda \in \mathbb{R} \mid u = \pi_{\rm s}(x) + \lambda(\pi_{\rm r}(x) - \pi_{\rm s}(x)),\ A_{R(x)}x + B_{R(x)}u + f_{R(x)} \in S_i\}$. Each Π_i represents the set of λ making $f_{\rm PWA}(x, u) \in S$, while the successor state $f_{\rm PWA}(x, u)$ is restrained in S_i.

If $U(\cdot)$ is polyhedral (ellipsoidal), $\lambda \in \Pi_u(x)$ is a linear (quadratic) constraint on λ. As a result, problem (3) can be expressed as

$$\lambda^*(x) = \max_{\lambda} \lambda \tag{5a}$$

$$\text{s.t. } a\lambda^2 + b\lambda \leq c \tag{5b}$$

$$\exists i \in \mathbb{N}_w^+,\ a^{(i)}\lambda^2 + b^{(i)}\lambda \leq c^{(i)}, \tag{5c}$$

where all the coefficients depend on x, and (5b) and (5c) are element-wise inequalities. If $U(x)$ is polyhedral, $a = 0$. If S_i is polyhedral, $a^{(i)} = 0$. The coefficients a, b and c are vectors. If S_i is ellipsoidal, $a^{(i)}$, $b^{(i)}$, and $c^{(i)}$ are scalars, otherwise, they are usually vectors. The constraint $\lambda \in [0, 1]$ is included in (5b) with the form of $[1\ \ -1]^T\lambda \leq [1\ 0]^T$. Due to the positive definiteness of E_i, a and $a^{(i)}$ are non-negative. For any x, the value of these coefficients can be easily obtained by substituting (3b) and the affine dynamics into the inequalities of $U(x)$ and S_i.

The following proposition characterizes the optimizer $\lambda^*(\cdot)$. A detailed proof is given in the appendix.

Proposition 1: Consider problem (5), the optimizer $\lambda^*(\cdot)$ has the representation:

$$\lambda^*(x) = \min\left\{\min(\lambda^{(0)}),\ \max_{i \in \mathbb{N}_w^+}\{\min(\lambda^{(i)})\}\right\}, \tag{6}$$

with

$$\lambda_j^0 = \begin{cases} c_j/b_j & \text{if} \quad a_j = 0,\ b_j > 0; \\ 1 & \text{if} \quad a_j = 0,\ b_j \leq 0; \\ \frac{-b_j + \sqrt{b_j^2 + 4a_j c_j}}{2a_j} & \text{if} \quad a_j > 0, \end{cases} \tag{7}$$

$$\lambda_j^{(i)} = \begin{cases} c_j^{(i)}/b_j^{(i)} & \text{if } a_j^{(i)} = 0,\ b_j^{(i)} > 0; \\ \text{step}(c_j^{(i)}) & \text{if } a_j^{(i)} = 0,\ b_j^{(i)} = 0; \\ \text{step}\left(c_j^{(i)} - b_j^{(i)} \min\left\{\min(\lambda^{(0)}), \min\limits_{k \text{ s.t. } \lambda_k^{(i)} \in (0,1)} \lambda_k^{(i)}\right\}\right) & \\ \qquad \text{if } a_j^{(i)} = 0,\ b_j^{(i)} < 0; & \\ 0 \qquad \text{if } a_j^{(i)} > 0,\ \delta_j^{(i)} \triangleq {b_j^{(i)}}^2 + 4a_j^{(i)}c_j^{(i)} < 0; & \\ \frac{-b_j^{(i)} + \sqrt{\delta_j^{(i)}}}{2a_j^{(i)}} \text{step}\left(2a_j^{(i)} \min(\lambda^{(0)}) + b_j^{(i)} + \sqrt{\delta_j^{(i)}}\right) & \\ \qquad \text{if } a_j^{(i)} > 0,\ \delta_j^{(i)} \geq 0. \end{cases} \tag{8}$$

In (6)-(8), λ^* is a scalar and λ^i, $i \in \mathbb{N}_w$ are vectors. The subscript j represents the jth element of a vector.

The equations (4), (6)-(8) constitute the proposed safe controller $\pi^*(\cdot)$, which makes the system (1) safe on S under Assumption 1. At the same time, $\pi^*(x)$ is the closest safe input along the line connecting $\pi_{\rm s}(x)$ and $\pi_{\rm r}(x)$ for any x. This means that if $\pi_{\rm r}(x)$ is a safe control input, i.e., $\pi_{\rm r}(x) \in U(x)$ and $f_{\rm PWA}(x, \pi_{\rm r}(x)) \in S$, we have $\pi^*(x) = \pi_{\rm r}(x)$. These mechanisms make (4) and (6)-(8) work as a safety filter for the system (1). Fig. 1 illustrates the geometric meanings of $\lambda^*(\cdot)$ and $\pi^*(\cdot)$.

To solve Problem P2, since we have obtained the closed-form of the mapping from x, $\pi_{\rm r}(x)$ and $\pi_{\rm s}(x)$ to $\lambda^*(x)$, we can easily involve (4) in the training loop of the learning-based policy $\pi_{\rm r}(\cdot)$. Compared to [7], which requires differentiating the optimizer of an optimization problem, our method enables efficient policy gradient computation by applying the chain rule to (4) and (6)-(8).

Our design draws inspiration from [13], in which the solution of convexly constrained optimization problems is approximated while constraints are strictly satisfied. In contrast, our method accommodates non-convex constraints (a

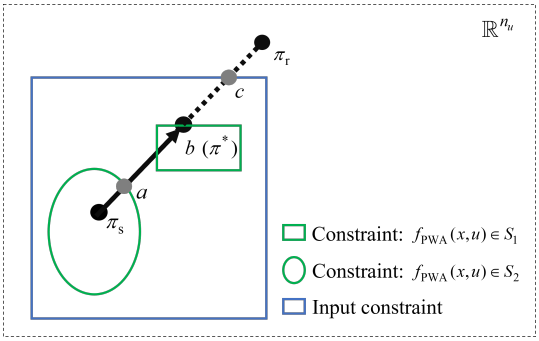

Fig. 1. Geometric interpretation of the proposed safety filter. We consider the case when $S = S_1 \cup S_2$. Each region with a green boundary represents the constraint on u such that $f_{\text{PWA}}(x,u) \in S_i,\ i = 1,2$. The input from the reference policy π_{r} is outside U and thus unsafe. We search for the least conservative input along the segment from π_{s} to π_{r}. According to (6)-(8), we have $\min(\lambda^{(1)}) = \frac{|a-\pi_{\text{s}}|}{|\pi_{\text{r}}-\pi_{\text{s}}|}$, $\min(\lambda^{(2)}) = \frac{|b-\pi_{\text{s}}|}{|\pi_{\text{r}}-\pi_{\text{s}}|}$, and $\min(\lambda^{(0)}) = \frac{|c-\pi_{\text{s}}|}{|\pi_{\text{r}}-\pi_{\text{s}}|}$. Then, $\lambda^* = \frac{|b-\pi_{\text{s}}|}{|\pi_{\text{r}}-\pi_{\text{s}}|}$, and the filtered input is b.

union of polyhedra and ellipsoids). Furthermore, unlike [13], we do not necessitate $\pi_{\text{s}}(x)$ to lie in the interior of each polyhedron and ellipsoid.

While the computational advantage of the proposed safe filter based on min-max operations cannot be formally or religiously proved, it is typically true in practice. Besides, in [16], it is also verified by simulation that the min-max based approach is faster than MPC for PWA systems.

IV. Synthesizing S and $\pi_{\text{s}}(\cdot)$

In this section, we discuss and present the methods of synthesizing the controlled invariant set S and the safe policy $\pi_{\text{s}}(\cdot)$ for the PWA system (1).

Existing methods of synthesizing S for PWA systems include the LMI approach [2], polyhedral iteration [6], and polytopic trees [14]. In this section, we first inner approximate the original state constraint by a polyhedral constraint. The above-mentioned approaches can then be adopted to compute a controlled invariant set S_0 under the inner approximated constraint. Then, we propose a PWA and PWQ classification approach to estimate a large S that is a union of polyhedra and ellipsoids. The approach can approximate the maximum control-invariant set under the original constraint.

A. Inner approximation of the state constraint

To apply the existing methods of synthesizing $\mathcal{S}$, we need to inner approximate the state constraint by a polyhedron $\{x \in \mathbb{R}^{n_x} \mid Gx \leq g\}$ with $G \in \mathbb{R}^{n_g \times n_x}$ and $g \in \mathbb{R}^{n_g}$. The matrix G should be determined in advance. The choice of G is problem-specific and the simplest choice is $G = [\mathbb{I}_{n_x}\ -\mathbb{I}_{n_x}]^T$, which makes the polyhedron a hyper-rectangle. Then, we decrease g until the following condition is verified:

$$x \in X,\ \forall x \text{ s.t. } Gx \leq g. \tag{9}$$

The constraint $x \in \mathcal{C}_i$ or $x \in X_i$ can be easily represented by linear or quadratic inequalities, and the state constraint is the intersection of $\cup_{i \in \mathbb{N}_s^+} \mathcal{C}_i$ and $\cup_{i=1}^{m_x} X_i$. With this knowledge, for the state constraint we can compute a validation function $h_X(\cdot) : \mathbb{R}^{n_x} \to \mathbb{R}$ (as defined below) through some min and max operators [17].

Definition 1: Given a closed set Z, a function $h_Z(\cdot) : \mathbb{R}^{n_z} \to \mathbb{R}$ is called a validation function for the constraint $z \in Z$ if $h_Z(z) \leq 0 \Leftrightarrow z \in Z$.

Then, the condition (9) is equivalent to

$$\max_{x \in \mathbb{R}^{n_x}} \{h_X(x),\ \text{s.t. } Gx \leq g\} \leq 0. \tag{10}$$

The included problem in (10) can be solved by mixed-integer linear or quadratic programming.

With a polyhedral inner approximation of the state constraint available, we can thereby apply the approaches [2], [6], [14] to obtain a union of polyhedra or union of ellipsoids controlled invariant set S_0 and a safe policy under the inner approximated constraint with guarantees.

B. A PWA and PWQ classification approach

In the previous subsection, there are several steps that can induce conservatism for S. Firstly, the state constraint is inner approximated. Besides, if the LMI approach [2] is used, we can only obtain S for the piecewise linear subsystem[1], rather than the original PWA system (1). Conservatism can also arise when applying the S-procedure to get tractable LMIs. These simplifications usually result in very small S. This observation motivates us to consider enlarging S.

According to the reachability analysis [1, Theorem 11.1], the T-step controllable set S_T of a controlled invariant set S_0 is still controlled invariant, and S_T expands to the maximum controlled invariant set as T grows to the infinity. Exactly computing S_T for PWA systems with complex state constraints is impossible. Therefore, we present a data-driven approach to approximate S_T. First, we need an oracle to tell whether a particular state is in S_T. To achieve this, we consider the following unconstrained optimization problem:

$$\begin{aligned} &B_T(x) \triangleq \\ &\min_{\{u_t,x_t\}_{t=0}^T} \max\left\{\max_{t \in \mathbb{N}_{T-1}} h_X(x_t), \max_{t \in \mathbb{N}_{T-1}} h_{U(x_t)}(u_t), h_{S_0}(x_T)\right\} \\ &\text{s.t. } x_0 = x,\ x_{t+1} = f_{\text{PWA}}(x_t,u_t),\ t = 0,1,...,T-1, \end{aligned} \tag{11}$$

representing a discrete-time version of the Hamilton-Jacobi reachability problem [18]. In (11), S_0 is a union of polyhedra and ellipsoids controlled invariant set, which can be obtained by the above-mentioned approaches [2], [6], [14]. The functions $h_X(\cdot)$, $h_{U(\cdot)}(\cdot)$, and $h_{S_0}(\cdot)$ are PWA or PWQ validation functions. Problem (11) can be straightforwardly cast as an MICP problem.

Lemma 1: The optimal value function $B_T(\cdot)$ of (11) is a validation function for the constraint $x \in S_T$, i.e., $S_T = \{x \in \mathbb{R}^{n_x} \mid B_T(x) \leq 0\}$.

Proof: According to (11), we can straightforwardly get the following equivalence: $B_T(x) \leq 0 \Leftrightarrow$ there exists $\{(u_t,x_t)\}_{t=0}^T$ such that $h_X(x_t) \leq 0$, $h_{U(x_t)}(u_t) \leq 0$, and $h_{S_0}(x_T) \leq 0 \Leftrightarrow x \in S_T$. ■

[1]The piecewise linear subsystem is obtained by removing every mode i of (1) such that $0 \notin \text{closure}(\mathcal{C}_i)$.

Lemma 1 implies that $B_T(\cdot)$ can serve as the oracle by testing whether $B_T(x) \leq 0$ for any particular state x. Then, we can sample the state space, solve the problem (11) to generate a data set $\{(x_i, \text{step}(-B_T(x_i)))\}_{i=1}^{N_x}$, where N_x is the number of samples. To learn S_T, existing literature has used a NN binary classification model [19] or support vector machine [20] to learn the mapping from x to $\text{step}(-B_T(x))$. These approaches, however, are not suitable in our case because we require the learned set to be a union of polyhedra and ellipsoids. To this end, we design the following parameterized function:

$$\hat{B}(x, \omega) = \min_{j \in \mathbb{N}_l^+} \{\hat{B}_j(x, \omega_j)\} - 1, \tag{12}$$

where $l \in \mathbb{N}^+$ and

$$\hat{B}_j(x, \omega_j) = ||P_j(x - c_j)||_\infty \text{ or } (x - c_j)^T P_j (x - c_j) \tag{13}$$

with the learning parameters $\omega_j = (P_j, c_j)$, $P_j \in \mathbb{R}^{n_x \times n_x}$ and $c_j \in \mathbb{R}^{n_x}$. All the parameters are condensed in ω. In (12), each $\hat{B}_j(\cdot, \omega_j)$ is called a component function. For each state, the smallest component function is called the active function. The equation (13) means that for each component, we can use either a PWA function or a quadratic function. Each P_j is full rank if $\hat{B}_j(\cdot, \omega_j)$ is PWA, or positively definite if $\hat{B}_j(\cdot, \omega_j)$ is quadratic. The ways of guaranteeing P_j to be full rank or positively definite can be found in [8], [21], which leverage singular value or Cholesky decomposition. The zero sub-level set of $\hat{B}(\cdot, \omega)$, denoted by $\hat{S}_B$, is a union of polyhedra and ellipsoids.

The parameters are trained to minimize the misclassification error over the state space. Intuitively, this can be formulated as the following unconstrained optimization problem:

$$\min_{\omega} \int_{S_T \setminus \hat{S}_B} 1 dx + \int_{\hat{S}_B \setminus S_T} 1 dx. \tag{14}$$

The objective function represents the volume of the regions that $\hat{B}(\cdot, \omega)$ misclassifies. However, it is hard to find the analytical relation between the objective function of (14) and ω. This means that to solve (14), one can only use numerical gradient-based or gradient-free optimization algorithms. Since w is composed of several matrices and vectors, these algorithms become inefficient when the total number N_x of samples is large and w has numerous components. To deal with this issue, we propose to train the parameters such that the following optimization problem is solved[2]

$$\min_{\omega} \int_{x \in \mathbb{R}^{n_x}} \frac{\text{sign}(B_T(x)) \max\{-B_T(x)\hat{B}(x, \omega), 0\}}{B_T(x)} \, dx, \tag{15}$$

where $\text{sign}(\cdot)$ returns the sign of a real number and specifically $\text{sign}(0)/0 = -1$.

Interpretation of (15). For any x, the term in the integral is a measure of the misclassification. In particular, if $x \in S_T \cap \hat{S}_B$ or if $x \notin S_T$ and $x_i \notin \hat{S}_B$, the term in the integral reaches its minimum (zero). Otherwise, the learning model (12) misclassifies x and the term in the integral is $-\text{sign}(B_T(x))\hat{B}(x, \omega)$. This indicates that the objective of (14) is to make $\hat{B}(x, \omega)$ have the same sign as $B_T(x)$ for all x.

As the integral in (15) is difficult to compute, like existing supervised learning methods [19], we approximate the objective of (15) by an empirical loss, i.e., the sum of the terms in the integral over a finite number of samples $\{x_i\}_{i=1}^{N_x}$. Then, we use stochastic gradient descent [22, Section 5.9] to update ω and the complexity is independent on N_x.

We employ two important strategies to avoid highly sub-optimal solutions. The first strategy involves multi-start optimization with random initialization of ω [23]. For the second strategy, we refer to it as multi/random gradient descent. When applying a classical stochastic gradient descent algorithm to (12), ω is susceptible to falling into sub-optimal points where many component functions of $\hat{B}(\cdot, \omega)$ remain inactive for all states. The multi/random gradient descent is thereby proposed as follows. For the sample x_i that satisfies $\hat{B}(x_i, \omega) \leq 0$ but $B_T(x_i) > 0$, our algorithm updates all non-negative $\hat{B}_j(\cdot, \omega_j)$. This targeted update can accelerate the learning process. Conversely, for the sample x_i that satisfies $\hat{B}(x_i, \omega) > 0$ but $B_T(x_i) \leq 0$, we randomly select several positive $\hat{B}_j(\cdot, \omega_j)$ to update. This randomized selection prevents the repetition of updating the same component during the learning phase, promoting a more diverse exploration of the parameter space.

It should be pointed out that the proposed classification method cannot guarantee the invariance of $\hat{S}_B$. However, as T increases, the method can approach the maximal controlled invariant set, thereby reducing the conservatism compared to existing methods [2], [14].

C. Learning π_s

After the training of $\hat{B}(\cdot, \omega)$ has been finished, we further learn a safe policy $\hat{\pi}_s(\cdot, \beta)$ with the parameter β. The choice of $\hat{\pi}_s(\cdot, \beta)$ is flexible, and can accommodate various forms such as NNs. According to Assumption 1, $\hat{\pi}_s(\cdot, \beta)$ is expected to satisfy $f_{\text{PWA}}(x, \hat{\pi}_s(x, \beta)) \in \hat{S}_B$ and $\hat{\pi}_s(x, \beta) \in U(x)$ for all $x \in \hat{S}_B$. This prompts us to optimize β to minimize the following objective:

$$\max_{x \in \hat{S}_B} \max \left\{ \hat{B}(f_{\text{PWA}}(x, \hat{\pi}_s(x, \beta), \omega)), \; h_{U(x)}(\hat{\pi}_s(x, \beta)) \right\}.$$

Similar to the proposed method for solving (15), we replace the above objective by an empirical loss and use stochastic gradient descent to update β.

V. Case Study

We consider the lateral and yaw dynamics of a bicycle model [24]:

$$\begin{aligned} m\ddot{y} &= -mv_x\dot{\psi} + 2(F_{y_f} + F_{y_r}) \\ I\ddot{\psi} &= 2(aF_{y_f} - bF_{y_r}), \end{aligned} \tag{16}$$

where $x = [\dot{y} \;\; \dot{\psi}]^T$ is the state, m is the mass, I is the inertia, a and b are the distances from the front and rear wheels to the center of gravity, ψ is the heading angle, v_x is

[2]It is hard to prove that any local optima of (15) is an optimum of (14), unless $B_T(\cdot)$ and $\hat{B}(\cdot, \omega)$ are $\mathbb{C}^1$ functions.

the longitudinal speed, which is assumed to be constant, F_{y_f} and F_{y_r} are front and rear tire lateral forces. Based on the results of [24], F_{y_f} and F_{y_r} can be approximated as PWA functions of $[\dot{y}\ \dot{\psi}\ \delta]^T$, where δ is the steering angle (input). The PWA functions are given by

$$F_{y_f} = \begin{cases} C_2(\frac{\dot{y}+a\dot{\psi}}{v_x} + \alpha^*) - C_1\delta - C_1\alpha^* & \text{for } \frac{\dot{y}+a\dot{\psi}}{v_x} < -\alpha^* \\ C_1(\frac{\dot{y}+a\dot{\psi}}{v_x} - \delta) & \text{for } \frac{\dot{y}+a\dot{\psi}}{v_x} \in [-\alpha^*, \alpha^*] \\ C_2(\frac{\dot{y}+a\dot{\psi}}{v_x} - \alpha^*) - C_1\delta + C_1\alpha^* & \text{for } \frac{\dot{y}+a\dot{\psi}}{v_x} > \alpha^*, \end{cases}$$

$$F_{y_r} = \begin{cases} C_2(\frac{\dot{y}-b\dot{\psi}}{v_x} + \alpha^*) - C_1\alpha^* & \text{for } \frac{\dot{y}-b\dot{\psi}}{v_x} < -\alpha^* \\ C_1\frac{\dot{y}-b\dot{\psi}}{v_x} & \text{for } \frac{\dot{y}-b\dot{\psi}}{v_x} \in [-\alpha^*,\ \alpha^*] \\ C_2(\frac{\dot{y}-b\dot{\psi}}{v_x} - \alpha^*) + C_1\alpha^* & \text{for } \frac{\dot{y}-b\dot{\psi}}{v_x} > \alpha^*, \end{cases}$$

where C_1, C_2 and α^* are constants. After discretizing (16) by Euler Discretization with the sampling time 0.05 s, we get a discrete-time PWA system (1) with 9 polyhedral regions. The system is subject to the input and state constraints: $|\delta| \le 0.17,\ x \in X_1 \cup X_2$ with $X_1 = \{x \mid |\dot{y}| \le 1,\ |\dot{\psi}| \le 1\},\ X_2 = \{x\ |\dot{y}^2 + \dot{\psi}^2 \le 1.2\}$.

The simulations are conducted in MATLAB R2021a on an AMD Core R7-5800H CPU @3.20GHz machine. All MICP problems are solved by Gurobi [25].

After applying the LMI approach, we obtain an ellipsoidal controlled invariant set $S_0 = \{x|\ x^T P x \le 1\}$ with $P = \begin{bmatrix} 4.606 & 0 \\ 0 & 7.597 \end{bmatrix}$. We sample the state space $\{x|\ |\dot{y}| \le 1.5,\ |\dot{\psi}| \le 1.5\}$ by a uniform grid and get 100^2 samples. Then, the problem (11) is solved with $T = 7$ for each state sample. Since the regions near the boundary of S_T are of more interest than the regions away from the boundary, we copy the samples x_i that satisfy $|B_T(x_i)| \le \epsilon$. Here ϵ is a small positive constant. As a result, we get the training data $\{(x_i,\ \text{step}(-B_T(x_i)))\}_{i=1}^{15215}$ for the proposed classifier (12).

Fig. 2 shows the performance of the proposed classifier (12) tested on 1000 samples that are randomly selected from the training samples. We take $N_\text{p} = N_\text{e} = 15$, where N_p (N_e) refers to the number of PWA (quadratic) component functions in (12). The classifier is trained by the proposed multi/random gradient descent. We observe that the proposed classifier can distinguish between the state samples inside and outside S_T with high accuracy. We also compare the multi/random gradient descent with the particle swarm optimization as employed in [26] for $N_\text{e} = 10$ and $N_\text{p} = 0$. The result indicates that the proposed method achieves an impressive 96.51% classification accuracy within a training time of 29 seconds, whereas the particle swarm optimization yields a significantly lower accuracy of 70.32% after 573 seconds of training.

Next, we verify the proposed safety filter. We consider three kinds of learning-based controllers: supervised learning of MPC [27], an approximate-dynamic-programming (ADP) controller [5], and supervised learning of MPC with the safety filter (4) included in the training loop. The learning-based controllers as well as the learned safe policy $\hat{\pi}_\text{s}(\cdot, \beta)$ are represented by NNs with three hidden layers with hyperbolic tangent activation functions. The three layers contain

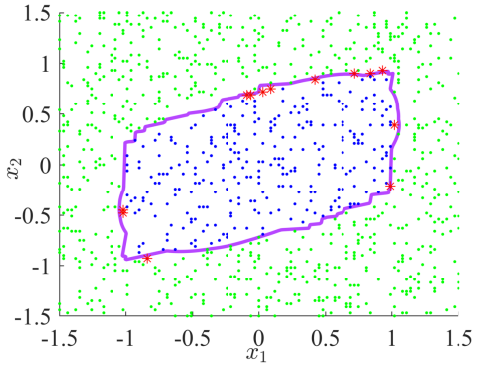

Fig. 2. Performance of the proposed classifier $\hat{B}(\cdot, \omega)$. The blue and green points represent the states within and outside S_T, the purple boundary is $\{x|\hat{B}(x, \omega) = 0\}$, and the red stars represent the states that $\hat{B}(\cdot, \omega)$ misclassifies.

16, 64, and 16 neurons. To improve the safety, $\hat{B}(\cdot, \omega)$ and $\pi_\text{s}(\cdot)$ are trained with 5% state constraint tightening [8].

Table 1 shows the comparative results. The performance metrics include the total cost $\sum_{t=0}^{50} x_t^T x_t + \delta_t^2$, the rate of safety and the average CPU time for generating the input per time step. For each $\hat{S}_B$, we compare three methods for generating safe control inputs. The first one is the proposed optimization-free safe controller (4). The last two ones are based on solving (2) (with S replaced by $\hat{S}_B$). "MIQCP" indicates that (2) is solved by mixed-integer quadratically constrained programming, while "nonlinear" indicates that (2) is solved by the active-set nonlinear programming algorithm, implemented using the MATLAB function "fmincon"[3].

Without safety filters, both learning MPC and ADP show lower safety rates compared to when safety filters are applied, with the proposed method and MIQCP almost achieving a 100% safety rate. The CPU time of computing the proposed filter (4) is less than 0.4 ms, significantly lower than that of solving MIQCP and nonlinear programming problems. Additionally, the proposed filter (4) generally performs competitively with MIQCP in terms of cost. These results suggest that the choice of safety filter and its parameters can significantly impact the safety and computational efficiency of learning MPC and ADP policies, with the proposed filter showing promise for achieving high safety rates at low computational costs. Additionally, when considering (4) in the training loop of learning MPC, the total cost is slightly lower than that of learning MPC without (4) in the training loop. Besides, although the proposed safe filter (4) outputs a suboptimal solution to (2), it does not mean that the policy π^* results in a higher total cost than π_p.

VI. Conclusions and Future Work

This paper has presented a new safety filter, called the optimization-free safety filter, which can enhance safety for PWA systems subject to the combination of polyhedral and ellipsoidal constraints. Compared to standard safety filters

[3]Since both f_PWA and $\hat{B}$ are continuous functions, we can treat them as general continuous nonlinear functions without considering the PWA property of them.

TABLE I

COMPARISON OF SAFETY FILTERS WITH DIFFERENT $\hat{S}_B$ AND DIFFERENT SOLVING ALGORITHMS.

Safety filter	Algorithm	Learning MPC			ADP			Learning MPC with (4) included		
		CPU time (ms)	Cost	Safety rate	CPU time	Cost	Safety rate	CPU time	Cost	Safety rate
No		0.004	4.776	19.53%	0.002	4.728	15.62%			
$N_\mathrm{p}=0$ $N_\mathrm{e}=30$	Proposed	0.209	4.798	100%	0.207	4.728	100%	0.218	4.732	100%
	MIQCP	84.5	4.798	100%	84.7	4.728	100%			
	Nonlinear	8.61	4.798	100%	8.72	4.728	100%			
$N_\mathrm{p}=15$ $N_\mathrm{e}=15$	Proposed	0.339	4.808	100%	0.339	4.738	100%	0.364	4.748	100%
	MIQCP	64.7	4.858	100%	64.4	4.787	100%			
	Nonlinear	8.71	13.07	96.88%	8.43	8.536	96.88%			
$N_\mathrm{p}=5$ $N_\mathrm{e}=5$	Proposed	0.177	4.963	97.66%	0.180	4.954	97.66%	0.164	4.770	100%
	MIQCP	44.2	5.823	97.66%	44.3	5.754	97.66%			
	Nonlinear	8.53	21.33	90.62%	9.20	16.37	90.62%			

that need to solve a usually nonconvex optimization problem online, the proposed safety filter has very small online computational overhead. To construct the safety filter, we have developed a new PWA and PWQ classification approach to learn a controlled invariant set from data. Simulation on a bicycle model with hybrid approximation has demonstrated that the proposed method can achieve zero constraint violations and generate control signals at the millisecond level. Topics for future work include handling more complex constraints, improving the applicability in large-scale PWA systems, and testing the computational advantages in more complex and realistic problems.

APPENDIX

Proof of Proposition 1: Let λ^+ denote the right-hand side of (6). The task is to prove $\lambda^+ = \lambda^*$.

(i) Feasibility of λ^+. Since $\pi_\mathrm{s}(x) \in U(x)$, we have $c \geq 0$. According to (7), we know $\lambda^{(0)} \geq 0$. Meanwhile, as the constraint $\lambda \in [0,1]$ is included in (5b), $\exists j$ such that $\lambda_j^0 = 1$. Therefore, we also have $\lambda^{(0)} \leq 1$. Since there exists at least one $i \in \mathbb{N}_s^+$ such that $f_\mathrm{PWA}(x, \pi_\mathrm{s}(x)) \in S_i$, we have that there exists at least one $i \in \mathbb{N}_s^+$ such that $c^{(i)} \geq 0$. Similarly, we know $\max_{i\in\mathbb{N}_w^+}\{\min(\lambda^{(i)})\} \geq 0$. Therefore, we get $\lambda^+ \geq 0$.

If $\lambda^+ = 0$, by Assumption 1 we obtain the feasibility of λ^+. In the remaining proof of the feasibility, we consider $\lambda^+ > 0$. Firstly, we consider the case when $a_j = 0$. This case could occur when $U(x)$ is polyhedral, or when $U(x)$ is ellipsoidal and $\pi_\mathrm{r}(x) = \pi_\mathrm{s}(x)$. If $b_j > 0$, $b_j\lambda^+ \leq b_j c_j / b_j \leq c_j$. If $b_j \leq 0$, $b_j\lambda^+ \leq 0 \leq c_j$. Then, we consider the case when $a_j > 0$, i.e., when $U(x)$ is ellipsoidal and $\pi_\mathrm{r}(x) \neq \pi_\mathrm{s}(x)$. As $\pi_\mathrm{s}(x) \in U(x)$, the quadratic equation $a_j\lambda^2 + b_j\lambda - c_j = 0$ has one non-negative root $\lambda_j^{(0)}$ and one non-positive root. As $\lambda^+ \in (0, \lambda_j^{(0)}]$, it follows that $a_j\lambda^{+2} + b_j\lambda^+ - c_j \leq 0$. Applying the above argument to all the element of a leads to the feasibility of λ^+ regarding (5b).

Let $i' = \arg\max_{i\in\mathbb{N}_w^+}\{\min(\lambda^{(i)})\}$. Next, we prove the feasibility of λ^+ regarding (5c) by verifying (5c) for $i = i'$. For the jth element $a_j^{(i')}$ of $a^{(i')}$, we define $f_j^{(i')}(\lambda) \triangleq a_j^{(i')}\lambda^2 + b_j^{(i')}\lambda - c_j^{(i')}$. We consider the five cases in (8).

Cases (1&2): $a_j^{(i')} = 0$ and $b_j^{(i')} \geq 0$. In these two cases, if $c_j^{(i')} < 0$, we have $\lambda^+ = \lambda_j^{(i')} = 0$, which is certainly feasible for problem (3) because of Assumption 1. If $c_j^{(i')} \geq 0$, as $f_j^{(i')}(\cdot)$ is monotonically non-decreasing, we have $f_j^{(i')}(\lambda^+) \leq f_j^{(i')}(\lambda_j^{(i')}) \leq 0$.

Case (3): $a_j^{(i')} = 0$ and $b_j^{(i')} < 0$. In this case, If $\min(\lambda^{(i')}) = 0$, we have $\lambda^+ = 0$ and the feasibility of λ^+. If $\min(\lambda^{(i')}) > 0$, it follows from (8) that

$$\kappa_j^{(i)} \triangleq c_j^{(i)} - b_j^{(i)} \min\left\{\min(\lambda^{(0)}), \min_{k \text{ s.t. } \lambda_k^{(i)}\in(0,1)} \lambda_k^{(i)}\right\} \geq 0 \text{ for } i = i'. \tag{17}$$

If $\min(\lambda^{(i')}) = 1$, we get $f_j^{(i')}(\lambda^+) = b_j^{(i')}\min(\lambda^{(0)}) - c_j^{(i')} \leq 0$ because of (17). If $\min(\lambda^{(i')}) < 1$, there must exist $k \in \mathbb{N}^+$ such that $\min(\lambda^{(i')}) = \lambda_k^{(i')} = c_k^{(i')}/b_k^{(i')}$, and consequently, $f_j^{(i')}(\lambda^+) = b_j^{(i')}\min\left\{\min(\lambda^{(0)}), \lambda_k^{(i')}\right\} - c_j^{(i')} \leq 0$ thanks to (17).

Case (4): $a_j^{(i')} > 0$ and $\delta_j^{(i')} < 0$. We have $\lambda^+ = 0$ and thus the feasibility of λ^+.

Case (5): $a_j^{(i')} > 0$ and $\delta_j^{(i')} \geq 0$. Similarly to Case (3), we only need to consider the case when

$$\tau_j^{(i)} \triangleq 2a_j^{(i')}\min(\lambda^{(0)}) + b_j^{(i)} + \sqrt{\delta_j^{(i)}} \geq 0 \text{ for } i = i', \tag{18}$$

because otherwise we have $\lambda^+ = 0$. The condition (18) implies that the smaller root of $f_j^{(i')}(\lambda) = 0$ is not larger than $\min(\lambda^{(0)})$, and that the larger root is $\lambda_j^{(i')}$. Note that for ellipsoidal $E_{i'}$, $j = 1$. Therefore, we have $f_j^{(i')}(\lambda^+) = f_j^{(i')}(\min\{\min(\lambda^{(0)}),\ \lambda_j^{(i')}\}) \leq 0$.

Combining the above 5 cases, we know that $f_j^{(i')}(\lambda^+) \leq 0$, which proves the feasibility of λ^+ w.r.t. (5c).

(ii) Optimality of λ^+. If $\lambda^+ = 1$, the optimality directly follows from the constraint $\lambda \leq 1$. In the remaining proof, we consider $\lambda^+ < 1$, and prove that $\lambda^+ + \Delta_\lambda$ is infeasible for problem (5) for any $\Delta_\lambda \in (0, 1-\lambda^+]$.

Case (1): $\lambda^+ = \min(\lambda^{(0)}) = 0$. In this case, $\exists j$ such that either $a_j = c_j = 0,\ b_j > 0$ or $a_j > 0,\ c_j = 0$. This implies that $\pi_\mathrm{s}(x) \in \partial U(x)$ and $\pi_\mathrm{r}(x) \notin U(x)$. As $U(x)$ is convex, 0 is the only feasible point and thus optimal for (3c).

Case (2): $\lambda^+ = \min(\lambda^{(0)}) \in (0,1)$. In this case, $\pi_\mathrm{s}(x)$ is in the interior of $U(x)$, and $\pi_\mathrm{r}(x) \notin U(x)$. From (7), we see

that $\min(\lambda^{(0)})$ represents the distance from $\pi_{\mathrm{s}}(x)$ to $\partial U(x)$. As $U(x)$ is convex, λ^+ is the optimal solution to (5).

Case (3): $\lambda^+ = \max_{i\in\mathbb{N}_w^+}\{\min(\lambda^{(i)})\} \in (0,1)$. In this case, $\forall i \in \mathbb{N}_w^+$, there exists j such that $\lambda_j^{(i)} \leq \lambda^+$. Then, we can distinguish five sub-cases based on (8):

- If $a_j^{(i)} = 0,\ b_j^{(i)} > 0$, then $f_j^{(i)}(\lambda^+ + \Delta_\lambda) \geq f_j^{(i)}(\lambda_j^{(i)} + \Delta_\lambda) \geq b_j^{(i)}\Delta_\lambda > 0$. This means $\lambda^+ + \Delta_\lambda$ violates the jth inequality of (5c).
- If $a_j^{(i)} = b_j^{(i)} = 0,\ c_j^{(i)} < 0$ or if $a_j^{(i)} > 0,\ \delta_j^{(i)} < 0$, we have $f_j^{(i)}(\lambda) = a_j^{(i)}\lambda^2 + b_j^{(i)}\lambda - c_j^{(i)} > 0$ holds for any $\lambda \in \mathbb{R}$, which means $\lambda^+ + \Delta_\lambda$ violates the jth inequality of (5c).
- If $a_j^{(i)} = 0,\ b_j^{(i)} < 0$ and $\kappa_j^{(i)} < 0$, then, for any $\Delta_\lambda \in (0, \min\{\min(\lambda^{(0)}), \min_{k \text{ s.t. } \lambda_k^{(i)}\in(0,1)} \lambda_k^{(i)}\} - \lambda^+]$, we have $f_j^{(i)}(\lambda^+ + \Delta_\lambda) = b_j^{(i)}(\lambda^+ + \Delta_\lambda) - c_j^{(i)} \geq -\kappa_j^{(i)} > 0$. For any $\Delta_\lambda > \min\big\{\min(\lambda^{(0)}), \min_{k \text{ s.t. } \lambda_k^{(i)}\in(0,1)} \lambda_k^{(i)}\} - \lambda^+$, we have either $\lambda^+ + \Delta_\lambda > \min(\lambda^{(0)})$ or $\lambda^+ + \Delta_\lambda > c_k^{(i)}/b_k^{(i)}$ for a k s.t. $\lambda_k^{(i)} \in (0,1)$. If $\lambda^+ + \Delta_\lambda > \lambda^{(0)}$, it is clear that $\lambda^+ + \Delta_\lambda$ violates the constraint (5b). If $\lambda^+ + \Delta_\lambda > c_k^{(i)}/b_k^{(i)}$, we have $f_k^{(i)}(\lambda^+ + \Delta_\lambda) = b_k^{(i)}(\lambda^+ + \Delta_\lambda) - c_k^{(i)} > 0$, which means $\lambda^+ + \Delta_\lambda$ violates the kth inequality of (5c). Therefore, any increase $\Delta_\lambda \in (0, 1 - \lambda^+]$ for λ^+ is infeasible for (5).
- If $a_j^{(i)} > 0,\ \delta_j^{(i)} \geq 0,\ \tau_j^{(i)} < 0$, then the smaller root of the equation $f_j^{(i)}(\lambda) = 0$, denoted by λ^-, is strictly larger than $\min\left(\lambda^{(0)}\right)$. As a result, if $\lambda^+ + \Delta_\lambda < \lambda^-$, we have $f_j^{(i)}(\lambda^+ + \Delta_\lambda) > 0$, i.e., $\lambda^+ + \Delta_\lambda$ is infeasible for (5c). Otherwise, we have $\lambda^+ + \Delta_\lambda \geq \lambda^- > \min\left(\lambda^{(0)}\right)$, which means $\lambda^+ + \Delta_\lambda$ is infeasible for (5b).
- If $a_j^{(i)} > 0,\ \delta_j^{(i)} \geq 0,\ \tau_j^{(i)} \geq 0$, then, $\lambda_j^{(i)}$ is the larger root of the equation $f_j^{(i)}(\lambda) = 0$, so we have $f_j^{(i)}(\lambda^+ + \Delta_\lambda) > f_j^{(i)}(\lambda_j^{(i)}) > 0$ holds for any $\Delta_\lambda \in (0, 1 - \lambda^+]$.

Combining the above five sub-cases and noticing the arbitrariness of i and the existence of j and k, we can conclude that $\lambda^+ + \Delta_\lambda$ is infeasible for problem (5) in the case (3).

REFERENCES

[1] F. Borrelli, A. Bemporad, and M. Morari, *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press, 2017.

[2] M. Lazar, M. Heemels, S. Weiland, and A. Bemporad, "Stabilizing model predictive control of hybrid systems," *IEEE Transactions on Automatic Control*, vol. 51, no. 11, pp. 1813–1818, 2006.

[3] T. Marcucci, R. Deits, M. Gabiccini, A. Bicchi, and R. Tedrake, "Approximate hybrid model predictive control for multi-contact push recovery in complex environments," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, 2017, pp. 31–38.

[4] T. Liu, Y. Gao, and M. Buss, "Adaptive output tracking control of piecewise affine systems with prescribed performance," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 9, pp. 5398–5410, 2021.

[5] K. He, S. Shi, T. v. d. Boom, and B. De Schutter, "Approximate dynamic programming for constrained piecewise affine systems with stability and safety guarantees," *arXiv preprint arXiv:2306.15723*, 2023.

[6] Y. Li, N. Li, H. E. Tseng, A. Girard, D. Filev, and I. Kolmanovsky, "Robust action governor for uncertain piecewise affine systems with non-convex constraints and safe reinforcement learning," *arXiv preprint arXiv:2207.08240*, 2022.

[7] W. Xiao, T.-H. Wang, R. Hasani, M. Chahine, A. Amini, X. Li, and D. Rus, "Barriernet: Differentiable control barrier functions for learning of safe robot control," *IEEE Transactions on Robotics*, vol. 29, no. 3, pp. 2289–2307, 2023.

[8] K. He, S. Shi, T. v. d. Boom, and B. De Schutter, "State-action control barrier functions: Imposing safety on learning-based control with low online computational costs," *arXiv preprint arXiv:2312.11255*, 2023.

[9] T. Marcucci and R. Tedrake, "Warm start of mixed-integer programs for model predictive control of hybrid systems," *IEEE Transactions on Automatic Control*, vol. 66, no. 6, pp. 2433–2448, 2020.

[10] D. Masti and A. Bemporad, "Learning binary warm starts for multiparametric mixed-integer quadratic programming," in *2019 18th European Control Conference (ECC)*, 2019, pp. 1494–1499.

[11] L. Zheng, Y. Shi, L. J. Ratliff, and B. Zhang, "Safe reinforcement learning of control-affine systems with vertex networks," in *Learning for Dynamics and Control*. PMLR, 2021, pp. 336–347.

[12] X. Zhao and Q. Xu, "Explicit reinforcement learning safety layer for computationally efficient inverter-based voltage regulation," in *2023 American Control Conference (ACC)*, 2023, pp. 4501–4506.

[13] J. Tordesillas, J. P. How, and M. Hutter, "RAYEN: Imposition of hard convex constraints on neural networks," *arXiv preprint arXiv:2307.08336*, 2023.

[14] S. Sadraddini and R. Tedrake, "Sampling-based polytopic trees for approximate optimal control of piecewise affine systems," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 7690–7696.

[15] C. Dawson, S. Gao, and C. Fan, "Safe control with learned certificates: A survey of neural Lyapunov, barrier, and contraction methods for robotics and control," *IEEE Transactions on Robotics*, 2023.

[16] B. De Schutter and T. J. van den Boom, "MPC for continuous piecewise-affine systems," *Systems & Control Letters*, vol. 52, no. 3-4, pp. 179–192, 2004.

[17] S. Chen and M. Fazlyab, "Learning performance-oriented control barrier functions under complex safety constraints and limited actuation," *arXiv preprint arXiv:2401.05629*, 2024.

[18] S. Bansal and C. J. Tomlin, "Deepreach: A deep learning approach to high-dimensional reachability," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 1817–1824.

[19] A. D. Bonzanini, J. A. Paulson, G. Makrygiorgos, and A. Mesbah, "Scalable estimation of invariant sets for mixed-integer nonlinear systems using active deep learning," in *2022 IEEE 61st Conference on Decision and Control (CDC)*, 2022, pp. 3431–3437.

[20] M. Srinivasan, A. Dabholkar, S. Coogan, and P. A. Vela, "Synthesis of control barrier functions using a supervised machine learning approach," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 7139–7145.

[21] H. Dai, B. Landry, L. Yang, M. Pavone, and R. Tedrake, "Lyapunov-stable neural-network control," *arXiv preprint arXiv:2109.14152*, 2021.

[22] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[23] A. Rinnooy Kan and G. Timmer, "Stochastic global optimization methods part II: Multi level methods," *Mathematical Programming*, vol. 39, pp. 57–78, 1987.

[24] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat, "Predictive active steering control for autonomous vehicle systems," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 3, pp. 566–580, 2007.

[25] L. Gurobi Optimization, "Gurobi optimizer reference manual," 2018.

[26] L. Gharavi, B. De Schutter, and S. Baldi, "Efficient MPC for emergency evasive maneuvers, part i: Hybridization of the nonlinear problem," *arXiv preprint arXiv:2310.00715*, 2023.

[27] B. Karg and S. Lucia, "Efficient representation and approximation of model predictive control laws via deep learning," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3866–3878, 2020.