NEURAL FEASIBILITY CERTIFICATES

Anonymous authors
Paper under double-blind review

ABSTRACT

Ensuring recursive feasibility is critical for safety-constrained online control under hard constraints. Since analytically constructing feasible regions and recovery policies is difficult for nonlinear systems, learning-based methods offer a flexible alternative. However, existing methods often rely on approximate boundaries and continuously constrain the nominal controller, leading to limited guarantees and unnecessary conservatism. To address these limitations, we propose a framework to synthesize and formally verify neural feasibility certificates for safety-constrained online control. Our contributions are threefold. First, we introduce a neural feasibility certificate defining an inner approximation of the instantaneous feasible set. The associated margin-triggered controller retains the original online controller when sufficient feasibility margin is available and activates a learned witness policy only near the boundary of the certified feasible region, where it drives the system back to the feasible interior in finite time. Second, we develop a synthesis and formal verification framework that jointly learns the feasibility certificate and witness policy and certifies recursive feasibility over the continuous state space. Third, we validate the proposed framework on adaptive cruise control, autonomous overtaking, and mobile robot navigation against multiple baselines, demonstrating improved feasibility with competitive safety, control performance, and computational efficiency.

1 INTRODUCTION

Safety-critical autonomous systems are increasingly required to operate in complex environments while simultaneously achieving task objectives and satisfying hard state and control constraints. A common approach is to formulate an online constrained control problem that optimizes control actions according to the current system state (Cheng et al., 2019; Xiao et al., 2023b; Zhou et al., 2026b). Such optimization-based safety controllers provide a flexible interface between task-oriented control and safety requirements, and have been widely adopted in autonomous driving (He et al., 2023), robotics (Cao et al., 2023), and other safety-critical applications.

A fundamental requirement for safety-constrained online control is recursive feasibility (Esterhuizen et al., 2020; Pin & Parisini, 2010). A state may satisfy all instantaneous safety constraints while already lacking sufficient control authority to maintain them in the future, causing the online control problem to eventually become infeasible. Therefore, feasibility is inherently a closed-loop property that depends on the system dynamics and control limits, and characterizing recursively feasible states remains challenging, particularly for unknown systems.

Several approaches have been developed to address feasibility loss. Predictive control (Esterhuizen et al., 2020; Pin & Parisini, 2010) can explicitly account for future constraints, but may incur substantial online computation for nonlinear systems and complex constraint sets. Viability- and reachability-based methods (Liniger & Lygeros, 2015; Song et al., 2026) can provide rigorous feasible regions, but their computation generally scales poorly with the state dimension. Analytical sufficient conditions can offer strong guarantees (Xiao et al., 2022), yet they are often difficult to derive beyond specific system structures. Constraint relaxation (Ames et al., 2014; 2016) can alleviate optimization infeasibility, but may weaken hard constraint satisfaction. More recently, learning-based methods (Xiao et al., 2020; 2023a; 2026) have been used to approximate feasible regions directly from data and incorporate the learned feasibility boundaries as additional barrier constraints in online control. However, these learning-based methods generally lack formally verified feasibility guarantees over the continuous state space and scale poorly to systems with a large number of

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

safety constraints. Only a few works have further studied formal verification and synthesis of feasibility or compatibility conditions (Jiang et al., 2026). However, these approaches rely on structured polynomial certificates and SOS programming, which can become computationally demanding for high-dimensional systems with multiple constraints.

Statement of Contributions. To address these challenges, we develop a neural feasibility certificate framework that learns and verifies an inner approximation of the instantaneous feasible set for safety-constrained online control. Our contributions are threefold. First, we introduce a margin-triggered recovery mechanism that retains the original online controller when sufficient feasibility margin is available and activates a learned witness policy only near the boundary of the certified feasible region. The witness policy is designed to drive the system back to the interior feasible region in finite time, thereby reducing unnecessary intervention and conservatism. Second, we jointly synthesize the neural feasibility certificate and witness policy and formally verify their closed-loop conditions over the continuous state space, establishing recursive feasibility beyond sampled training data. Third, we validate the proposed framework on adaptive cruise control, autonomous overtaking with moving obstacles, and mobile robot navigation through narrow passages. Extensive simulations demonstrate the effectiveness of the proposed method.

2 Related Work

This work makes contributions in two main research areas: (i) feasibility in safety-constrained control, and (ii) neural certificates and formal verification.

2.1 Feasibility in Safety-Constrained Control

Maintaining feasibility under hard state and input constraints has been studied through predictive control, viability and reachability analysis, analytical sufficient conditions, and constraint relaxation (Esterhuizen et al., 2020; Pin & Parisini, 2010; Liniger & Lygeros, 2015; Song et al., 2026; Xiao et al., 2022; Ames et al., 2014; 2016). Related backup CBF methods (Chen et al., 2021) use a backup controller and a recoverable safe set to preserve safety near constraint boundaries, but typically focus on safety invariance rather than recursive feasibility of the underlying online control problem. More recently, learning-based approaches approximate feasible regions from sampled system behavior and incorporate the learned constraints into online controllers (Xiao et al., 2020; 2023a). However, these methods generally lack formally verified guarantees over the continuous state space and face scalability challenges as the number of safety constraints grows. Recent compatibility-based approaches provide formal feasibility verification for multiple certificate constraints, but rely on structured polynomial representations and SOS programming (Jiang et al., 2026). To the best of our knowledge, formally verified and scalable learning-based methods for recursive feasibility remain largely unexplored.

2.2 Neural Certificates and Formal Verification

To provide guarantees for learning-based controllers, numerous works train Lyapunov- or barrier-type neural certificates (Dai et al., 2021; Clark, 2021; Qin et al., 2021; Yang et al., 2024; 2023; Zhang et al., 2023a;b; Schlaginhaufen et al., 2021; Hu et al., 2024; Tayal et al., 2024; Zhang et al., 2025; Mandal et al., 2024a; Liu et al., 2024; Nadali et al., 2024; Zhou et al., 2025; Neustroev et al., 2025; Yu et al., 2025; Wang et al., 2024; Zhou et al., 2026a). These certificates can be formally verified using control-theoretic conditions (Anand & Zamani, 2023; Tayal et al., 2024; Zhang et al., 2024; Henzinger et al., 2025; Ren et al., 2025) and neural network verification tools (Wang et al., 2021; Lopez et al., 2023; Wu et al., 2024), enabling counterexample-guided synthesis (Ding et al., 2022; Mandal et al., 2024b; Zhao et al., 2024) or certified training (Mueller et al., 2023). Beyond individual certificate properties, recent work has also considered the joint synthesis and verification of compatible control certificates (Jiang et al., 2026). However, to the best of our knowledge, existing neural certificate methods primarily focus on stability, safety, or set invariance, while neural certificates for formally guaranteeing recursive feasibility have not yet been systematically studied.

3 PRELIMINARIES AND PROBLEM STATEMENT

This section introduces the system dynamics and reviews high-order discrete-time control barrier functions in Section 3.1. Section 3.2 then formulates the online control problem and the recursive feasibility objective.

3.1 PRELIMINARIES

We consider the discrete-time control-affine system

$$x_{k+1} = f(x_k) + g(x_k)u_k, \tag{1}$$

where $x_k \in \mathcal{X} \subseteq \mathbb{R}^n$ and $u_k \in \mathcal{U} \subseteq \mathbb{R}^p$ denote the state and control input, respectively. The set $\mathcal{U}$ represents the admissible inputs under the system's control limits. The system must satisfy N safety constraints, expressed as $h_{\text{safe},i}(x_k) \geq 0$ for $i = 1, \ldots, N$, where each function $h_{\text{safe},i} : \mathcal{X} \to \mathbb{R}$ characterizes a safety requirement.

Control barrier functions provide a systematic way to translate state safety requirements into constraints on the control input. High-order discrete-time control barrier functions extend this approach to safety constraints that are affected by the current input only after multiple time steps. For each $h_{\text{safe},i}$, let m_i denote its discrete relative degree, meaning that the current input first appears in its m_i-th forward difference.

Definition 1 (High-Order Control Barrier Function (HOCBF) for Discrete-time Systems)**.** *For each constraint $h_{\text{safe},i}(x_k) \geq 0$, define $\psi_{i,0}(x_k) := h_{\text{safe},i}(x_k)$ and recursively*

$$\psi_{i,j}(x_k) = \Delta\psi_{i,j-1}(x_k, u_k) + \gamma_{i,j}\psi_{i,j-1}(x_k), \qquad j = 1, \ldots, m_i, \tag{2}$$

where $\gamma_{i,j} \in (0, 1]$ and $\Delta\psi_{i,j-1}(x_k, u_k) := \psi_{i,j-1}\big(f(x_k) + g(x_k)u_k\big) - \psi_{i,j-1}(x_k)$. For $j < m_i$, the relative-degree condition makes $\psi_{i,j}$ independent of the current input. At the final order, we write $\psi_{i,m_i}(x_k, u_k)$ to indicate its input dependence. Define $\mathcal{C}_{i,j} := \{x \in \mathcal{X} \mid \psi_{i,j}(x) \geq 0\}$ for $j = 0, \ldots, m_i - 1$. Then $h_{\text{safe},i}$ is a high-order discrete-time control barrier function if, for every $x_k \in \bigcap_{j=0}^{m_i-1} \mathcal{C}_{i,j}$, there exists $u_k \in \mathcal{U}$ such that

$$\psi_{i,m_i}(x_k, u_k) \geq 0. \tag{3}$$

For an initial state $x_0 \in \bigcap_{i=1}^{N} \bigcap_{j=0}^{m_i-1} \mathcal{C}_{i,j}$, enforcing $\psi_{i,m_i}(x_k, u_k) \geq 0$ for all i at every time step preserves the safety constraints, provided that the trajectory remains in $\mathcal{X}$. This follows recursively from $\psi_{i,j-1}(x_{k+1}) \geq (1-\gamma_{i,j})\psi_{i,j-1}(x_k)$ for $j = 1, \ldots, m_i$. These guarantees require the HOCBF constraints to admit a common admissible input at every time step.

3.2 PROBLEM STATEMENT

Given a stage cost $\ell : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$, we consider the safety-constrained online control problem

$$\begin{aligned} u^\star(x_k) \in \arg\min_{u_k} \quad & \ell(x_k, u_k) \\ \text{s.t.} \quad & \psi_{i,m_i}(x_k, u_k) \geq 0, \qquad i = 1, \ldots, N, \\ & u_k \in \mathcal{U}. \end{aligned} \tag{4}$$

Here, $u^\star(x_k)$ denotes an optimal control input, assumed to exist whenever the problem is feasible. If multiple minimizers exist, one is selected by a fixed rule. The selected input is applied to system equation 1 at each time step.

The HOCBF constraints may conflict with one another or with the input limits. Consequently, a common admissible input need not exist, even when the current state satisfies the original safety constraints. Define the instantaneous feasible set as $\mathcal{X}_{\text{feas}} := \{x \in \mathcal{X} \mid \exists u \in \mathcal{U} : \psi_{i,m_i}(x, u) \geq 0,\ \forall i\}$.

Even if $x_k \in \mathcal{X}_{\text{feas}}$, problem equation 4 does not require the next state $f(x_k) + g(x_k)u^\star(x_k)$ to remain in $\mathcal{X}_{\text{feas}}$. Thus, an input that minimizes the current cost while satisfying all current constraints may leave no feasible input at the next time step. Once the online problem becomes infeasible, the controller can no longer provide an input satisfying all HOCBF and input constraints, and the associated safety guarantee no longer applies. This motivates the following definition.

Definition 2 (Recursive Feasibility)**.** *The online control problem equation 4 is recursively feasible from $x_0 \in \mathcal{X}_{\text{feas}}$ under a control law $u_k = \pi(x_k)$ if the resulting closed-loop trajectory satisfies $x_k \in \mathcal{X}_{\text{feas}}$ for all $k \geq 0$. It is recursively feasible on a set $\mathcal{X}_0 \subseteq \mathcal{X}_{\text{feas}}$ under π if this property holds for every $x_0 \in \mathcal{X}_0$.*

Our objective is to construct a control law and establish a nonempty set of initial states $\mathcal{X}_0 \subseteq \mathcal{X}_{\text{feas}}$ from which the online problem remains recursively feasible, while the applied input satisfies all HOCBF and input constraints at every time step. These guarantees must hold for every initial state in $\mathcal{X}_0$. For initial states satisfying the HOCBF initialization conditions in Section 3.1, this also ensures satisfaction of the original safety constraints.

Remark 1. *Problem equation 4 allows general stage costs and nonlinear constraints. The Farkas-based inclusion test in Lemma 4 additionally requires an equivalent representation $A(x)u \leq b(x)$. For nonlinear constraints without such a representation, set inclusion requires a different verification procedure.*

4 Feasibility Certificates

The main idea is to identify a certified feasible set and keep the trajectory inside it, thereby maintaining recursive feasibility. First, we introduce the feasibility certificate and switching mechanism and establish finite-step recovery under the witness policy. Second, we select a switching margin that prevents the original controller from leaving the certified set in one step. Finally, we combine these results to establish recursive feasibility.

(1) Feasibility certificate and switching mechanism. The witness policy is designed to restore feasibility margins. We therefore use a switching mechanism that retains the original controller whenever the feasibility margin is sufficiently large and activates the witness policy near the boundary.

Let $\pi_w : \mathcal{X} \to \mathcal{U}$ denote a witness policy and $h_{\text{feas}} : \mathcal{X} \to \mathbb{R}$ a feasibility certificate. Define the certified set as $\mathcal{S}_0 := \{x \in \mathcal{X} \mid h_{\text{feas}}(x) \geq 0\}$. The certificate is required to satisfy $\mathcal{S}_0 \subseteq \mathcal{X}_{\text{feas}}$, so that $\mathcal{S}_0$ is an inner approximation of the instantaneous feasible set.

For a margin $\delta > 0$, define $\mathcal{S}_\delta := \{x \in \mathcal{X} \mid h_{\text{feas}}(x) \geq \delta\}$, $\mathcal{B}_\delta := \{x \in \mathcal{X} \mid 0 \leq h_{\text{feas}}(x) < \delta\}$, and $\mathcal{S}_{\text{neg}} := \{x \in \mathcal{X} \mid h_{\text{feas}}(x) < 0\}$, which represent the certified interior, recovery region, and uncertified region, respectively. In particular, $\mathcal{S}_0 = \mathcal{S}_\delta \cup \mathcal{B}_\delta$.

The controller switches between the original online controller and the witness policy according to

$$u_k = \begin{cases} u^\star(x_k), & x_k \in \mathcal{S}_\delta, \\ \pi_w(x_k), & x_k \in \mathcal{B}_\delta. \end{cases} \tag{5}$$

Thus, the original controller pursues its performance objective in the certified interior, while the witness policy restores the feasibility margin near the boundary. Once the state returns to $\mathcal{S}_\delta$, the original controller resumes.

The following definition formalizes the feasibility and recovery conditions required of the certificate and witness policy.

Definition 3 (Finite-Time Feasibility Certificate)**.** *A continuously differentiable function $h_{\text{feas}} : \mathcal{X} \to \mathbb{R}$ is a finite-time feasibility certificate if there exist a witness policy $\pi_w : \mathcal{X} \to \mathcal{U}$ and constants $\delta, \kappa > 0$ such that the following conditions hold.*

a) Certified feasible inner approximation: *The set $\mathcal{S}_0$ is nonempty, and for every $x \in \mathcal{X}$,*

$$h_{\text{feas}}(x) \geq 0 \quad \Longrightarrow \quad \exists u \in \mathcal{U} : \psi_{i,m_i}(x, u) \geq 0, \quad \forall i. \tag{6}$$

b) Boundary recovery: *For every $x_k \in \mathcal{B}_\delta$, the witness input satisfies $\psi_{i,m_i}(x_k, \pi_w(x_k)) \geq 0$ for all i. Moreover, the successor state $x_{k+1} = f(x_k) + g(x_k)\pi_w(x_k)$ belongs to $\mathcal{X}$ and satisfies*

$$h_{\text{feas}}(x_{k+1}) - h_{\text{feas}}(x_k) \geq \min\{\kappa, \delta - h_{\text{feas}}(x_k)\}. \tag{7}$$

Condition equation 7 requires the witness policy to increase the certificate value whenever the state lies in $\mathcal{B}_\delta$, while satisfying all original HOCBF and input constraints.

Once the state enters $\mathcal{B}_\delta$, the recovery condition ensures finite-step return to the certified interior, as summarized below.

Lemma 1 (Finite-Step Backup Recovery). *Suppose the boundary recovery condition in Definition 3 holds. Then every $x_0 \in \mathcal{B}_\delta$ reaches $\mathcal{S}_\delta$ under π_w within at most $\lceil(\delta - h_{\text{feas}}(x_0))/\kappa\rceil$ steps.*

Lemma 1 bounds the duration of each recovery phase. During recovery, the certificate value increases, so the trajectory remains in $\mathcal{S}_0$ until it reaches $\mathcal{S}_\delta$.

(2) Switching margin selection. We select δ to ensure that, under the original controller, every $x_k \in \mathcal{S}_\delta$ satisfies $x_{k+1} \in \mathcal{S}_0$. Since the online control problem is feasible throughout $\mathcal{S}_0$, we use $u^\star$ directly and restrict verification to this set. Define $F_\star(x) := f(x) + g(x)u^\star(x)$ for $x \in \mathcal{S}_0$.

Assumption 1 (Lipschitz Continuity). *The set $\mathcal{X}$ is compact and $\mathcal{U}$ is bounded. The function h_{feas} is Lipschitz continuous on $\mathcal{X}$ with constant L_h. The mappings $F_\star$ and $F_w(x) := f(x) + g(x)\pi_w(x)$ are Lipschitz continuous on $\mathcal{S}_0$ with constants $L_{F_\star}$ and L_{F_w}, respectively. All Lipschitz bounds use the same norm $\|\cdot\|$. Moreover, $F_\star(x), F_w(x) \in \mathcal{X}$ for every $x \in \mathcal{S}_0$.*

Remark 2. *The Lipschitz constants in Assumption 1 can be computed offline over the compact verification domain. Neural-network bounds for h_{feas} and π_w can be obtained using standard Lipschitz bounding methods (Xu & Sivaranjani, 2024), while L_{F_w} and $L_{F^\star}$ follow from the corresponding closed-loop mappings, with $L_{F^\star}$ additionally requiring regularity of the optimizer $u^\star(x)$.*

Definition 4 (Grid Sampling of the Certified Set). *Let $\mathcal{X}_{\text{grid}} \subseteq \mathcal{S}_0$ be a finite sampling set. For every $x \in \mathcal{S}_0$, let $\hat{x} \in \arg\min_{\tilde{x}\in\mathcal{X}_{\text{grid}}} \|x - \tilde{x}\|$ be a nearest sampling point. The sampling set is chosen such that $\|x - \hat{x}\| \le \eta_x$ for every $x \in \mathcal{S}_0$, where $\eta_x > 0$ is a prescribed sampling radius.*

We evaluate the next-step certificate at the sampling points and use Lipschitz bounds to account for sampling errors. The switching margin is then chosen to exclude regions where these checks cannot certify one-step feasibility.

Lemma 2 (Switching Margin Selection). *Suppose Assumption 1 holds and the sampling set satisfies Definition 4. Define $\mathcal{I}_{\text{grid}} := \{\hat{x} \in \mathcal{X}_{\text{grid}} \mid h_{\text{feas}}(F_\star(\hat{x})) < L_h L_{F_\star}\eta_x\}$. If $\mathcal{I}_{\text{grid}} \neq \varnothing$, choose $\delta = \max_{\hat{x}\in\mathcal{I}_{\text{grid}}} h_{\text{feas}}(\hat{x}) + L_h\eta_x + \varepsilon_\delta$ with $\varepsilon_\delta > 0$. Then $F_\star(x) \in \mathcal{S}_0$ for every $x \in \mathcal{S}_\delta$. If $\mathcal{I}_{\text{grid}} = \varnothing$, the same conclusion holds for any $\delta > 0$.*

The selected margin allows the original controller to enter $\mathcal{B}_\delta$ without leaving $\mathcal{S}_0$ in one step.

(3) Recursive feasibility. Combining finite-step recovery under the witness policy with the interior non-penetration condition yields recursive feasibility under the switched controller.

Theorem 3 (Recursive Feasibility with Margin-Triggered Backup). *Suppose h_{feas} satisfies Definition 3, and the same margin δ satisfies the conditions of Lemma 2. Then, for every $x_0 \in \mathcal{S}_0 = \mathcal{S}_\delta \cup \mathcal{B}_\delta$, the switched controller equation 5 guarantees*

$$x_k \in \mathcal{S}_0 \subseteq \mathcal{X}_{\text{feas}}, \qquad \forall k \ge 0. \tag{8}$$

The applied input satisfies the original HOCBF and input constraints at every time step.

Theorem 3 establishes recursive feasibility through two complementary mechanisms. In the certified interior, the selected margin ensures that the original controller maps $\mathcal{S}_\delta$ into $\mathcal{S}_0$. In the recovery region, the witness policy preserves $\mathcal{S}_0$ and returns the state to $\mathcal{S}_\delta$ in finite steps. Thus, $\mathcal{S}_0$ is a certified recursively feasible inner approximation under the switched controller.

5 Synthesis and Verification Framework

This section presents the synthesis and verification framework for the proposed neural feasibility certificates. Section 5.1 introduces a Lipschitz-based method for verification over the entire verification domain. Section 5.2 describes the joint synthesis of the feasibility certificate and witness policy. Additional scalability analysis for systems with multiple safety constraints under certain assumptions is provided in Appendix C.

5.1 Verification Formulation

This subsection develops verification conditions for certified-set inclusion and finite-step recovery. Since directly constructing a feasibility certificate and a witness policy satisfying the required conditions is difficult, we represent both using neural networks and learn them jointly. However, training

on sampled states alone does not guarantee these properties throughout the required domains. We therefore formulate certified-set inclusion as an infeasibility query and verify the recovery inequality using Lipschitz bounds. The proofs are provided in Appendix B.

(1) Certified-set inclusion. Suppose the original HOCBF and input constraints can be expressed as $A(x)u \le b(x)$, where $A(x) \in \mathbb{R}^{q\times p}$ and $b(x) \in \mathbb{R}^q$. Thus, $x \in \mathcal{X}_{\text{feas}}$ if and only if there exists $u \in \mathbb{R}^p$ satisfying these inequalities. The following result uses Farkas' lemma to characterize states that are certified by h_{feas} but admit no feasible input.

Lemma 4 (Verification of Certified-Set Inclusion)**.** *Suppose the original HOCBF and input constraints are equivalent to $A(x)u \le b(x)$. Then $\mathcal{S}_0 \subseteq \mathcal{X}_{\text{feas}}$ if and only if the following system has no solution $(x, \lambda) \in \mathcal{X} \times \mathbb{R}^q$:*

$$h_{\text{feas}}(x) \ge 0, \quad \lambda \ge 0, \quad A(x)^\top \lambda = 0, \quad b(x)^\top \lambda = -1. \tag{9}$$

Query equation 9 can be submitted to a formal verifier such as α-β-CROWN (Wang et al., 2021) or Marabou (Wu et al., 2024) when its expressions admit a supported encoding. An UNSAT result for an exact encoding, or for a sound overapproximation of the query, certifies the inclusion. Marabou can be used when the network and constraint expressions admit such an encoding.

(2) Finite-step recovery. We use the Lipschitz bounds in Assumption 1 and the sampling set in Definition 4 to verify the recovery inequality throughout $\mathcal{B}_\delta$. The grid checks include a nearest sampling point for every state in this region, and the Lipschitz margin accounts for the sampling error.

Lemma 5 (Verification of the Recovery Inequality)**.** *Suppose Assumption 1 holds and the sampling set satisfies Definition 4. For $\delta, \kappa > 0$, suppose that every $\hat{x} \in \mathcal{X}_{\text{grid}}$ satisfying $h_{\text{feas}}(\hat{x}) \le \delta + L_h\eta_x$ also satisfies*

$$h_{\text{feas}}(F_w(\hat{x})) - h_{\text{feas}}(\hat{x}) \ge \min\{\kappa, \delta - h_{\text{feas}}(\hat{x})\} + L_h(L_{F_w} + 1)\eta_x. \tag{10}$$

Then $h_{\text{feas}}(F_w(x)) - h_{\text{feas}}(x) \ge \min\{\kappa, \delta - h_{\text{feas}}(x)\}$ for every $x \in \mathcal{B}_\delta$.

The two verification results, together with witness-policy admissibility and the switching-margin condition, yield the following guarantee.

Theorem 6 (Verified Recursive Feasibility)**.** *Suppose $\mathcal{S}_0 \neq \varnothing$ and Lemma 4 applies, with query equation 9 certified to be unsatisfiable. Suppose the conditions of Lemma 5 hold, and the same margin δ satisfies Lemma 2. If the witness policy satisfies $A(x)\pi_w(x) \le b(x)$ for every $x \in \mathcal{B}_\delta$, then, for every $x_0 \in \mathcal{S}_0$, controller equation 5 guarantees*

$$x_k \in \mathcal{S}_0 \subseteq \mathcal{X}_{\text{feas}}, \qquad \forall k \ge 0. \tag{11}$$

The applied input satisfies all original HOCBF and input constraints at every time step.

5.2 Neural Certificate Synthesis

We jointly learn the neural feasibility certificate h_{feas} and witness policy π_w while keeping the original HOCBF constraints fixed. Input admissibility is enforced by constructing $\pi_w(x) \in \mathcal{U}$. The overall training objective is

$$\mathcal{L} = \mathcal{L}_{\text{feas}} + \lambda_{\text{adm}}\mathcal{L}_{\text{adm}} + \lambda_{\text{rec}}\mathcal{L}_{\text{rec}}, \tag{12}$$

with the three terms defined as

$$\begin{aligned}
\mathcal{L}_{\text{feas}} &= \mathbb{E}_{x\sim\mathcal{D}}\Big[\max\{0, \delta - h_{\text{feas}}(x)\}\mathbf{1}_{\text{feas}}(x) + \max\{0, h_{\text{feas}}(x) + \varepsilon_h\}\mathbf{1}_{\text{infeas}}(x)\Big], \\
\mathcal{L}_{\text{adm}} &= \mathbb{E}_{x\sim\mathcal{D}}\left[\sum_{i=1}^{N}\max\{0, -\psi_{i,m_i}(x, \pi_w(x))\}\mathbf{1}_{\{0\le h_{\text{feas}}(x)<\delta\}}\right], \\
\mathcal{L}_{\text{rec}} &= \mathbb{E}_{x\sim\mathcal{D}}\Big[\max\big\{0,\ \min\{h_{\text{feas}}(x) + \kappa, \delta\} \\
&\qquad + L_h(L_{F_w} + 1)\eta_x - h_{\text{feas}}(F_w(x))\big\} \cdot \mathbf{1}_{\{0\le h_{\text{feas}}(x)\le\delta+L_h\eta_x\}}\Big],
\end{aligned} \tag{13}$$

where $\mathcal{D}$ is the training set, $\mathbf{1}_{\mathrm{feas}}$ and $\mathbf{1}_{\mathrm{infeas}}$ indicate feasibility and infeasibility of the original online problem, $\mathbf{1}_{\{\cdot\}}$ denotes an indicator function, $\varepsilon_h > 0$ is an exclusion margin, and $\lambda_{\mathrm{adm}}, \lambda_{\mathrm{rec}} > 0$ are weighting coefficients. The three losses encourage coverage of feasible states and exclusion of infeasible states, witness-policy constraint satisfaction, and finite-step recovery, respectively. The recovery loss includes the Lipschitz margin from equation 10. Training may use estimated Lipschitz bounds, whereas formal verification requires valid bounds for the resulting candidate.

We use counterexample-guided inductive synthesis (CEGIS) to alternate between training and verification, as summarized in Algorithm 1 in Appendix D. Counterexamples and failed grid-check samples are added to the training set, with feasibility labels determined by the original online problem. Only candidates satisfying all required verification conditions are accepted.

6 Experiments

This section evaluates the proposed framework from both feasibility and control-performance perspectives. Section 6.1 introduces the simulation scenarios and the compared methods. Section 6.2 evaluates recursive feasibility across closed-loop rollouts, and Section 6.3 compares task performance, safety margins, and computational efficiency against the baselines. Additional results on the offline synthesis and formal verification cost are provided in Appendix E.

6.1 Simulation Setup

We evaluate the proposed neural feasibility certificate framework on three representative safety-critical control scenarios, with detailed settings given in Appendix A.1:

(1) Adaptive cruise control (ACC) (Xiao et al., 2022): a low-dimensional longitudinal control problem with hard safety and input constraints, used to examine recursive feasibility under limited braking authority.

(2) Mobile robot navigation (Xiao et al., 2026): a multi-constraint navigation task through narrow passages, where several safety constraints can become simultaneously active.

(3) Overtake (Xiao et al., 2026): a dynamic obstacle-avoidance problem with moving surrounding vehicles, used to evaluate feasibility under time-varying interactions.

We compare the proposed method with the following baselines:

(1) Standard (Ames et al., 2016). The original safety-constrained controller, representing the case without an explicit recursive-feasibility mechanism.

(2) MPC (Qie et al., 2022). A finite-horizon constrained controller, representing predictive handling of future feasibility.

(3) Feasibility-guided optimization (FGO) (Xiao et al., 2026). A learned SVM feasibility constraint, representing learning-based feasible-region approximation.

(4) Backup CBF (Chen et al., 2021). A backup-controller-based CBF-QP, representing recovery with a predefined backup strategy.

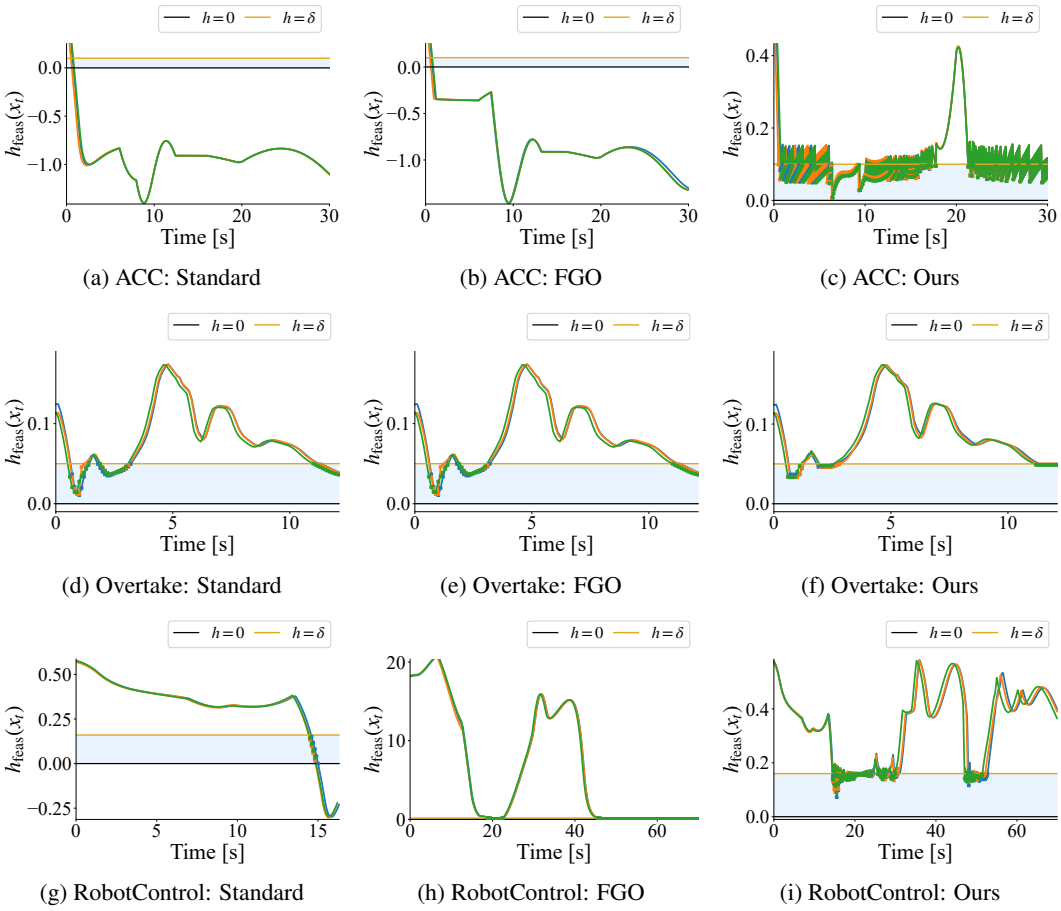
(5) Ours. The proposed neural feasibility certificate with margin-triggered learned recovery and formal verification.

6.2 Feasibility Performance

Table 1 reports the empirical feasibility rates over 1000 paired closed-loop rollouts for each system, conditioned on $h_{\mathrm{feas}}(x_0) \geq 0$. Our method achieves a 100% feasibility rate across all three systems. In comparison, the standard controller achieves 29.9%, 55.8%, and 0.2%, FGO achieves 100.0%, 59.8%, and 100.0%, MPC achieves 92.1%, 95.6%, and 98.9%, and Backup CBF achieves 100.0%, 99.4%, and 100.0% for ACC, Overtake, and RobotControl, respectively. These results show that the proposed method maintains full recursive feasibility across all three scenarios, while the baselines exhibit performance degradation in at least one task.

Table 1: Feasibility rates over 1000 paired rollouts per method and system, conditional on $h_{\text{feas}}(x_0) \geq 0$.

System	Standard	FGO	MPC	Backup CBF	**Ours**
ACC	29.9%	100.0%	92.1%	100.0%	**100.0%**
Overtake	55.8%	59.8%	95.6%	99.4%	**100.0%**
RobotControl	0.2%	100.0%	98.9%	100.0%	**100.0%**



(a) ACC: Standard (b) ACC: FGO (c) ACC: Ours

(d) Overtake: Standard (e) Overtake: FGO (f) Overtake: Ours

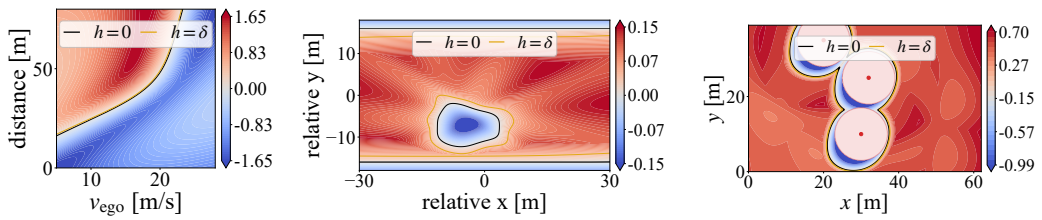
(g) RobotControl: Standard (h) RobotControl: FGO (i) RobotControl: Ours

Figure 1: Control performance comparison over three systems. Each plot shows the neural feasibility certificate value $h_{\text{feas}}(x_t)$ evaluated along closed-loop trajectories generated by the corresponding controller. The shaded region indicates $0 < h_{\text{feas}}(x_t) < \delta$.

Figures 1 and 2 further illustrate the learned feasibility certificates and their closed-loop behavior. Figure 1 shows that the standard and FGO controllers may drive the system outside the certified region, whereas our method keeps the trajectories within the certified feasible set by activating the learned recovery policy near the certificate boundary. Figure 2 visualizes the learned certificate over representative state slices, where $h_{\text{feas}}(x) \geq 0$ defines the certified feasible inner approximation and $h_{\text{feas}}(x) = \delta$ separates the interior region from the recovery region. The learned boundaries adapt to the different constraint geometries across ACC, Overtake, and RobotControl.

6.3 Control Performance

Table 2 compares the closed-loop control performance over the same 10 seeds. Detailed definitions of the evaluation metrics are provided in Appendix A.2. Overall, the proposed method achieves competitive task performance while maintaining low computational cost and positive safety margins. In



(a) ACC (b) Overtake (c) RobotControl

Figure 2: Visualization of learned feasibility certificates for the three systems. Nonnegative certificate values indicate the certified feasible inner approximation.

Table 2: Control performance over the 10 seeds (mean $\pm$ standard deviation). Computation time is in milliseconds and reaching time is in seconds.

Metric	Standard	FGO	MPC	Backup CBF	**Ours**
			ACC		
Track. err.	0.151 ± 0.022	0.148 ± 0.024	0.142 ± 0.023	0.153 ± 0.022	0.144 ± 0.023
Comp. time	0.087 ± 0.005	0.445 ± 0.022	4.440 ± 0.122	0.758 ± 0.017	0.077 ± 0.005
Safety margin	-3.488 ± 0.390	3.342 ± 0.513	0.500 ± 0.000	$(3.37 \pm 0.68) \times 10^{-4}$	2.591 ± 0.115
			Overtake		
Track. err.	1.164 ± 0.178	1.567 ± 0.604	0.915 ± 0.168	1.208 ± 0.160	1.444 ± 0.340
Comp. time	2.476 ± 0.092	6.051 ± 1.326	3.847 ± 0.211	0.893 ± 0.036	2.328 ± 0.347
Min obs. dist.	5.810 ± 1.070	13.126 ± 3.678	5.690 ± 0.208	5.265 ± 0.181	8.411 ± 1.000
			RobotControl		
Reach. time	34.710 ± 1.064	47.420 ± 19.437	29.460 ± 0.582	34.890 ± 0.296	34.670 ± 0.912
Comp. time	3.353 ± 0.040	2.591 ± 0.110	9.277 ± 0.147	1.578 ± 0.063	3.321 ± 0.051
Min obs. dist.	7.014 ± 0.002	7.383 ± 0.330	7.088 ± 0.006	8.894 ± 0.004	7.177 ± 0.013

ACC, our method attains a tracking error close to MPC with the lowest computation time among all methods, while maintaining a large positive safety margin. In Overtake, it achieves moderate tracking error and computation time while maintaining a larger minimum obstacle-center distance than the standard controller, MPC, and Backup CBF, without requiring an additional explicitly designed obstacle-avoidance backup path. In RobotControl, our method reaches the goal in a time comparable to the standard and Backup CBF controllers, with moderate computation cost and a mean minimum obstacle-center distance above the required separation. Together with the feasibility results, these findings demonstrate that the proposed feasibility mechanism improves reliability while retaining competitive control efficiency.

7 Conclusion

This paper developed a neural feasibility certificate framework for maintaining recursive feasibility in safety-constrained online control. First, we introduced a neural feasibility certificate that defines a certified inner approximation of the instantaneous feasible set, together with a margin-triggered recovery mechanism that retains the original controller in the certified interior and activates a learned witness policy near the boundary. Second, we developed a synthesis and formal verification framework to jointly learn the feasibility certificate and witness policy and certify recursive feasibility over the continuous state space, with additional compositional constructions for systems with multiple constraints. Third, experiments on adaptive cruise control, autonomous overtaking, and mobile robot navigation demonstrated reliable feasibility, competitive control performance, and low online computational overhead.

AI USE STATEMENT

In this work, we used generative AI tools to assist with discussions of theoretical derivations, formulation and refinement of mathematical claims, experimental design, interpretation and presentation of results, and manuscript writing and revision. We also used generative AI tools for recommended tasks including brainstorming, improving readability and structure, refining technical descriptions, and assisting with literature organization. We did not use generative AI tools to generate synthetic datasets and implement the proposed methods. All AI-assisted theoretical arguments, mathematical claims, experimental choices, and written content were independently reviewed, verified, and finalized by the authors. In particular, all proofs were checked by the authors for mathematical correctness, and all experimental results were produced and validated using the authors' own implementations. We take full responsibility for the final content of this work, including any text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work adheres to the ICLR Code of Ethics. No human subjects, animal experiments, or personally identifiable information were involved in this study. All experiments were conducted in simulated control environments. The study does not rely on private or sensitive datasets and does not involve data collection that raises privacy concerns.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the system dynamics, constraints, neural network architectures, training and verification procedures, evaluation metrics, random seeds, and hardware/software settings in the paper and appendices. All experiments are conducted in simulation, with paired initial conditions or matched random seeds across methods where applicable.

REFERENCES

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

Aaron D Ames, Xiangru Xu, Jessy W Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE transactions on automatic control*, 62(8): 3861–3876, 2016.

Mahathi Anand and Majid Zamani. Formally verified neural network control barrier certificates for unknown systems. *IFAC-PapersOnLine*, 56(2):2431–2436, 2023.

Huanhui Cao, Hao Xiong, Weifeng Zeng, Hantao Jiang, Zhiyuan Cai, Liang Hu, Lin Zhang, and Wenjie Lu. Safe reinforcement learning-based motion planning for functional mobile robots suffering uncontrollable mobile robots. *IEEE Transactions on Intelligent Transportation Systems*, 25(5):4346–4363, 2023.

Yuxiao Chen, Mrdjan Jankovic, Mario Santillo, and Aaron D Ames. Backup control barrier functions: Formulation and comparative study. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pp. 6835–6841. IEEE, 2021.

Richard Cheng, Gábor Orosz, Richard M Murray, and Joel W Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 3387–3395, 2019.

Andrew Clark. Verification and synthesis of control barrier functions. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pp. 6105–6112. Ieee, 2021.

Hongkai Dai, Benoit Landry, Lujie Yang, Marco Pavone, and Russ Tedrake. Lyapunov-stable neural-network control. *Robotics: Science and Systems*, 2021.

Mi Ding, Kaipeng Lin, Wang Lin, and Zuohua Ding. A novel counterexample-guided inductive synthesis framework for barrier certificate generation. In *2022 IEEE 33rd International Symposium on Software Reliability Engineering (ISSRE)*, pp. 263–273. IEEE, 2022.

Willem Esterhuizen, Karl Worthmann, and Stefan Streif. Recursive feasibility of continuous-time model predictive control without stabilising constraints. *IEEE Control Systems Letters*, 5(1):265–270, 2020.

Xiangkun He, Jingda Wu, Zhiyu Huang, Zhongxu Hu, Jun Wang, Alberto Sangiovanni-Vincentelli, and Chen Lv. Fear-neuro-inspired reinforcement learning for safe autonomous driving. *IEEE transactions on pattern analysis and machine intelligence*, 46(1):267–279, 2023.

Thomas A Henzinger, Konstantin Kueffner, and Emily Yu. Formal verification of neural certificates done dynamically. In *International Conference on Runtime Verification*, pp. 54–72. Springer, 2025.

Hanjiang Hu, Yujie Yang, Tianhao Wei, and Changliu Liu. Verification of neural control barrier functions with symbolic derivative bounds propagation. In *Proceedings of The 8th Conference on Robot Learning*, volume 270, pp. 1797–1814. PMLR, 2024.

Chuanrui Jiang, Hongkai Dai, and Andrew Clark. Verification and synthesis of compatible control lyapunov and multiple high-order control barrier functions. *IEEE Transactions on Automatic Control*, 2026.

Alexander Liniger and John Lygeros. A viability approach for fast recursive feasible finite horizon path planning of autonomous rc cars. In *Proceedings of the 18th international conference on hybrid systems: computation and control*, pp. 1–10, 2015.

Jun Liu, Yiming Meng, Maxwell Fitzsimmons, and Ruikun Zhou. Compositionally verifiable vector neural lyapunov functions for stability analysis of interconnected nonlinear systems. In *2024 American Control Conference (ACC)*, pp. 4789–4794. IEEE, 2024.

Diego Manzanas Lopez, Sung Woo Choi, Hoang-Dung Tran, and Taylor T Johnson. Nnv 2.0: the neural network verification tool. In *International Conference on Computer Aided Verification*, pp. 397–412. Springer, 2023.

Udayan Mandal, Guy Amir, Haoze Wu, Ieva Daukantas, Fletcher Lee Newell, Umberto Ravaioli, Baoluo Meng, Michael Durling, Kerianne Hobbs, Milan Ganai, et al. Safe and reliable training of learning-based aerospace controllers. In *2024 AIAA DATC/IEEE 43rd Digital Avionics Systems Conference (DASC)*, pp. 1–10. IEEE, 2024a.

Udayan Mandal, Guy Amir, Haoze Wu, Ieva Daukantas, Fletcher Lee Newell, Umberto J Ravaioli, Baoluo Meng, Michael Durling, Milan Ganai, Tobey Shim, et al. Formally verifying deep reinforcement learning controllers with lyapunov barrier certificates. *arXiv preprint arXiv:2405.14058*, 2024b.

Mark Niklas Mueller, Franziska Eckert, Marc Fischer, and Martin Vechev. Certified training: Small boxes are all you need. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=7oFuxtJtUMH.

Alireza Nadali, Vishnu Murali, Ashutosh Trivedi, and Majid Zamani. Neural closure certificates. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 21446–21453, 2024.

Grigory Neustroev, Mirco Giacobbe, and Anna Lukina. Neural continuous-time supermartingale certificates. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 27538–27546, 2025.

Gilberto Pin and Thomas Parisini. Networked predictive control of uncertain constrained nonlinear systems: Recursive feasibility and input-to-state stability analysis. *IEEE Transactions on Automatic Control*, 56(1):72–87, 2010.

Tianqi Qie, Weida Wang, Chao Yang, Ying Li, Yuhang Zhang, Wenjie Liu, and Changle Xiang. An improved model predictive control-based trajectory planning method for automated driving vehicles under uncertainty environments. *IEEE Transactions on Intelligent Transportation Systems*, 24(4):3999–4015, 2022.

Zengyi Qin, Kaiqing Zhang, Yuxiao Chen, Jingkai Chen, and Chuchu Fan. Learning safe multi-agent control with decentralized neural barrier certificates. In *International Conference on Learning Representations*, 2021.

Dejin Ren, Yiling Xue, Taoran Wu, and Bai Xue. Efficient verification and falsification of relu neural barrier certificates. *arXiv preprint arXiv:2511.10015*, 2025.

Andreas Schlaginhaufen, Philippe Wenk, Andreas Krause, and Florian Dorfler. Learning stable deep dynamics models for partially observed or delayed dynamical systems. *Advances in Neural Information Processing Systems*, 34:11870–11882, 2021.

Chao Song, Xizhe Zang, Boyang Chen, Yan Liu, Le Qi, and Jie Zhao. Capturability as controlled-invariant sets: Recursive feasibility for variable-stepping time s2s nmpc. *IEEE Robotics and Automation Letters*, 2026.

Manan Tayal, Hongchao Zhang, Pushpak Jagtap, Andrew Clark, and Shishir Kolathaya. Learning a formally verified control barrier function in stochastic environment. In *2024 IEEE 63rd Conference on Decision and Control (CDC)*, pp. 4098–4104, 2024. doi: 10.1109/CDC56724.2024.10886052.

Shiqi Wang, Huan Zhang, Kaidi Xu, Xue Lin, Suman Jana, Cho-Jui Hsieh, and J Zico Kolter. Beta-crown: Efficient bound propagation with per-neuron split constraints for neural network robustness verification. *Advances in neural information processing systems*, 34:29909–29921, 2021.

Xinyu Wang, Luzia Knoedler, Frederik Baymler Mathiesen, and Javier Alonso-Mora. Simultaneous synthesis and verification of neural control barrier functions through branch-and-bound verification-in-the-loop training. In *2024 European Control Conference (ECC)*, pp. 571–578, 2024. doi: 10.23919/ECC64448.2024.10591251.

Haoze Wu, Omri Isac, Aleksandar Zeljić, Teruhiro Tagomori, Matthew Daggitt, Wen Kokke, Idan Refaeli, Guy Amir, Kyle Julian, Shahaf Bassan, et al. Marabou 2.0: a versatile formal analyzer of neural networks. In *International Conference on Computer Aided Verification*, pp. 249–264. Springer, 2024.

Wei Xiao, Calin A Belta, and Christos G Cassandras. Feasibility-guided learning for constrained optimal control problems. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pp. 1896–1901. IEEE, 2020.

Wei Xiao, Calin A Belta, and Christos G Cassandras. Sufficient conditions for feasibility of optimal control problems using control barrier functions. *Automatica*, 135:109960, 2022.

Wei Xiao, Christos G Cassandras, and Calin A Belta. Learning feasibility constraints for control barrier functions. In *2023 European Control Conference (ECC)*, pp. 1–6. IEEE, 2023a.

Wei Xiao, Tsun-Hsuan Wang, Ramin Hasani, Makram Chahine, Alexander Amini, Xiao Li, and Daniela Rus. Barriernet: Differentiable control barrier functions for learning of safe robot control. *IEEE Transactions on Robotics*, 2023b.

Wei Xiao, Calin Belta, and Christos G Cassandras. Learning for feasible and safe control with control barrier functions: A tutorial. *Cybernetics and Intelligence*, 2026.

Yuezhu Xu and S Sivaranjani. Eclipse: Efficient compositional lipschitz constant estimation for deep neural networks. *Advances in Neural Information Processing Systems*, 37:10414–10441, 2024.

Lujie Yang, Hongkai Dai, Zhouxing Shi, Cho-Jui Hsieh, Russ Tedrake, and Huan Zhang. Lyapunov-stable neural control for state and output feedback: A novel formulation. In *Forty-first International Conference on Machine Learning*, 2024.

Yujie Yang, Yuxuan Jiang, Yichen Liu, Jianyu Chen, and Shengbo Eben Li. Model-free safe reinforcement learning through neural barrier certificate. *IEEE Robotics and Automation Letters*, 8 (3):1295–1302, 2023.

Emily Yu, Djordje Zikelic, and Thomas A Henzinger. Neural control and certificate repair via runtime monitoring. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 26409–26417, 2025.

Hongchao Zhang, Junlin Wu, Yevgeniy Vorobeychik, and Andrew Clark. Exact verification of relu neural control barrier functions. *Advances in neural information processing systems*, 36:5685–5705, 2023a.

Hongchao Zhang, Zhizhen Qin, Sicun Gao, and Andrew Clark. Seev: Synthesis with efficient exact verification for relu neural barrier functions. *Advances in Neural Information Processing Systems*, 37:101367–101392, 2024.

Songyuan Zhang, Yumeng Xiu, Guannan Qu, and Chuchu Fan. Compositional neural certificates for networked dynamical systems. In *Learning for Dynamics and Control Conference*, pp. 272–285. PMLR, 2023b.

Songyuan Zhang, Oswin So, Kunal Garg, and Chuchu Fan. Gcbf+: A neural graph control barrier function framework for distributed safe multi-agent control. *IEEE Transactions on Robotics*, 2025.

Hanrui Zhao, Niuniu Qi, Mengxin Ren, Xia Zeng, Zhenbing Zeng, and Zhengfeng Yang. Neural barrier certificates synthesis of nn-controlled continuous systems via counterexample-guided learning. In *Proceedings of the 61st ACM/IEEE Design Automation Conference*, pp. 1–6, 2024.

Jingyuan Zhou, Haoze Wu, Longhao Yan, and Kaidi Yang. Synthesis and verification of string stable control for interconnected systems via neural siss certificate. In *ICLR 2025 Workshop: VerifAI: AI Verification in the Wild*, 2025.

Jingyuan Zhou, Yuexuan Wang, and Kaidi Yang. Neural vector lyapunov–razumikhin certificates for delayed interconnected systems. In *Forty-third International Conference on Machine Learning*, 2026a. URL https://openreview.net/forum?id=mJEVzcuDNF.

Jingyuan Zhou, Longhao Yan, Jinhao Liang, and Kaidi Yang. Enforcing cooperative safety for reinforcement learning-based mixed-autonomy platoon control. *IEEE Transactions on Intelligent Transportation Systems*, 27(1):1592–1605, 2026b. doi: 10.1109/TITS.2025.3627592.

A Simulation Setting

A.1 System Dynamics

Adaptive cruise control. We first consider a one-dimensional vehicle-following task. The augmented system state is

$$x_k = \begin{bmatrix} d_k & v_{e,k} & v_{l,k} & \zeta_{l,k}^\top \end{bmatrix}^\top, \tag{14}$$

where d_k is the distance between the ego vehicle and the lead vehicle, $v_{e,k}$ is the ego vehicle speed, and $v_{l,k}$ is the lead vehicle speed. The additional state $\zeta_{l,k}$ generates the prescribed lead motion, with known update $\zeta_{l,k+1} = q_l(\zeta_{l,k})$ and lead acceleration $a_{l,k} = a_l(\zeta_{l,k})$. The control input is the ego vehicle acceleration

$$u_k = a_{e,k}, \qquad a_{\min} \le u_k \le a_{\max}. \tag{15}$$

The discrete-time dynamics are given by

$$d_{k+1} = d_k + \Delta t \left(v_{l,k} - v_{e,k}\right), \tag{16}$$

$$v_{e,k+1} = v_{e,k} + \Delta t\, u_k, \tag{17}$$

$$v_{l,k+1} = v_{l,k} + \Delta t\, a_{l,k}, \tag{18}$$

where $\Delta t > 0$ is the sampling period. The safety constraint is defined by a time-headway condition

$$h_{\text{safe}}(x_k) = d_k - d_{\min} - T_h v_{e,k} \ge 0, \tag{19}$$

where $d_{\min}$ is the minimum standstill distance and T_h is the desired time headway. This scenario is used to visualize the difference between the safe set and the recursively feasible set. In particular, when the ego vehicle speed is high and the available braking input is limited, some states may still satisfy $h_{\text{safe}}(x_k) \geq 0$ but no longer admit a feasible safety control action.

Mobile robot navigation through narrow passages. The second scenario considers a nonholonomic mobile robot navigating from an initial position to a target position while avoiding obstacles. The robot state and control input are

$$x_k = \begin{bmatrix} p_{x,k} & p_{y,k} & \theta_k & v_k \end{bmatrix}^\top, \qquad u_k = \begin{bmatrix} \omega_k & a_k \end{bmatrix}^\top, \tag{20}$$

where $(p_{x,k}, p_{y,k})$ is the robot position, θ_k is the heading angle, v_k is the linear speed, ω_k is the angular velocity, and a_k is the acceleration. The discrete-time dynamics are

$$p_{x,k+1} = p_{x,k} + \Delta t\, v_k \cos\theta_k, \tag{21}$$
$$p_{y,k+1} = p_{y,k} + \Delta t\, v_k \sin\theta_k, \tag{22}$$
$$\theta_{k+1} = \theta_k + \Delta t\, \omega_k, \tag{23}$$
$$v_{k+1} = v_k + \Delta t\, a_k. \tag{24}$$

The input and speed constraints are

$$\omega_{\min} \leq \omega_k \leq \omega_{\max}, \qquad a_{\min} \leq a_k \leq a_{\max}, \qquad 0 \leq v_k \leq v_{\max}. \tag{25}$$

For each circular obstacle with center $o_i = (o_{x,i}, o_{y,i})$ and safety radius r_i, the safety constraint is

$$h_{\text{safe},i}(x_k) = (p_{x,k} - o_{x,i})^2 + (p_{y,k} - o_{y,i})^2 - r_i^2 \geq 0. \tag{26}$$

We construct narrow-passage environments by placing multiple circular obstacles close to each other. This setting creates multiple simultaneous HOCBF constraints and often leads to infeasible online control problems when the robot approaches the passage with high speed or unfavorable heading. The proposed method is expected to activate the backup policy in the boundary recovery region and drive the state back to the interior feasible set $\mathcal{S}_\delta$.

Autonomous driving with moving obstacles. The third scenario considers an ego vehicle interacting with moving vehicles. The augmented system state is

$$x_k = \begin{bmatrix} p_{x,k} & p_{y,k} & \psi_k & v_k & \zeta_{o,k}^\top \end{bmatrix}^\top, \tag{27}$$

where $(p_{x,k}, p_{y,k})$ is the vehicle position, ψ_k is the heading angle, and v_k is the speed. The state $\zeta_{o,k}$ contains the surrounding vehicles' positions and the states of their prescribed motion models, with known update $\zeta_{o,k+1} = q_o(\zeta_{o,k})$. The control input is

$$u_k = \begin{bmatrix} s_k & a_k \end{bmatrix}^\top, \tag{28}$$

where $s_k := \tan\vartheta_k$, ϑ_k is the steering angle, and a_k is the longitudinal acceleration. This exact input reparameterization gives the control-affine kinematic bicycle model

$$p_{x,k+1} = p_{x,k} + \Delta t\, v_k \cos\psi_k, \tag{29}$$
$$p_{y,k+1} = p_{y,k} + \Delta t\, v_k \sin\psi_k, \tag{30}$$
$$\psi_{k+1} = \psi_k + \Delta t\, \frac{v_k}{L} s_k, \tag{31}$$
$$v_{k+1} = v_k + \Delta t\, a_k, \tag{32}$$

where L is the vehicle wheelbase. The input and speed constraints are

$$\tan\vartheta_{\min} \leq s_k \leq \tan\vartheta_{\max}, \qquad a_{\min} \leq a_k \leq a_{\max}, \qquad 0 \leq v_k \leq v_{\max}. \tag{33}$$

The steering limits satisfy $-\pi/2 < \vartheta_{\min} < \vartheta_{\max} < \pi/2$, and the physical steering angle is recovered as $\vartheta_k = \arctan s_k$. Each surrounding vehicle is treated as a moving obstacle with position $p_{i,k} = (p_{x,i,k}, p_{y,i,k})$. The collision-avoidance constraint is

$$h_{\text{safe},i}(x_k) = (p_{x,k} - p_{x,i,k})^2 + (p_{y,k} - p_{y,i,k})^2 - r_i^2 \geq 0. \tag{34}$$

The position components of the prescribed obstacle model satisfy

$$p_{i,k+1} = p_{i,k} + \Delta t\, v_{i,k}, \tag{35}$$

where $v_{i,k}$ is the obstacle velocity determined by $\zeta_{o,k}$. This scenario is used to evaluate the proposed method under dynamic obstacles and tight steering and acceleration limits, where standard online control problems may become infeasible during overtaking or lane change maneuvers.

A.2 Control Performance Metrics

Each metric is computed per rollout and reported as the mean $\pm$ sample standard deviation over the same 10 seeds. Let N denote the number of evaluated time steps.

Tracking error. For ACC, we use the dimensionless normalized speed MSE:

$$E_{\mathrm{ACC}} = \frac{1}{N}\sum_{k=0}^{N-1}\left(\frac{v_{\mathrm{ego},k} - v_{\mathrm{des}}}{v_{\mathrm{des}}}\right)^2, \tag{36}$$

where $v_{\mathrm{des}} = 24\,\mathrm{m/s}$. For Overtake, we use the lateral tracking RMSE in meters:

$$E_{\mathrm{OT}} = \sqrt{\frac{1}{N}\sum_{k=0}^{N-1}(y_k - y_{\mathrm{ref},k})^2}. \tag{37}$$

Goal reaching time. For RobotControl, reaching time is the first sampled time at which the robot enters the goal region:

$$T_{\mathrm{goal}} = \min\{t_k : \|p_k - p_{\mathrm{goal}}\|_2 \le 1\,\mathrm{m}\}. \tag{38}$$

Non-arrivals are assigned the 70 s evaluation horizon; this capped value does not indicate successful arrival.

Computation time. We measure the mean wall-clock time per online controller call within each rollout, in milliseconds. This includes certificate evaluation, optimization, and backup computation when invoked, but excludes simulation updates, initialization, JIT warm-up, and one-time path planning.

Safety-related metrics. For ACC, the minimum safety margin is

$$M_{\mathrm{ACC}} = \min_k(d_k - \tau v_{\mathrm{ego},k} - d_0), \tag{39}$$

where $\tau = 1.8\,\mathrm{s}$ and $d_0 = 5\,\mathrm{m}$. A negative value indicates a violation of the prescribed headway constraint, not necessarily a collision.

For Overtake and RobotControl, minimum obstacle distance is

$$D_{\min} = \min_{k,j}\|p_k - p_{j,k}^{\mathrm{obs}}\|_2, \tag{40}$$

where $p_{j,k}^{\mathrm{obs}}$ denotes the center of obstacle j. Both distance metrics are reported in meters and evaluated at sampled time steps. Obstacle distance measures center-to-center separation rather than boundary clearance; safety must therefore be assessed against the required separation, not against zero.

A.3 Implementation Details

Hardware and software. All experiments were conducted on a Linux workstation running Ubuntu 24.04, equipped with an Intel Core Ultra 9 285K CPU (24 cores), 125 GB RAM, and two NVIDIA GeForce RTX 5090 GPUs (32 GB memory each). The system uses Linux kernel 6.14.0 with CUDA 12.8 and NVIDIA driver 570.153.02.

Models and training. The ACC and RobotControl certificate networks use three hidden layers with 96 units per layer and ReLU activations. RobotControl evaluates a shared obstacle-relative certificate for the three obstacles and uses the minimum value for recovery decisions. Overtake uses a compositional certificate combining front-obstacle and lane components, together with a bounded neural backup policy.

Simulation and evaluation. The sampling periods are $0.05\,\mathrm{s}$ for ACC and $0.1\,\mathrm{s}$ for Overtake and RobotControl, with maximum horizons of 30, 12, and 70 s, respectively. The latest control-performance comparison uses ten seeds (0–9) and reports the mean and sample standard deviation of rollout-level metrics.

B THEORETICAL RESULTS

B.1 PROOF OF LEMMA 1

Proof. Take any $x_0 \in \mathcal{B}_\delta$ and apply the witness policy $u_k = \pi_w(x_k)$ until the trajectory reaches $\mathcal{S}_\delta$. By the boundary recovery condition in Definition 3, whenever $x_k \in \mathcal{B}_\delta$, the next state $x_{k+1} = f(x_k) + g(x_k)\pi_w(x_k)$ belongs to $\mathcal{X}$ and satisfies

$$h_{\text{feas}}(x_{k+1}) \geq h_{\text{feas}}(x_k) + \min\{\kappa, \delta - h_{\text{feas}}(x_k)\}. \tag{41}$$

Since $h_{\text{feas}}(x_k) \geq 0$, this implies $x_{k+1} \in \mathcal{S}_0$. Thus, the next state either reaches $\mathcal{S}_\delta$ or remains in $\mathcal{B}_\delta$, where the recovery condition continues to apply.

Define the remaining margin deficit as $e_k := \max\{0, \delta - h_{\text{feas}}(x_k)\}$. Before the trajectory reaches $\mathcal{S}_\delta$, the recovery condition gives

$$e_{k+1} \leq \max\{0, e_k - \kappa\}. \tag{42}$$

Iterating this inequality up to the first entry into $\mathcal{S}_\delta$ yields

$$e_k \leq \max\{0, e_0 - k\kappa\}. \tag{43}$$

Hence, the trajectory must reach $\mathcal{S}_\delta$ within at most $\lceil e_0/\kappa \rceil = \lceil (\delta - h_{\text{feas}}(x_0))/\kappa \rceil$ steps. □

B.2 PROOF OF LEMMA 2

Proof. For any $x \in \mathcal{S}_0$, choose a nearest sampling point $\hat{x} \in \mathcal{X}_{\text{grid}}$. By Definition 4, $\|x - \hat{x}\| \leq \eta_x$. Assumption 1 then gives $h_{\text{feas}}(x) \leq h_{\text{feas}}(\hat{x}) + L_h \eta_x$. Moreover, since $F_\star(x), F_\star(\hat{x}) \in \mathcal{X}$, we have

$$\begin{aligned} &|h_{\text{feas}}(F_\star(x)) - h_{\text{feas}}(F_\star(\hat{x}))| \\ &\quad \leq L_h \|F_\star(x) - F_\star(\hat{x})\| \\ &\quad \leq L_h L_{F_\star} \eta_x. \end{aligned} \tag{44}$$

Thus, $h_{\text{feas}}(F_\star(x)) \geq h_{\text{feas}}(F_\star(\hat{x})) - L_h L_{F_\star} \eta_x$.

First suppose $\mathcal{I}_{\text{grid}} \neq \varnothing$ and take any $x \in \mathcal{S}_\delta \subseteq \mathcal{S}_0$. If its nearest sampling point $\hat{x}$ belongs to $\mathcal{I}_{\text{grid}}$, then the choice of δ implies

$$\begin{aligned} h_{\text{feas}}(x) &\leq h_{\text{feas}}(\hat{x}) + L_h \eta_x \\ &\leq \delta - \varepsilon_\delta < \delta, \end{aligned} \tag{45}$$

contradicting $h_{\text{feas}}(x) \geq \delta$. Hence, $\hat{x} \notin \mathcal{I}_{\text{grid}}$. By the definition of $\mathcal{I}_{\text{grid}}$, $h_{\text{feas}}(F_\star(\hat{x})) \geq L_h L_{F_\star} \eta_x$, and therefore

$$\begin{aligned} h_{\text{feas}}(F_\star(x)) &\geq h_{\text{feas}}(F_\star(\hat{x})) - L_h L_{F_\star} \eta_x \\ &\geq 0. \end{aligned} \tag{46}$$

Together with $F_\star(x) \in \mathcal{X}$, this yields $F_\star(x) \in \mathcal{S}_0$.

If $\mathcal{I}_{\text{grid}} = \varnothing$, every sampling point satisfies $h_{\text{feas}}(F_\star(\hat{x})) \geq L_h L_{F_\star} \eta_x$. The same Lipschitz bound then gives $h_{\text{feas}}(F_\star(x)) \geq 0$ for every $x \in \mathcal{S}_0$. Consequently, $F_\star(x) \in \mathcal{S}_0$ for every $x \in \mathcal{S}_\delta$ and any $\delta > 0$. □

B.3 PROOF OF THEOREM 3

Proof. We prove the result by induction. The initial condition gives $x_0 \in \mathcal{S}_0 = \mathcal{S}_\delta \cup \mathcal{B}_\delta$. Suppose $x_k \in \mathcal{S}_0$ for some $k \geq 0$. We show that the applied input satisfies all HOCBF and input constraints and that $x_{k+1} \in \mathcal{S}_0$.

Case 1: $x_k \in \mathcal{S}_\delta$**.** By the certified feasible inner approximation condition in Definition 3, problem equation 4 is feasible at x_k. The switched controller equation 5 therefore applies $u_k = u^\star(x_k)$, which satisfies $u_k \in \mathcal{U}$ and $\psi_{i,m_i}(x_k, u_k) \geq 0$ for all i. Since the same margin δ satisfies the conditions of Lemma 2, the next state satisfies

$$x_{k+1} = F_\star(x_k) \in \mathcal{S}_0. \tag{47}$$

Case 2: $x_k \in \mathcal{B}_\delta$**.** The switched controller applies $u_k = \pi_w(x_k)$. By Definition 3, $\pi_w(x_k) \in \mathcal{U}$ and $\psi_{i,m_i}(x_k, \pi_w(x_k)) \geq 0$ for all i. Moreover, the next state $x_{k+1} = f(x_k) + g(x_k)\pi_w(x_k)$ belongs to $\mathcal{X}$ and satisfies

$$\begin{aligned} h_{\text{feas}}(x_{k+1}) &\geq h_{\text{feas}}(x_k) + \min\{\kappa, \delta - h_{\text{feas}}(x_k)\} \\ &\geq h_{\text{feas}}(x_k) \geq 0. \end{aligned} \tag{48}$$

Thus, $x_{k+1} \in \mathcal{S}_0$.

In both cases, the applied input satisfies all HOCBF and input constraints, and the next state remains in $\mathcal{S}_0$. By induction and the certified feasible inner approximation condition, we obtain

$$x_k \in \mathcal{S}_0 \subseteq \mathcal{X}_{\text{feas}}, \qquad \forall k \geq 0. \tag{49}$$

Hence, the online control problem is recursively feasible under the switched controller from every $x_0 \in \mathcal{S}_0$, with all HOCBF and input constraints satisfied at every time step. □

B.4 Proof of Lemma 4

Proof. For any fixed $x \in \mathcal{X}$, Farkas' lemma states that $A(x)u \leq b(x)$ is infeasible if and only if there exists $\lambda \geq 0$ such that $A(x)^\top \lambda = 0$ and $b(x)^\top \lambda < 0$. By positive scaling, the last condition can be normalized to $b(x)^\top \lambda = -1$. Hence, equation 9 has a solution if and only if there exists $x \in \mathcal{S}_0 \setminus \mathcal{X}_{\text{feas}}$. □

B.5 Proof of Lemma 5

Proof. Take any $x \in \mathcal{B}_\delta$ and choose a nearest sampling point $\hat{x} \in \mathcal{X}_{\text{grid}}$. Since $\|x - \hat{x}\| \leq \eta_x$, we have $h_{\text{feas}}(\hat{x}) \leq h_{\text{feas}}(x) + L_h\eta_x < \delta + L_h\eta_x$. Thus, $\hat{x}$ satisfies equation 10.

Using the Lipschitz bounds and the identity $h + \min\{\kappa, \delta - h\} = \min\{h + \kappa, \delta\}$, we obtain

$$\begin{aligned} h_{\text{feas}}(F_w(x)) &\geq h_{\text{feas}}(F_w(\hat{x})) - L_h L_{F_w} \eta_x \\ &\geq \min\{h_{\text{feas}}(\hat{x}) + \kappa, \delta\} + L_h\eta_x \\ &\geq \min\{h_{\text{feas}}(x) + \kappa, \delta\}. \end{aligned} \tag{50}$$

The last inequality follows from $h_{\text{feas}}(x) \leq h_{\text{feas}}(\hat{x}) + L_h\eta_x$. Subtracting $h_{\text{feas}}(x)$ establishes the result. □

B.6 Proof of Theorem 6

Proof. Lemma 4 establishes $\mathcal{S}_0 \subseteq \mathcal{X}_{\text{feas}}$. In $\mathcal{S}_\delta$, the original controller therefore provides a feasible input, and Lemma 2 ensures that the next state belongs to $\mathcal{S}_0$.

In $\mathcal{B}_\delta$, the witness input satisfies the original constraints by assumption. Lemma 5 gives $h_{\text{feas}}(F_w(x)) \geq h_{\text{feas}}(x) \geq 0$. Together with $F_w(x) \in \mathcal{X}$ from Assumption 1, this implies $F_w(x) \in \mathcal{S}_0$. Induction establishes the claimed invariance, recursive feasibility, and constraint satisfaction. □

C Scalability Analysis

This appendix establishes conditions for combining local feasibility certificates while preserving recursive feasibility. Instead of learning a single certificate for all constraints, we consider local certificates $\{h_i\}_{i=1}^N$ and the composite set $\mathcal{C}_{\text{comp}} := \bigcap_{i=1}^N \mathcal{C}_i$, where $\mathcal{C}_i := \{x \in \mathcal{X} \mid h_i(x) \geq 0\}$.

For each i, define $\mathcal{S}_i := \{x \in \mathcal{X} \mid h_i(x) \geq \delta_i\}$, $\mathcal{B}_i := \{x \in \mathcal{X} \mid 0 \leq h_i(x) < \delta_i\}$, and $\mathcal{S}_{\text{neg},i} := \{x \in \mathcal{X} \mid h_i(x) < 0\}$, with $\delta_i, \kappa_i > 0$. Assume $\mathcal{C}_{\text{comp}} \neq \varnothing$, and write $F(x, u) := f(x) + g(x)u$.

(1) Independent input channels. Local controllers can be combined directly when each input block affects only its corresponding certificate and HOCBF constraint.

Assumption 2 (Independent Local Control)**.** *The input is partitioned as $u = (u_1, \ldots, u_N)$ with $\mathcal{U} = \prod_{i=1}^N \mathcal{U}_i$. For every $x \in \mathcal{C}_{\text{comp}}$, $h_i(F(x, u))$ and $\psi_{i,m_i}(x, u)$ depend on u_i but not on the other input blocks.*

Each local controller selects $u_i \in \mathcal{U}_i$ satisfying $\psi_{i,m_i}(x,u) \geq 0$ and

$$h_i(F(x,u)) \geq \begin{cases} 0, & x \in \mathcal{S}_i, \\ h_i(x) + \min\{\kappa_i, \delta_i - h_i(x)\}, & x \in \mathcal{B}_i. \end{cases} \tag{51}$$

These conditions may be checked with the other input blocks fixed at admissible reference values. The successor under the composed input belongs to $\mathcal{X}$.

Theorem 7 (Recursive Feasibility via Orthogonal Actuation)**.** *Suppose Assumption 2 holds. Combining the local control inputs guarantees $x_k \in \mathcal{C}_{\text{comp}} \subseteq \mathcal{X}_{\text{feas}}$ for every $x_0 \in \mathcal{C}_{\text{comp}}$ and $k \geq 0$. The applied input satisfies all original HOCBF and input constraints. Thus, the online problem is recursively feasible on $\mathcal{C}_{\text{comp}}$.*

The independence condition ensures that combining the inputs preserves each local guarantee.

(2) Spatially separated recovery regions. When control inputs affect multiple certificates, composition is still possible if recovering one certificate cannot drive the state into another certificate's uncertified region.

Assumption 3 (Spatially Separated Recovery)**.** *A common interior controller maps $\bigcap_{i=1}^{N} \mathcal{S}_i$ into $\mathcal{C}_{\text{comp}}$. For each i, a witness policy $\pi_{w,i}$ satisfies*

$$h_i(F(x, \pi_{w,i}(x))) - h_i(x) \geq \min\{\kappa_i, \delta_i - h_i(x)\} \tag{52}$$

for every $x \in \mathcal{C}_{\text{comp}} \cap \mathcal{B}_i$. Both the interior controller and each active witness satisfy all original HOCBF and input constraints and produce successors in $\mathcal{X}$.

The recovery regions $\mathcal{B}_i$ are pairwise disjoint. During recovery, each step satisfies $\|x_{k+1} - x_k\| \leq L_{\text{dyn}}$. Moreover,

$$\operatorname{dist}(\mathcal{B}_i, \mathcal{S}_{\text{neg},j}) \geq \Delta > \max_{\ell} T_{r,\ell} L_{\text{dyn}}, \qquad i \neq j, \tag{53}$$

where $T_{r,i} := \lceil \delta_i / \kappa_i \rceil$.

Theorem 8 (Recursive Feasibility via Spatial Isolation)**.** *Suppose Assumption 3 holds. Use the common interior controller when all certificates are in their interior regions. When $x_k \in \mathcal{B}_i$, apply $\pi_{w,i}$ until the first entry into $\mathcal{S}_i$, then select the controller again. This guarantees $x_k \in \mathcal{C}_{\text{comp}} \subseteq \mathcal{X}_{\text{feas}}$ for every $x_0 \in \mathcal{C}_{\text{comp}}$ and $k \geq 0$. Each recovery phase takes at most $T_{r,i}$ steps, and all original HOCBF and input constraints are satisfied.*

The distance condition ensures that the state cannot reach another uncertified region before recovery ends. Together, these constructions support certificate reuse when either input independence or spatial separation holds. Computational savings arise when the local certificates use smaller state subsets or reusable structures. The proofs are given below.

C.1 Proof of Theorem 7

Proof. Take any $x \in \mathcal{C}_{\text{comp}}$ and let $u = (u_1, \ldots, u_N)$ be the composed input selected by the local controllers. Since each $u_i \in \mathcal{U}_i$ and $\mathcal{U} = \prod_{i=1}^{N} \mathcal{U}_i$, we have $u \in \mathcal{U}$.

By Assumption 2, $h_i(F(x,u))$ and $\psi_{i,m_i}(x,u)$ are independent of the other input blocks. Thus, replacing the admissible reference inputs used in each local check by the selected inputs from the other controllers preserves that check. Consequently, the composed input satisfies $\psi_{i,m_i}(x,u) \geq 0$ for every i. Since x was arbitrary, this also proves $\mathcal{C}_{\text{comp}} \subseteq \mathcal{X}_{\text{feas}}$.

For each i, the state x belongs to either $\mathcal{S}_i$ or $\mathcal{B}_i$. If $x \in \mathcal{S}_i$, equation 51 gives $h_i(F(x,u)) \geq 0$. If $x \in \mathcal{B}_i$, the same condition gives

$$\begin{aligned} h_i(F(x,u)) &\geq h_i(x) + \min\{\kappa_i, \delta_i - h_i(x)\} \\ &\geq h_i(x) \geq 0. \end{aligned} \tag{54}$$

Hence, all local certificates remain nonnegative. Together with $F(x,u) \in \mathcal{X}$ from Assumption 2, this yields $F(x,u) \in \mathcal{C}_{\text{comp}}$.

Starting from any $x_0 \in \mathcal{C}_{\text{comp}}$, induction therefore gives

$$x_k \in \mathcal{C}_{\text{comp}} \subseteq \mathcal{X}_{\text{feas}}, \qquad \forall k \geq 0. \tag{55}$$

The composed input satisfies all original HOCBF and input constraints at every step, establishing recursive feasibility under the composed controller. □

C.2 Proof of Theorem 8

Proof. For any $x \in \mathcal{C}_{\text{comp}}$, either $x \in \bigcap_{i=1}^{N} \mathcal{S}_i$ or $x \in \mathcal{B}_i$ for exactly one i, because the recovery regions are pairwise disjoint. In the first case, the common interior controller provides an input satisfying all original constraints. In the second case, $\pi_{w,i}(x)$ provides such an input. Thus, Assumption 3 implies $\mathcal{C}_{\text{comp}} \subseteq \mathcal{X}_{\text{feas}}$.

We first prove that the state remains in $\mathcal{C}_{\text{comp}}$. Suppose $x_k \in \mathcal{C}_{\text{comp}}$. If all certificates are in their interior regions, the common interior controller gives $x_{k+1} \in \mathcal{C}_{\text{comp}}$ by assumption. Otherwise, $x_k \in \mathcal{B}_i$ for a unique i, and the controller applies $\pi_{w,i}$. The recovery condition yields

$$\begin{aligned} h_i(x_{k+1}) &\geq h_i(x_k) + \min\{\kappa_i, \delta_i - h_i(x_k)\} \\ &\geq h_i(x_k) \geq 0. \end{aligned} \tag{56}$$

Moreover, $x_{k+1} \in \mathcal{X}$. For every $j \neq i$, spatial separation gives

$$\begin{aligned} &\operatorname{dist}(x_k, \mathcal{S}_{\text{neg},j}) \geq \Delta, \\ &\|x_{k+1} - x_k\| \leq L_{\text{dyn}} < \Delta, \end{aligned} \tag{57}$$

where the strict inequality follows from equation 53 and $T_{r,\ell} \geq 1$. Therefore, x_{k+1} cannot belong to $\mathcal{S}_{\text{neg},j}$, and $h_j(x_{k+1}) \geq 0$ for every $j \neq i$. It follows that $x_{k+1} \in \mathcal{C}_{\text{comp}}$.

While the i-th recovery phase continues, the state remains in $\mathcal{B}_i$, so the same one-step argument applies at each step. After the first entry into $\mathcal{S}_i$, the controller is selected again according to the current state. Thus, induction gives $x_k \in \mathcal{C}_{\text{comp}}$ for all $k \geq 0$, with all original HOCBF and input constraints satisfied.

It remains to bound the duration of each recovery phase. Suppose a phase starts at time k in $\mathcal{B}_i$, and define $e_s := \max\{0, \delta_i - h_i(x_{k+s})\}$. Before the first entry into $\mathcal{S}_i$, equation 52 gives

$$e_{s+1} \leq \max\{0, e_s - \kappa_i\}. \tag{58}$$

If the phase did not terminate within $T_{r,i}$ steps, this inequality would hold throughout those steps and yield

$$e_{T_{r,i}} \leq \max\{0, e_0 - T_{r,i}\kappa_i\} = 0, \tag{59}$$

since $e_0 \leq \delta_i$ and $T_{r,i} = \lceil \delta_i/\kappa_i \rceil$. This contradicts the state remaining outside $\mathcal{S}_i$. Hence, the actual first entry occurs within $T_{r,i}$ steps, and the controller is selected again at that time. Combining invariance with $\mathcal{C}_{\text{comp}} \subseteq \mathcal{X}_{\text{feas}}$ establishes recursive feasibility. □

D CEGIS Algorithm

Algorithm 1 summarizes the CEGIS procedure for learning and verifying the feasibility certificate and witness policy. Each iteration trains the networks using equation 12 and checks certified-set inclusion, witness admissibility, and recovery with the selected switching margin. Counterexamples and failed grid-check samples guide subsequent training, while unresolved verification checks are refined. A candidate is accepted only after all required conditions and assumptions are verified; otherwise, the procedure returns an unverified status when the iteration budget is exhausted.

Algorithm 1 Counterexample-Guided Synthesis and Verification

Require: Training set $\mathcal{D}$, initial margin $\delta > 0$, recovery parameter $\kappa > 0$, and iteration budget K
Ensure: Verified $(h_{\mathrm{feas}}, \pi_w, \delta)$, or an unverified status

```
Initialize the networks and Lipschitz estimates
for t = 1, ..., K do
    Train the networks using equation 12
    Attempt to establish valid Lipschitz bounds, construct the grid, and select δ using Lemma 2
    if these steps fail or S_δ ≠ ∅ cannot be established then
        Refine the candidate or grid and continue
    end if
    Verify set inclusion, witness admissibility, and recovery as required by Theorem 6
    if all required conditions and assumptions are certified then
        return (h_feas, π_w, δ)
    end if
    Add counterexamples and failed grid-check samples to D and determine their feasibility labels
    Update training margins and refine unresolved checks
end for
return Unverified status
```

E ADDTIONAL EXPERIMENTS

Table 3 summarizes the offline computational cost over three sampling seeds per scenario. All nine runs successfully passed formal verification within their specified domains. Mean verification times were 221.27s for ACC, 208.36s for Overtake, and 231.89s for RobotControl, with small variation across seeds. Each scenario required two CEGIS rounds across all three seeds. Importantly, this computational cost is incurred entirely offline. After verification, the certified controller does not require repeated formal verification during online execution. These results indicate that the proposed synthesis and verification pipeline can obtain formally verified certificates within a few minutes for the considered scenarios.

Table 3: Offline training and formal verification of feasibility certificates over three seeds. Times are mean $\pm$ sample standard deviation, accumulated over all CEGIS rounds.

Scenario	Training (s)	Verification (s)	CEGIS rounds	Verified
ACC	180.00 ± 0.00	221.27 ± 3.68	2	3/3
Overtake	180.00 ± 0.00	208.36 ± 0.58	2	3/3
RobotControl	180.00 ± 0.00	231.89 ± 3.19	2	3/3