Verification Framework for the Union of Control Barrier Functions

Chuanrui Jiang and Andrew Clark

***Abstract*— Control Barrier Functions (CBFs) have been proposed to ensure safety of autonomous systems. This paper considers control policies that switch between CBF constraints. Under this approach, we represent a complex non-convex safe region as a union of sets that are computationally tractable to verify. We denote this framework as union-CBFs and make the following contributions. First, considering switching CBF-QP controllers, we propose a sufficient condition that ensures (i) the system undergoes a finite number of switches in any finite time interval and ensures (ii) the forward invariance of the closed-loop system in between switches. Second, we consider two types of switching strategies and propose union-CBFs conditions for each strategy to satisfy (i) and (ii). Third, we formulate Sum-of-Squares (SOS) algorithms to verify the conditions. The experiments show that our union-CBFs framework allows safe control policies that are less conservative than existing methods. We also show the efficiency of the verification algorithms using a polynomial system model.**

I. Introduction

Safety is a critical property of control systems and is usually defined as the forward invariance of a set of safe states [1]. Control Barrier Functions (CBFs) have been proposed to ensure safety [2], [3]. Mathematically, CBFs are state evaluation functions that output negative values for unsafe states and output positive values for some safe states. The safety of a system can be ensured by placing a CBF constraint on the control input so that the closed-loop system is forward invariant within the 0-super level set of the CBF.

In order to ensure that the system is able to perform needed tasks or satisfy stability properties, it is usually desirable to maximize the volume of the safe region [4]–[6]. In many safe control scenarios, the set of safe states is non-convex with irregular shapes [7], [8]. In such cases, the CBF needed to express this safety constraint is a high-degree polynomial [9] or neural network [10]–[12], both of which require significant computational overhead to synthesize. Furthermore, verifying the safety properties of such complex CBFs is computationally prohibitive [13].

An alternative to expressing complex safety constraints with a single CBF is to describe the safe region as a union of simpler subsets, with each subset defined by a distinct CBF [7], [14], [15]. We call this idea *union-CBFs* in this paper. Union-CBFs ensure safety by letting the safe controller switch between different CBF constraints. In [15], Boolean Non-smooth CBF (BNCBF) is proposed as the maximum over all the distinct CBFs in the union. However, in order to ensure existence of solutions to the closed-loop system, the BNCBF framework requires that *multiple* CBF constraints be satisfied simultaneously at certain points in state space, and hence can be conservative. The recent work [16] proposes necessary and sufficient conditions for safety using switching CBFs, but assumes the uniqueness of trajectories of the closed-loop system. According to [17], [18], formal CBF-based safety also requires that the safe controller has non-trivial regularity properties (e.g. existence of a feasible control input [17], and smoothness [19] or local Lipschitz [18] properties). These regularity properties are not guaranteed for optimization-based controllers, such as CBF-Quadratic Programming [2] (CBF-QP), and hence must be verified [13], [17].

In this paper, we propose a union-CBFs framework with switching CBF-QP controllers. The first result is a sufficient condition for union-CBFs that ensures (i) the number of switches in any finite time interval is finite and ensures (ii) forward invariance of the system with switching CBF-QP. The proposed union-CBFs condition leads to less conservative safe control policies while satisfying needed regularity properties. Based on the first result, we propose two types of switching strategies. For each strategy, we derive conditions for forward invariance and develop Sum-Of-Squares-based verification algorithms to verify that the conditions are satisfied. In the experiments, we show that (i) the union-CBFs framework is more efficient than a single high-degree CBF, and (ii) our proposed sufficient condition for union-CBFs is less conservative than existing method. Efficiency of our proposed SOS verification methods are also analyzed using a nonlinear polynomial system example.

The rest of the paper is organized as follows. We introduce preliminaries in Section II. We propose sufficient conditions for forward invariance under union-CBFs in Section III. The union-CBFs conditions under two switching strategies are introduced in Section IV. The SOS verification frameworks are derived in Section V. Experiments are presented in Section VI, and Section VII concludes the paper.

Notations: Throughout this paper, we use un-bold letters x and X to denote scalars, use bold lower case letters $\mathbf{x}$ to denote vectors, and use bold upper case letters $\mathbf{X}$ to present matrices. The bold $\mathbf{0}$ denotes a vector of all zeros. For a vector $\mathbf{x}$, $\mathbf{x}_{(i)}$ denotes the i-th element. $\mathbf{x}^2$ denotes element-wise squared vector of $\mathbf{x}$ such that $(\mathbf{x}^2)_{(i)} = (\mathbf{x}_{(i)})^2, i = 1, \dots, n$. For a vector $\mathbf{x}$ and a scalar constant ϵ, $\mathbf{y} = \mathbf{x} - \epsilon$ means $\mathbf{y}_{(i)} = \mathbf{x}_{(i)} - \epsilon$ for all $i = 1, \dots, n$. For two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, $\mathbf{x} \leq \mathbf{y}$ denotes that $\mathbf{x}_{(i)} \leq \mathbf{y}_{(i)}$ for all $i = 1, ..., n$. For a matrix $\mathbf{X}$, $\mathbf{X} \succ 0$ means $\mathbf{X}$ is positive definite. We also use upper case letters in Cali-graphic font $\mathcal{X}$ to present sets.

Chuanrui Jiang and Andrew Clark are with the Electrical and Systems Engineering Department, McKelvey School of Engineering, Washington University in St. Louis, St. Louis, MO 63130. Email: {chaunrui,andrewclark}@wustl.edu

For Lie derivatives, we let $L_{\mathbf{f}}h(\mathbf{x}) = \frac{\partial h(\mathbf{x})}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x})$.

II. Preliminaries

In this section, we first introduce the system model. Then, we introduce the concepts of Control Barrier Functions (CBFs). Finally, we introduce a Sum-Of-Squares method that verifies the emptiness of a set.

A. System Model

We consider a control affine system defined as

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{G}(\mathbf{x}(t))\mathbf{u}(t) \tag{1}$$

where the state $\mathbf{x}(t) \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$, the control input $\mathbf{u}(t) \in \mathcal{U} \subseteq \mathbb{R}^{n_u}$, $\mathcal{X}$ denotes the state space, and $\mathcal{U}$ denotes the admissible control set. In this paper, we assume that both $\mathbf{f}(\mathbf{x})$ and $\mathbf{G}(\mathbf{x})$ are polynomials of $\mathbf{x}$, and the state space $\mathcal{X}$ is a bounded set, meaning that we rule-out the case where $||\mathbf{x}(t)|| \to \infty$. We also assume that the admissible control set is $\mathcal{U} = \{\mathbf{u}|\mathbf{A}\mathbf{u} \leq \mathbf{c}\}$, where $\mathbf{A}$ and $\mathbf{c}$ are constant matrix and vector such that $\mathcal{U}$ is compact. The state space $\mathcal{X}$ has a safe region $\mathcal{X}_s$ containing all the safe states, and an unsafe region $\mathcal{X}_u = \mathcal{X} \setminus \mathcal{X}_s$ containing all the unsafe states. The system (1) is safe if there exists a controller and a subset $\mathcal{C} \subseteq \mathcal{X}_s$ such that, $\forall \mathbf{x}(0) \in \mathcal{C}$, the closed loop system satisfies $\mathbf{x}(t) \in \mathcal{C}, \forall t \geq 0$, namely, system (1) is controlled forward invariant with respect to $\mathcal{C}$.

B. Control Barrier Functions

In order to ensure safety, we define a function $h : \mathcal{X} \to \mathbb{R}$. We let $\mathcal{C}$ be its 0-super-level set $\{\mathbf{x}|h(\mathbf{x}) \geq 0\}$ so that the safety can be certified by control barrier functions.

Definition 1 (CBF [3]): A continuously differentiable function $h(\mathbf{x})$ is a CBF for system (1) if there exists an extended class-κ function $\kappa_h()$ such that for all $\mathbf{x} \in \mathcal{C}$:

$$\sup_{\mathbf{u}\in\mathcal{U}}\{L_{\mathbf{f}}h(\mathbf{x}) + L_{\mathbf{G}}h(\mathbf{x})\mathbf{u} + \kappa(h(\mathbf{x}))\} \geq 0. \tag{2}$$

Note that $L_{\mathbf{f}}h(\mathbf{x})$ is a scalar, $L_{\mathbf{G}}h(\mathbf{x})$ is a 1-by-n_u row vector, and $\kappa(.)$ is an extended class κ function, which is monotonically increasing with $\kappa(0) = 0$. A CBF is a minimal CBF [20] if the class-κ function is also a minimal function [20, Theorem 2]. In this paper, we consider linear extended class-κ functions as the minimal function such that $\kappa(h(\mathbf{x})) = \kappa h(\mathbf{x})$ with $\kappa > 0$.

The safe controller $\mathbf{k} : \mathcal{X} \to \mathcal{U}$ considered in this paper is based on the CBF-Quadratic Program shown below

$$\mathbf{k}(\mathbf{x}) = arg\min_{\mathbf{u}} \frac{1}{2}\mathbf{u}^T\mathbf{H}(\mathbf{x})\mathbf{u} + \mathbf{c}^T(\mathbf{x})\mathbf{u} \tag{3a}$$

$$\text{s.t. } L_{\mathbf{f}}h(\mathbf{x}) + L_{\mathbf{G}}h(\mathbf{x})\mathbf{u} \geq -\kappa h(\mathbf{x}) \tag{3b}$$

$$\mathbf{A}\mathbf{u} \leq \mathbf{c} \tag{3c}$$

where $\mathbf{H}(\mathbf{x}) \succ 0$, $\forall \mathbf{x} \in \mathcal{X}$. By [18, Theorem 5.4], the following theorem provides a regularity property for the CBF-QP above to ensure formal safety.

Theorem 1 ([18]): The closed loop system (1) with controller $\mathbf{u} = \mathbf{k}(\mathbf{x})$ is forward invariant with respect to $\mathcal{C}$ if $h(\mathbf{x})$ is a minimal CBF and the controller $\mathbf{k}(\mathbf{x})$ is continuous.

By [18], the controller $\mathbf{k}(\mathbf{x})$ is point-Lipschitz and thus continuous if the following Slater's Condition holds: $\forall \mathbf{x} \in \mathcal{C}$, $\exists \mathbf{u}$ such that $\mathbf{A}\mathbf{u} < \mathbf{c}$ and the inequality constraint (3b) strictly holds.

C. Algebraic Background

In this subsection, we first introduce Farkas' lemma, which gives a necessary and sufficient condition for the existence of solutions to linear constraints. Then, we introduce a Sum-Of-Squares (SOS) method from [5] and [21] to verify emptiness of a set.

Lemma 1 (Farkas' Lemma [22]): Let $\mathbf{\Lambda} \in \mathbb{R}^{m\times n}$ be a matrix and $\boldsymbol{\xi} \in \mathbb{R}^m$ be a vector. $\exists \mathbf{x}$ with $\mathbf{\Lambda}\mathbf{x} \leq \boldsymbol{\xi}$ if and only if the set $\{\mathbf{z}|\mathbf{z} \geq \mathbf{0}, \mathbf{\Lambda}^T\mathbf{z} = \mathbf{0}, \boldsymbol{\xi}^T\mathbf{z} = -1\}$ is empty.

In this paper, we use the equivalent condition: $\mathbf{\Lambda}\mathbf{x} \leq \boldsymbol{\xi}$ has a solution $\mathbf{x}$ if and only if the set $\{\mathbf{y}|\mathbf{\Lambda}^T\mathbf{y}^2 = \mathbf{0}, \boldsymbol{\xi}^T\mathbf{y}^2 = -1\}$ is empty.

Now we introduce the theorem in [21] that verifies the emptiness of a given set. Note that a polynomial $s(\mathbf{x})$ is an SOS if $s(\mathbf{x})$ can be decomposed as $s(\mathbf{x}) = \sum_{i=1}^N p_i^2(\mathbf{x})$ for a set of polynomials $p_1(\mathbf{x}), \ldots, p_N(\mathbf{x})$.

Theorem 2 ([21]): For some given polynomials $\phi_i(\mathbf{x})$, $i = 1, \ldots, N$, and polynomials $\psi_j(\mathbf{x})$, $j = 1, \ldots, M$, we define the set $\mathcal{S}$ as follows

$$\mathcal{S} = \left\{ \mathbf{x} \in \mathbb{R}^n \;\middle|\; \begin{array}{l} \phi_i(\mathbf{x}) = 0, \forall i = 1, \ldots, N \\ \psi_j(\mathbf{x}) \geq 0, \forall j = 1, \ldots, M \end{array} \right\}$$

Set $\mathcal{S}$ is empty if the following SOS program is feasible

$$\begin{aligned} &\text{Find } s_1(\mathbf{x}), \ldots, s_N(\mathbf{x}) \text{ and } q_1(\mathbf{x}), \ldots, q_M(\mathbf{x}) \\ &\text{s.t. } -1 - \sum_{i=1}^N s_i(\mathbf{x})\phi_i(\mathbf{x}) - \sum_{j=1}^M q_j(\mathbf{x})\psi_j(\mathbf{x}) \text{ is SOS} \\ &\quad s_1(\mathbf{x}), \ldots, s_N(\mathbf{x}) \text{ are SOS} \end{aligned}$$

where $q_1(\mathbf{x}), \ldots, q_M(\mathbf{x})$ are polynomials.

Theorem 2 provides a sufficient condition for a set to be empty. With this SOS method, the verification of an empty set can be transformed into an SOS program that can be solved by SOSTOOLS [23] or MOSEK [24] solvers.

III. Forward Invariance Conditions

This section presents our first result of this paper. Let $h_1(\mathbf{x}), \ldots, h_{N_h}(\mathbf{x})$ be a given collection of differentiable functions. We define set $\mathcal{P} = \{1, \ldots, N_h\}$, and define $\mathcal{X}_p = \{\mathbf{x}|h_p(\mathbf{x}) \geq 0\}$ for all $p \in \mathcal{P}$. Our goal is to keep the system (1) controlled forward invariant with respect to the set $\mathcal{X}_c = \cup_{p=1}^{N_h}\mathcal{X}_p$. We first introduce the considered switching CBF-QP, then give sufficient conditions for controlled forward invariance under union-CBFs.

A. Switching CBF-QPs

To achieve controlled forward invariance, we let the control policy switch between controllers $\mathbf{k}_1(\mathbf{x}), ..., \mathbf{k}_{N_h}(\mathbf{x})$, where each $\mathbf{k}_p(\mathbf{x}), p \in \mathcal{P}$ is computed as

$$\mathbf{k}_p(\mathbf{x}) = arg\min_{\mathbf{u}} \frac{1}{2}\mathbf{u}^T\mathbf{H}(\mathbf{x})\mathbf{u} + \mathbf{c}^T(\mathbf{x})\mathbf{u} \tag{4a}$$

$$\text{s.t. } L_{\mathbf{f}}h_p(\mathbf{x}) + L_{\mathbf{G}}h_p(\mathbf{x})\mathbf{u} \geq -\kappa h_p(\mathbf{x}) \tag{4b}$$

$$\mathbf{A}\mathbf{u} \leq \mathbf{c} \tag{4c}$$

where $\mathbf{H}(\mathbf{x}) \succ 0$ for all $\mathbf{x} \in \mathcal{X}$. Let $\sigma : [0, \infty) \to \mathcal{P}$ be the switching policy. A switching CBF-QP controller is then denoted as $\mathbf{u}(t) = \mathbf{k}_{\sigma(t)}(\mathbf{x}(t))$. Next, we consider the control system (1) with a switching CBF-QP controller and propose sufficient conditions that keep (1) controlled forward invariant with respect to $\mathcal{X}_c$.

B. Sufficient Conditions For Union-CBFs

Proposition 1: Consider control affine model (1) with switching CBF-QP controller $\mathbf{u}(t) = \mathbf{k}_{\sigma(t)}(\mathbf{x}(t))$ suppose that:

P1-1 $\sigma(t)$ is right continuous, piecewise constant, and only have finite number of switches for any finite time horizon $[0, T]$.

P1-2 $\forall t \geq 0$, $\sigma(t)$ satisfies $h_{\sigma(t)}(\mathbf{x}(t)) \geq 0$, and $\exists \mathbf{u} \in Int(\mathcal{U})$ with

$$L_{\mathbf{f}} h_{\sigma(t)}(\mathbf{x}(t)) + L_{\mathbf{G}} h_{\sigma(t)}(\mathbf{x}(t))\mathbf{u} > -\kappa h_{\sigma(t)}(\mathbf{x}(t)) \quad (5)$$

where κ is a positive constant.

Then, $\forall \mathbf{x}' \in \mathcal{X}_c$, $\mathbf{x}(0) = \mathbf{x}'$ implies $\mathbf{x}(t) \in \mathcal{X}_c, \forall t \geq 0$.

Proof: For any $T > 0$, we divide the time range $[0, T]$ into $[0, T] = \cup_{k=1}^{K+1} [t_{k-1}, t_k)$ where $t_0 = 0$, $t_{K+1} = T$, and $\sigma(t)$ is constant in time interval $[t_{k-1}, t_k)$, $\forall k = 1, \ldots, K+1$. Let $\mathcal{K} = \{1, ..., K\}$. So that switches happen at $t_k, \forall k \in \mathcal{K}$. For all $t \in [t_{k-1}, t_k)$ where $k = 0, \ldots, K+1$, $\sigma(t) = \sigma(t_{k-1}) \in \mathcal{P}$. According to P1-2, constraints (4b) and (4c) always satisfy the Slater's condition when $\sigma(t) = p$. This means for each $k = 0, \ldots, K$ and for all $t \in [t_k, t_{k+1})$, the controller $\mathbf{k}_{\sigma(t)}(\mathbf{x})$ is point-Lipschitz. Hence, the closed-loop system (1) with controller $\mathbf{u} = \mathbf{k}_{\sigma(t)}(\mathbf{x})$ is a switched system [25, Part-III] whose dynamics is continuous with respect to $\mathbf{x}$ in between switches. This implies that the closed-loop trajectory $\mathbf{x}(t)$ is continuously differentiable in between switches. Hence, the closed-loop dynamics would be piecewise continuous with respect to t for all $t \geq 0$. Hence, we have that the closed-loop trajectory $\mathbf{x}(t)$ is absolutely continuous for all $t \geq 0$ [25, Chapter 1.2].

During the time interval $[t_0, t_1)$: By condition P1-1, $\forall t \in [t_0, t_1)$, $\sigma(t) = \sigma(t_0)$, and the controller for the system (1) is $\mathbf{u} = \mathbf{k}_p(\mathbf{x})$ with $p \in \mathcal{P}$ such that $h_p(\mathbf{x}(t_0)) \geq 0$. By condition 1-2, the constraints for controller $\mathbf{k}_p(\mathbf{x})$ satisfy Slater's Conditions. Also, the CBF $h_p(\mathbf{x})$ is a minimal CBF since (5) requires a linear extended class-κ function. Hence, by Theorem 1, $\mathbf{x}(t_0) \in \mathcal{X}_c$ implies $\mathbf{x}(t) \in \mathcal{X}_p \subseteq \mathcal{X}_c$, $\forall t \in [t_0, t_1)$, where p satisfies $h_p(\mathbf{x}(t_0)) \geq 0$.

During the time interval $[t_k, t_{k+1})$: We assume that $\mathbf{x}(t) \in \mathcal{X}_c$ for all $t \in [t_{k-1}, t_k)$. Since $\mathbf{x}(t)$ is absolutely continuous, then $\lim_{t \to t_k^+} \mathbf{x}(t) = \lim_{t \to t_k^-} \mathbf{x}(t) = \mathbf{x}(t_k) \in \mathcal{X}_c$. According to P1-1, $\sigma(t) = \sigma(t_k) = p' \in \mathcal{P}$ for all $t \in [t_k, t_{k+1})$, where p' satisfies $h_{p'}(\mathbf{x}(t_k)) \geq 0$. By P1-2, the control constraints for $\mathbf{k}_{p'}(\mathbf{x})$ also satisfy Slater's Condition, and $h_{p'}(\mathbf{x})$ is a minimal CBF. Hence, $\mathbf{x}(t) \in \mathcal{X}_{p'} \subseteq \mathcal{X}_c$ for all $t \in [t_k, t_{k+1})$

By induction, $\mathbf{x}(t) \in \mathcal{X}_c$ for all $t \in [0, T]$, and for all $T > 0$. ■

The conditions in Proposition 1 not only depends on $h_1(\mathbf{x}), ..., h_{N_h}(\mathbf{x})$, but also depends on the switching signal $\sigma(t)$. Hence, in the next section, in order to verify Proposition 1, we first specify the switching strategy, then propose some conditions for $h_1(\mathbf{x}), \ldots, h_{N_h}(\mathbf{x})$ such that the switching signal $\sigma(t)$ together with $h_1(\mathbf{x}), \ldots, h_{N_h}(\mathbf{x})$ satisfy P1-1 to P1-2.

IV. Forward Invariance Conditions Under Different Switching Strategies

In this section we provide two switching strategies and provide conditions for forward invariance under these switching strategies. All sufficient conditions proposed in this section depends only on $h_1(\mathbf{x})$, ..., $h_{N_h}(\mathbf{x})$.

Before we introduce the details of the switching strategies and the corresponding forward invariance conditions, we first provide a result that will be used during the following discussion of this paper.

Theorem 3: Let $f : \mathbb{R}^{n_x} \to \mathbb{R}$ be a locally Lipschitz function with Lipschitz constant L on a compact set $\mathcal{W} \subseteq \mathbb{R}^{n_x}$. Let $\eta > 0$ be a constant. For all $\mathbf{x}, \mathbf{x}' \in \mathcal{W}$, we have that

$$\inf \{||\mathbf{x}' - \mathbf{x}|| : f(\mathbf{x}) \geq \eta, f(\mathbf{x}') \leq 0\} > \frac{\eta}{L}$$

Proof: According to the local Lipschitz property of the function $f(\mathbf{x})$, for all $\mathbf{x}, \mathbf{x}' \in \mathcal{W}$ we have:

$$\begin{aligned} &|f(\mathbf{x}') - f(\mathbf{x})| \leq L||\mathbf{x}' - \mathbf{x}|| \\ \Rightarrow &- L||\mathbf{x}' - \mathbf{x}|| \leq f(\mathbf{x}') - f(\mathbf{x}) \leq L||\mathbf{x}' - \mathbf{x}|| \\ \Rightarrow &f(\mathbf{x}') - L||\mathbf{x}' - \mathbf{x}|| \leq f(\mathbf{x}) \leq f(\mathbf{x}') + L||\mathbf{x}' - \mathbf{x}|| \end{aligned}$$

Since $\mathbf{x}'$ satisfies $f(\mathbf{x}') \leq 0$, then we have:

$$f(\mathbf{x}) \leq f(\mathbf{x}') + L||\mathbf{x}' - \mathbf{x}|| \leq L||\mathbf{x}' - \mathbf{x}||$$

Also, since $f(\mathbf{x}) \geq \eta > 0$, we have

$$||\mathbf{x}' - \mathbf{x}|| \geq \frac{\eta}{L}.$$

Hence, $\frac{\eta}{L}$ is the lower bound for the distance from any $\mathbf{x} \in \mathcal{W}$ with $f(\mathbf{x}) \geq \eta$ to any other $\mathbf{x}' \in \mathcal{W}$ with $f(\mathbf{x}') \leq 0$. ■

A. Switching Strategy I

We give the first switching strategy as follows. Let $\eta > 0$ be a constant. The signal $\sigma(t)$ switches from $p \in \mathcal{P}$ to another $p' \in \mathcal{P}$ at time t if the following conditions are satisfied.

S1-1 $h_p(\mathbf{x}(t)) \geq 0$ and $h_{p'}(\mathbf{x}(t)) \geq 0$.

S1-2 At state $\mathbf{x} = \mathbf{x}(t)$,

$$\sup_{\mathbf{u} \in Int(\mathcal{U})} \{L_{\mathbf{f}} h_p(\mathbf{x}) + L_{\mathbf{G}} h_p(\mathbf{x})\mathbf{u} + \kappa h_p(\mathbf{x})\} \leq 0.$$

S1-3 At state $\mathbf{x} = \mathbf{x}(t)$, $\exists \mathbf{u} \in Int(\mathcal{U})$ such that

$$L_{\mathbf{f}} h_{p'}(\mathbf{x}) + L_{\mathbf{G}} h_{p'}(\mathbf{x})\mathbf{u} + \kappa h_{p'}(\mathbf{x}) \geq \eta.$$

The following theorem gives conditions for forward invariance under this switching strategy.

Theorem 4: Let $h_1(\mathbf{x})$, ..., $h_{N_h}(\mathbf{x})$ be a collection of given differentiable functions with their first order derivatives being locally Lipschitz on $\mathcal{X}_c$. If $\forall \mathbf{x} \in \mathcal{X}_c$, $\exists p \in \mathcal{P}, \mathbf{u} \in Int(\mathcal{U})$ with

C1-1 $h_p(\mathbf{x}) \geq 0$.

C1-2 $L_{\mathbf{f}}h_p(\mathbf{x}) + L_{\mathbf{G}}h_p(\mathbf{x})\mathbf{u} + \kappa h_p(\mathbf{x}) \geq \eta$.

Then, any switching signal $\sigma(t)$ following conditions S1-1 to S1-3 satisfies P1-1 and P1-2.

Proof: We first prove P1-1 and then prove P1-2.

Proof of P1-1: Assume $\sigma(t)$ switches to p at time t_k, and the next switch of $\sigma(t)$ from p to another index in $\mathcal{P}$ happens at t_{k+1}. In order to prove P1-1, we first prove that there exists a uniform lower bound for $||\mathbf{x}(t_{k+1}) - \mathbf{x}(t_k)||$, then we show that there exists a uniform lower bound on $t_{k+1} - t_k$ that holds for all $k = 0, 1, \ldots$. According to the problem setup, $\mathbf{f}(\mathbf{x})$ and $\mathbf{G}(\mathbf{x})$ of system (1) are locally Lipschitz for all $\mathbf{x} \in \mathcal{X}$. According to condition S1-3 of Switching Strategy 1, at $\mathbf{x}(t_k)$, there exists a $\mathbf{u}_0 \in Int(\mathcal{U})$ such that $L_{\mathbf{f}}h_p(\mathbf{x}(t_k)) + L_{\mathbf{G}}h_p(\mathbf{x}(t_k))\mathbf{u}_0 + \kappa h_p(\mathbf{x}(t_k)) \geq \eta$. We now define $f_0(\mathbf{x}) := L_{\mathbf{f}}h_p(\mathbf{x}) + L_{\mathbf{G}}h_p(\mathbf{x})\mathbf{u}_0 + \kappa h_p(\mathbf{x})$. Since $\mathbf{f}(\mathbf{x})$, $\mathbf{G}(\mathbf{x})$ and $\partial h_p(\mathbf{x})/\partial \mathbf{x}$ are locally Lipchitz on $\mathcal{X}$, then the function $f_0(\mathbf{x})$ is also locally Lipschitz on $\mathcal{X}$. Let the Lipschitz constant of $f_0(\mathbf{x})$ be L_0. By Theorem 3, for any $\mathbf{x}' \in \mathcal{X}_p$ such that $f_0(\mathbf{x}') \leq 0$, we have $||\mathbf{x}' - \mathbf{x}(t_k)|| \geq \frac{\eta}{L_0}$. Define the following two sets:

$$\mathcal{R} := \{\mathbf{x}| \max_{\mathbf{u} \in Int(\mathcal{U})} \{L_{\mathbf{f}}h_p(\mathbf{x}) + L_{\mathbf{G}}h_p(\mathbf{x})\mathbf{u} + \kappa h_p(\mathbf{x})\} \leq 0\}$$

$$\mathcal{R}_0 := \{\mathbf{x}| L_{\mathbf{f}}h_p(\mathbf{x}) + L_{\mathbf{G}}h_p(\mathbf{x})\mathbf{u}_0 + \kappa h_p(\mathbf{x}) \leq 0\}$$

where $\mathbf{u}_0 \in Int(\mathcal{U})$. It is obvious that $\mathcal{R} \subseteq \mathcal{R}_0$. Since we have $\mathbf{x}(t_k) \notin \mathcal{R}$, $\mathbf{x}(t_k) \notin \mathcal{R}_0$, and $\mathbf{x}(t_{k+1}) \in \mathcal{R}$, we have that

$$||\mathbf{x}(t_{k+1}) - \mathbf{x}(t_k)|| \geq ||\mathbf{x}' - \mathbf{x}(t_k)|| \geq \frac{\eta}{L_0}$$

Hence, we have a uniform lower bound for the term $||\mathbf{x}(t_{k+1}) - \mathbf{x}(t_k)|| \geq \frac{\eta}{L_0}$. Since we also assume that $\mathcal{X}$ and $\mathcal{U}$ are bounded sets, then there exists a constant $D > 0$ such that $||\mathbf{f}(\mathbf{x}(t)) + \mathbf{G}(\mathbf{x}(t))\mathbf{u}(t)|| \leq D$, $\forall t \geq 0$. Hence, we now have a uniform lower bound for $||t_{k+1} - t_k|| \geq \frac{\eta}{DL_0}$ for all $k = 0, 1, \ldots$. Condition P1-1 is now proved.

Proof of P1-2: If conditions C1-1 and C1-2 are satisfied, then all possible switching signals under Switching Strategy 1 satisfy P1-2. Hence, P1-2 is proved. ■

B. Switching Strategy II

We consider another switching strategy as follows. Let $\eta_h, \tau > 0$ be positive constants. The switching signal $\sigma(t)$ is initialized as $p_0 \in \mathcal{P}$ such that $h_{p_0}(\mathbf{x}(0)) \geq 0$, then, $\sigma(t) = p_0$ for all $t \in [0, \tau]$. After that, $\sigma(t)$ switches from a $p \in \mathcal{P}$ to another $p' \in \mathcal{P}$ at time t if $h_p(\mathbf{x}(t)) = 0$ and $h_{p'}(\mathbf{x}(t)) \geq \eta_h$.

Under the Switching Strategy 2, we have the union CBF condition provided in the following theorem.

Theorem 5: Let $h_1(\mathbf{x}), \ldots, h_{N_h}(\mathbf{x})$ be a collection of given polynomials. Let $\eta > 0$ be a constant. If $\forall p \in \mathcal{P}$ and $\forall \mathbf{x} \in \mathcal{X}_p$, $\exists \mathbf{u} \in Int(\mathcal{U})$ such that

C2-1 $L_{\mathbf{f}}h_p(\mathbf{x}) + L_{\mathbf{G}}h_p(\mathbf{x})\mathbf{u} + \kappa h_p(\mathbf{x}) \geq \eta$.

Then, any switching signal $\sigma(t)$ following the Switching Strategy 2 satisfies conditions P1-1 and P1-2.

Proof: We prove Theorem 5 by considering the following two cases.

(a) $\forall \mathbf{x} \in \mathcal{X}_c$, $\nexists p, p' \in \mathcal{P}$ such that $p \neq p'$, $h_p(\mathbf{x}) = 0$, and $h'_p(\mathbf{x}) \geq \eta_h$.

(b) $\exists \mathbf{x} \in \mathcal{X}_c$ such that $h_p(\mathbf{x}) = 0$ and $h_{p'}(\mathbf{x}) \geq \eta_h$ for some $p, p' \in \mathcal{P}$ and $p \neq p'$.

Proof of P1-1: According to condition C2-1, in case (a), once the switching signal $\sigma(t)$ is initialized as p_0 such that $h_{p_0}(\mathbf{x}(0)) \geq 0$, the closed-loop system (1) with CBF-QP controller $\mathbf{u}_{p_0}(\mathbf{x})$ is forward invariant to set $\{\mathbf{x}|h_{p_0}(\mathbf{x}) \geq 0\} \subseteq \mathcal{X}_c$. Hence, the switching signal keeps its value $\sigma(t) = p_0$ for all $t \geq 0$.

In case (b), the switching signal $\sigma(t)$ may switch at some $t > \tau$. In this case, the positive constant τ ensures that $\sigma(t)$ does not have a switch in an arbitrary small time range after the initialization $\sigma(0) = p_0$. After the time τ, we let $t_k > \tau$ be the time slot when $\sigma(t)$ switches to $p \in \mathcal{P}$ such that $h_p(\mathbf{x}(t_k)) \geq \eta_h > 0$. We also denote t_{k+1} as the time when $h_p(\mathbf{x}(t_{k+1})) = 0$ and $\sigma(t)$ switches away from p. Since, $\forall p \in \mathcal{P}$, $h_p(\mathbf{x})$ is a polynomial and is locally Lipschitz on $\mathcal{X}$. Let the Lipschitz constant be L_p, and let $L_h = \max_{p \in \mathcal{P}} L_p$. By Theorem 3, we always have that $||\mathbf{x}(t_{k+1}) - \mathbf{x}(t_k)|| \geq \frac{\eta_h}{L_p}$. Since we assumed that $\mathcal{X}$ and $\mathbf{U}$ are bounded sets, then there exists constant $D > 0$ such that $||\mathbf{f}(\mathbf{x}(t)) + \mathbf{G}(\mathbf{x}(t))\mathbf{u}(t)|| \leq D$, $\forall t \geq 0$. Hence, we now have a uniform lower bound for $||t_{k+1} - t_k|| \geq \frac{\eta_h}{DL_h}$ for all $k = 0, 1, \ldots$. Hence, we prove the P1-1 for Theorem 5 in case (b).

Proof of P1-2: Since $\sigma(t) \in \mathcal{P}$ for all $t \geq 0$, then, condition C2-1 of Theorem 3 implies P1-2 in both (a) and (b) cases. ■

Remark: Since condition C2-1 also implies C1-2, the union-CBFs condition in Theorem 5 is also a sufficient condition for Theorem 4. This means if the condition in Theorem 5 is true for a given collection of CBFs $h_1(\mathbf{x}), \ldots, h_{N_h}(\mathbf{x})$, then we can also use the switching CBF-QP with strategy I to keep system (1) forward invariant with respect to $\mathcal{X}_c$.

V. Verification Framework

This section introduces SOS frameworks to verify sufficient conditions in Theorem 4 and Theorem 5. Given a polynomial system dynamics (1) and a collection of CBFs $h_1(\mathbf{x}), \ldots, h_{N_h}(\mathbf{x})$, verifying the union of these CBFs is to verify (a) Theorem 4 or Theorem 5 is true, and (b) validity of these CBFs, meaning that $\mathcal{X}_c \subseteq \mathcal{X}_s$. Hence, in what follows, we first introduce the SOS verification frameworks for Theorem 4. Then we introduce the verification framework for Theorem 5. Finally, we verify that $\mathcal{X}_c \subseteq \mathcal{X}_s$.

We use $\mathcal{S}^{\mathcal{P}}$ to denote the collection of all possible subsets of $\mathcal{P}$ except $\emptyset$. During the verification of Theorem 4 and Theorem 5, we let $\bar{\mathcal{U}} = \{\mathbf{u}|\mathbf{A}\mathbf{u} \leq \mathbf{c} - \epsilon\}$ with a given constant $\epsilon > 0$ so that $\bar{\mathcal{U}} \subset Int(\mathcal{U})$. For simplicity, we also define matrix $\mathbf{\Lambda}_p(\mathbf{x})$ and vector $\boldsymbol{\xi}_p(\mathbf{x})$ such that

$$\mathbf{\Lambda}_p(\mathbf{x}) = \begin{bmatrix} -L_{\mathbf{G}}h_p(\mathbf{x}) \\ \mathbf{A} \end{bmatrix}; \boldsymbol{\xi}_p(\mathbf{x}) = \begin{bmatrix} L_{\mathbf{f}}h_p(\mathbf{x}) + \kappa h_p(\mathbf{x}) - \eta \\ \mathbf{c} - \epsilon \end{bmatrix} \tag{6}$$

where η, ϵ are specified positive constants. Hence, for any $\mathbf{u}$ with $\mathbf{\Lambda}_p(\mathbf{x})\mathbf{u} \leq \boldsymbol{\xi}_p(\mathbf{x})$, $\mathbf{u}$ strictly satisfies constraints (4b) and (4c).

A. Framework For Switching Strategy I

According the defined symbols above, we verify Theorem 4 by solving the following problem.

Verification Problem 1: Given polynomials $h_1(\mathbf{x}),\ldots,h_{N_h}(\mathbf{x})$, system model (1), positive constants κ, η, ϵ and constant matrix $\mathbf{A}$ and vector $\mathbf{c}$, verify that, $\forall \mathbf{x}\in\mathcal{X}_c$, $\exists p\in\mathcal{P}$ and $\mathbf{u}\in\mathbb{R}^{n_u}$ such that $h_p(\mathbf{x})\geq 0$ and $\Lambda_p(\mathbf{x})\mathbf{u}\leq\xi_p(\mathbf{x})$.

We solve Problem 1 by three steps. First, we partition the set $\mathcal{X}_c$. Next, we collect all the non-empty subsets of $\mathcal{X}_c$. Finally, for each of these non-empty subsets, we verify that, for all states $\mathbf{x}$ in the subset, there exists $p\in\mathcal{P}$ and $\mathbf{u}\in\mathbb{R}^{n_u}$ such that $h_p(\mathbf{x})\geq 0$ and $\Lambda_p(\mathbf{x})\mathbf{u}\leq\xi_p(\mathbf{x})$.

Step 1: We partition the set $\mathcal{X}_c$ into subsets such that $\mathcal{X}_c=\cup_{\mathcal{N}\in\mathcal{S}^{\mathcal{P}}}\mathcal{X}_{\mathcal{N}}$, where $\mathcal{X}_{\mathcal{N}}$ for all $\mathcal{N}\in\mathcal{S}^{\mathcal{P}}$ is defined as

$$\mathcal{X}_{\mathcal{N}}=\{\mathbf{x}|h_p(\mathbf{x})\geq 0,\forall p\in\mathcal{N};h_{p'}(\mathbf{x})<0,\forall p'\in\mathcal{P}\setminus\mathcal{N}\}.$$

To solve Problem 1, we consider the following two statements.

T1: $\forall\mathcal{N}\in\mathcal{S}^{\mathcal{P}}$ and $\forall\mathbf{x}\in\mathcal{X}_{\mathcal{N}}$, $\exists p\in\mathcal{N}$ and $\mathbf{u}\in\mathbb{R}^{n_u}$ such that $\Lambda_p(\mathbf{x})\mathbf{u}\leq\xi_p(\mathbf{x})$.

T2: $\forall\mathbf{x}\in\mathcal{X}_{\mathcal{N}}$, $\exists p\in\mathcal{N}$ and $\mathbf{u}\in\mathbb{R}^{n_u}$ such that $\Lambda_p(\mathbf{x})\mathbf{u}\leq\xi_p(\mathbf{x})$.

According to the partition for $\mathcal{X}_c$ and definition of $\mathcal{X}_{\mathcal{N}}$, verifying T1 is sufficient to verify Theorem 4, and is equivalent to iteratively verifying T2 over all $\mathcal{N}\in\mathcal{S}^{\mathcal{P}}$.

Step 2: In the most general case where $\mathcal{X}_{\mathcal{N}}\neq\emptyset$ for all $\mathcal{N}\in\mathcal{S}^{\mathcal{P}}$, the verification of T2 is repeated for 2^N-1 times. However, in cases where $\mathcal{X}_1,...,\mathcal{X}_{N_h}$ are located sparsely, there are some $\mathcal{N}\in\mathcal{S}^{\mathcal{P}}$ such that $\mathcal{X}_{\mathcal{N}}=\emptyset$. Verifying T1 only requires to repeat the verification of T2 for all $\mathcal{X}_{\mathcal{N}}\neq\emptyset$. In this step, we check whether $\mathcal{X}_{\mathcal{N}}=\emptyset$ for all $\mathcal{N}\in\mathcal{S}^{\mathcal{P}}$ so that, at the end of this step, we should have the collection $\mathcal{S}_{ne}=\{\mathcal{N}\in\mathcal{S}^{\mathcal{P}}|\mathcal{X}_{\mathcal{N}}\neq\emptyset\}$ where the subscript ne stands for "non-empty".

The following lemma provides an SOS program to check whether $\mathcal{X}_{\mathcal{N}}=\emptyset$.

Lemma 2: $\forall\mathcal{N}\in\mathcal{S}^{\mathcal{P}}$, $\mathcal{X}_{\mathcal{N}}=\emptyset$ if the following SOS program is feasible.

$$\text{Find } s_p(\mathbf{x},\mathbf{c}),\forall p\in\mathcal{N},\text{ and } s_{p'}(\mathbf{x},\mathbf{c}),\forall p'\in\mathcal{P}\setminus\mathcal{N} \tag{7a}$$

$$\text{s.t. } -1-\sum_{p\in\mathcal{N}}s_p(\mathbf{x},\mathbf{c})h_p(\mathbf{x}) - \sum_{p'\in\mathcal{P}\setminus\mathcal{N}}q_{p'}(\mathbf{x},\mathbf{c})[c_{p'}^2h_{p'}(\mathbf{x})+1]\text{ is SOS} \tag{7b}$$

$$s_p(\mathbf{x},\mathbf{c}),\forall p\in\mathcal{N}\text{ are SOS} \tag{7c}$$

where $\mathbf{c}$ is a vector of scalar variables $c_{p'}$ with $p'\in\mathcal{P}\setminus\mathcal{N}$, and each $q_{p'}(\mathbf{x},\mathbf{c})$is a polynomial of $\mathbf{x}$ and $\mathbf{c}$.

Proof: For a function $h:\mathbb{R}^{n_x}\to\mathbb{R}$, $\exists\mathbf{x}$ with $h(\mathbf{x})<0$ if and only if $\exists(\mathbf{x},c)\in\mathbb{R}^{n_x+1}$ such that $c^2h(\mathbf{x})=-1$. Hence, $\mathcal{X}_{\mathcal{N}}$ is empty if and only if the following set is empty.

$$\{(\mathbf{x},\mathbf{c})|h_p(\mathbf{x})\geq 0,\forall p\in\mathcal{N};c_{p'}^2h_{p'}(\mathbf{x})+1=0,\forall p'\in\mathcal{P}\setminus\mathcal{N}\}$$

where $\mathbf{c}$ is a vector of scalar variables $c_{p'}$ with $p'\in\mathcal{P}\setminus\mathcal{N}$. Then, by Theorem 2, we have the SOS program (7). ■

Step 3: In this step, we verify T2 for each $\mathcal{X}_{\mathcal{N}}$ with $\mathcal{N}\in\mathcal{S}_{ne}$. We can see from Lemma 2 that the terms $h_{p'}(\mathbf{x})<0$ in set $\mathcal{X}_{\mathcal{N}}$ could introduce extra variables to SOS programs. This complicates polynomial structures and makes SOS program harder to solve. Hence, instead of consider $\mathcal{X}_c$, in this step, we consider $\mathcal{X}'_{\mathcal{N}}$ defined as

$$\mathcal{X}'_{\mathcal{N}}=\{\mathbf{x}|h_p(\mathbf{x})\geq 0,\forall p\in\mathcal{N};h_{p'}(\mathbf{x})\leq 0,\forall p'\in\mathcal{P}\setminus\mathcal{N}\}.$$

Instead of verifying T2, we verify the following statement.

T3: $\forall\mathbf{x}\in\mathcal{X}'_{\mathcal{N}}$, $\exists p\in\mathcal{N}$ and $\mathbf{u}\in\mathbb{R}^{n_u}$ such that $\Lambda_p(\mathbf{x})\mathbf{u}\leq\xi_p(\mathbf{x})$.

Since $\mathcal{X}_{\mathcal{N}}\subset\mathcal{X}'_{\mathcal{N}}$, then verifying T3 also verifies T2.

The following theorem provides the SOS program to verify T3 for all $\mathcal{N}\in\mathcal{S}_{ne}$.

Theorem 6: For all $\mathcal{N}\in\mathcal{S}_{ne}$, $\mathcal{X}'_{\mathcal{N}}$ satisfies T3 if the following SOS program is feasible.

$$\text{Find } s_p(\mathbf{x},\mathbf{y}),\mathbf{q}_p(\mathbf{x},\mathbf{y}),r_p(\mathbf{x},\mathbf{y}),\forall p\in\mathcal{N}\text{ and } s_{p'}(\mathbf{x},\mathbf{y}),\forall p'\in\mathcal{P}\setminus\mathcal{N} \tag{8a}$$

$$\text{s.t. } -1-\sum_{p\in\mathcal{N}}s_p(\mathbf{x},\mathbf{y})h_p(\mathbf{x})+\sum_{p'\in\mathcal{P}\setminus\mathcal{N}}s_{p'}(\mathbf{x},\mathbf{y})h_{p'}(\mathbf{x}) - \sum_{p\in\mathcal{N}}\Big(\mathbf{q}_p^T(\mathbf{x},\mathbf{y})\mathbf{\Lambda}_p^T(\mathbf{x})\mathbf{y}_p^2+r_p(\mathbf{x},\mathbf{y})(\boldsymbol{\xi}_p(\mathbf{x})\mathbf{y}_p^2+1)\Big)\text{ is SOS} \tag{8b}$$

$$s_p(\mathbf{x},\mathbf{y}),\forall p\in\mathcal{N}\text{ are SOS} \tag{8c}$$

$$s_{p'}(\mathbf{x},\mathbf{y}),\forall p'\in\mathcal{P}\setminus\mathcal{N}\text{ are SOS} \tag{8d}$$

where $\mathbf{y}$ is a vector of variables concatenated by all $\mathbf{y}_p$ for all $p\in\mathcal{N}$, each $\mathbf{q}_p(\mathbf{x},\mathbf{y})$ is a vector of polynomials, and each $r_p(\mathbf{x},\mathbf{y})$ is a polynomial.

Proof: The the following T4 statement is the negation of T3:

T4: $\exists\mathbf{x}\in\mathcal{X}'_{\mathcal{N}}$ such that, $\forall p\in\mathcal{N}$, $\nexists\mathbf{u}_p$ with $\mathbf{\Lambda}_p(\mathbf{x})\mathbf{u}_p\leq\boldsymbol{\xi}_p(\mathbf{x})$

By Lemma 1, T4 is equivalent to the *non-emptiness* of the following set.

$$\mathcal{V}=\left\{(\mathbf{x},\mathbf{y})\;\middle|\;\begin{array}{l}h_p(\mathbf{x})\geq 0,\forall p\in\mathcal{N}\\ -h_{p'}(\mathbf{x})\geq 0,\forall p'\in\mathcal{P}\setminus\mathcal{N}\\ \mathbf{\Lambda}_p^T(\mathbf{x})\mathbf{y}_p^2=\mathbf{0},\boldsymbol{\xi}_p^T(\mathbf{x})\mathbf{y}_p^2+1=0,\forall p\in\mathcal{N}\end{array}\right\}$$

where the $\mathbf{y}$ is a vector of variables concatenated by all $\mathbf{y}_p$ for all $p\in\mathcal{N}$. The first two lines of $\mathcal{V}$ correspond to $\mathcal{X}'_{\mathcal{N}}$, and the last line of $\mathcal{V}$ corresponds to the result of Farkas' Lemma for each $p\in\mathcal{N}$. Since T4 is the negation of T3, and T4 is equivalent to the non-emptiness of $\mathcal{V}$, then T3 is equivalent to the emptiness of $\mathcal{V}$. Now, applying Theorem 2, we get the SOS program (8) that verifies T3. ■

To summarize, in this subsection, we verify sufficient conditions for Theorem 4. We first compute all possible subsets of $\mathcal{X}_c$. Second, we use Program (7) to discard all the empty subsets of $\mathcal{X}_c$ and returns all the non-empty subsets. Finally, we verify a sufficient condition for T3 using Program (8) for all the non-empty subsets.

B. Framework For Switching Strategy II

In this subsection, we propose the SOS programs that verifies Theorem 5.

Verification Problem 2: Given a collection of polynomials $h_1(\mathbf{x}), \ldots, h_{N_h}(\mathbf{x})$, system model (1), positive constants $\kappa, \eta > 0$, and constant matrix and vector $\mathbf{A}, \mathbf{c}$, verify that for all $p \in \mathcal{P}$ and $\mathbf{x} \in \mathcal{X}_p$, there exists $\mathbf{u} \in \mathbb{R}^{n_u}$ with $\mathbf{\Lambda}_p(\mathbf{x})\mathbf{u} \leq \boldsymbol{\xi}_p(\mathbf{x})$.

Theorem 7: Problem 2 can be verified by checking the feasibility of the following SOS program for all $p \in \mathcal{P}$.

$$\text{Find } s(\mathbf{x},\mathbf{y}), \mathbf{q}(\mathbf{x},\mathbf{y}), r(\mathbf{x},\mathbf{y}) \tag{9a}$$

$$\text{s.t. } -1 - s(\mathbf{x},\mathbf{y})h_p(\mathbf{x}) - \mathbf{q}(\mathbf{x},\mathbf{y})\mathbf{\Lambda}_p^T(\mathbf{x})\mathbf{y}^2 - r(\mathbf{x},\mathbf{y})(\boldsymbol{\xi}_p^T(\mathbf{x})\mathbf{y}^2 + 1) \text{ is SOS} \tag{9b}$$

$$s(\mathbf{x},\mathbf{y}) \text{is SOS} \tag{9c}$$

where $\mathbf{y}$ is a vector of variables, $\mathbf{q}(\mathbf{x},\mathbf{y})$ is a vector of polynomials, and $r(\mathbf{x},\mathbf{y})$ is a polynomial.

Proof: By Lemma 1, for all $\mathbf{x} \in \mathcal{X}_p$, $\exists \mathbf{u} \in \mathbb{R}^{n_u}$ such that $\mathbf{\Lambda}_p(\mathbf{x})\mathbf{u} \leq \boldsymbol{\xi}_p(\mathbf{x})$ if and only if the following set is empty.

$$\mathcal{V}_s = \{(\mathbf{x},\mathbf{y}) | h_p(\mathbf{x}) \geq 0, \mathbf{\Lambda}_p^T(\mathbf{x})\mathbf{y}^2 = \mathbf{0}, \boldsymbol{\xi}_p^T(\mathbf{x})\mathbf{y}^2 + 1 = 0\}$$

Applying Theorem 2 to $\mathcal{V}_s$, we get the SOS Program (9). Problem 2 is verified if Program (9) is feasible for all $p \in \mathcal{P}$. ■

Remark: Comparing Program (9) to Program (8), we can see the verification of Problem 2 is in fact a special case for verification of Problem 1. During the verification of Problem 1, if we replace $\mathcal{N}$ as the index set $\{p\}$, and replace $\mathcal{X}_\mathcal{N}$ as $\mathcal{X}_p = \{\mathbf{x} | h_p(\mathbf{x}) \geq 0\}$, then Program (8) and Program (9) would be the same. This observation coincides with the remark in Section IV-B that Theorem 5 is a sufficient condition of Theorem 4.

C. Verifying Validity of CBFs

In this subsection, we verify that $\mathcal{X}_c \subseteq \mathcal{X}_s$. Assume the unsafe region $\mathcal{X}_u = \mathcal{X} \setminus \mathcal{X}_s$ is identified by a collection of polynomials such that $\mathcal{X}_u = \{\mathbf{x} | \mathbf{l}(\mathbf{x}) \leq 0\}$ where $\mathbf{l}(\mathbf{x})$ is a vector of polynomials. In order to verify that $\mathcal{X}_c \subseteq \mathcal{X}_s$, we need to verify that $\mathcal{X}_c \cap \mathcal{X}_u = \emptyset$. Since $\mathcal{X}_c = \cup_{p \in \mathcal{P}} \mathcal{X}_p$, then verifying $\mathcal{X}_c \cap \mathcal{X}_u = \emptyset$ is equivalent to verify $\mathcal{X}_p \cap \mathcal{X}_u = \emptyset$ for all $p \in \mathcal{P}$. Hence, to verify $\mathcal{X}_c \subseteq \mathcal{X}_s$, we verify that the set $\{\mathbf{x} | h_p(\mathbf{x}) \geq 0, -\mathbf{l}(\mathbf{x}) \geq 0\}$ is empty for all $p \in \mathcal{P}$. By Theorem 2, this is to solve the following SOS feasibility program.

$$\text{Find } s(\mathbf{x}), \mathbf{s}_l(\mathbf{x}) \tag{10a}$$

$$\text{s.t. } -1 - s(\mathbf{x})h_p(\mathbf{x}) - \mathbf{s}_l^T(\mathbf{x})\mathbf{l}(\mathbf{x}) \text{ is SOS} \tag{10b}$$

$$s(\mathbf{x}), \mathbf{s}_l(\mathbf{x}) \text{ are SOS} \tag{10c}$$

where $\mathbf{s}_l(\mathbf{x})$ is a vector of SOS with the same size as vector $\mathbf{l}(\mathbf{x})$.

VI. Experiments

This section presents experiments that support our motivation and compares the efficiency of our proposed verification frameworks. The SOS verification programs are solved using MOSEK [24] for all examples. For simplicity, we name the verification methods for Theorem 4 and Theorem 5 as Verif-I and Verif-II, respectively.

A. Motivating Example

In this example, we show that a union of linear CBFs covers a larger safe region than a degree-6 single CBF, which has a longer verification time.

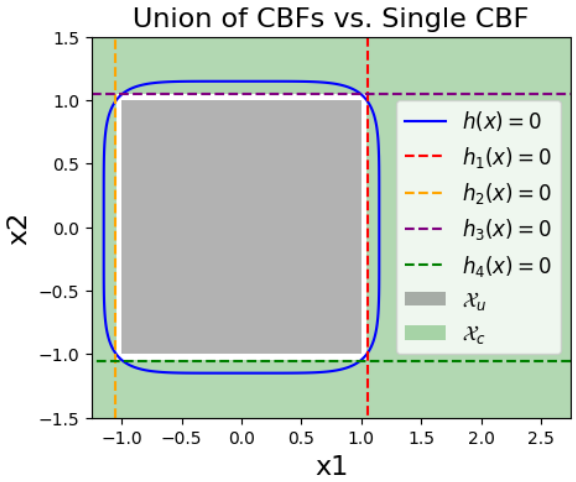


Fig. 1: Safe states coverage comparison between a union of 4 linear CBFs and a degree-6 CBF. Although the boundary of $\mathcal{X}_h = \{\mathbf{x} | h(\mathbf{x}) \geq 0\}$ almost touches the boundary of $\mathcal{X}_u$, the union region $\mathcal{X}_c$ still provides a better approximation of $\mathcal{X}_s$ and covers more safe states than $\mathcal{X}_h$.

We consider a 2D single integrator $\dot{\mathbf{x}} = \mathbf{u}$ with control limits $-1 \leq \mathbf{u}_{(1)} \leq 1$ and $-1 \leq \mathbf{u}_{(2)} \leq 1$. The unsafe region has a square shape such that $\mathcal{X}_u = \{\mathbf{x} | -1 \leq \mathbf{x}_{(1)} \leq 1, -1 \leq \mathbf{x}_{(2)} \leq 1\}$. We choose 4 linear CBFs as $h_1(\mathbf{x}) = \mathbf{x}_1 - 1.05$, $h_1(\mathbf{x}) = -\mathbf{x}_1 - 1.05$, $h_1(\mathbf{x}) = \mathbf{x}_2 - 1.05$, and $h_1(\mathbf{x}) = -\mathbf{x}_2 - 1.05$. The degree-6 CBF is chosen as $h(\mathbf{x}) = \mathbf{x}_1^6 + \mathbf{x}_2^6 - 2.30$, and we let $\mathcal{X}_h = \{\mathbf{x} | h(\mathbf{x}) \geq 0\}$. The boundary of these CBFs, the region $\mathcal{X}_c$ and the region $\mathcal{X}_u$ are shown in Fig. 1. We can see that the union of low degree CBFs covers more safe states than a high-degree polynomial.

Next, we verify the union of $h_1(\mathbf{x}), \ldots, h_4(\mathbf{x})$ using Verif-I and Verif-II. Then, we verify $h(\mathbf{x})$ using Program (9). For verifying the union-CBFs, Verif-I takes 48.95 seconds, while Verif-II only takes 0.03 seconds. On the other hand, the verification of $h(\mathbf{x})$ takes 159.97 seconds.

B. Comparison To Existing Method

In this subsection, we compare the formulation of union of CBFs with a previously proposed BNCBF [15]. We construct an example in which our union of CBFs framework is less conservative compared to the BNCBF approach.

We consider the same 2D single integrator system model as Section VI-A. The goal of the control is to reach the state $\mathbf{x}_{goal} = \mathbf{0}$ while satisfying $\mathbf{x}(t) \in \mathcal{X}_c$ for all t. In this example, we let $\mathcal{X}_c = \cup_{i=1}^{2}\{\mathbf{x}|h_i(\mathbf{x}) \geq 0\}$, where $h_1(\mathbf{x}) = \frac{1}{2}(\mathbf{x}_{(1)}+3)(\mathbf{x}_{(1)}+1)-\mathbf{x}_{(2)}$, and $h_2(\mathbf{x}) = \frac{1}{2}(\mathbf{x}_{(1)}+3)(\mathbf{x}_{(1)}+1) + \mathbf{x}_{(2)}$. We consider a CBF-QP based safety filter for nominal controller $\mathbf{k}_d(\mathbf{x}) = 2(\mathbf{x}_{goal} - \mathbf{x})$. Hence, the safe controller $\mathbf{k}_{\sigma(t)}(\mathbf{x})$ switches between

$$\begin{aligned}\mathbf{k}_p(\mathbf{x}) = arg\min_{\mathbf{u}} &\frac{1}{2}||\mathbf{u} - \mathbf{k}_d(\mathbf{x})||^2 \\ \text{s.t. } &L_{\mathbf{f}}h_p(\mathbf{x}) + L_{\mathbf{G}}h_p(\mathbf{x})\mathbf{u} \geq -\kappa h_p(\mathbf{x}) \\ &\mathbf{A}\mathbf{u} \leq \mathbf{c}\end{aligned}$$

for $p = 1, 2$, and $\kappa = 0.1$.

According to the setup above, at $\mathbf{x}' = [-3, 0]^T$, we have $h_1(\mathbf{x}') = h_2(\mathbf{x}') = 0$, and the $\mathbf{k}_d(\mathbf{x}') = [6, 0]^T$. By formulation of BNCBF, at the state $\mathbf{x}'$, the control $\mathbf{u}$ should satisfy both the following conditions:

$$L_{\mathbf{f}}h_1(\mathbf{x}') + L_{\mathbf{G}}h_1(\mathbf{x}')\mathbf{u} \geq -\kappa h_1(\mathbf{x}') \quad (12a)$$
$$L_{\mathbf{f}}h_2(\mathbf{x}') + L_{\mathbf{G}}h_2(\mathbf{x}')\mathbf{u} \geq -\kappa h_2(\mathbf{x}') \quad (12b)$$

This means the BNCBF formulation requires $\mathbf{u}_{(2)} \leq -\mathbf{u}_{(1)}$ and $\mathbf{u}_{(2)} \geq \mathbf{u}_{(1)}$, which implies that $\mathbf{u}_{(1)} \leq 0$. Since the objective of CBF-QP is to minimize $\frac{1}{2}||\mathbf{u}-\mathbf{k}_d(\mathbf{x})||^2$, then the controller with BNCBF only provides control signal $\mathbf{u} = \mathbf{0}$ at state $\mathbf{x}'$. This means the single integrator controlled by BNCBF stops at $\mathbf{x}'$.

However, for union of CBFs, no matter which switching strategy we choose (strategy I or II in Section V), the control $\mathbf{u}$ only needs to satisfy (12a) *or* (12b). This means our sufficient condition for forward invariance only requires $\mathbf{u}_{(2)} \leq -\mathbf{u}_{(1)}$ or $\mathbf{u}_{(2)} \geq -\mathbf{u}_{(1)}$ at state $\mathbf{x}'$, and $\mathbf{u}_{(0)} > 0$ is allowed in either cases. Hence, at $\mathbf{x}'$ the single integrator controlled by union of CBFs continues to move closer to $\mathbf{x}_{goal}$.

C. Computation Time Comparison

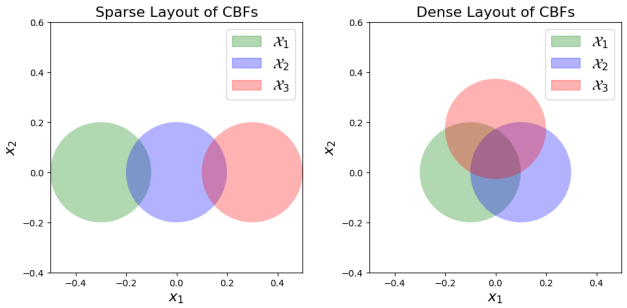
This subsection compares the efficiency between Verif-I and Verif-II. In Section IV-B, we have mentioned that Theorem 5 is a sufficient condition of Theorem 4. In this subsection, we show that, in order to efficiently verify the union of a given set of CBFs, we can first use Verif-II to verify the Theorem 5, then use Verif-I to verify Theorem 4 if needed.

We consider the polynomial control system from [9].

$$\begin{bmatrix}\dot{x}_1 \\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}x_2 \\ x_1 + \frac{1}{3}x_1^3 + x_2\end{bmatrix} + \begin{bmatrix}(0.2x_1^2 + 0.2x_2 + 1)u_1 \\ (-0.2x_2^2 + 0.2x_1 + 4)u_2\end{bmatrix}$$

with control limits $-5 \leq u_1 \leq 5$ and $-5 \leq u_2 \leq 5$. We first show that Verif-I is sensitive to CBFs' layout of the union, while Verif-II is not. Then, we show that the total verification time of Verif-I grows rapidly as the number of CBFs increases, while the verification time of Verif-II grows linearly.

We first focus on relative positions of CBFs' super level sets, which is called *CBF layout*. Depending on the number of empty intersections between super level sets, we consider two types of CBF layouts, namely sparse layout and dense layout, shown in Fig. 2. In both layout cases, each CBF is $h_i(\mathbf{x}) = 0.04 - ||\mathbf{x} - \mathbf{x}_i||^2$, $i = 1, 2, 3$. In the sparse case $\mathbf{x}_1 = [-0.3, 0]$, $\mathbf{x}_2 = [0, 0]$, and $\mathbf{x}_3 = [0.3, 0]$. In the dense case $\mathbf{x}_1 = [-0.1, 0]$, $\mathbf{x}_2 = [0.1, 0]$, and $\mathbf{x}_3 = [0, 0.17]$. In the sparse case shown in Fig. (2a), Verif-I takes 76.12 seconds and Verif-II takes 1.12 seconds. In the dense case shown in Fig. (2b), Verif-I takes 463.10 seconds while Verif-II only takes 1.14 seconds. This time comparison shows that Verif-I is sensitive to the layout of the CBFs in the union. According to Section V-A, the total number of non-empty subsets computed in the step 2 of Verif-I depends on the layout of CBFs, and it directly decides the number of times we repeat the Program (8). On the other hand, Verif-II only repeats the Program (9) for each of the CBFs, thus not sensitive to the layout of the CBFs.



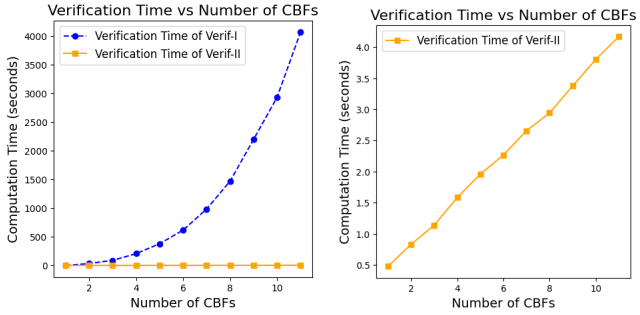
(a) (b)

Fig. 2: Union of 3 CBFs with different layouts. Fig. (2a) is the spares layout such that $\mathcal{X}_c$ only has 5 subsets $\mathcal{X}_\mathcal{N}$. Fig. (2b) is the dense layout such that $\mathcal{X}_c$ has 7 subsets $\mathcal{X}_\mathcal{N}$.

The next comparison shows that, in the sparse layout, for the same number of CBFs, Verif-I takes longer time than Verif-II. We consider the sparse layout and compare the computation time of Verif-I and Verif-II for unions with different number of CBFs. The comparison result is shown in Fig. 3. Compared to Verif-II, the computation time of Verif-I grow rapidly as the number of CBFs grows. This is caused by the increasing number of $\mathbf{y}$ variables in the Program (8). When the Program (8) is used to verify the overlapping region of two 0-super-level sets, the number of $\mathbf{y}$ variables would be doubled compared to verifying a single 0-super-level set. On the contrary, Verif-II only repeats the Program (9b) for each of the 0-super-level set, and the number of $\mathbf{y}$ variables in Program (9) never changes. Hence, the total computation time for Verif-II grows linearly with respect to the number of CBFs.

VII. Conclusions And Future Work

This paper focuses on ensuring forward invariance of the union of super level sets of CBFs (union-CBFs). We proposed sufficient conditions for keeping the closed-loop system with switching CBF-QP controller safe. The proposed condition ensures (i) finite switches in any finite time interval and (ii) forward invariance of the closed-loop system. By



(a) (b)

Fig. 3: Computation time for Verif-I and Verif-II. Fig. (3a) compares the computation time of Verif-I and Verif-II in the sparse case with unions of different number of CBFs. Fig. (3b) shows that the computation time of Verif-II only grows linearly as the number of CBFs grows.

utilizing the concept of minimal CBF and Slater's condition, our proposed sufficient condition allows less conservative safe control policies while satisfying the point-Lipschitz property. We considered two types of switching strategies and proposed sufficient conditions on CBFs for each of the strategies. We also derived SOS verification frameworks for union-CBFs conditions under each switching strategy. The experiments show that our union-CBFs condition is less conservative than existing BNCBF methods. However, the methods proposed in this paper only consider CBFs with relative degree 1, and hence our future work will extend the discussion to high-relative-degree cases.

REFERENCES

[1] B. Franco and M. Stefano, "Set-theoretic methods in control," *Systems & control: Foundations & applications*, 2008.

[2] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2016.

[3] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European control conference (ECC)*, pp. 3420–3431, IEEE, 2019.

[4] 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)*, pp. 7139–7145, Ieee, 2020.

[5] H. Dai and F. Permenter, "Convex synthesis and verification of control-Lyapunov and barrier functions with input constraints," in *2023 American Control Conference (ACC)*, pp. 4116–4123, IEEE, 2023.

[6] 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. 39, no. 3, pp. 2289–2307, 2023.

[7] P. Glotfelter, J. Cortés, and M. Egerstedt, "Nonsmooth barrier functions with applications to multi-robot systems," *IEEE control systems letters*, vol. 1, no. 2, pp. 310–315, 2017.

[8] T. G. Molnar and A. D. Ames, "Composing control barrier functions for complex safety specifications," *IEEE Control Systems Letters*, vol. 7, pp. 3615–3620, 2023.

[9] H. Wang, K. Margellos, and A. Papachristodoulou, "Safe and stable filter design using a relaxed compatibitlity control barrier–lyapunov condition," *arXiv preprint arXiv:2407.00414*, 2024.

[10] W. Jin, Z. Wang, Z. Yang, and S. Mou, "Neural certificates for safe control policies," *arXiv preprint arXiv:2006.08465*, 2020.

[11] A. Robey, H. Hu, L. Lindemann, H. Zhang, D. V. Dimarogonas, S. Tu, and N. Matni, "Learning control barrier functions from expert demonstrations," in *2020 59th IEEE Conference on Decision and Control (CDC)*, pp. 3717–3724, Ieee, 2020.

[12] 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*, vol. 39, no. 3, pp. 1749–1767, 2023.

[13] H. Zhang, J. Wu, Y. Vorobeychik, and A. Clark, "Exact verification of relu neural control barrier functions," *Advances in neural information processing systems*, vol. 36, pp. 5685–5705, 2023.

[14] P. Glotfelter, J. Cortés, and M. Egerstedt, "Boolean composability of constraints and control synthesis for multi-robot systems via nonsmooth control barrier functions," in *2018 IEEE Conference on Control Technology and Applications (CCTA)*, pp. 897–902, IEEE, 2018.

[15] P. Glotfelter, J. Cortés, and M. Egerstedt, "A nonsmooth approach to controller synthesis for boolean specifications," *IEEE Transactions on Automatic Control*, vol. 66, no. 11, pp. 5160–5174, 2020.

[16] L. Long, C. Huang, and Z. Sun, "Safety control of switched systems via multiple barrier functions," *IEEE Transactions on Automatic Control*, 2025.

[17] A. Clark, "Verification and synthesis of control barrier functions," in *2021 60th IEEE Conference on Decision and Control (CDC)*, pp. 6105–6112, Ieee, 2021.

[18] P. Mestres, A. Allibhoy, and J. Cortés, "Regularity properties of optimization-based controllers," *European Journal of Control*, vol. 81, p. 101098, 2025.

[19] B. J. Morris, M. J. Powell, and A. D. Ames, "Continuity and smoothness properties of nonlinear optimization-based feedback controllers," in *2015 54th IEEE Conference on Decision and Control (CDC)*, pp. 151–158, IEEE, 2015.

[20] R. Konda, A. D. Ames, and S. Coogan, "Characterizing safety: Minimal control barrier functions from scalar comparison systems," *IEEE Control Systems Letters*, vol. 5, no. 2, pp. 523–528, 2020.

[21] H. Dai, C. Jiang, H. Zhang, and A. Clark, "Verification and synthesis of compatible control Lyapunov and control barrier functions," in *2024 IEEE Conference on Decision and Control (CDC)*, pp. 8178–8185, IEEE, 2024.

[22] J. Matousek and B. Gärtner, *Understanding and Using Linear Programming*. Springer Science & Business Media, 2006.

[23] S. Prajna, A. Papachristodoulou, and P. A. Parrilo, "Introducing sostools: A general purpose sum of squares programming solver," in *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.*, vol. 1, pp. 741–746, IEEE, 2002.

[24] M. ApS, "Mosek optimizer api for python," *Version*, vol. 9, no. 17, pp. 6–4, 2022.

[25] D. Liberzon, *Switching in systems and control*, vol. 190. Springer, 2003.