State-Action Control Barrier Functions: Imposing Safety on Learning-Based Control With Low Online Computational Costs

Kanghui He, *Graduate Student Member, IEEE*, Shengling Shi, *Member, IEEE*, Ton van den Boom, and Bart De Schutter, *Fellow, IEEE*

***Abstract*—Learning-based control with safety guarantees usually requires real-time safety certification and modifications of possibly unsafe learning-based policies. The control barrier function (CBF) method uses a safety filter (SF) containing a constrained optimization problem to produce safe policies. However, finding a valid CBF for a general nonlinear system requires a complex function parameterization, which in general makes the policy optimization problem difficult to solve in real time. For nonlinear systems with nonlinear state constraints, this article proposes the novel concept of state-action CBFs (SACBFs), which do not only characterize the safety at each state but also evaluate the control inputs taken at each state. SACBFs, in contrast to CBFs, enable a flexible parameterization, resulting in an SF that involves a convex quadratic optimization problem, which significantly alleviates the online computational burden. We propose a learning-based approach to synthesize SACBFs. The effect of learning errors on the effectiveness of SACBFs is addressed by constraint tightening and introducing a new concept called contractive-set CBFs. This ensures formal safety guarantees for the learned CBFs and control policies. Simulation results on an inverted pendulum with elastic walls validate the proposed CBFs in terms of constraint satisfaction and CPU time.**

***Index Terms*—Constrained control, control barrier functions (CBFs), machine learning, nonlinear control.**

I. Introduction

Learning-based control methods have demonstrated extensive success in control systems. However, learning-based controllers may provide unsafe control actions that result in undesirable or even destructive effects on the system. In control systems, safety means that the trajectories of the closed-loop system should satisfy state and input constraints for the entirety of the system's evolution. There has been an increasing interest in designing controllers for safety-critical systems. One notable approach is using supervised learning to approximate model predictive control (MPC) policies. While traditional online MPC can be computationally intensive for nonlinear constrained systems [1], recent work has demonstrated that neural networks (NNs) trained to mimic MPC behavior can achieve satisfactory performance in both safety and real-time efficiency [2]. However, approximate MPC typically relies on expert data and lacks adaptability to new scenarios, whereas reinforcement learning (RL) allows agents to explore and learn optimal control strategies through interaction with the environment, making it more flexible and scalable for complex tasks.

Safe reinforcement learning (safe RL) is a subfield of RL focusing on ensuring safety during or after the learning phase. Techniques in safe RL generally fall into two main categories: constrained policy optimization during learning and policy refinement after learning. Constrained policy optimization during learning uses penalties [3], multiplier methods [4], or constraint elimination [5] when training the RL policy. These approaches are often straightforward to implement and compatible with standard RL algorithms. However, they typically involve a tradeoff between safety and optimal performance. Policy refinement after learning uses a safety filter (SF) accompanied by a safety certificate to modify unsafe control actions. Examples include invariant sets [6] and control barrier functions (CBFs) [7]. While policy refinement can provide formal safety guarantees, it usually requires prior knowledge on safety certificates, which can be difficult to obtain [8], due to the nonconvex and nonlinear nature of the underlying problem.

Among policy refinement methods, CBFs offer a particularly attractive solution for enforcing safety for nonlinear systems. Similar to energy-based functions, CBFs define sublevel sets that represent safe regions of the state space and they can be used to construct SFs. CBFs provide a theoretically sound way to ensure that long-term safety constraints are respected by a single CBF constraint. SFs adjust control inputs in real time to enforce compliance with the CBF constraint. In general, SFs are more online computationally efficient than some other safe controllers (such as MPC) that need long-term prediction. In addition to CBF-based SFs, there are also predictive SFs [8] and SFs that involve projection onto invariant sets [9]. In this article, we will use the term "CBF-SF" specifically to refer to a "CBF-based SF."

In the realm of CBF-based methodologies, even though the state and input constraints are known, there is a lack of universally applicable methods for generating valid CBFs, necessitating reliance on manually designed or problem-specific CBFs. For some particular systems such as linear, piecewise affine [10], or polynomial systems [11], CBFs can be computed by solving convex optimization problems. For nonlinear systems with general constraints, using learning-based algorithms accompanied by advanced function approximators has been explored in several contexts [12], [13], [14], [15].

No matter how the CBF is learned, to control a given system with safety guarantees, it is inevitable to solve an online optimization problem with CBF-based constraints, which is usually nonconvex for discrete-time nonlinear systems [16]. To obtain a satisfactory approximation accuracy, sophisticated function approximators such as deep NNs are commonly used to parameterize the CBF [7], [15]. However, this will inherently cause nonconvexity and increase the complexity of the optimization problem. As a result, the increased online computational load makes the CBF-based approach unsuitable for situations where fast computation of control inputs is required. Besides, the effect of approximation errors on the validity of the learned CBFs and the resulting control policies has not yet been

Received 21 October 2024; revised 1 June 2025; accepted 16 November 2025. Date of publication 24 November 2025; date of current version 28 April 2026. This work was supported by a project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under Grant 101018826 - CLariNet. Recommended by Associate Editor J. Zhou. *(Corresponding author: Shengling Shi.)*

The authors are with the Delft Center for Systems and Control, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: k.he@tudelft.nl; shengling.shi@tudelft.nl.; a.j.j.vandenBoom@tudelft.nl; b.deschutter@tudelft.nl).

Digital Object Identifier 10.1109/TAC.2025.3636804

addressed [7]. Note that Choi et al. [14] introduce control barrier-value functions by unifying Hamilton–Jacobi (HJ) reachability and CBFs to address issues of finding valid CBFs. However, Choi et al. [14] focus on theoretical guarantees of robust safety to disturbances but does not address learning errors.

This article has the following main contributions.

1) *Computationally efficient safety enforcement using state-action CBFs (SACBFs):* We introduce the novel concept of SACBFs. Unlike standard CBFs [14], [16], which are often computationally intensive when ensuring safety in nonlinear systems, SACBFs provide a flexible parametrization. The flexibility lies in the ability to enforce safety constraints through a convex quadratic optimization problem, significantly reducing the computational burden during online operations.
2) *Handling approximation errors in learned CBFs:* We propose a constraint tightening approach alongside the novel concept of contractive-set CBFs to address approximation errors in learned CBFs. This ensures that the invariance property of CBFs is maintained even when approximation errors occur, which is not yet achieved in existing work on learning CBFs [12], [14], [15], [17]. In addition, we examine the relationship between SACBFs and contractive-set CBFs, and develop a new learning-based method to approximate SACBFs. This ensures the safety of the policy filtered by SACBFs, provided the approximation error is sufficiently small. Besides, we discuss the tradeoff between online computational efficiency and safety when learning the proposed SACBFs.

II. Preliminaries and Problem Formulation

A. Preliminaries

We consider a deterministic discrete-time nonlinear system

$$x_{t+1} = f(x_t, u_t), \quad t = 0, 1, \dots \tag{1}$$

where $x_t \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$ and $u_t \in \mathcal{U} \subseteq \mathbb{R}^{n_u}$ are the state and the input at time step t, and $f(\cdot,\cdot) : \mathcal{X} \times \mathcal{U} \to \mathcal{X}$ is a continuous function satisfying $f(0,0) = 0$. We consider a constrained optimal control problem in which the states and inputs should satisfy time-invariant constraints: $x_t \in X := \{x \in \mathcal{X} | h(x) \leq 0\}$ and $u_t \in U \subseteq \mathcal{U}$. Here, $h(\cdot) : \mathcal{X} \to \mathbb{R}$ is a continuous function that defines the state constraint.[1] In most parts of the article, we assume that f is fully known. However, in Section V-C, we briefly discuss the applicability of our method when there is system uncertainty. Besides, we assume that X is compact and that U is a polytope.

The control objective is to regulate a predefined control policy, which could be a policy learned through various methods such as RL [18], learning-based MPC [9], or any other control technique that needs safety guarantees. The primary concern is ensuring constraint satisfaction *after* learning, rather than safe exploration [19], which is not the focus here.

B. Control Barrier Functions

To achieve the objective, we need to design a control policy that can ensure constraint satisfaction at all time steps. A set of states for which such a policy exists, needs to be defined. Usually, this class of sets is called controlled-invariant sets or safe sets. For high-dimensional systems, however, some controlled-invariant sets could have complex representations, making the controller synthesis difficult. A CBF uses the sublevel set of it to conveniently define the safe set.

Definition 1 (CBF): A continuous function $B(\cdot) : \mathcal{X} \to \mathbb{R}$ is called a CBF with a corresponding safe set $\mathcal{S}_B := \{x \in \mathbb{R}^{n_x} | B(x) \leq 0\} \subseteq \mathcal{X}$, if $\mathcal{S}_B$ is nonempty

$$h(x) \leq 0 \ \forall x \in \mathcal{S}_B, \text{ and} \tag{2}$$

$$\forall x \in \mathcal{S}_B, \ \exists u \in U \text{ s.t. } B(f(x,u)) \leq 0. \tag{3}$$

Furthermore, a CBF is called an exponential CBF if $h(x) \leq B(x) \ \forall x \in \mathcal{S}_B$ and if there exists a $\beta \in (0,1)$ such that

$$\forall x \in \mathcal{S}_B, \ \exists u \in U \text{ s.t. } B(f(x,u)) \leq \beta B(x). \tag{4}$$

Condition (2) is equivalent to $\mathcal{S}_B \subseteq X$, which is usually assumed in literature [16]. With a CBF available, one can generate a safe control policy in $\mathcal{S}_B$ by using the following optimization-based approach:

$$\begin{aligned} \pi_{\text{safe}}(x) = \arg\min_{u \in U} \ & ||u - \pi_0(x)||_2 \\ \text{s.t. } & B(f(x,u)) \leq 0, \text{if } B \text{ is a CBF (not exponential)} \\ \text{or } & B(f(x,u)) \leq \beta B(x), \text{if } B \text{ is an exponential CBF} \end{aligned} \tag{5}$$

which serves as a CBF-SF [6] for any unsafe policy π_0.

Definition 2: A policy $\pi(\cdot) : \mathcal{X} \to \mathcal{U}$ is safe in $\mathcal{S} \subseteq X$ for system (1) under the state and input constraints $x \in X, \ u \in U$, if $\pi(x) \in U \ \forall x \in \mathcal{S}$, and for any initial state in $\mathcal{S}$, the state trajectory of (1) steered by π will always stay in $\mathcal{S}$.

For general nonlinear systems with state and input constraints, synthesizing a nonconservative CBF is a difficult task. Hand-crafted or application-specific heuristics are mostly used in literature to design candidate CBFs, which can be either unsafe or overly conservative (see [20, Fig. 1] for an illustrative example). To deal with this issue, using advanced function approximators to learn a CBF certificate has received much attention (see [7] for a comprehensive survey).

C. Limitations of Existing CBFs

1) Problem P1: High Online Computational Complexity: One main limitation of (5) is that it needs to solve a usually nonconvex optimization problem in real time. If a CBF is represented by a complex function approximator $B_\theta(\cdot)$ with the parameter θ, solving (5) may take much online computation time and result in very suboptimal solutions. Even if a convex B_θ is formed, the constraint in (5) could be nonconvex due to the nonlinearity of f.

2) Problem P2: Effects of Approximation Errors: Another limitation of (5) is that the approximation error of $B(\cdot)$ may affect the safety of the system controlled by the optimizer of (5) with $B(\cdot)$ replaced by $B_\theta(\cdot)$. Besides, the recursive feasibility of (5) is also not guaranteed with $B_\theta(\cdot)$.

III. State-Action Control Barrier Function (SACBF)

To deal with Problem P1, motivated by Q-learning in RL [18], we propose a novel SF (SACBF-SF) in the following form:

$$\pi_{\text{safe}}(x) = \arg\min_{u \in U} \{||u - \pi_0(x)||_2, \text{ s.t. } Q(x,u) \leq 0\} \tag{6}$$

[1] For the constraint defined by multiple inequalities $h_i(x) \leq 0, \ i = 1, 2, \dots, I$, we can let $h(x) = \max_{i \in \{1,\dots,I\}} h_i(x)$. The set $\{x | h(x) \leq 0\}$ is then identical to $\{x | h_i(x) \leq 0, \ i = 1, 2, \dots, I\}$, and h will be continuous if each h_i is continuous.

where $Q(\cdot,\cdot): \mathcal{X} \times U \to \mathbb{R}$ is a function of states and actions. We will analyze how to enforce the safety of π_{safe} by imposing conditions on Q. To achieve this, we introduce the definition of SACBFs as follows.

Definition 3 (SACBF): A continuous function $Q(\cdot,\cdot): \mathcal{X} \times U \to \mathbb{R}$ is called an SACBF with a corresponding safe set $\mathcal{S}_Q$, if the pair $(Q,\ \mathcal{S}_Q)$ satisfies

(i) $\mathcal{S}_Q$ is non-empty, and $h(x) \le 0\ \forall x \in \mathcal{S}_Q$;

(ii) $\forall x \in \mathcal{S}_Q,\ \exists u \in U$ s.t. $Q(x,u) \le 0$;

(iii) for any $x \in \mathcal{S}_Q$, any $u \in U$ that satisfies $Q(x,u) \le 0$ will make $f(x,u) \in \mathcal{S}_Q$.

Unlike the definition of standard CBFs, we do not prescribe the form of $\mathcal{S}_Q$ based solely on Q. Besides, condition (i) imposes $\mathcal{S}_Q \subseteq X$. The following lemma builds the connection between standard CBFs and SACBFs, and provides an explicit form of $\mathcal{S}_Q$ when the SACBF is derived from a standard CBF.

Lemma 1: (i) For the SACBF-SF (6), any SACBF Q will render π_{safe} safe in $\mathcal{S}_Q$, i.e., $\mathcal{S}_Q$ is control-invariant.

(ii) If B is a CBF, $Q(\cdot,\cdot) = B(f(\cdot,\cdot))$ will be an SACBF with the safe set $\mathcal{S}_B$, i.e., $\mathcal{S}_Q = \mathcal{S}_B$.

Proof: (i) Based on the condition (ii) of Definition 3, for any $x_0 \in \mathcal{S}_Q$, there exists a $u_0 \in U$ such that $Q(x,u) \le 0$. This means that problem (6) is feasible when $x = x_0$. As $Q(x_0,u_0) \le 0$ implies $f(x_0,u_0) \in \mathcal{S}_Q$, the control-invariance of $\mathcal{S}_Q$ follows. Consequently, problem (6) is recursively feasible for the initial state x_0. Due to the arbitrariness of x_0, π_{safe} is safe according to Definition 3.

(ii) By specifying $Q(\cdot,\cdot) = B(f(\cdot,\cdot))$ and $\mathcal{S}_Q = \mathcal{S}_B$, we have that $\mathcal{S}_Q$ and Q satisfy the conditions (i) and (ii) of Definition 3. Furthermore, for any $x_0 \in \mathcal{S}_Q$, consider any u_0 such that $Q(x_0,u_0) \le 0$. We have $B(f(x_0,u_0)) \le 0$, which further implies $f(x_0,u_0) \in \mathcal{S}_Q$. In addition, as B and f are continuous, Q is also continuous. □

Similarly to designing CBF, the challenge of using (6) is that the explicit form of an SACBF cannot be directly obtained based on Definition 3. We will present a learning-based approach to synthesize SACBFs in Section V.

The advantage of directly learning Q over learning B is that we can design a specific structure for the approximation Q_θ of Q to simplify the constraint in (6) so that the online computational cost of solving (6) is reduced. To achieve this, we can specify the following parameterization:

$$Q_\theta(x,u) = q_{1,\theta}(x) + q_{2,\theta}(x)u + u^T Q_{3,\theta}(x)u,\ Q_{3,\theta} \succeq 0 \tag{7}$$

which will make problem (6) a convex QP with linear and convex quadratic constraints. In (7), $q_{1,\theta}(\cdot): \mathbb{R}^{n_x} \to \mathbb{R}$, $q_{2,\theta}(\cdot): \mathbb{R}^{n_x} \to \mathbb{R}^{n_u}$ and $Q_{3,\theta}(\cdot): \mathbb{R}^{n_x} \to \mathbb{R}^{n_u \times n_u}$ are parameterized functions with all parameters condensed in θ.

Our proposed concept of SACBFs is inspired by the definition of state-action (Q) value functions in RL [18]. Both kinds of functions not only characterize the energy of each state but also quantify the quality of taking each action in each state. In Section V, we will demonstrate that the Q value function for a specific finite-horizon optimal control problem is an SACBF.

IV. Constraint Tightening and Contractive-Set CBFs

To cope with Problem P2, we propose a method to compute conservative CBFs based on state constraint tightening. In particular, we consider two kinds of conservative CBFs. The first kind includes the CBFs of system (1) under the tightened state constraint $X_\lambda := \{x \in \mathcal{X} | h(x) + \lambda \le 0\}$ where λ is a positive constraint backoff. In this situation, the CBFs still satisfy (3) or (4), while condition (2) is tightened to $h(x) + \lambda \le B(x)\ \forall x \in \mathcal{S}_B$. The second one, called the contractive-set CBF, is defined as follows.

Definition 4 (Contractive-set CBF): A CBF $B(\cdot): \mathcal{X} \to \mathbb{R}$ is a λ-contractive-set CBF[2] if there exists a $\lambda > 0$ such that

$$\min_{u \in U} B(f(x,u)) \le -\lambda\ \ \forall x \in \mathcal{S}_B. \tag{8}$$

Similarly, an exponential CBF $B(\cdot): \mathcal{X} \to \mathbb{R}$ is a λ-contractive-set exponential CBF if there exist $\lambda > 0$ and $\beta \in (0,1)$ such that

$$\min_{u \in U} B(f(x,u)) \le \beta B(x) - \lambda\ \ \forall x \in \mathcal{S}_B. \tag{9}$$

The definition of contractive-set CBFs is introduced to endow the safe set $\mathcal{S}_B$ with a contractive property. It enforces an energy decay when the state is on the boundary of the safe set. In particular, if $B(\cdot)$ is a λ-contractive-set CBF, for any $x \in \partial\mathcal{S}_B$, there exists an input $u \in U$ such that $B(f(x,u)) \le -\lambda$, which means that $x^+ = f(x,u)$ is in the interior of $\mathcal{S}_B$. Such a property can guarantee that any approximation of $B(\cdot)$ is still a valid CBF if the approximation error is sufficiently small.

V. Approximating SACBFs

In this section, we propose a method for approximating the SACBF. Inspired by HJ reachability [14] and predictive CBF [15], we propose a comprehensive framework that uses the optimal value functions of a sequence of optimization problems to implicitly represent the CBF sequence $\{B_k\}_{k=1}^{\infty}$. By tuning some parameters, this framework can compute standard CBFs, exponential CBFs, as well as contractive-set CBFs. Although the exact expression of each B_k is in general intractable to compute, we can use learning-based approaches such as supervised learning and RL [18], and then, use the results of Lemma 1 to approximate SACBFs.

A. CBF Generator Based on Reachability Analysis

We consider each B_k as the value function of the following optimization problem:

$$\begin{aligned} B_k(x) := \min_{\{x_t,u_t\}_{t=0}^{k}} \max\left\{ \max_{t \in \mathcal{I}(k-1)} \alpha^t (h(x_t) + \lambda_t), \alpha^k B_0(x_k) \right\} \\ \text{s.t. } x_{t+1} = f(x_t,u_t),\ \ u_t \in U,\ t \in \mathcal{I}(k-1) \\ x_0 = x \end{aligned} \tag{10}$$

where $\mathcal{I}(k-1) = \{0,1,\ldots,k-1\}$, $\alpha \ge 1$, and B_0 is a CBF, which, however, could have a very small safe set. We take $\alpha = 1$ if the CBF B_0 is not an exponential CBF, or $\alpha = 1/\beta$ if B_0 is an exponential CBF. To solve (10), we need to know the explicit formulation of B_0. An LMI-based method that can compute a local quadratic B_0 for the nonlinear system (1) is reported in [22]. In [23, Appendix D], this method is extended to our situations where additional conditions such as (8), (9), (11), and (12) in the subsequent Theorem 1 are required. The LMI method to obtain B_0 relies on the linearization of both the system and the constraints. The resulting B_0 is in general very conservative. In contrast, by introducing the reachability problem (10), we get less conservative CBFs B_k, as stated in Theorem 1.

In (10), $\lambda_t \ge 0,\ t \in \mathcal{I}(k-1)$ are tuning parameters. We consider the following three options for choosing λ_t: *Option 1:* $\lambda_t = 0$; *Option 2:* $\lambda_t = \lambda$, where $\lambda > 0$ is a constant; and *Option 3:* $\lambda_t = t\lambda$, where $\lambda > 0$ is a constant.

[2]Note that the definition of contractive-set CBFs given here is not consistent with the common definition of contraction mappings [18]. Formally speaking, the CBF we define should be called a CBF with a contractive safe set [21]. However, for compactness, we adopt "contractive-set CBF" in the article.

Option 1 means that we are constructing CBFs for the system (1) under the original state constraints. If Option 2 is chosen, it is seen from (10) that we increase the function h to $h+\lambda$, i.e., we tighten the original state constraints to $x \in X_\lambda = \{x \in \mathcal{X} | h(x)+\lambda \leq 0\}$. More conservatively, in Option 3, we require a linear decrease rate for h w.r.t. the time step to construct contractive-set CBFs.

In the original HJ reachability analysis [14], $B_0(\cdot)$ is chosen as $h(\cdot)$, which is not a CBF. It has been proven in [14] that only when $k=\infty$, B_∞ is a CBF. Although this result shows a strong connection between CBFs and HJ reachability, it cannot be used in practice since we cannot solve (10) with $k=\infty$. In our case, we require the initial function B_0 to be a CBF. As a result, any $B_k,\ k=1,2,\ldots$ will become a CBF, with a nonshrinking safe set as k increases.

Theorem 1: Consider B_k from (10). Suppose that B_0 is a (exponential) CBF for (1) with the state constraint $x \in X$.

(i) Let $\lambda_t = 0,\ t \in \mathcal{I}(k-1)$. Then, $B_k,\ k=1,2,\ldots$ is a (exponential) CBF for (1) with the state constraint $x \in X$.

(ii) Let $\lambda_t = \lambda > 0,\ t \in \mathcal{I}(k-1)$. If λ, B_0 satisfy

$$h(x)+\lambda \leq B_0(x)\ \forall x \in \mathcal{S}_{B_0} \tag{11}$$

then $B_k,\ k=1,2,\ldots$ is a (exponential) CBF for (1) with the tightened state constraint $x \in X_\lambda = \{x | h(x)+\lambda \leq 0\}$.

(iii) Let $\lambda_t = t\lambda,\ t \in \mathcal{I}(k-1),\ \lambda > 0$. If B_0 is a λ-contractive (exponential) CBF and k, λ, B_0 satisfy

$$h(x)+k\lambda \leq B_0(x)\ \forall x \in \mathcal{S}_{B_0} \tag{12}$$

then $B_k,\ k=1,2,\ldots$ is a λ-contractive (exponential) CBF for system (1) with the state constraint $x \in X$.

(iv) In the statements (i)–(iii), $B_k(\cdot)$ is a continuous function in $\mathcal{X}$, and the safe sets satisfy $\mathcal{S}_{B_0} \subseteq \mathcal{S}_{B_1} \subseteq \ldots \subseteq \mathcal{S}_{B_\infty}$. Furthermore, if f, h, and B_0 are Lipschitz continuous, B_k is Lipschitz continuous.

Proof: We will consider the case when B_0 is an exponential CBF, i.e., CBF satisfying the additional condition (4). The case when B_0 is a general CBF satisfying (3) can be analyzed similarly to the case when B_0 is an exponential CBF.

The proofs of (i) and (ii) can be combined, by considering $\lambda_t = \lambda$ and $\lambda \geq 0$. For any x_0 such that $B_k(x_0) \leq 0$, letting $(x^*_{t+1}, u^*_t),\ t = 0,1,\ldots,k-1$ be any one of the optimal solutions to (10) when $x = x_0$, we have

$$h(x^*_t)+\lambda \leq B_k(x_0) \leq 0,\ t=1,2,\ldots,k-1 \text{ and } B_0(x^*_k) \leq 0. \tag{13}$$

As B_0 is an exponential CBF and $B_0(x^*_k) \leq 0$, we have (i) $h(x^*_k)+\lambda \leq B_0(x^*_k)$ based on (11), and (ii) there exists an input $u^*_k \in U$, such that $x^*_{k+1} = f(x^*_k, u^*_k)$ will make $B_0(x^*_{k+1}) \leq \beta B_0(x^*_k)$. Now, we consider the value of $B_k(x^*_1)$. Based on the aforementioned discussion, it is clear that the trajectory $(x^*_{t+1}, u^*_t),\ t=1,2,\ldots,k$ is feasible for problem (10) when $x = x^*_1$. Due to the optimality of $B_k(x^*_1)$, we have

$$\begin{aligned} B_k(x^*_1) &\leq \max\left\{\max_{t\in\mathcal{I}(k-1)} \alpha^t(h(x^*_{t+1})+\lambda), \alpha^k B_0(x^*_{k+1})\right\} \\ &\leq \frac{1}{\alpha}\max\left\{\max_{t\in\mathcal{I}(k)} \alpha^t(h(x^*_t)+\lambda), \alpha^{k+1} B_0(x^*_{k+1})\right\} \\ &\leq \frac{1}{\alpha}\max\left\{\max_{t\in\mathcal{I}(k)} \alpha^t(h(x^*_t)+\lambda), \alpha^k B_0(x^*_k)\right\} \\ &\leq \frac{1}{\alpha}\max\left\{B_k(x_0), \alpha^k B_0(x^*_k)\right\} \leq \beta B_k(x_0). \end{aligned} \tag{14}$$

Here, the second inequality is true because we add the term $h(x^*_0)+\lambda$ to the inner max block. The third inequality follows from $B_0(x^*_{k+1}) \leq \frac{1}{\alpha}B_0(x^*_k)$. The fourth inequality holds because $\max\{\max_{t\in\mathcal{I}(k-1)} \alpha^t(h(x^*_t)+\lambda), \alpha^k B_0(x^*_k)\} = B_k(x_0)$. The last inequality holds because $\alpha^k B_0(x^*_k) \leq B_k(x_0)$, according to (10). From (14), we can conclude that $\min_{u\in U} B_k(f(x_0,u)) \leq \beta B_k(x_0)$. This, together with (13), shows that B_k is an exponential CBF for system (1) under the state constraint $x \in X_\lambda = \{x | h(x)+\lambda \leq 0\}$, since x_0 is selected arbitrarily in $\mathcal{S}_{B_k}$. Therefore, by specifying $\lambda = 0$ we prove the first statement, and by letting $\lambda > 0$ we prove (ii).

Next, we will prove (iii) of Theorem 1. Similarly to the proof of the first two statements, we consider any x_0 such that $B_k(x_0) \leq 0$. Denoting by $(x^*_{t+1}, u^*_t),\ t \in \mathcal{I}(k-1)$ any one of the optimal solutions to (10) when $x = x_0$, we have

$$h(x^*_t)+\lambda_t \leq B_k(x_0) \leq 0,\ t=1,2,\ldots,k-1 \text{ and } B_0(x^*_k) \leq 0. \tag{15}$$

Since B_0 is a λ-contractive-set exponential CBF and $B_0(x^*_k) \leq 0$, we have (i) $h(x^*_k)+k\lambda \leq B_0(x^*_k)$, and (ii) there is a control input $u^*_k \in U$ such that the successor state $x^*_{k+1} = f(x^*_k, u^*_k)$ ensures that $B_0(x^*_{k+1}) \leq \beta B_0(x^*_k) - \lambda$. Then, similar to (14), we can get $B_k(x^*_1) \leq -\lambda + \beta B_k(x_0)$ (see [23] for the detail). Together with (15), the aforementioned inequality implies that B_k is a λ-contractive-set exponential CBF for (1) with the state constraint $x \in X$.

Finally, we prove the last statement of Theorem 1. In all the cases of $\lambda_t = 0, \lambda_t = \lambda$, and $\lambda_t = t\lambda$, we consider the state and input sequences $\{x^*_t\}_{t=0}^{k+1}, \{u^*_t\}_{t=0}^{k}$, where $x^*_0 = x_0$. The sequences satisfy

$$B_k(x^*_0) \leq 0,\ h(x^*_t)+\lambda_t \leq 0,\ t \in \mathcal{I}(k),\ B_0(x^*_{k+1}) \leq 0. \tag{16}$$

From the last inequality in (16) and Definitions 1 and 4, we know that $h(x^*_{k+1})+\lambda_{k+1} \leq 0$ and that there exists a $u^*_{k+1} \in U$ such that the value of B_0 at $x^*_{k+2} = f(x^*_{k+1}, u^*_{k+1})$ is smaller than or equal to zero. Meanwhile, note that the sequences $\{x^*_t\}_{t=0}^{k+2}$ and $\{u^*_t\}_{t=0}^{k+1}$ constitute a feasible solution to problem (10) with $k+1$, so we get an upper bound on $B_{k+1}(x_0)$ as $B_{k+1}(x_0) \leq 0$. Due to the arbitrariness of x_0 in $\mathcal{S}_{B_k}$, we can conclude that $\mathcal{S}_{B_k} \subseteq \mathcal{S}_{B_{k+1}}$, which further by recursion proves that $\mathcal{S}_{B_0} \subseteq \mathcal{S}_{B_1} \subseteq \ldots \subseteq \mathcal{S}_{B_\infty}$.

For any $x_0, y_0 \in \mathcal{X}$, let $(x^*_{t+1}, u^{x*}_t), (y^*_{t+1}, u^{y*}_t),\ t \in \mathcal{I}(k-1)$ be any one of the optimal solutions to (10), with $x = x_0$ and $x = y_0$ respectively. For the continuity of B_k, following [24, Thm. 4], we have

$$\begin{aligned} &B_k(x_0) - B_k(y_0) \\ &\leq \max\left\{\max_{t\in\mathcal{I}(k-1)} \alpha^t h(x'_t) - (h(y^*_t)), \alpha^k(B_0(x'_k) - B_0(y^*_k))\right\} \end{aligned} \tag{17}$$

where $x'_t,\ t=0,1,\ldots,k$ is the state trajectory starting from x_0 and applying u^{y*}_t. Since (i) the trajectories x'_t and $y^*_t,\ t=0,1,\ldots,k$ are obtained by applying the same control inputs, (ii) f and h are continuous in their domains, and (iii) the maximum of continuous functions of x yields a continuous function of x, there exists a $\delta > 0$ such that $B_k(x_0) - B_k(y_0) < \epsilon$ whenever $||x_0 - y_0||_2 < \delta$. A mirror argument proves that $B_k(y_0) - B_k(x_0) < \epsilon$ whenever $||x_0 - y_0||_2 < \delta$. The continuity of B_k w.r.t. x thus follows.

Furthermore, if f, h, and B_0 are Lipschitz continuous, with Lipschitz constants L_f, L_h, and L_{B_0}, respectively, from (17) for any $x_0,\ y_0$ such that $||x_0 - y_0||_2 \leq \delta$, we have $B_k(x_0) - B_k(y_0) \leq L_{B_k}\delta$ where $L_{B_k} = \max\{\max_{t\in\mathcal{I}(k-1)} \alpha^t L_h L_f^t, \alpha^k L_{B_0} L_f^k\}$. Similarly, we get that $B_k(y_0) - B_k(x_0) \leq L_{B_k}\delta$ whenever $||x_0 - y_0||_2 \leq \delta$. Thus, we conclude the Lipschitz continuity of B_k w.r.t. x. □

If B_k and f are (Lipschitz) continuous in their domains, then the function Q defined by

$$Q(x,u) = B_k(f(x,u)) \tag{18}$$

is also (Lipschitz) continuous in $\mathcal{X} \times U$.

The condition (12) indicates that the horizon k should not be selected too large when generating a contractive-set CBF.

B. Learning SACBFs From CBF Samples

We assume that the prediction model in the reachability problem (10) is known. To approximate SACBFs, we opt for supervised learning, which is more computationally efficient than RL because it can solve (10) exactly and efficiently for each state sample.

In this work, we use NNs to approximate the SACBF. For the parameterization (7), we use three NNs to represent $q_{1,\theta}$, $q_{2,\theta}$, and $Q_{3,\theta}$ in (7), respectively. There are several ways to guarantee the positive-semidefiniteness of $Q_{3,\theta}$. For example, we can further parameterize $Q_{3,\theta}$ by $Q_{3,\theta} = L_\theta L_\theta^T$, where $L_\theta \in \mathbb{R}^{n_u \times n_u}$ is a lower triangular matrix with nonnegative diagonal entries. Alternatively, we can parameterize $Q_{3,\theta}$ by $Q_{3,\theta} = P_0 \text{diag}(r_\theta) \text{diag}(r_\theta) P_0^T$, where $r_\theta \in \mathbb{R}^{n_u}$ is the output of an NN and $P_0 \in \mathbb{R}^{n_u \times n_u}$ is a given matrix.

Finally, θ is optimized to minimize the MSE between $B_k(f)$ and Q_θ over all state-action samples, using classical NN training algorithms such as stochastic gradient descent [25].

C. Performance Analysis

After the approximation Q_θ has been obtained, it can be integrated into the SACBF-SF (6), i.e., Q in (6) is replaced by Q_θ. In general, the safety of the policy generated from (6) will not be ensured due to the approximation error. However, we will now demonstrate that we can guarantee safety in the presence of a small approximation error. To achieve this, we need an assumption on the boundedness of this error.

Assumption 1: There exists a constant $\Delta \in [0, \infty)$ such that

$$|Q_\theta(x,u) - Q(x,u)| \leq \Delta \; \forall (x,u) \in \Omega \times U \tag{19}$$

where Ω is a compact set and satisfies $\mathcal{S}_Q \subseteq \Omega$.

Assumption 1 is common in the literature studying performance guarantees of learning-based control [2], [15], [26], [27]. If the chosen function approximator represents a continuous function, since Q is also continuous, the approximation error is always upper bounded in any compact region. To obtain an upper bound Δ, we can first get the error bound for a finite number of samples, and then, extract a statistical estimate by using concentration inequalities [2] or compute a deterministic bound for the whole region by using the Lipschitz property of Q and Q_θ [26].

With Assumption 1, we modify the SACBF-SF (6) to

$$\begin{aligned} \pi_\theta(x) = \arg\min_{u \in U} \; & ||u - \pi_0(x)||_2 \\ \text{s.t. } & Q_\theta(x,u) \leq 0 \text{ for Options 1 and 2} \\ & Q_\theta(x,u) \leq -\lambda + \Delta \text{ for Option 3.} \end{aligned} \tag{20}$$

Theorem 2 (Safety and recursive feasibility): Consider the system (1) controlled by π_θ, the SACBF Q from (18), and the SACBF-SF (20). Suppose that Assumption 1 holds.

(i) If B_k is a CBF for the original state constraint, for any initial state $x_0 \in \mathcal{S}_{B_k}$ that makes problem (20) recursively feasible, the closed-loop system $x_{t+1} = f(x_t, \pi_\theta(x_t))$, $t = 0, 1, \ldots$ has the maximum constraint violation Δ.

(ii) If B_k is a CBF for the tightened state constraint $x \in X_\lambda$ and $\Delta \leq \lambda$, for any $x_0 \in \mathcal{S}_{B_k}$ that makes (20) recursively feasible, the closed-loop system $x_{t+1} = f(x_t, \pi_\theta(x_t))$ always satisfies the state constraint $x \in X$.

(iii) If B_k is a λ-contractive-set CBF for the original state constraint and $\Delta \leq \lambda/2$, (20) will be recursively feasible for all initial states in $\mathcal{S}_{B_k}$ and the closed-loop system $x_{t+1} = f(x_t, \pi_\theta(x_t))$ always satisfies the state constraint $x \in X$.

Proof: The proofs of (i) and (ii) can be combined. From Assumption 1 and (20), we know that if (20) is recursively feasible for the initial state x_0, we have $Q(x_t, \pi_\theta(x_t)) \leq \Delta \; \forall t \in \mathcal{I}(\infty)$, which further implies $B_k(x_t) \leq \Delta \; \forall t \in \mathcal{I}(\infty)$. If B_k is a CBF for (1) with the original state constraint $x \in X$, we get $h(x_t) \leq B_k(x_t) \leq \Delta$. Similarly, if B_k is a CBF for (1) with the tightened state constraint $x \in X_\lambda$ and $\lambda \geq \Delta$, we get $h(x_t) \leq B_k(x_t) - \lambda \leq 0$.

Next, we consider (iii). For any initial state $x_0 \in \mathcal{S}_{B_k}$, there exists a u_0 such that $Q(x_0, u_0) \leq -\lambda$ according to (8) and (9). Since Q_θ satisfies Assumption 1, u_0 makes $Q_\theta(x_0, u_0) \leq -\lambda + \Delta$, which means problem (20) is feasible at x_0. Then, we consider any feasible solution u_0' to (20) when $x = x_0$, i.e., any $u_0' \in U$ such that $Q_\theta(x_0, u_0') \leq -\lambda + \Delta$. It follows from Assumption 1 and $\Delta \leq \lambda/2$ that $Q(x_0, u_0') \leq -\lambda + 2\Delta \leq 0$. This means that $B_k(f(x_0, u_0')) \leq 0$, i.e., the next state $x_1' = f(x_0, u_0')$ is in $\mathcal{S}_{B_k}$. As we have proved that problem (20) is feasible for any $x_0 \in \mathcal{S}_{B_k}$, the recursive feasibility of (20) is thus proved. A direct consequence is that $B_k(x_t) \leq 0 \; \forall t \in \mathcal{I}(\infty)$, where x_t is the state trajectory of the system $x_{t+1} = f(x_t, \pi_\theta(x_t))$. Moreover, since B_k is a CBF, we have $h(x_t) \leq 0 \; \forall t \in \mathcal{I}(\infty)$. This finishes the proof of (iii). □

In Option 2 of (10), we get a CBF B_k for the tightened constraint $x \in X_\lambda$. The policy π_θ will always make the system satisfy the original state constraint if (20) is always feasible. However, the feasibility of (20) is only guaranteed for the initial state, i.e., (20) can be infeasible for some subsequent states. Practically, ones need to perform offline some statistical or deterministic verification [2] to analyze the recursive feasibility of (20). In comparison, in Option 3 of (10), we get a λ-contractive-set CBF B_k. An induced feature is the recursive feasibility of problem (20). Under the sufficiently small approximation error, the SACBF approximation Q_θ will render π_θ a safe policy according to Definition 2.

The proposed constraint tightening approach naturally has robustness to model uncertainty. Suppose that (1) is a nominal system, and the real system is $x_{t+1} = f_r(x_t, u_t)$. The mismatch between f and f_r can be handled similarly to the learning error of Q. Formally, we have the following corollary.

Corollary 1 (Robustness to model uncertainty): Suppose that Assumption 1 is fulfilled and the nominal system satisfies $|f(x,u) - f_r(x,u)| \leq \kappa \; \forall (x,u) \in \Omega \times U$, where $\kappa \geq 0$. Then, if B_k is a λ-contractive-set CBF for the nominal system under the original state constraint and $\lambda \geq 2\Delta + \text{L}_B \kappa$ with L_B the Lipschitz constant of B_k, problem (20) will be recursively feasible for the real system with all initial states in $\mathcal{S}_{B_k}$ and the closed-loop system $x_{t+1} = f_r(x_t, \pi_\theta(x_t))$ always satisfies the state constraint $x \in X$, i.e., $\max_{t \in \mathcal{I}(\infty)}\{h(x_t)\} \leq 0$.

D. Discussion on SACBFs

In addition to the computational benefit of SACBFs, the similarity between SACBFs and Q value functions in RL indicates that it is possible to use RL methods [28] such as Q value iteration or Q-learning to learn SACBFs directly without knowing the explicit model. Even though there are several data-driven methods to synthesize standard CBFs from black-box models [12], [29], a prediction model is necessary to implement the CBF-SF (5). In comparison, the proposed SACBF provides the possibility of bridging the gap between model-free learning control and CBF-SFs (see also [7] and[8]).

One limitation of the parameterization (7) is that it sacrifices the universal approximation property [25] of NNs, since it restricts Q_θ to be quadratic on u. This may result in a large error bound Δ and cause constraint violations according to Theorem 2. As our main motivation for introducing SACBFs is to improve online computational efficiency, such a sacrifice is acceptable. Conversely, fully parameterizing Q_θ with a continuous NN, possessing universal approximation capabilities, increases the online computational complexity and introduces challenges in finding the global optimum of (20). In conclusion, when applying

TABLE I
PERFORMANCE OF DIFFERENT CONTROLLERS

Safety filters	Basic control policies								
	Learning MPC			ADP			LQR		
	Safety rate	CPU time	Cost	Safety rate	CPU time	Cost	Safety rate	CPU time	Cost
No filter	63.52%	0.024 ms	99.69	36.66%	0.024 ms	125.11	56.99%	0.023 ms	122.10
Standard CBF	63.52%	2.3 ms	126.18	39.93%	2.4 ms	161.82	57.17%	2.3 ms	120.00
Quad. SACBF	70.05%	1 ms	131.76	47.19%	1.1 ms	161.83	65.52%	0.9 ms	115.80
Nonlin. SACBF	65.15%	2.5 ms	132.22	40.19%	3.0 ms	168.19	60.80%	2.9 ms	137.30
Quad tightened SACBF	98.73%	1.0 ms	97.29	98.73%	1.0 ms	99.44	92.92%	1.0 ms	91.47
Nonlin. tightened SACBF	82.40%	2.7 ms	117.26	82.03%	2.7 ms	116.56	90.56%	3.0 ms	116.92
Quad. contr. SACBF	100%	0.9 ms	98.84	100%	1.0 ms	101.91	100%	0.9 ms	93.10
Nonlin. contr. SACBF	96.91%	2.1 ms	110.44	92.20%	2.0 ms	128.99	94.57%	3.0 ms	109.97
MPC	Horizon=5			Horizon=6			Horizon=7		
	70.24%	38.2 ms	98.55	95.83%	37.6 ms	97.43	100%	38.5 ms	96.86

an approximation to the SACBF, there is typically a natural tradeoff between online computational efficiency and safety.

Note that a parametrization of an SACBF Q, instead of parameterizing a standard CBF B, may increase the approximation error on $B(f)$, especially when Q is parameterized in a quadratic form. This error can result in the learned Q failing to satisfy the conditions of a valid SACBF. To address this issue, we have proposed the constraint tightening approach (see Section IV) to ensure that the learning-based approach can still generate a valid SACBF under mild approximation errors. However, a quadratic parametrization of Q typically requires more stringent constraint tightening, which can lead to increased conservatism in the learned barrier certificates.

The conservatism of the learned SACBF Q_θ and the policy π_θ is strongly influenced by Δ. According to Theorem 2, to guarantee the recursive feasibility of (20) and the safety of π_θ, a larger Δ requires a larger λ in (10). This, in turn, increases the conservatism of the target Q. In practice, to minimize the conservatism, one can start with a small λ and iteratively increase it until Assumption 1 and $\Delta \leq \lambda/2$ are satisfied. Increasing the approximation capability of the chosen function approximator, as discussed in [30], can also be beneficial.

VI. CASE STUDY

We validate the proposed methods for an active inverted pendulum interacting with elastic walls, which is a piecewise affine system [3]. The detailed model and solvers we use can be found in Appendix E of the supplementary material [23]. The system contains the state $x = [\theta\ \dot{\theta}]^T$ where θ is the pendulum angle, and the input torque u. The constraints are $|x_1| \leq 0.15$, $|x_2| \leq 1$, and $|u| \leq 4$. The system is discretized using the explicit Euler method with time step 0.05 s.

Using the approach in Appendix D of the supplementary material [23] and letting $\lambda = 0.2$ for Option 2 and $\lambda = 0.05$ for Option 3, we obtain the initial CBF $B_0(x) = x^T P x - 1$ with $P = \begin{bmatrix} 124.74 & 7.44 \\ 7.44 & 2.24 \end{bmatrix}$, $\begin{bmatrix} 283.40 & 20.25 \\ 20.25 & 4.69 \end{bmatrix}$, and $\begin{bmatrix} 522.39 & 25.69 \\ 25.69 & 5.39 \end{bmatrix}$ for Options 1, 2, and 3, respectively. We sample from the state-input space $\{(x,u)\big|\ |x_1| \leq 0.16, |x_2| \leq 1.1, |u| \leq 4\}$ on a uniform grid and we obtain 40^3 samples $\{(x_i^{(\mathrm{s})}, u_i^{(\mathrm{s})})\}_{i=1}^{40^3}$. The successor state $f(x_i^{(\mathrm{s})}, u_i^{(\mathrm{s})})$ of each sample $(x_i^{(\mathrm{s})}, u_i^{(\mathrm{s})})$ is fed to the CBF generator (10) with $k = 7$. Only those samples satisfying $B_k(f(x_i^{(\mathrm{s})}, u_i^{(\mathrm{s})})) \leq 10$ are collected. After this procedure, 42 109, 38 609 and 37 027 samples are collected in each option.

We consider the following four different kinds of safety certificates:

i) SACBF for the original state constraint;
ii) SACBF for the tightened state constraint $|x_1| \leq 0.14$, $|x_2| \leq 0.9$;
iii) contractive-set SACBF;
iv) standard CBF for the original state constraint.

For the learning, we use NNs with three hidden hyperbolic tangent layers containing 16, 64, and 8 neurons. We parameterize each standard CBF by an NN. For the parameterization of each SACBF, we use (7) or fully parameterize it by an NN. Besides, we also consider four different control policies. The first three include the following:

i) supervised learning of the MPC policy [9];
ii) the approximate dynamic programming (ADP) policy [3];
iii) the LQR policy for the linearized system.

These policies may be unsafe and are thus taken as π_0 in (5) and (20). The learning-based control policies (i) and (ii) are also represented by NNs. The main reason for choosing these three particular policies is that they can achieve suboptimal control objectives using very limited computational resources. However, they generally cannot guarantee constraint satisfaction. Therefore, we use these three policies as the baseline to demonstrate that the proposed SACBF-SF can enhance the rate of constraint satisfaction for these policies. The fourth control policy is implicit MPC. One can refer to Appendix E of the supplementary material [23] for a detailed description of these policies.

To test all the control methods, we uniformly randomly sample initial states from the state sample set $\{x_i^{(\mathrm{s})}\}_{i=1}^{40^3}$ and select a total of 551 initial states x that satisfy $B_k(x) \in (-0.1, 0)$. The simulation results are shown in Table I. The performance metrics include the rate of safety, the average CPU time for computing the control input per time step, and the total cost $\sum_{t=0}^{50} C(x_t, u_t)$ with the stage cost C the same as that in the considered basic controllers. The safety rate is defined as the number of initial states leading to a safe closed-loop trajectory divided by the total number of initial states. Overall, the table indicates that incorporating appropriate SFs, especially quadratic and nonlinear SACBFs, can significantly improve the safety performance of control policies without excessively compromising on CPU time and total cost. The online computational benefit of using the SACBF with the quadratic parametrization (7) primarily stems from the fact that solving a convex quadratic program is typically faster than solving a nonlinear program with the same number of constraints and decision variables. The maximum performance loss is about 34%, which occurs when applying the nonlinear SACBF to the ADP controller. Considering that the main objective of our work is to impose safety using low computational resources, such a performance degradation is acceptable. The inferior performance of the nonlinear contractive-set SACBF compared to its quadratic counterpart with respect to constraint satisfaction is primarily due to the suboptimal solutions when solving (20).

Fig. 1(a) and (b) shows the state-trajectories of the closed-loop system with the policy π_θ from (20), with π_0 being the learning MPC controller and Q_θ being the nonlinear SACBF or the nonlinear contractive-set SACBF. Without constraint tightening, the system evolution records several constraint violations, as evidenced by Table I. Conversely, the contractiveness property of SACBFs ensures that almost all state trajectories remain within its safe set. To further demonstrate the proposed method's effectiveness under nonlinear constraints, we consider a quadratic state constraint given by $x \in X := \{x \in \mathbb{R}^2 | \frac{\theta^2}{0.15^2} + \frac{\dot{\theta}^2}{1^2} \leq 1\}$. The resulting state-trajectories of the

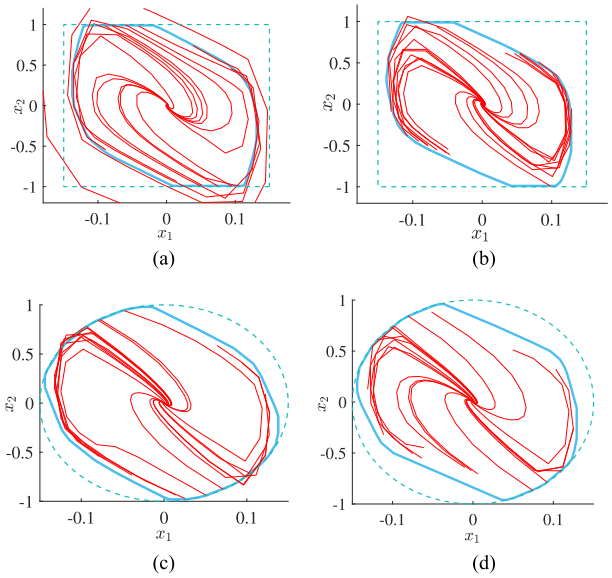

Fig. 1. Closed-loop trajectories of the learning MPC controller under the SACBF-SF. The red curves represent the trajectories starting from different initial states in the safe set. The blue curves refer to the boundaries of the safe set. The green dashed curves represent the boundaries of the state constraint set. (a) State trajectories and the safe set for the SACBF under the linear state constraint. (b) State trajectories and the safe set for the contractive-set SACBF under the linear state constraint. (c) State trajectories and the safe set for the SACBF under the nonlinear state constraint. (d) State trajectories and the safe set for the contractive-set SACBF under the nonlinear state constraint.

closed-loop system under the policy π_θ are displayed in Fig. 1(c) and (d), from which we observe that the contractive-set SACBF guarantees safety for all simulated trajectories under this quadratic constraint.

VII. Conclusion

This article has presented a new type of CBFs, called SACBFs, that can synthesize safe controllers for constrained nonlinear systems with a very small online computational overhead. We have also developed a new approach to learn the proposed CBF from data, and proposed a constraint tightening approach to improve the robustness of the learned CBFs to learning errors. Future work will focus on computational efficiency for large-scale problems, guiding policy learning with the SACBFs, investigating potential undesired equilibria induced by the SF, and applying the SACBF-SF in model-free safe learning for safe control.

References

[1] M. Schwenzer, M. Ay, T. Bergs, and D. Abel, "Review on model predictive control: An engineering perspective," *Int. J. Adv. Manuf. Technol.*, vol. 117, no. 5, pp. 1327–1349, 2021.

[2] M. Hertneck, J. Köhler, S. Trimpe, and F. Allgöwer, "Learning an approximate model predictive controller with guarantees," *IEEE Contr. Syst. Lett.*, vol. 2, no. 3, pp. 543–548, Jul. 2018.

[3] K. He, S. Shi, T. van den Boom, and B. De Schutter, "Approximate dynamic programming for constrained piecewise affine systems with stability and safety guarantees," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 55, no. 3, pp. 1722–1734, Mar. 2025.

[4] S. Paternain, M. Calvo-Fullana, L. F. Chamon, and A. Ribeiro, "Learning safe policies via primal-dual methods," in *Proc. IEEE 58th Conf. Decis. Control*, 2019, pp. 6491–6497.

[5] L. Zheng, Y. Shi, L. J. Ratliff, and B. Zhang, "Safe reinforcement learning of control-affine systems with vertex networks," in *Proc. Conf. Learn. Dyn. Control*, 2021, pp. 336–347.

[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," 2022, *arXiv:2207.08240*.

[7] 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 Trans. Robot.*, vol. 39, no. 3, pp. 1749–1767, Jun. 2023.

[8] K. P. Wabersich et al., "Data-driven safety filters: Hamilton-Jacobi reachability, control barrier functions, and predictive methods for uncertain systems," *IEEE Control Syst. Mag.*, vol. 43, no. 5, pp. 137–177, Oct. 2023.

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

[10] M. Lazar, W. Heemels, S. Weiland, and A. Bemporad, "Stabilizing model predictive control of hybrid systems," *IEEE Trans. Autom. Control*, vol. 51, no. 11, pp. 1813–1818, Nov. 2006.

[11] S. Prajna, A. Jadbabaie, and G. J. Pappas, "A framework for worst-case and stochastic safety verification using barrier certificates," *IEEE Trans. Autom. Control*, vol. 52, no. 8, pp. 1415–1428, Aug. 2007.

[12] A. Robey et al., "Learning control barrier functions from expert demonstrations," in *Proc. IEEE 59th Conf. Decis. Control*, 2020, pp. 3717–3724.

[13] H. Dai, B. Landry, M. Pavone, and R. Tedrake, "Counter-example guided synthesis of neural network Lyapunov functions for piecewise linear systems," in *Proc. IEEE 59th Conf. Decis. Control*, 2020, pp. 1274–1281.

[14] J. J. Choi, D. Lee, K. Sreenath, C. J. Tomlin, and S. L. Herbert, "Robust control barrier–value functions for safety-critical control," in *Proc. IEEE 60th Conf. Decis. Control*, 2021, pp. 6814–6821.

[15] A. Didier, R. C. Jacobs, J. Sieber, K. P. Wabersich, and M. N. Zeilinger, "Approximate predictive control barrier functions using neural networks: A computationally cheap and permissive safety filter," in *Proc. Eur. Control Conf.*, 2023, pp. 1–7.

[16] A. Agrawal and K. Sreenath, "Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation," in *Proc. Conf. Robot. Sci. Syst.*, Cambridge, MA, USA, 2017, pp. 1–10.

[17] M. Srinivasan, A. Dabholkar, S. Coogan, and P. A. Vela, "Synthesis of control barrier functions using a supervised machine learning approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 7139–7145.

[18] D. P. Bertsekas, *Reinforcement Learning and Optimal Control*, Belmont, MA, USA: Athena Sci., 2019.

[19] M. H. Cohen, and C. Belta, "Safe exploration in model-based reinforcement learning using control barrier functions," *Automatica*, vol. 147, 2023, Art. no. 110684.

[20] S. Tonkens and S. Herbert, "Refining control barrier functions through Hamilton-Jacobi reachability," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 13355–13362.

[21] A. Alessio, M. Lazar, A. Bemporad, and W. Heemels, "Squaring the circle: An algorithm for generating polyhedral invariant sets from ellipsoidal ones," *Automatica*, vol. 43, no. 12, pp. 2096–2103, 2007.

[22] K. P. Wabersich and M. N. Zeilinger, "Predictive control barrier functions: Enhanced safety mechanisms for learning-based control," *IEEE Trans. Autom. Control*, vol. 68, no. 5, pp. 2638–2651, May 2023.

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

[24] M. Korda, "Computing controlled invariant sets from data using convex optimization," *SIAM J. Control Optim.*, vol. 58, no. 5, pp. 2871–2899, 2020.

[25] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[26] K. He, S. Shi, T. van den Boom, and B. De Schutter, "Approximate dynamic programming for constrained linear systems: A piecewise quadratic approximation approach," *Automatica*, vol. 160, 2024, Art. no. 111456.

[27] A. Abate, A. Edwards, and M. Giacobbe, "Neural abstractions," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 26432–26447.

[28] J. F. Fisac, N. F. Lugovoy, V. Rubies-Royo, S. Ghosh, and C. J. Tomlin, "Bridging Hamilton-Jacobi safety analysis and reinforcement learning," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 8550–8556.

[29] 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 *Proc. IEEE 61st Conf. Decis. Control*, 2022, pp. 3431–3437.

[30] A. R. Barron, "Approximation and estimation bounds for artificial neural networks," *Mach. Learn.*, vol. 14, pp. 115–133, 1994.