Computers and Chemical Engineering 213 (2026) 109777

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/cace

Control Lyapunov-barrier function-based reinforcement learning of nonlinear processes

Xiaodong Cui [a], Arthur Khodaverdian [a], Panagiotis D. Christofides [a,b,*]

[a] Department of Chemical and Biomolecular Engineering, University of California, Los Angeles, CA 90095, USA
[b] Department of Electrical and Computer Engineering, University of California, Los Angeles, CA 90095-1592, USA

ARTICLE INFO

Keywords:
Reinforcement learning
Process safety
Control Lyapunov-barrier functions
Model predictive control
Nonlinear systems

ABSTRACT

Control Lyapunov-barrier function-based model predictive control (CLBF-MPC) provides a systematic way to optimize control performance while enforcing stability and safety constraints for nonlinear systems; however, solving the corresponding constrained optimization problem at every sampling time can be computationally expensive for real-time implementation. This work develops a control Lyapunov-barrier function-based reinforcement learning (CLBF-RL) control framework for learning a feedback policy that improves closed-loop performance while satisfying the required control Lyapunov-barrier function (CLBF) constraints. Instead of training an unconstrained policy, the proposed actor network is equipped with a CLBF gate that enforces the active CLBF condition during control action generation. As a result, the implemented control action satisfies the required constraints and the CLBF condition, while online implementation requires only neural-network evaluation and constraint checking, rather than solving the optimization problem. It is proved that the proposed CLBF-RL policy is feasible at every sampling time, keeps the closed-loop state inside the safe region, avoids the unsafe set, and drives the state to a robust inner CLBF level set. The proposed method is demonstrated on a non-isothermal continuous stirred-tank reactor (CSTR) with an embedded unsafe region. The results demonstrate that, for the case study considered, the CLBF-RL controller achieves a lower closed-loop control cost and requires less online computation time than CLBF-MPC, while maintaining the required safety and stability properties.

1. Introduction

Safety is a central requirement in the control of nonlinear systems, especially when the system dynamics are complex and operating constraints are tight (Ames et al., 2019). For such systems, the controller is required not only to meet the control objective, but also to ensure that the state trajectory remains within a prescribed admissible region. For instance, chemical processes are representative nonlinear systems in which operational safety is particularly critical (Albalawi et al., 2017). In practical process operation, physical and safety-related limits are often imposed on critical variables such as temperature, concentration, pressure, and flow rate to keep the process within a safe operating envelope (Stauffer and Chastain-Knight, 2021; Cui et al., 2024; Peters et al., 2026). Violation of these limits may result in off-specification production, equipment degradation, environmental release, or hazardous operating conditions (Lees, 2012; Crowl and Louvar, 2011). In industrial practice, these requirements are often represented by safe operating limits that define an admissible operating envelope for the process (Stauffer and Chastain-Knight, 2021); however, maintaining operation inside this envelope can be challenging because performance-oriented control objectives may drive the closed-loop state toward constraint boundaries or unsafe regions. Therefore, there is a strong need for feedback control strategies that can explicitly enforce safety requirements by design while maintaining desirable closed-loop performance for nonlinear systems.

Model predictive control (MPC) provides a systematic framework for the control of constrained nonlinear systems because it can explicitly incorporate input and state constraints into an online optimization problem while optimizing predicted closed-loop behavior (Qin and Badgwell, 2003; Rawlings et al., 2017; Mhaskar et al., 2006). Owing to its ability to directly include constraints in the controller design, MPC-based methods have been widely developed for safety-critical control problems. For example, Safeness Index-based MPC methods quantify the proximity of the process state to unsafe operating regions and incorporate safety-related requirements into the online optimization problem (Albalawi et al., 2017); however, although such methods can

* Corresponding author.
E-mail address: pdc@seas.ucla.edu (P.D. Christofides).

https://doi.org/10.1016/j.compchemeng.2026.109777
Received 31 May 2026; Received in revised form 16 June 2026; Accepted 22 June 2026
Available online 29 June 2026

guide the controller away from unsafe regions, they cannot guarantee that the closed-loop state trajectory will not enter unsafe regions. To address this issue, control barrier function (CBF)-based MPC formulations impose barrier-based constraints to prevent the state from entering unsafe regions, while control Lyapunov-barrier function (CLBF)-based MPC formulations further combine safety and stabilization requirements within the MPC optimization problem (Zeng et al., 2021; Wu et al., 2019). These developments demonstrate that MPC can explicitly account for safety constraints in the controller design; however, MPC-based approaches may face practical limitations in real-time nonlinear control applications. In particular, MPC requires solving a constrained optimization problem at each sampling time, and the computational cost can increase significantly with system nonlinearity, prediction horizon, and problem size (Chen et al., 2000; Khodaverdian et al., 2025; Cui et al., 2026). In addition, the finite prediction horizon may lead to conservative or locally myopic decisions when long-term closed-loop performance is important (Liu and Liu, 2016). For nonlinear systems with tight constraints, numerical optimization may also suffer from infeasibility, convergence issues, or excessive solution times, which can limit the real-time use of MPC-based safety controllers (Cui et al., 2026; Löfberg, 2012). These limitations motivate the development of alternative control architectures that can retain safety and stability guarantees while reducing the required online computation.

Reinforcement learning (RL) provides a promising learning-based approach for addressing this need because it can learn feedback policies from data and optimize long-term closed-loop performance through interaction with the system or a simulated environment (Sutton and Barto, 2018; Haarnoja et al., 2018; Kaelbling et al., 1996; Y. Wang et al., 2024). Once trained, the learned policy can generate control actions through direct policy evaluation, which is typically much faster than solving a constrained optimization problem at each sampling time (Faria et al., 2022; Levine et al., 2020; Elmaz et al., 2023). This feature makes RL attractive for real-time nonlinear control applications where repeated online optimization may be computationally expensive; however, standard RL algorithms are usually designed to maximize cumulative reward and do not inherently enforce state or input constraints (Garcıa and Fernández, 2015; Berkenkamp et al., 2017; Osinenko et al., 2022). As a result, a learned neural-network policy may improve the reward while driving the closed-loop state toward constraint boundaries or unsafe regions, making it difficult to directly certify prescribed safety and stability properties (Osinenko et al., 2022). Therefore, although RL offers significant advantages in terms of online computational efficiency and long-term performance optimization, additional mechanisms are needed to ensure safe and stable operation of nonlinear systems.

To overcome these limitations, several studies have incorporated Lyapunov and barrier conditions into RL-based control. One common direction is to use CBF safety filters, where the learned policy first proposes a nominal action and a quadratic program (QP) modifies this action when the safety condition may be violated (Cheng et al., 2019; Emam et al., 2022). This type of method can enforce CBF constraints during implementation, but it still requires solving a QP online, and its safety guarantee depends on the feasibility and accuracy of the filter. Related work has also combined CLF and CBF constraints with RL under model uncertainty by using a CLF-CBF-QP to compute the applied input (Choi et al., 2020); however, in some formulations, the QP is constructed for an auxiliary variable μ rather than directly for the original nonlinear control input u. When μ is mapped back to u, constraints imposed on μ do not necessarily imply that the original input constraints on u are satisfied unless the mapping and admissible input set are explicitly considered. In addition, QP filters are most direct when the safety and stability conditions can be written as affine inequalities in the decision variable, which may limit their use for more general nonlinear constraints. More recently, CLBF-related RL methods have been developed to encode stability and safety requirements through the learned critic or performance index (Wang and Wu, 2024). These methods provide useful theoretical insights, but they often require the critic to satisfy restrictive conditions, and training such a critic accurately can be challenging. Moreover, approximation errors in the critic may affect the rigorous satisfaction of the desired CLBF conditions, and the cost function may need to be designed in a specific form to support the theoretical analysis. These limitations motivate the development of a CLBF-based RL framework that reduces the need for online optimization while more directly learning admissible feedback policies for nonlinear systems.

In this work, we propose a CLBF-based RL (CLBF-RL) control framework for nonlinear systems with input constraints and unsafe operating regions. The controller trains an actor to improve closed-loop performance, but the actor is not allowed to output arbitrary neural-network actions. Instead, each candidate action is checked against the active CLBF-based constraint and the actuator limits before it can be implemented. If the candidate action is inadmissible, the controller applies a known admissible fallback input. In this way, learning is used to improve performance within a CLBF certified action structure, rather than to replace the CLBF certificate. The closed-loop properties of the resulting controller are analyzed under bounded disturbances and sample-and-hold implementation. The proposed method is then applied to a non-isothermal CSTR with an embedded unsafe region to evaluate its safety, control performance, and online computation time. The remainder of this paper is organized as follows. Section 2 introduces the class of nonlinear systems, the unsafe-region characterization, the CLBF preliminaries, CLBF-MPC, and RL-based control. Section 3 presents the proposed CLBF-RL controller, including the CLBF gate, the learning formulation, and the closed-loop analysis. Section 4 demonstrates the proposed method on a CSTR example and compares its performance with benchmark controllers. Section 5 concludes the paper.

2. Preliminaries

2.1. Notation

The actual state and the nominal predicted state are denoted by x and $\tilde{x}$, respectively. The transpose and Euclidean norm of a vector x are denoted by $x^\top$ and $\|x\|$, respectively. The set of real numbers and nonnegative real numbers are denoted by $\mathbb{R}$ and $\mathbb{R}_{\geq 0}$, respectively. Set difference is denoted by $\Omega_1 \backslash \Omega_2$. The boundary and closure of a set Ω are denoted by $\partial\Omega$ and $\overline{\Omega}$, respectively. A function is said to be of class C^1 if it is continuously differentiable. For a scalar C^1 function $V : \mathbb{R}^n \to \mathbb{R}$, the Lie derivatives of V along f and g are denoted by $L_f V(x) = \frac{\partial V(x)}{\partial x} f(x)$ and $L_g V(x) = \frac{\partial V(x)}{\partial x} g(x)$, respectively. Thus, for the input-affine systems considered below, the nominal time derivative of V under input u is written as $\dot{V}(x,u) = L_f V(x) + L_g V(x) u$. For a given positive real number δ and point $x_c \in \mathbb{R}^n$, $\mathcal{B}_\delta(x_c) := \{x \in \mathbb{R}^n : \|x - x_c\| < \delta\}$ denotes the open ball centered at x_c with radius δ. The set of piecewise-constant functions with period Δ is denoted by $S(\Delta)$. The initial time is denoted by t_0, and sampling instants are denoted by t_k, with $t_{k+1} = t_k + \Delta$. For a given measured state $x(t_k)$ and admissible input trajectory $u \in S(\Delta)$, $\tilde{x}(t \mid t_k)$ denotes the nominal predicted state trajectory initialized from $\tilde{x}(t_k \mid t_k) = x(t_k)$. When a constant input u_k is applied over one sampling interval, $\tilde{x}(t \mid t_k; x(t_k), u_k)$ is used when the dependence on $x(t_k)$ and u_k needs to be made explicit. The unsafe region and the safe operating region are denoted by D and $\mathcal{X}_s$, respectively. In the RL setting, s denotes the observed state, a denotes the action proposed by a policy, and π denotes a policy mapping s to a. The replay buffer is denoted by $\mathcal{D}_{\mathrm{RL}}$.

2.2. Class of systems

This work considers a class of nonlinear continuous-time systems described by the following input-affine state-space form:

$$\dot{x} = F(x,u) + w(x,u,t) = f(x) + g(x)u + w(x,u,t), \qquad x(t_0) = x_0 \tag{1}$$

where $x = [x_1, x_2, \ldots, x_n]^\top \in \mathbb{R}^n$ is the state vector, $u = [u_1, u_2, \ldots, u_m]^\top \in \mathbb{R}^m$ is the manipulated input vector, and $F(x,u) := f(x) + g(x)u$ represents the nominal nonlinear process model. The input is constrained to satisfy $u \in U$, where $U \subset \mathbb{R}^m$ is assumed to be nonempty and compact, reflecting physical actuator limits. The functions $f(\cdot)$ and $g(\cdot)$ are assumed to be locally Lipschitz and continuously differentiable on an operating domain $\mathcal{X} \subset \mathbb{R}^n$ containing the origin. The term $w(x,u,t) : \mathcal{X} \times U \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ represents model mismatch and external disturbances in the real process. It is assumed to be uniformly bounded on the compact operating region considered in the closed-loop analysis, i.e., there exists $W_{\max} > 0$ such that $\|w(x,u,t)\| \leq W_{\max}$ for all relevant $x \in \mathcal{X}$, $u \in U$, and $t \geq t_0$. The system is written in deviation-variable form with respect to a nominal steady state, and the nominal model satisfies $F(0,0) = 0$. Thus, the origin is a steady state of the nominal system in Eq. (1) when $w = 0$. The state $x(t)$ is assumed to be available for feedback at each sampling instant t_k.

Remark 1. In process control applications, the compact input set U is specified directly from physical actuator limits, such as valve capacities, pump limits, heating/cooling capacities, or safety-related operating bounds. The disturbance bound $W_{\max}$ can be estimated conservatively from historical process data, model-identification residuals, operating experience, or high-fidelity simulation over the process operating region. The smoothness assumptions on $f(\cdot)$ and $g(\cdot)$ can be checked from the process model. For first-principles chemical process models derived from mass and energy balances, these vector fields are typically continuously differentiable on compact operating regions that avoid physical singularities (like phase change).

2.3. Characterization of unsafe regions

We assume the existence of a set $D \subset \mathcal{X}$ that represents unsafe operating states of the system of Eq. (1). The corresponding safe operating region is defined as $\mathcal{X}_s := \mathcal{X} \backslash D$, and it is assumed that $0 \in \mathcal{X}_s$. The unsafe region D may be identified through safety analysis based on first-principles process knowledge, operational data, or both. In this work, D is an unsafe region embedded in the operating region. Specifically, D is assumed to be a bounded open set satisfying $\overline{D} \subset \mathcal{X}$ and $0 \notin \overline{D}$. This assumption excludes the nominal steady state from the unsafe region and allows the unsafe region to have a general geometry that is not necessarily expressible as simple component-wise bounds on the state variables.

Definition 1 (*Safety (Wang and Wu, 2024)*)**.** Consider the system of Eq. (1) with input constraints $u \in U$, an unsafe region $D \subset \mathcal{X}$, and the safe operating region $\mathcal{X}_s = \mathcal{X} \backslash D$. A feedback policy $u = \Phi(x) \in U$ is said to ensure system safety with respect to an initial set $\mathcal{X}_{\text{init}} \subseteq \mathcal{X}_s$ if, for every initial condition $x(t_0) \in \mathcal{X}_{\text{init}}$, the corresponding closed-loop trajectory satisfies $x(t) \in \mathcal{X}_s$ for all $t \geq t_0$, or equivalently, $x(t) \notin D$ for all $t \geq t_0$.

2.4. Control Lyapunov-barrier function

The definitions in this subsection are stated for the nominal system of Eq. (1) with $w = 0$, i.e., $\dot{x} = F(x,u) = f(x) + g(x)u$. The disturbance term is handled separately in the robustness analysis through the bound $\|w(x,u,t)\| \leq W_{\max}$.

Definition 2 (*Control Lyapunov Function*)**.** A continuously differentiable, proper, and positive definite function $V_L : \mathcal{X} \to \mathbb{R}_{\geq 0}$ is called a control Lyapunov function for the nominal system if:

$$L_f V_L(x) < 0, \qquad \forall x \in \{z \in \mathcal{X} \backslash \{0\} \mid L_g V_L(z) = 0\} \tag{2}$$

and V_L satisfies the small-control property (Lin and Sontag, 1991). Specifically, for every $\epsilon > 0$, there exists $\delta > 0$ such that, for all $x \in B_\delta(0)$, there exists an input u satisfying $\|u\| < \epsilon$ and $L_f V_L(x) + L_g V_L(x) u < 0$.

A CLF characterizes the stabilizability of the nominal system. In this work, we assume that a stabilizing reference feedback law $\Phi(x) \in U$ is available for the nominal system on the operating region considered in the analysis. Such a controller may be constructed using a CLF-based design, for example, a saturated Sontag-type feedback law when the corresponding assumptions are satisfied (Lin and Sontag, 1991).

Definition 3 (*Control Barrier Function (Wu et al., 2019)*)**.** Given an unsafe region $D \subset \mathcal{X}$, a C^1 function $B : \mathcal{X} \to \mathbb{R}$ is called a control barrier function if it satisfies:

$$B(x) > 0, \qquad \forall x \in D \tag{3a}$$

$$L_f B(x) \leq 0, \qquad \forall x \in \{z \in \mathcal{X} \backslash D \mid L_g B(z) = 0\} \tag{3b}$$

$$X_B = \{x \in \mathcal{X} \mid B(x) \leq 0\} \neq \emptyset \tag{3c}$$

The set X_B represents a certified safe set associated with the CBF. Since $B(x) > 0$ on D and $B(x) \leq 0$ on X_B, it follows that $X_B \cap D = \emptyset$. While a CLF encodes stabilization and a CBF encodes safety, a control Lyapunov-barrier function combines these two roles.

Definition 4 (*Constrained Control Lyapunov-Barrier Function (Wu et al., 2019)*)**.** Given an unsafe region $D \subset \mathcal{X}$, a proper, lower-bounded, and C^1 function $W_c : \mathcal{X} \to \mathbb{R}$ is called a constrained control Lyapunov-barrier function on a region $\phi_{uc} \subset \mathcal{X}$ if W_c has a minimum at the origin and there exist a constant $\rho_c \in \mathbb{R}$ and a class-$\mathcal{K}$ function $r(\cdot)$ such that:

$$W_c(x) > \rho_c, \qquad \forall x \in D \subset \phi_{uc} \tag{4a}$$

$$\left\| \frac{\partial W_c(x)}{\partial x} \right\| \leq r(\|x\|), \qquad \forall x \in \phi_{uc} \tag{4b}$$

$$L_f W_c(x) < 0, \qquad \forall x \in \{z \in \phi_{uc} \backslash (D \cup \{0\} \cup X_e) \mid L_g W_c(z) = 0\} \tag{4c}$$

$$U_{\rho_c} = \{x \in \phi_{uc} \mid W_c(x) \leq \rho_c\} \neq \emptyset \tag{4d}$$

where $X_e = \{x \in \phi_{uc} \backslash (D \cup \{0\}) \mid \partial W_c(x)/\partial x = 0\}$ denotes the set of stationary points of W_c other than the origin.

The definitions above specify the properties required of CLFs, CBFs, and constrained CLBFs. The construction of a specific CLBF used in the proposed controller is given in Section 3.1, where a CLF and a CBF are combined and the resulting function is verified to satisfy the constrained CLBF properties needed for the subsequent analysis.

Remark 2. The constrained CLBF definition used in this work contains the conditions needed for the initial set considered in the closed-loop analysis, namely U_{ρ_c}. In the more general CLBF definition, an additional separation condition is imposed when initial conditions in $\phi_{uc} \backslash (D \cup U_{\rho_c})$ are also included (Wu et al., 2019):

$$\overline{\phi_{uc} \backslash (D \cup U_{\rho_c})} \cap \overline{D} = \emptyset \tag{5}$$

Since the stability and safety theorem in this work assumes $x(t_0) \in U_{\rho_c}$, this additional condition is not required for the subsequent analysis. If initial conditions outside U_{ρ_c} are considered, then Eq. (5) should also be imposed.

2.5. CLBF-based MPC

CLBF-based MPC computes the control action online by solving a finite-horizon optimization problem using the nominal model $F(x,u)$ as the prediction model. In this subsection, we assume that a constrained CLBF W_c satisfying Definition 4 is available. The construction of the CLBF used in the proposed CLBF-based RL framework will be presented later in Section 3.1. The role of the CLBF-MPC formulation here is to introduce the nominal CLBF-based constraints that are later embedded into the proposed constraint-informed policy structure.

Let $\rho^W_{\min}$ be a CLBF level satisfying $W_c(0) < \rho^W_{\min} < \rho_c$, where ρ_c is the CLBF level separating the unsafe region from the admissible CLBF sublevel set. We denote the corresponding sublevel set by $U_{\rho^W_{\min}} := \{x \in$

$\phi_{uc} \mid W_c(x) \le \rho_{\min}^W\}$. This set describes a local CLBF sublevel region around the origin. In the CLBF-MPC formulation, once the state enters $U_{\rho_{\min}^W}$, the nominal predicted trajectory is constrained to remain in $U_{\rho_{\min}^W}$.

At each sampling instant t_k, given the measured state $x(t_k)$, the CLBF-MPC optimization problem is formulated as follows:

$$\mathcal{J}^* = \min_{u \in S(\Delta)} \int_{t_k}^{t_k + P_N \Delta} L(\tilde{x}(t), u(t))\, dt \tag{6a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = F(\tilde{x}(t), u(t)), \qquad t \in [t_k, t_k + P_N \Delta) \tag{6b}$$

$$u(t) \in U, \qquad t \in [t_k, t_k + P_N \Delta) \tag{6c}$$

$$\tilde{x}(t_k) = x(t_k) \tag{6d}$$

$$\frac{\partial W_c(x(t_k))}{\partial x} F(x(t_k), u(t_k)) \le \frac{\partial W_c(x(t_k))}{\partial x} F(x(t_k), \Phi(x(t_k))), \tag{6e}$$

$$\text{if } x(t_k) \notin U_{\rho_{\min}^W} \text{ and } x(t_k) \notin \mathcal{B}_\delta(x_e)$$

$$W_c(\tilde{x}(t)) \le \rho_{\min}^W, \qquad t \in [t_k, t_k + P_N \Delta), \tag{6f}$$

$$\text{if } x(t_k) \in U_{\rho_{\min}^W} \text{ and } x(t_k) \notin \mathcal{B}_\delta(x_e)$$

$$W_c(\tilde{x}(t)) < W_c(x(t_k)), \qquad t \in (t_k, t_k + P_N \Delta), \tag{6g}$$

$$\text{if } x(t_k) \in \mathcal{B}_\delta(x_e)$$

where $\tilde{x}(t)$ denotes the nominal predicted state trajectory, P_N is the number of sampling periods in the prediction horizon, $S(\Delta)$ denotes the set of piecewise-constant input functions with period Δ, and $L(\tilde{x}, u)$ is the stage cost. Eq. (6b) uses the nominal model for prediction, Eq. (6c) enforces the input constraint, and Eq. (6d) initializes the predicted state at the measured state.

The CLBF constraints in Eq. (6) are imposed according to the current measured state $x(t_k)$. If $x(t_k) \notin U_{\rho_{\min}^W}$ and $x(t_k) \notin \mathcal{B}_\delta(x_e)$, Eq. (6e) is activated and requires the first control action to decrease W_c at least as fast as the reference controller $\Phi(x)$. If $x(t_k) \in U_{\rho_{\min}^W}$ and $x(t_k) \notin \mathcal{B}_\delta(x_e)$, Eq. (6f) is activated and requires $\tilde{x}(t) \in U_{\rho_{\min}^W}$ over the prediction horizon. If $x(t_k) \in \mathcal{B}_\delta(x_e)$, Eq. (6g) is activated and requires the predicted trajectory to move to a lower CLBF level so that the state does not remain near x_e. After solving Eq. (6), only the first control action $u^*(t_k)$ is applied to the actual system over $[t_k, t_{k+1})$, and the optimization problem is solved again at the next sampling instant. Finally, with respect to the role of the CLBF derivative constraint, it is used to ensure closed-loop stability for an appropriate set of initial conditions.

2.6. Reinforcement learning-based control

In contrast to MPC, which computes the control action online by solving an optimization problem at each sampling instant, RL-based control seeks to learn a policy π that maps the observed state to an action directly. In a sampled-data control setting, the agent observes a state s_k, applies an action $a_k = \pi(s_k)$, receives a reward r_k, and transitions to a successor state s_{k+1}. The resulting experience tuples are stored in a replay buffer, denoted by $\mathcal{D}_{\text{RL}} := \{(s_i, a_i, r_i, s_i')\}_{i=1}^N$, which can be used to train the policy and the associated value-function approximators. In this notation, the subscript i denotes the index of a sample in the replay buffer.

The objective in RL is to learn a policy that maximizes the expected discounted cumulative reward. For a given policy π, the value function is defined as the expected return starting from state s, and it satisfies the Bellman relation:

$$V^\pi(s) = \mathbb{E}\left[r(s, \pi(s)) + \gamma V^\pi(s')\right] \tag{7}$$

where $0 < \gamma < 1$ is the discount factor and the expectation is taken over the state transition induced by the system dynamics and the policy. Many RL methods also use an action-value function $Q^\pi(s, a)$, which evaluates the expected return obtained by applying action a at state s and then following policy π. The corresponding Bellman relation can be written as:

$$Q^\pi(s, a) = \mathbb{E}\left[r(s, a) + \gamma Q^\pi(s', \pi(s'))\right] \tag{8}$$

For continuous-state and continuous-action control problems, value-based methods such as Q-learning are difficult to apply directly because the action space is continuous. Actor-critic methods address this issue by using one approximator to represent the policy and another approximator to estimate the action-value function. Among these methods, deep deterministic policy gradient (DDPG) is a representative algorithm for deterministic continuous-action control; however, DDPG may overestimate Q-values during learning, which can lead to poor policy updates. Twin Delayed Deep Deterministic Policy Gradient (TD3) mitigates this issue by using two critic networks, delayed policy updates, and target policy smoothing. These modifications improve learning stability and control performance in continuous-action settings (Fujimoto et al., 2018). Therefore, TD3 is adopted in this work as the RL algorithm for policy learning.

3. Control Lyapunov-barrier function-based reinforcement learning

3.1. CLBF design

We construct a constrained CLBF following the design procedure in Wu and Christofides (2019, 2020). The main idea is to first design a CLF $V_L(x)$ for stabilization and a barrier function $B(x)$ associated with D, and then combine them through a weighted sum to obtain a CLBF $W_c(x)$. The construction is carried out for the nominal system of Eq. (1) with $w = 0$, i.e., $\dot{x} = F(x, u)$.

Let $H \subset \phi_{uc}$ be a bounded open connected set with compact closure satisfying $\overline{D} \subset H$, $\overline{H} \subset \phi_{uc}$, and $0 \notin \overline{H}$. The set H is an auxiliary set enclosing D and is used only in the CLBF construction. We choose a C^1 CLF $V_L : \mathcal{X} \to \mathbb{R}_{\ge 0}$ satisfying the quadratic bounds $c_{V,\ell}\|x\|^2 \le V_L(x) \le c_{V,u}\|x\|^2$ on the compact operating region considered in the analysis, where $c_{V,u} > c_{V,\ell} > 0$. In addition, $B : \mathcal{X} \to \mathbb{R}$ is chosen as a C^1 barrier function satisfying $B(x) = -\eta < 0$ for all $x \in \mathcal{X} \backslash H$, $B(x) \ge -\eta$ for all $x \in H$, and $B(x) > 0$ for all $x \in D$, where $\eta > 0$. Since $0 \notin \overline{H}$, it follows that $0 \in \mathcal{X} \backslash H$, and hence $B(0) = -\eta$.

The constrained CLBF is constructed as:

$$W_c(x) = V_L(x) + \mu B(x) + \nu \tag{9}$$

Define $c_H = \sup_{x \in \partial H} \|x\|^2$ and $c_D = \min_{x \in \partial D} \|x\|^2$. Since $\overline{H}$ is compact, $c_H < \infty$. Since D is bounded and $0 \notin \overline{D}$, the boundary ∂D is compact and separated from the origin, and therefore $c_D > 0$. The parameters μ and ν are selected as:

$$\mu > \frac{c_{V,u} c_H - c_{V,\ell} c_D}{\eta} \tag{10a}$$

$$\nu = \rho_c - c_{V,\ell} c_D \tag{10b}$$

To make the verification of the CLBF properties explicit, we use the following component-level sufficient conditions on the compact operating region considered in the closed-loop analysis. In addition to the quadratic bounds on V_L, suppose that there exist positive constants $c_{V,g}$, $c_{V,d}$, $c_{B,u}$, and $c_{B,g}$ such that:

$$\left\| \frac{\partial V_L(x)}{\partial x} \right\| \le c_{V,g}\|x\|, \qquad \forall x \in \phi_{uc} \tag{11a}$$

$$\frac{\partial V_L(x)}{\partial x} F(x, \Phi(x)) \le -c_{V,d}\|x\|^2, \qquad \forall x \in \phi_{uc} \backslash \mathcal{B}_\delta(x_e) \tag{11b}$$

$$0 \le B(x) + \eta \le c_{B,u}\|x\|^2, \qquad \forall x \in \phi_{uc} \tag{11c}$$

$$\left\| \frac{\partial B(x)}{\partial x} \right\| \le c_{B,g}\|x\|, \qquad \forall x \in \phi_{uc} \tag{11d}$$

The selected localized barrier is also required to be compatible with the reference feedback controller in the sense that:

$$\frac{\partial B(x)}{\partial x} F(x, \Phi(x)) \le 0, \qquad \forall x \in \phi_{uc} \backslash \mathcal{B}_\delta(x_e). \tag{12}$$

This compatibility condition is natural for localized barriers because $B(x)$ is constant outside H, and therefore $\partial B(x)/\partial x = 0$ outside the region where the barrier varies. Thus, the nontrivial verification is restricted to the compact buffer region around the unsafe set. Alternatively, when the implemented reference controller is saturated, the final CLBF decrease condition can be checked directly by verifying the residual:

$$\frac{\partial W_c(x)}{\partial x} F(x, \Phi(x)) + c_{V,d}\|x\|^2 \le 0, \qquad \forall x \in \phi_{uc} \backslash \mathcal{B}_\delta(x_e). \tag{13}$$

This residual check is useful because component-wise saturation does not automatically preserve the decrease property of an unsaturated stabilizing feedback.

We now verify the remaining constrained CLBF properties. Since $V_L(x)$ and $B(x)$ are C^1, $W_c(x)$ is also C^1. Moreover, $W_c(x)$ attains its global minimum at the origin. Specifically, $W_c(0) = -\mu\eta + \nu$. For any $x \neq 0$, if $x \in \mathcal{X}\backslash H$, then $B(x) = -\eta$, and thus $W_c(x) - W_c(0) = V_L(x) > 0$. If $x \in H$, then $B(x) \ge -\eta$, and thus $W_c(x) - W_c(0) = V_L(x) + \mu(B(x)+\eta) > 0$. Therefore, the global minimum of $W_c(x)$ is achieved at the origin.

Next, for all $x \in D$, since $B(x) > 0$, $\|x\|^2 \ge c_D$, and $\nu = \rho_c - c_{V,\ell} c_D$, it follows that:

$$W_c(x) = V_L(x) + \mu B(x) + \nu > c_{V,\ell} c_D + \rho_c - c_{V,\ell} c_D = \rho_c \tag{14}$$

Thus, $D \subset \{x \in \phi_{uc} \mid W_c(x) > \rho_c\}$.

Furthermore, since H is open, $\partial H \subset \mathcal{X}\backslash H$, and thus $B(x) = -\eta$ for all $x \in \partial H$. Also, $\|x\|^2 \le c_H$ for all $x \in \partial H$ by the definition of c_H. Therefore,

$$\begin{aligned} W_c(x) &= V_L(x) + \mu B(x) + \nu \\ &\le c_{V,u} c_H - \mu\eta + \rho_c - c_{V,\ell} c_D \\ &< \rho_c, \qquad \forall x \in \partial H \end{aligned} \tag{15}$$

where the last inequality follows from Eq. (10a). Hence, $\partial H \subset \{x \in \phi_{uc} \mid W_c(x) \le \rho_c\}$, which implies that this CLBF sublevel set is nonempty.

Define the CLBF-enlarged unsafe region as $D_H = \{x \in H \mid W_c(x) > \rho_c\}$. Since $D \subset D_H$, avoiding D_H also guarantees $x(t) \notin D$. Combining the above results with the component-level gradient bounds and the compatibility or residual decrease verification, the designed function $W_c(x) = V_L(x) + \mu B(x) + \nu$ satisfies the constrained CLBF conditions used in Definition 4 with D_H replacing D.

For later use, let $\rho_0 = W_c(0)$, define the shifted CLBF as $W_c^{\mathrm{sh}}(x) = W_c(x) - \rho_0$, and define the shifted admissible level as $\rho_c^{\mathrm{sh}} = \rho_c - \rho_0$. For any $\rho > 0$, define the shifted CLBF sublevel set as $U_\rho^{\mathrm{sh}} = \{x \in \phi_{uc} \mid W_c^{\mathrm{sh}}(x) \le \rho\}$. The admissible CLBF sublevel set associated with ρ_c can then be written in shifted form as $U_{\rho_c^{\mathrm{sh}}}^{\mathrm{sh}}$. We select an inner nominal shifted CLBF level $\rho_{\min}$ satisfying $0 < \rho_{\min} < \rho_c^{\mathrm{sh}}$. The set $U_{\rho_{\min}}^{\mathrm{sh}}$ is used later in the CLBF-based RL constraints. The relationship among D, H, D_H, $U_{\rho_c^{\mathrm{sh}}}^{\mathrm{sh}}$, $U_{\rho_{\min}}^{\mathrm{sh}}$, and the neighborhood $\mathcal{B}_\delta(x_e)$ is illustrated in Fig. 1.

The component-level conditions above imply the local CLBF bounds used later in the closed-loop analysis. Since $B(0) = -\eta$, one has:

$$W_c(0) = \nu - \mu\eta \tag{16}$$

Therefore, the shifted CLBF satisfies:

$$\begin{aligned} W_c^{\mathrm{sh}}(x) &= W_c(x) - W_c(0) \\ &= V_L(x) + \mu(B(x) + \eta) \end{aligned} \tag{17}$$

Using $V_L(x) \ge c_{V,\ell}\|x\|^2$, $V_L(x) \le c_{V,u}\|x\|^2$, and $0 \le B(x) + \eta \le c_{B,u}\|x\|^2$, we obtain:

$$c_{V,\ell}\|x\|^2 \le W_c^{\mathrm{sh}}(x) \le \left(c_{V,u} + \mu c_{B,u}\right)\|x\|^2 \tag{18}$$

Moreover, by the chain rule and the compatibility condition in Eq. (12),

$$\begin{aligned} \frac{\partial W_c(x)}{\partial x} F(x, \Phi(x)) &= \frac{\partial V_L(x)}{\partial x} F(x, \Phi(x)) + \mu \frac{\partial B(x)}{\partial x} F(x, \Phi(x)) \\ &\le -c_{V,d}\|x\|^2 \end{aligned} \tag{19}$$

Finally, by the triangle inequality,

$$\begin{aligned} \left\| \frac{\partial W_c(x)}{\partial x} \right\| &\le \left\| \frac{\partial V_L(x)}{\partial x} \right\| + \mu \left\| \frac{\partial B(x)}{\partial x} \right\| \\ &\le \left(c_{V,g} + \mu c_{B,g}\right)\|x\| \end{aligned} \tag{20}$$

Thus, the following local bounds hold with $c_1 = c_{V,\ell}$, $c_2 = c_{V,u} + \mu c_{B,u}$, $c_3 = c_{V,d}$, and $c_4 = c_{V,g} + \mu c_{B,g}$:

$$c_1\|x\|^2 \le W_c^{\mathrm{sh}}(x) \le c_2\|x\|^2 \tag{21a}$$

$$\frac{\partial W_c(x)}{\partial x} F(x, \Phi(x)) \le -c_3\|x\|^2, \qquad \forall x \in \phi_{uc} \backslash \mathcal{B}_\delta(x_e) \tag{21b}$$

$$\left\| \frac{\partial W_c(x)}{\partial x} \right\| \le c_4\|x\| \tag{21c}$$

where $c_j > 0$, $j = 1, 2, 3, 4$, are positive constants, and $x_e \in X_e$ is the stationary point considered in the CLBF construction. These properties are stronger than the basic constrained CLBF definition and are used later to establish sample-and-hold feasibility, stability, and safety properties under bounded disturbances.

Furthermore, since $F(\cdot,\cdot)$ and $W_c(\cdot)$ are sufficiently smooth, U is compact, and $U_{\rho_c^{\mathrm{sh}}}^{\mathrm{sh}}$ is compact, there exist positive constants M, L_x, and L_x' such that, for all $x, x' \in U_{\rho_c^{\mathrm{sh}}}^{\mathrm{sh}}$ and $u \in U$:

$$\|F(x, u)\| \le M \tag{22a}$$

$$\|F(x, u) - F(x', u)\| \le L_x\|x - x'\| \tag{22b}$$

$$\left| \frac{\partial W_c(x)}{\partial x} F(x, u) - \frac{\partial W_c(x')}{\partial x} F(x', u) \right| \le L_x'\|x - x'\| \tag{22c}$$

These bounds will be used in the sample-and-hold and robustness analysis.

Remark 3. The CLBF construction above is performed offline. In practice, the certified operating region and the unsafe region D are specified from process safety limits, first-principles knowledge, historical operating data, or safety analysis. The auxiliary set H is selected as a buffer set enclosing D, and a stabilizing Lyapunov component $V_L(x)$ and a localized barrier component $B(x)$ are then constructed. After $W_c(x) = V_L(x) + \mu B(x) + \nu$ is formed, μ and ν are tuned to satisfy the level-separation conditions in Eq. (14) and (15). The component-level bounds, the compatibility condition in Eq. (12), or the residual condition in Eq. (13) are then verified over the compact certified operating region using analytical bounds, dense sampling, interval verification, or a validated process model/simulator. If these checks fail, H, $B(x)$, μ, ν, or $\Phi(x)$ can be retuned and the verification repeated. Once certified, $W_c(x)$ is fixed during RL training and online implementation; the RL policy only learns performance-improving actions within the CLBF-certified admissible action structure.

3.2. Constraint-informed policy network design

Neural-network policies provide low online computational cost because the control action is obtained through direct forward evaluation. However, a standard neural-network output does not necessarily satisfy actuator constraints or the CLBF-based stability and safety constraints required by the controller design. A common remedy is to append a projection operation or a quadratic-programming safety filter after the neural-network output (Kasaura et al., 2023; Choi et al., 2020). Although such post-processing can enforce constraints, it introduces an additional online optimization step and may reduce useful gradient information when the projected action lies on an active constraint boundary. In particular, the projected action may become locally insensitive to directions normal to the active constraint, which can make it difficult for the actor to learn how to modify its proposal to improve performance while avoiding constraint violations.

A gate-based policy structure is used to preserve the low inference cost of neural-network control while enforcing the prescribed CLBF-based action constraints. Let $x = x(t_k)$ be the measured state at the

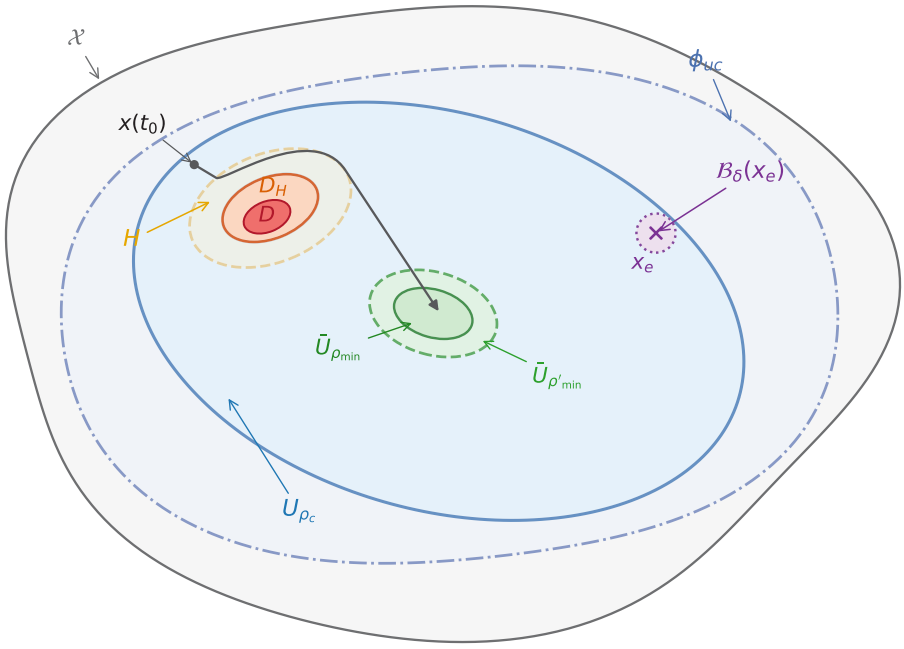


Fig. 1. Schematic illustration of the CLBF-related regions.

sampling instant t_k. The active CLBF-related constraints are written compactly as $h_j(x,u) \geq 0$, $j \in \mathcal{I}(x)$, where $\mathcal{I}(x)$ denotes the set of constraints that are active at x. The functions $h_j(x,u)$ may be algebraic in u, or they may depend on a nominal state prediction generated over one sampling interval. We assume that a known admissible input $u_f(x) \in U$ is available for the active mode and satisfies $h_j(x,u_f(x)) \geq 0$ for all $j \in \mathcal{I}(x)$. The input $u_f(x)$ may be obtained from the reference controller, from an auxiliary backup controller, or from a separate feasibility calculation associated with the active CLBF mode.

The actor network generates a candidate input using the measured state, the current admissible input, and normalized information about the active CLBF constraint. In the one-step case, the current admissible input is $u_f(x)$. In the refinement case described below, it is the current admissible iterate. The final layer uses a bounded activation function followed by affine scaling, so the nominal actor proposal is generated within the actuator range. In addition, the gate explicitly checks input admissibility, which is useful for numerical robustness and is consistent with the implementation.

The candidate input is accepted only if it satisfies the actuator constraint and all active CLBF-related constraints. Let $\mathbf{1}_U(u)$ be equal to one when $u \in U$ and zero otherwise. The hard step function is taken as $H(z) = 1$ when the checked margin is positive and $H(z) = 0$ otherwise. Equivalently, a small positive tolerance can be used in numerical implementation to make the acceptance test conservative. The hard acceptance gate is defined as:

$$\alpha(x,\hat{u}_\theta) = \mathbf{1}_U(\hat{u}_\theta) \prod_{j\in\mathcal{I}(x)} H\left(h_j(x,\hat{u}_\theta)\right) \tag{23}$$

The constraint-informed actor output is then defined by:

$$\pi_\theta(x) = u_f(x) + \alpha(x,\hat{u}_\theta)\left(\hat{u}_\theta - u_f(x)\right) \tag{24}$$

If the neural-network candidate satisfies the actuator constraint and all active CLBF-related constraints, then $\alpha = 1$, and the candidate action is returned. If at least one active constraint is violated, then $\alpha = 0$, and the actor returns the known admissible input $u_f(x)$. Therefore, the actor output is either an admissible neural-network proposal or the known admissible fallback input.

The hard gate in Eq. (23) is used in the forward pass to preserve the exact accept-or-reject decision. Since the step function is not differentiable, a straight-through estimator is used in the backward pass. Specifically, the forward value of the gate remains binary, while the backward pass uses the derivative of a smooth surrogate, chosen in this implementation as $\tanh(\mathrm{softplus}_\beta(z))$, where $\beta > 0$ controls the sharpness of the approximation. This allows gradient-based actor training while keeping the deployed actor as a hard constraint-gated map. The smooth approximation is used only for backpropagation and is not used to generate the implemented control action.

The same construction can be used in an optional refinement architecture. Starting from $u^{(0)} = u_f(x)$, the actor generates a proposal $\hat{u}_\theta^{(\ell)}(x,u^{(\ell)},c^{(\ell)})$ at refinement step ℓ, where $c^{(\ell)}$ denotes the normalized active-mode constraint information evaluated at the current iterate. The refinement update is:

$$u^{(\ell+1)} = u^{(\ell)} + \alpha^{(\ell)}\left(\hat{u}_\theta^{(\ell)}(x,u^{(\ell)},c^{(\ell)}) - u^{(\ell)}\right), \qquad c^{(\ell+1)} = c(x,u^{(\ell+1)}) \tag{25}$$

where:

$$\alpha^{(\ell)} = \mathbf{1}_U\left(\hat{u}_\theta^{(\ell)}\right) \prod_{j\in\mathcal{I}(x)} H\left(h_j(x,\hat{u}_\theta^{(\ell)})\right) \tag{26}$$

After N_r refinement steps, the internally gated actor output is $\pi_\theta(x) = u^{(N_r)}$. Since $u^{(0)}$ is admissible and each refinement step either accepts another admissible input or retains the previous admissible input, the final actor output satisfies $\pi_\theta(x) \in U$ and $h_j(x,\pi_\theta(x)) \geq 0$ for all $j \in \mathcal{I}(x)$. This property does not require the active constraint set to be convex because the forward pass uses a binary accept-or-reject decision, not a fractional interpolation between a feasible and an infeasible input. If the proposal generator does not depend on the current iterate or updated constraint information, repeated refinement steps reduce to the one-step gate in Eq. (24).

For the CLBF-based RL controller, the constraints $h_j(x,u) \geq 0$ are selected from the first-move constraints of the CLBF-MPC formulation. The shifted CLBF $W_c^{\mathrm{sh}}(x) = W_c(x) - \rho_0$ and shifted admissible level $\rho_c^{\mathrm{sh}} = \rho_c - \rho_0$ are used in the one-step CLBF constraints, where $\rho_0 = W_c(0)$. Let $0 < \rho_{\min} < \rho_c^{\mathrm{sh}}$ be the inner shifted CLBF level used in the nominal one-step constraints. The active mode is selected with priority given to the stationary-point mode as follows:

$$\mathcal{I}(x) = \begin{cases} \mathcal{I}_g, & x \in \mathcal{B}_\delta(x_e) \\ \mathcal{I}_e, & x \notin \mathcal{B}_\delta(x_e),\ W_c^{\mathrm{sh}}(x) > \rho_{\min} \\ \mathcal{I}_f, & x \notin \mathcal{B}_\delta(x_e),\ W_c^{\mathrm{sh}}(x) \leq \rho_{\min} \end{cases} \tag{27}$$

In the outer mode, corresponding to the CLBF-MPC constraint in Eq. (6e), the active constraint requires the proposed input to decrease W_c at least as fast as the reference controller $\Phi(x)$. This condition is written as:

$$h_e(x,u) = \frac{\partial W_c(x)}{\partial x} F(x,\Phi(x)) - \frac{\partial W_c(x)}{\partial x} F(x,u) \geq 0 \tag{28}$$

Equivalently, the proposed input satisfies $\frac{\partial W_c(x)}{\partial x} F(x,u) \leq \frac{\partial W_c(x)}{\partial x} F(x,\Phi(x))$.

In the inner mode, corresponding to the CLBF-MPC constraint in Eq. (6f), the nominal trajectory generated by the proposed sample-and-hold input is required to remain inside the inner CLBF level set over the first sampling interval. Let $\tilde{x}(t \mid t_k; x, u)$ denote the nominal state trajectory initialized at $\tilde{x}(t_k \mid t_k; x, u) = x$ and generated by the constant input u over $[t_k, t_{k+1}]$. The inner-mode constraint is written as:

$$h_f(x,u) = \rho_{\min} - \max_{t\in[t_k,t_{k+1}]} W_c^{\mathrm{sh}}\big(\tilde{x}(t \mid t_k; x, u)\big) \geq 0 \tag{29}$$

This condition enforces one-step nominal invariance of the inner CLBF level set. In numerical implementation, the maximum can be evaluated over the discretized nominal rollout used by the controller.

In the stationary-point mode, corresponding to the CLBF-MPC constraint in Eq. (6g), the nominal trajectory is required to remain in an admissible CLBF level during the first sampling interval and to achieve a strict endpoint decrease. To obtain a robust decrease condition under sample-and-hold implementation and bounded disturbances, the strict decrease is enforced with a positive margin $\eta_e > 0$. With a level $\rho_e > 0$, the stationary-point constraints are:

$$h_{g,1}(x,u) = \rho_e - \max_{t\in[t_k,t_{k+1}]} W_c^{\mathrm{sh}}\big(\tilde{x}(t \mid t_k; x, u)\big) \geq 0 \tag{30a}$$

$$h_{g,2}(x,u) = W_c^{\mathrm{sh}}(x) - \eta_e - W_c^{\mathrm{sh}}\big(\tilde{x}(t_{k+1} \mid t_k; x, u)\big) \geq 0 \tag{30b}$$

The first constraint keeps the nominal prediction inside a prescribed CLBF level, while the second constraint is a margin-based form of the strict CLBF decrease condition used in CLBF-MPC. The margin η_e is used to preserve the endpoint decrease after the nominal-to-actual trajectory mismatch caused by bounded disturbances is taken into account.

The fallback input $u_f(x)$ is selected before RL training and is not learned by the actor. Its choice depends on the active CLBF mode. In the outer mode, $u_f(x) = \Phi(x)$, where $\Phi(x)$ is the stabilizing reference controller used in the CLBF decrease condition. A practical construction of $\Phi(x)$ is a saturated Sontag-type feedback law associated with the designed Lyapunov or CLBF certificate. In the inner mode, $\Phi(x)$ is used when its one-step nominal rollout satisfies the inner-level CLBF constraint; otherwise, another certified backup input obtained from an offline one-step feasibility calculation can be used. Near the stationary point x_e, an auxiliary input $u_{\mathrm{aux}}(x)$ is selected and verified offline to satisfy the stationary-point level and endpoint-decrease constraints. These fallback candidates are checked over the compact certified operating region to verify the corresponding active CLBF constraints and actuator bounds. These fallback feasibility requirements are stated explicitly because the gate can preserve feasibility only when at least one admissible input is available for the active mode.

Thus, the policy network does not solve the full CLBF-MPC problem online. Instead, the CLBF-MPC constraints are used as evaluable first-move checks on the actor proposal. The constraints in the inner and stationary-point modes may require a one-step nominal rollout, but they do not require optimizing over a finite-horizon input sequence. If the actor proposal passes the active check, it is used; otherwise, the actor returns a known admissible action for the active mode.

This design differs from penalty-based RL. In a penalty-based method, constraint satisfaction is encouraged through the reward, but the trained policy may still violate constraints during deployment. In the proposed structure, the constraint check is part of the actor output map. Therefore, under the availability of $u_f(x)$, the internally gated actor output satisfies the actuator constraints and the active CLBF-MPC first-move constraint by construction. Since the constraint logic is implemented through hard gates rather than an online optimization layer, the controller preserves the low inference cost of a neural-network policy while using the CLBF-MPC constraints to guide the generated action. During training, any exploration noise added after the internally gated actor output is handled by the outer constraint check described in the next subsection before the input is applied to the process.

3.3. CLBF-based reinforcement learning

The constraint-informed neural network developed in Section 3.2 is used as the actor network in the proposed RL controller. Thus, the actor is not an unconstrained neural network. It contains the internal CLBF gate in Eq. (24) and (25), and its output is designed to satisfy the input constraints and the active CLBF-based constraints. The role of RL is to train this constraint-informed actor to improve closed-loop performance within the admissible action structure, rather than to learn a new Lyapunov or barrier certificate from data.

During training, exploration noise is added after the internally gated actor output. This noise is useful for action-space exploration, but it can move an otherwise admissible actor output outside the input constraints or the active CLBF constraint. Therefore, the proposed CLBF-RL training uses two layers of constraint enforcement. The first layer is the internal gate inside the actor network, as described in Section 3.2. The second layer is an outer constraint check before the input is applied to the process. If the noisy submitted action passes the outer check, it is implemented. If it violates the active constraints, it is replaced by a known admissible input for the active mode, such as the reference controller $\Phi(x)$ or the auxiliary input $u_{\mathrm{aux}}(x)$. When this replacement occurs, a correction penalty is added to prevent the RL update from assigning the favorable transition produced by the admissible replacement input to the inadmissible submitted action.

At each sampling instant t_k, the RL state is chosen as $s_k = x(t_k)$. The internally gated actor first generates $u_{\theta,k} = \pi_\theta(s_k)$, where $\pi_\theta(\cdot)$ is the constraint-informed actor defined in Section 3.2. During policy evaluation without exploration noise, the submitted action is $a_k = u_{\theta,k}$. During training, bounded exploration noise is added after the actor output, so that $a_k = u_{\theta,k} + \zeta_k$, where ζ_k denotes the exploration noise.

The submitted action a_k is then passed through the outer constraint check. Using the active constraints and active mode defined in Section 3.2, the implemented input is

$$u_k = \mathcal{E}_{\mathrm{out}}(s_k, a_k; u_f) = \begin{cases} a_k, & a_k \in U,\ h_j(s_k,a_k) \geq 0,\ j \in \mathcal{I}(s_k) \\ u_f(s_k), & \text{otherwise} \end{cases} \tag{31}$$

where $u_f(s_k)$ is the known admissible input associated with the active mode, as introduced in Section 3.2. In the outer mode, the reference controller Φ is used because it satisfies the CLBF decrease constraint in Eq. (28). In the inner mode, Φ or another certified backup input is used when it satisfies Eq. (29). Near the stationary point x_e, the auxiliary input u_{aux} is used when required to satisfy Eq. (30a) and (30b). Therefore, the input implemented on the process is u_k, not necessarily the submitted action a_k. The implemented input is applied in sample-and-hold form over $[t_k, t_{k+1})$, and the successor RL state is $s_{k+1} = x(t_{k+1})$. The actual process evolves according to Eq. (1), while the internal actor gate and the outer constraint check evaluate the CLBF constraints using the nominal model $F(x,u)$.

The one-step reward is defined as

$$r_k = -\int_{t_k}^{t_{k+1}} L(x(t),u_k)\,dt - \lambda_c \left\|a_k - u_k\right\|_{R_c}^2 \tag{32}$$

where $L(x,u)$ is the stage cost, $\lambda_c \geq 0$ is the correction-penalty weight, and R_c is a positive semidefinite weighting matrix. The first term evaluates the closed-loop cost under the input actually applied to the process. The second term is zero when the submitted action passes the outer constraint check, because $u_k = a_k$. If the submitted action violates the active constraints and the outer check replaces it with the

admissible input $u_f(s_k)$, the term $\lambda_c\|a_k-u_k\|^2_{R_c}$ penalizes this correction. Thus, the penalty is associated with the outer replacement mechanism during RL training, not with the internal actor gate itself. The penalty guides policy learning, while feasibility and safety are enforced by the internal gate, the outer check, and the availability of u_f.

The interaction between the internally gated actor, the outer constraint check, and the correction penalty is summarized in Algorithm 1.

Algorithm 1: CLBF-RL training interaction with two-layer constraint enforcement

Input : Current state $x(t_k)$, constraint-informed actor π_θ, admissible input set U, active constraints h_j, active index set $\mathcal{I}(x(t_k))$, admissible input u_f, correction-penalty weight λ_c

Set RL state $s_k \leftarrow x(t_k)$;
Compute internally gated actor output $u_{\theta,k} \leftarrow \pi_\theta(s_k)$;
Add exploration noise during training: $a_k \leftarrow u_{\theta,k} + \zeta_k$;
if $a_k \in U$ *and* $h_j(s_k, a_k) \geq 0,\ j \in \mathcal{I}(s_k)$ **then**
  Set implemented input $u_k \leftarrow a_k$;
  Set correction penalty $p_k \leftarrow 0$;
else
  Set implemented input $u_k \leftarrow u_f(s_k)$;
  Set correction penalty $p_k \leftarrow \lambda_c\|a_k - u_k\|^2_{R_c}$;
Apply $u(t) = u_k$ for all $t \in [t_k, t_{k+1})$;
Observe $s_{k+1} = x(t_{k+1})$;
Compute reward $r_k \leftarrow -\int_{t_k}^{t_{k+1}} L(x(t), u_k)\,dt - p_k$;
Store $(s_k, a_k, r_k, s_{k+1}, d_k)$ in $\mathcal{D}_{\mathrm{RL}}$;

The transition data are stored in the replay buffer as $(s_i, a_i, r_i, s_i', d_i)$, where a_i is the submitted action after exploration and d_i is the terminal indicator. The input applied to the process is $u_i = \mathcal{E}_{\mathrm{out}}(s_i, a_i; u_f)$. Therefore, the state transition and the control-cost component of the reward are generated by the implemented input, while the correction penalty records whether the submitted action required replacement by the admissible input.

The construction above is independent of the specific continuous-action RL algorithm. In this work, TD3 is used, as introduced in Section 2.6. Let $Q_{\psi_1}(s,a)$ and $Q_{\psi_2}(s,a)$ denote the two critic networks, and let $\pi_\theta(s)$ denote the internally gated actor. The corresponding target networks are denoted by $Q_{\psi_1}^{\mathrm{targ}}$, $Q_{\psi_2}^{\mathrm{targ}}$, and $\pi_{\theta^{\mathrm{targ}}}$. For a mini-batch sampled from $\mathcal{D}_{\mathrm{RL}}$, the target actor first generates an internally gated target action. Target policy smoothing noise is then added, and the resulting action is passed through the same outer constraint check:

$$a_i' = \mathcal{E}_{\mathrm{out}}\left(s_i', \pi_{\theta^{\mathrm{targ}}}(s_i') + \epsilon_i; u_f\right), \qquad \epsilon_i \sim \mathrm{clip}\left(\mathcal{N}(0,\sigma_\epsilon^2), -\epsilon_{\mathrm{clip}}, \epsilon_{\mathrm{clip}}\right) \tag{33}$$

This step ensures that target policy smoothing does not create an inadmissible target action. The critic target is computed as

$$y_i = r_i + \gamma(1-d_i)\min_{\ell=1,2} Q_{\psi_\ell}^{\mathrm{targ}}(s_i', a_i') \tag{34}$$

where $0<\gamma<1$ is the discount factor. The critic networks are updated using the standard TD3 critic loss with the target in Eq. (34). The actor is updated less frequently than the critics, also following TD3. Since $\pi_\theta(s_i)$ already includes the internal CLBF gate, the actor update is performed using the internally gated actor output. The target networks are updated by Polyak averaging.

For the CLBF-based RL controller, the final implemented input satisfies $u_k \in U$ and $h_j(s_k, u_k) \geq 0$ for all $j \in \mathcal{I}(s_k)$. When $s_k \notin \mathcal{B}_\delta(x_e)$ and $W_c^{\mathrm{sh}}(s_k) > \rho_{\min}$, this condition corresponds to the CLBF decrease constraint in Eq. (28). When $s_k \notin \mathcal{B}_\delta(x_e)$ and $W_c^{\mathrm{sh}}(s_k) \leq \rho_{\min}$, it corresponds to the one-step inner-level invariance constraint in Eq. (29). When $s_k \in \mathcal{B}_\delta(x_e)$, it corresponds to the stationary-point level and endpoint-decrease constraints in Eq. (30a) and (30b).

During deployment, no exploration noise is added. The submitted action is then the internally gated actor output, $a_k = \pi_\theta(s_k)$, and the outer check is kept as a final verification step. If the actor output satisfies the active constraints, it is applied directly; otherwise, the admissible input u_f is applied. Therefore, the implemented input remains admissible by construction.

The critic functions Q_{ψ_1} and Q_{ψ_2} are RL value approximators used for policy improvement. They are not Lyapunov or CLBF functions. Stability and safety are enforced through the prescribed CLBF $W_c(x)$, the active CLBF-MPC first-move constraints, the internal actor gate, and the outer constraint check. The role of RL is to optimize accumulated performance among the admissible inputs generated by this two-layer action structure.

Remark 4. The outer constraint check in Eq. (31) is kept not only during training, but also during online implementation. Although the actor contains the internal CLBF gate, the deployed neural network is evaluated using finite-precision arithmetic, and the constraint functions are also evaluated numerically. Therefore, when the actor output lies very close to an active constraint boundary, small numerical errors may lead to a slight constraint violation. In the CSTR implementation in Section 4, such cases were rare, accounting for around 0.5% of the evaluated actions, but the outer check prevents these numerical violations from being applied to the process. If the final check is not passed, the controller applies the admissible input u_f, such as the reference controller Φ in the corresponding CLBF mode. An alternative implementation is to tighten the internal gate by requiring a positive margin in the CLBF constraint, so that small numerical errors in neural-network evaluation or constraint checking still leave the original constraint satisfied. In this work, we keep the explicit outer check because it is simple, inexpensive, and directly preserves the admissibility of the implemented input.

3.4. Feasibility, stability and safety analysis

We establish the feasibility, practical stability, and safety properties of the proposed CLBF-based RL controller under sample-and-hold implementation and bounded disturbances. Recall that $W_c^{\mathrm{sh}}(x) = W_c(x) - \rho_0$ and $\rho_c^{\mathrm{sh}} = \rho_c - \rho_0$, and define:

$$U_\rho^{\mathrm{sh}} = \{x \in \phi_{uc} \mid W_c^{\mathrm{sh}}(x) \leq \rho\}$$

Thus, $U_{\rho_c} = U_{\rho_c^{\mathrm{sh}}}^{\mathrm{sh}}$. Let $0 < \rho_s < \rho_{\min} < \rho'_{\min} < \rho_c^{\mathrm{sh}}$, where $\rho_{\min}$ is the nominal inner shifted CLBF level used in the one-step CLBF constraint and $\rho'_{\min}$ is the corresponding robust inner level for the actual disturbed closed-loop system. The robust inner level is selected as:

$$\rho'_{\min} = \rho_{\min} + c_4\left(\sqrt{\frac{\rho_{\min}}{c_1}} + \frac{W_{\max}}{L_x}\left(e^{L_x\Delta}-1\right)\right)\frac{W_{\max}}{L_x}\left(e^{L_x\Delta}-1\right) < \rho_c^{\mathrm{sh}} \tag{35}$$

Theorem 1. *Consider the system of Eq. (1) in closed loop under the proposed CLBF-based RL controller, implemented in sample-and-hold fashion with sampling period $\Delta > 0$. Suppose that $x(t_0) \in U_{\rho_c}$. The constraint-informed action map uses the following mode priority: if $x(t_k) \in \mathcal{B}_\delta(x_e)$, the stationary-point mode is used; otherwise, the mode is selected according to whether $W_c^{\mathrm{sh}}(x(t_k)) > \rho_{\min}$ or $W_c^{\mathrm{sh}}(x(t_k)) \leq \rho_{\min}$.*

In the outer mode, i.e., when $x(t_k) \notin \mathcal{B}_\delta(x_e)$ and $W_c^{\mathrm{sh}}(x(t_k)) > \rho_{\min}$, the active CLBF constraint is:

$$\frac{\partial W_c(x(t_k))}{\partial x}F(x(t_k), u_k) \leq \frac{\partial W_c(x(t_k))}{\partial x}F(x(t_k), \Phi(x(t_k))) \tag{36}$$

In the inner mode, i.e., when $x(t_k) \notin \mathcal{B}_\delta(x_e)$ and $W_c^{\mathrm{sh}}(x(t_k)) \leq \rho_{\min}$, the active CLBF constraint is:

$$W_c^{\mathrm{sh}}(\tilde{x}(t \mid t_k)) \leq \rho_{\min}, \qquad t \in [t_k, t_{k+1}] \tag{37}$$

In the stationary-point mode, i.e., when $x(t_k) \in \mathcal{B}_\delta(x_e)$, the active CLBF constraints are:

$$W_c^{\mathrm{sh}}(\tilde{x}(t \mid t_k)) \leq \rho_e, \qquad t \in [t_k, t_{k+1}] \tag{38a}$$

$$W_c^{\mathrm{sh}}(\tilde{x}(t_{k+1} \mid t_k)) \le W_c^{\mathrm{sh}}(x(t_k)) - \eta_e \tag{38b}$$

The level ρ_e satisfies:

$$\rho_e + c_4\left(\sqrt{\frac{\rho_e}{c_1}} + \frac{W_{\max}}{L_x}\left(e^{L_x\Delta} - 1\right)\right)\frac{W_{\max}}{L_x}\left(e^{L_x\Delta} - 1\right) < \rho_c^{\mathrm{sh}} \tag{39}$$

and the stationary-point decrease margin satisfies:

$$\eta_e > c_4\left(\sqrt{\frac{\rho_e}{c_1}} + \frac{W_{\max}}{L_x}\left(e^{L_x\Delta} - 1\right)\right)\frac{W_{\max}}{L_x}\left(e^{L_x\Delta} - 1\right) \tag{40}$$

Assume that $\Phi(x)$ is used as the known admissible input for the outer and inner modes, and that an auxiliary input $u_{\mathrm{aux}}(x) \in U$ is available in $\mathcal{B}_\delta(x_e)$ and satisfies Eq. (38). Suppose that $\rho_{\min}$ is selected to satisfy:

$$\rho_s + c_4\left(\sqrt{\frac{\rho_s}{c_1}} + M\Delta\right)M\Delta < \rho_{\min} \tag{41}$$

Furthermore, suppose that there exists $\epsilon_w > 0$ such that:

$$-\frac{c_3}{c_2}\rho_s + L_x'(M + W_{\max})\Delta + c_4\left(\sqrt{\frac{\rho_c^{\mathrm{sh}}}{c_1}} + (M + W_{\max})\Delta\right)W_{\max} \le -\epsilon_w \tag{42}$$

Finally, assume that the robust inner level set is separated from the stationary-point neighborhood:

$$U^{\mathrm{sh}}_{\rho'_{\min}} \cap \mathcal{B}_\delta(x_e) = \emptyset \tag{43}$$

Then, for every $x(t_0) \in U_{\rho_c}$, the following properties hold:

(i) The constraint-informed CLBF-based RL policy map is feasible at every sampling instant and returns an implemented input $u_k \in U$ satisfying the active CLBF-based constraint.

(ii) The closed-loop state satisfies $x(t) \in U_{\rho_c}$ for all $t \ge t_0$, and there exists a finite time $t_s \ge t_0$ such that $x(t) \in U^{\mathrm{sh}}_{\rho'_{\min}}$ for all $t \ge t_s$.

(iii) The closed-loop state satisfies $x(t) \notin D_H$ and $x(t) \notin D$ for all $t \ge t_0$.

Proof. We first prove feasibility. Fix a sampling instant t_k and suppose that $x(t_k) \in U_{\rho_c}$. The constraint-informed action map accepts a submitted input only when it satisfies the active constraint. If the submitted input does not satisfy the active constraint, the map returns the known admissible input associated with the active mode. Therefore, it is sufficient to verify that the known admissible input satisfies the corresponding active constraint in each mode.

If $x(t_k) \notin \mathcal{B}_\delta(x_e)$ and $W_c^{\mathrm{sh}}(x(t_k)) > \rho_{\min}$, the known admissible input is $u_f(x(t_k)) = \Phi(x(t_k))$. Substituting $u_f = \Phi$ into Eq. (36) gives equality. Therefore, the outer-mode constraint is feasible.

If $x(t_k) \notin \mathcal{B}_\delta(x_e)$ and $W_c^{\mathrm{sh}}(x(t_k)) \le \rho_{\min}$, the known admissible input is again $u_f(x(t_k)) = \Phi(x(t_k))$. Consider the nominal prediction generated by the sample-and-hold input $\Phi(x(t_k))$. Along this nominal prediction:

$$\begin{aligned}\frac{d}{dt}W_c(\tilde{x}(t)) &= \frac{\partial W_c(\tilde{x}(t))}{\partial x}F(\tilde{x}(t), \Phi(x(t_k))) \\ &= \frac{\partial W_c(x(t_k))}{\partial x}F(x(t_k), \Phi(x(t_k))) \\ &\quad + \left[\frac{\partial W_c(\tilde{x}(t))}{\partial x}F(\tilde{x}(t), \Phi(x(t_k))) - \frac{\partial W_c(x(t_k))}{\partial x}F(x(t_k), \Phi(x(t_k)))\right]\end{aligned} \tag{44}$$

Using Eq. (22c), this gives:

$$\frac{d}{dt}W_c(\tilde{x}(t)) \le \frac{\partial W_c(x(t_k))}{\partial x}F(x(t_k), \Phi(x(t_k))) + L_x'\|\tilde{x}(t) - x(t_k)\| \tag{45}$$

On any interval where the nominal prediction remains in $U^{\mathrm{sh}}_{\rho_{\min}}$, it also remains in $U^{\mathrm{sh}}_{\rho_c^{\mathrm{sh}}}$. Hence, $\|F(\tilde{x}(t), \Phi(x(t_k)))\| \le M$, and:

$$\begin{aligned}\|\tilde{x}(t) - x(t_k)\| &= \left\|\int_{t_k}^{t} F(\tilde{x}(\tau), \Phi(x(t_k)))\, d\tau\right\| \\ &\le M\Delta\end{aligned} \tag{46}$$

The robust decrease condition in Eq. (42) implies:

$$\begin{aligned}-\frac{c_3}{c_2}\rho_s + L_x'M\Delta &= \left[-\frac{c_3}{c_2}\rho_s + L_x'(M + W_{\max})\Delta + c_4\left(\sqrt{\frac{\rho_c^{\mathrm{sh}}}{c_1}} + (M + W_{\max})\Delta\right)W_{\max}\right] \\ &\quad - L_x'W_{\max}\Delta - c_4\left(\sqrt{\frac{\rho_c^{\mathrm{sh}}}{c_1}} + (M + W_{\max})\Delta\right)W_{\max} \\ &\le -\epsilon_w \le 0\end{aligned} \tag{47}$$

If $W_c^{\mathrm{sh}}(x(t_k)) \ge \rho_s$, then Eq. (21a) implies $\|x(t_k)\|^2 \ge W_c^{\mathrm{sh}}(x(t_k))/c_2 \ge \rho_s/c_2$. Therefore, Eq. (21b) gives:

$$\frac{\partial W_c(x(t_k))}{\partial x}F(x(t_k), \Phi(x(t_k))) \le -c_3\|x(t_k)\|^2 \le -\frac{c_3}{c_2}\rho_s \tag{48}$$

Combining the previous estimates gives:

$$\frac{d}{dt}W_c(\tilde{x}(t)) \le -\frac{c_3}{c_2}\rho_s + L_x'M\Delta \le 0 \tag{49}$$

Thus, the nominal CLBF level cannot increase before any possible violation of $W_c^{\mathrm{sh}}(\tilde{x}(t)) \le \rho_{\min}$. A first-violation argument then gives $W_c^{\mathrm{sh}}(\tilde{x}(t \mid t_k)) \le \rho_{\min}$ for all $t \in [t_k, t_{k+1}]$.

If $W_c^{\mathrm{sh}}(x(t_k)) < \rho_s$, then Eq. (21a) gives $\|x(t_k)\| < \sqrt{\rho_s/c_1}$. Using the mean-value theorem, Eq. (21c), and $\|\tilde{x}(t) - x(t_k)\| \le M\Delta$, for any possible first-violation time of $W_c^{\mathrm{sh}}(\tilde{x}(t)) \le \rho_{\min}$, one obtains:

$$\begin{aligned}W_c^{\mathrm{sh}}(\tilde{x}(t \mid t_k)) &\le W_c^{\mathrm{sh}}(x(t_k)) + c_4\left(\sqrt{\frac{\rho_s}{c_1}} + M\Delta\right)M\Delta \\ &< \rho_s + c_4\left(\sqrt{\frac{\rho_s}{c_1}} + M\Delta\right)M\Delta \\ &< \rho_{\min}\end{aligned} \tag{50}$$

where the last inequality follows from Eq. (41). Therefore, such a first violation cannot occur, and Eq. (37) is feasible under $u_f = \Phi(x(t_k))$.

If $x(t_k) \in \mathcal{B}_\delta(x_e)$, the known admissible input is $u_f(x(t_k)) = u_{\mathrm{aux}}(x(t_k))$. By assumption, $u_{\mathrm{aux}}(x(t_k)) \in U$ satisfies Eq. (38). Hence, in all modes, the policy map returns an input $u_k \in U$ satisfying the active CLBF-based constraint. This proves feasibility.

We next prove forward invariance of U_{ρ_c}. Fix a sampling instant t_k and suppose that $x(t_k) \in U_{\rho_c}$. We show that $x(t) \in U_{\rho_c}$ for all $t \in [t_k, t_{k+1}]$. Suppose, by contradiction, that the trajectory leaves U_{ρ_c} during this sampling interval. Let $\tau_1 \in (t_k, t_{k+1}]$ be the first time at which the trajectory reaches the boundary of U_{ρ_c} before leaving it. Then:

$$W_c^{\mathrm{sh}}(x(\tau_1)) = \rho_c^{\mathrm{sh}}, \qquad W_c^{\mathrm{sh}}(x(t)) \le \rho_c^{\mathrm{sh}}, \quad t \in [t_k, \tau_1] \tag{51}$$

Therefore, $x(t) \in U^{\mathrm{sh}}_{\rho_c^{\mathrm{sh}}}$ for all $t \in [t_k, \tau_1]$, and the bounds in Eq. (22) can be used on this interval.

Along the actual disturbed system under sample-and-hold implementation:

$$\frac{d}{dt}W_c^{\mathrm{sh}}(x(t)) = \frac{\partial W_c(x(t))}{\partial x}F(x(t), u_k) + \frac{\partial W_c(x(t))}{\partial x}w(x(t), u_k, t) \tag{52}$$

The first term satisfies:

$$\begin{aligned}\frac{\partial W_c(x(t))}{\partial x}F(x(t), u_k) &= \frac{\partial W_c(x(t_k))}{\partial x}F(x(t_k), u_k) \\ &\quad + \left[\frac{\partial W_c(x(t))}{\partial x}F(x(t), u_k) - \frac{\partial W_c(x(t_k))}{\partial x}F(x(t_k), u_k)\right] \\ &\le \frac{\partial W_c(x(t_k))}{\partial x}F(x(t_k), u_k) + L_x'\|x(t) - x(t_k)\|\end{aligned} \tag{53}$$

For the disturbance term, Eq. (21c) and $\|w(x, u, t)\| \le W_{\max}$ give:

$$\frac{\partial W_c(x(t))}{\partial x}w(x(t), u_k, t) \le c_4\|x(t)\|W_{\max} \tag{54}$$

Thus, for $t \in [t_k, \tau_1]$:

$$\frac{d}{dt} W_c^{\mathrm{sh}}(x(t)) \le \frac{\partial W_c(x(t_k))}{\partial x} F(x(t_k), u_k) + L_x' \|x(t) - x(t_k)\| + c_4 \|x(t)\| W_{\max} \tag{55}$$

Also:

$$\begin{aligned} \|x(t) - x(t_k)\| &= \left\| \int_{t_k}^{t} \left(F(x(\tau), u_k) + w(x(\tau), u_k, \tau) \right) d\tau \right\| \\ &\le (M + W_{\max}) \Delta \end{aligned} \tag{56}$$

Since $x(t_k) \in U_{\rho_c} = U^{\mathrm{sh}}_{\rho_c^{\mathrm{sh}}}$, Eq. (21a) gives $\|x(t_k)\| \le \sqrt{\rho_c^{\mathrm{sh}}/c_1}$, and hence:

$$\|x(t)\| \le \sqrt{\frac{\rho_c^{\mathrm{sh}}}{c_1}} + (M + W_{\max}) \Delta \tag{57}$$

Consider first the outer mode, where $x(t_k) \notin \mathcal{B}_\delta(x_e)$ and $W_c^{\mathrm{sh}}(x(t_k)) > \rho_{\min}$. Since $\rho_{\min} > \rho_s$, Eq. (21a) gives $\|x(t_k)\|^2 \ge \rho_s / c_2$. From Eq. (36) and (21b):

$$\begin{aligned} \frac{\partial W_c(x(t_k))}{\partial x} F(x(t_k), u_k) &\le \frac{\partial W_c(x(t_k))}{\partial x} F(x(t_k), \Phi(x(t_k))) \\ &\le -c_3 \|x(t_k)\|^2 \\ &\le -\frac{c_3}{c_2} \rho_s \end{aligned} \tag{58}$$

Substituting this estimate into Eq. (55) yields, for all $t \in [t_k, \tau_1]$:

$$\begin{aligned} \frac{d}{dt} W_c^{\mathrm{sh}}(x(t)) &\le -\frac{c_3}{c_2} \rho_s + L_x'(M + W_{\max}) \Delta + c_4 \left(\sqrt{\frac{\rho_c^{\mathrm{sh}}}{c_1}} + (M + W_{\max}) \Delta \right) W_{\max} \\ &\le -\epsilon_w \end{aligned} \tag{59}$$

where the last inequality follows from Eq. (42). Integrating over $[t_k, \tau_1]$ gives:

$$W_c^{\mathrm{sh}}(x(\tau_1)) \le W_c^{\mathrm{sh}}(x(t_k)) - \epsilon_w (\tau_1 - t_k) < \rho_c^{\mathrm{sh}} \tag{60}$$

This contradicts $W_c^{\mathrm{sh}}(x(\tau_1)) = \rho_c^{\mathrm{sh}}$. Hence, the trajectory cannot leave U_{ρ_c} in the outer mode.

Next consider the inner mode, where $x(t_k) \notin \mathcal{B}_\delta(x_e)$ and $W_c^{\mathrm{sh}}(x(t_k)) \le \rho_{\min}$. Let $\tilde{x}(t \mid t_k)$ be the nominal predicted state generated from $x(t_k)$ under the same held input u_k. The actual and nominal trajectories satisfy:

$$\frac{d}{dt} \left(x(t) - \tilde{x}(t \mid t_k) \right) = F(x(t), u_k) - F(\tilde{x}(t \mid t_k), u_k) + w(x(t), u_k, t) \tag{61}$$

Therefore:

$$\frac{d}{dt} \|x(t) - \tilde{x}(t \mid t_k)\| \le L_x \|x(t) - \tilde{x}(t \mid t_k)\| + W_{\max} \tag{62}$$

Since $x(t_k) = \tilde{x}(t_k \mid t_k)$, Gronwall's inequality gives:

$$\begin{aligned} \|x(t) - \tilde{x}(t \mid t_k)\| &\le \frac{W_{\max}}{L_x} \left(e^{L_x (t - t_k)} - 1 \right) \\ &\le \frac{W_{\max}}{L_x} \left(e^{L_x \Delta} - 1 \right) \end{aligned} \tag{63}$$

The one-step constraint gives $W_c^{\mathrm{sh}}(\tilde{x}(t \mid t_k)) \le \rho_{\min}$ for $t \in [t_k, t_{k+1}]$. Hence, Eq. (21a) gives $\|\tilde{x}(t \mid t_k)\| \le \sqrt{\rho_{\min}/c_1}$. Using the mean-value theorem and Eq. (21c), at $t = \tau_1$ one obtains:

$$\begin{aligned} W_c^{\mathrm{sh}}(x(\tau_1)) &\le W_c^{\mathrm{sh}}(\tilde{x}(\tau_1 \mid t_k)) + c_4 \left(\sqrt{\frac{\rho_{\min}}{c_1}} + \frac{W_{\max}}{L_x} \left(e^{L_x \Delta} - 1 \right) \right) \frac{W_{\max}}{L_x} \left(e^{L_x \Delta} - 1 \right) \\ &\le \rho_{\min}' < \rho_c^{\mathrm{sh}} \end{aligned} \tag{64}$$

This contradicts $W_c^{\mathrm{sh}}(x(\tau_1)) = \rho_c^{\mathrm{sh}}$. Hence, the trajectory cannot leave U_{ρ_c} in the inner mode.

Finally, consider the stationary-point mode, where $x(t_k) \in \mathcal{B}_\delta(x_e)$. The nominal trajectory satisfies $W_c^{\mathrm{sh}}(\tilde{x}(t \mid t_k)) \le \rho_e$ for $t \in [t_k, t_{k+1}]$. By the same actual-to-nominal estimate used in the inner mode:

$$\|x(t) - \tilde{x}(t \mid t_k)\| \le \frac{W_{\max}}{L_x} \left(e^{L_x \Delta} - 1 \right) \tag{65}$$

Using the mean-value theorem and Eq. (21c), at $t = \tau_1$ one obtains:

$$\begin{aligned} W_c^{\mathrm{sh}}(x(\tau_1)) &\le \rho_e + c_4 \left(\sqrt{\frac{\rho_e}{c_1}} + \frac{W_{\max}}{L_x} \left(e^{L_x \Delta} - 1 \right) \right) \frac{W_{\max}}{L_x} \left(e^{L_x \Delta} - 1 \right) \\ &< \rho_c^{\mathrm{sh}} \end{aligned} \tag{66}$$

where the last inequality follows from Eq. (39). This contradicts $W_c^{\mathrm{sh}}(x(\tau_1)) = \rho_c^{\mathrm{sh}}$. Hence, the trajectory cannot leave U_{ρ_c} in the stationary-point mode.

The three cases show that $x(t_k) \in U_{\rho_c}$ implies $x(t) \in U_{\rho_c}$ for all $t \in [t_k, t_{k+1}]$. Since $x(t_0) \in U_{\rho_c}$, induction over k gives $x(t) \in U_{\rho_c}$ for all $t \ge t_0$.

We also record the endpoint decrease obtained in the stationary-point mode. From Eq. (38b), $W_c^{\mathrm{sh}}(\tilde{x}(t_{k+1} \mid t_k)) \le W_c^{\mathrm{sh}}(x(t_k)) - \eta_e$. Using the same actual-to-nominal CLBF bound at t_{k+1}:

$$\begin{aligned} W_c^{\mathrm{sh}}(x(t_{k+1})) &\le W_c^{\mathrm{sh}}(\tilde{x}(t_{k+1} \mid t_k)) + c_4 \left(\sqrt{\frac{\rho_e}{c_1}} + \frac{W_{\max}}{L_x} \left(e^{L_x \Delta} - 1 \right) \right) \frac{W_{\max}}{L_x} \left(e^{L_x \Delta} - 1 \right) \\ &\le W_c^{\mathrm{sh}}(x(t_k)) - \eta_e + c_4 \left(\sqrt{\frac{\rho_e}{c_1}} + \frac{W_{\max}}{L_x} \left(e^{L_x \Delta} - 1 \right) \right) \frac{W_{\max}}{L_x} \left(e^{L_x \Delta} - 1 \right) \\ &< W_c^{\mathrm{sh}}(x(t_k)) \end{aligned} \tag{67}$$

where the last inequality follows from Eq. (40).

We now prove practical stability. Suppose that the trajectory has not yet reached $U^{\mathrm{sh}}_{\rho_{\min}'}$. If $x(t_k) \notin \mathcal{B}_\delta(x_e)$, then $W_c^{\mathrm{sh}}(x(t_k)) > \rho_{\min}' > \rho_{\min}$, and the outer-mode estimate gives $\frac{d}{dt} W_c^{\mathrm{sh}}(x(t)) \le -\epsilon_w$ over the sampling interval as long as the trajectory has not reached $U^{\mathrm{sh}}_{\rho_{\min}'}$. If $x(t_k) \in \mathcal{B}_\delta(x_e)$, then Eq. (67) shows that the actual CLBF level strictly decreases at the next sampling time. Moreover, the decrease in the stationary-point mode is at least:

$$\eta_e - c_4 \left(\sqrt{\frac{\rho_e}{c_1}} + \frac{W_{\max}}{L_x} \left(e^{L_x \Delta} - 1 \right) \right) \frac{W_{\max}}{L_x} \left(e^{L_x \Delta} - 1 \right) > 0 \tag{68}$$

by Eq. (40). Since W_c^{sh} is lower bounded by zero, the trajectory cannot remain outside $U^{\mathrm{sh}}_{\rho_{\min}'}$ forever while undergoing these decreases. Therefore, there exists a finite sampling time $t_s \ge t_0$ such that $x(t_s) \in U^{\mathrm{sh}}_{\rho_{\min}'}$.

It remains to show that $U^{\mathrm{sh}}_{\rho_{\min}'}$ is invariant after it is reached. By Eq. (43), the stationary-point mode is not active inside $U^{\mathrm{sh}}_{\rho_{\min}'}$. If $W_c^{\mathrm{sh}}(x(t_k)) \le \rho_{\min}$, the inner-mode estimate gives:

$$W_c^{\mathrm{sh}}(x(t)) \le \rho_{\min}', \qquad t \in [t_k, t_{k+1}] \tag{69}$$

If $\rho_{\min} < W_c^{\mathrm{sh}}(x(t_k)) \le \rho_{\min}'$, the outer-mode estimate gives $\frac{d}{dt} W_c^{\mathrm{sh}}(x(t)) \le -\epsilon_w$ before any possible exit from $U^{\mathrm{sh}}_{\rho_{\min}'}$. A first-exit argument therefore shows that the trajectory cannot leave $U^{\mathrm{sh}}_{\rho_{\min}'}$ through its boundary. Hence, $x(t) \in U^{\mathrm{sh}}_{\rho_{\min}'}$ for all $t \ge t_s$.

Finally, we prove safety. Since $x(t) \in U_{\rho_c}$ for all $t \ge t_0$, it follows that $W_c(x(t)) \le \rho_c$ for all $t \ge t_0$. The CLBF-enlarged unsafe region is $D_H = \{x \in H \mid W_c(x) > \rho_c\}$, and the CLBF construction gives $D \subset D_H$. Therefore, $x(t) \notin D_H$ and $x(t) \notin D$ for all $t \ge t_0$. This completes the proof. □

Remark 5. Eq. (41) is a conservative sufficient condition for the one-step nominal feasibility requirement in the inner mode. It is obtained from the bounds:

$$\|\tilde{x}(t \mid t_k) - x(t_k)\| \le M \Delta \tag{70a}$$

$$\|x(t_k)\| \le \sqrt{\frac{\rho_s}{c_1}} \tag{70b}$$

Table 1
Parameter values of the CSTR model.

Variable	Value	Variable	Value
T_0	310 K	F	100×10^{-3} m^3 min^{-1}
V_L	0.1 m^3	E	8.314×10^4 kJ kmol^{-1}
k_0	72×10^9 min^{-1}	ΔH	-4.78×10^4 kJ kmol^{-1}
C_p	0.239 kJ kg^{-1} K^{-1}	R	8.314 kJ kmol^{-1} K^{-1}
ρ_L	1,000 kg m^{-3}	$C_{A0,s}$	1.0 kmol m^{-3}
$\dot{Q}_s$	0.0 kJ min^{-1}	$C_{A,s}$	0.57 kmol m^{-3}
T_s	395.3 K	$T_{0,s}$	310 K

when $W_c^{\mathrm{sh}}(x(t_k)) \le \rho_s$. In implementation, a less conservative choice of $\rho_{\min}$ can be obtained by simulating the nominal closed-loop system Eq. (1) with $w = 0$ from initial states $x_k \in U_{\rho_s}^{\mathrm{sh}}$ over one sampling period $t \in [0, \Delta]$, and estimating the largest value of $W_c^{\mathrm{sh}}(\tilde{x}(t))$. Any $\rho_{\min}$ larger than this offline one-step nominal maximum, with a suitable numerical margin, and satisfying Eq. (35), can be used.

Remark 6. The sampling period Δ affects the conservativeness of the sufficient conditions in Theorem 1. Specifically, the sample-and-hold and nominal-to-actual mismatch terms in Eq. (35) and (39)–(42) decrease as Δ decreases. Therefore, a smaller feasible sampling period makes the robust inner level $\rho'_{\min}$ closer to the nominal shifted level $\rho_{\min}$, relaxes the required stationary-point decrease margin, and helps satisfy the robust CLBF decrease condition. This does not remove the effect of bounded disturbances, but it reduces the part of the bounds caused by holding the input constant over one sampling interval. Since the proposed CLBF-RL controller only requires neural-network evaluation and constraint checking online, it can more readily support a smaller sampling period than optimization-based CLBF-MPC, which is beneficial for both computation and the theoretical bounds above.

4. Application to a chemical process example

4.1. Process description

The model chemical process considered in this study is a well-mixed, non-isothermal continuous stirred-tank reactor (CSTR), where an irreversible first-order exothermic reaction takes place. The reaction converts reactant A to product B through $A \to B$. An external heat input is used to supply or remove heat from the reactor. The dynamic model of the CSTR, derived from material and energy balances, is given by:

$$\frac{dC_A}{dt} = \frac{F}{V_L}\left(C_{A0} - C_A\right) - k_0 \exp\left[-\frac{E}{RT}\right] C_A \tag{71a}$$

$$\frac{dT}{dt} = \frac{F}{V_L}\left(T_0 - T\right) - \frac{\Delta H k_0}{\rho_L C_p} \exp\left[-\frac{E}{RT}\right] C_A + \frac{\dot{Q}}{\rho_L C_p V_L} \tag{71b}$$

where C_A is the concentration of reactant A in the reactor, T is the reactor temperature, and $\dot{Q}$ denotes the heat supply/removal rate. The feed to the reactor contains reactant A with concentration C_{A0}, temperature T_0, and volumetric flow rate F. The liquid volume is denoted by V_L, and the liquid density and heat capacity are denoted by ρ_L and C_p, respectively. The parameters k_0, E, and ΔH denote the pre-exponential factor, activation energy, and enthalpy of reaction, respectively. The parameter values used in the simulations are listed in Table 1.

4.2. Control problem

The control objective is to operate the CSTR around the nominal steady state $C_{A,s} = 0.57\,\mathrm{kmol\,m^{-3}}$ and $T_s = 395.3\,\mathrm{K}$, while driving the state toward this steady state and avoiding the unsafe region. Using deviation variables, the state and input vectors are defined as $x^\top = \left[C_A - C_{A,s},\ T - T_s\right]$ and $u^\top = \left[\Delta C_{A0},\ \Delta\dot{Q}\right]$, where $\Delta C_{A0} = C_{A0} - C_{A0,s}$ and $\Delta\dot{Q} = \dot{Q} - \dot{Q}_s$. The input constraints are $|\Delta C_{A0}| \le 1\,\mathrm{kmol\,m^{-3}}$ and $|\Delta\dot{Q}| \le 0.0167\,\mathrm{kJ\,min^{-1}}$.

A quadratic control Lyapunov function is chosen as $V(x) = x^\top P x$, where:

$$P = \begin{bmatrix} 9.35 & 0.41 \\ 0.41 & 0.02 \end{bmatrix} \tag{72}$$

The unsafe region is defined as:

$$D := \left\{ x \in \mathbb{R}^2 \;\middle|\; F_D(x) = \frac{(x_1 + 0.22)^2}{1} + \frac{(x_2 - 4.6)^2}{1 \times 10^4} < 2 \times 10^{-4} \right\} \tag{73}$$

and the auxiliary set is defined as $H := \left\{ x \in \mathbb{R}^2 \;\middle|\; F_D(x) < 4 \times 10^{-4} \right\}$. The corresponding closure is $\overline{H} = \left\{ x \in \mathbb{R}^2 \;\middle|\; F_D(x) \le 4 \times 10^{-4} \right\}$, which is compact and satisfies $\overline{D} \subset H$. The barrier function is selected as:

$$B(x) = \begin{cases} \exp\left(\frac{a F_D^2(x)}{F_D(x) - 4\times 10^{-4}}\right) - \exp\left(-2a \times 10^{-4}\right), & x \in H \\ -\exp\left(-2a \times 10^{-4}\right), & x \notin H \end{cases} \tag{74}$$

where $a = 0.001$. The CLBF is constructed as $W_c(x) = V(x) + \mu B(x) + \nu$, with $\rho_c = 0$, $c_{V,\ell} = 0.001$, $c_{V,u} = 10$, $c_H = 34.8$, $c_D = 16.85$, $\nu = -1.685 \times 10^{-2}$, and $\mu = 5{,}000$. For this CLBF construction, the saddle point is calculated as $x_e = (-0.235,\ 4.83)$.

The reference controller $\Phi(x)$ used in the CLBF constraints is chosen as a saturated Sontag-type feedback law. Define $a_W(x) = L_f W_c(x)$ and $b_W(x) = L_g W_c(x)$, where $b_W(x) \in \mathbb{R}^{1\times m}$. The unconstrained Sontag-type input is defined as:

$$\Phi_0(x) = \begin{cases} -\frac{a_W(x) + \sqrt{a_W^2(x) + \gamma \|b_W(x)\|^4}}{\|b_W(x)\|^2} b_W^\top(x), & \|b_W(x)\| > 0 \\ 0, & \|b_W(x)\| = 0 \end{cases} \tag{75}$$

where $\gamma > 0$ is a design parameter. The reference controller applied in the simulations is obtained by component-wise saturation:

$$[\Phi(x)]_i = \min\left\{u_{i,\max}, \max\left\{u_{i,\min}, [\Phi_0(x)]_i\right\}\right\}, \qquad i = 1, \ldots, m \tag{76}$$

For the CSTR example, $m = 2$, $\gamma = 1$, and the saturation limits are the input bounds $|\Delta C_{A0}| \le 1\,\mathrm{kmol\,m^{-3}}$ and $|\Delta\dot{Q}| \le 0.0167\,\mathrm{kJ\,min^{-1}}$. This saturated Sontag-type controller is used as the reference controller in the CLBF decrease condition and as the fallback input in the corresponding CLBF-RL mode.

In the implementation, the inner shifted CLBF threshold is set to $\rho_{\min} = 1 \times 10^{-6}$, i.e., $W_c^{\mathrm{sh}}(x) \le 1 \times 10^{-6}$. Equivalently, if the same condition is written in terms of the unshifted CLBF, the corresponding level is $\rho_{\min}^W = W_c(0) + \rho_{\min} = W_c(0) + 1\times 10^{-6}$, where $W_c(0) = -\mu \exp(-2a \times 10^{-4}) + \nu$. Numerical tests showed that shifted CLBF thresholds larger than 1×10^{-10} were admissible for the present implementation; therefore, $\rho_{\min} = 1 \times 10^{-6}$ is used in the simulations. Since the corresponding local region lies outside H, the barrier term is constant in this region and $W_c^{\mathrm{sh}}(x) = V(x)$ holds there.

The resulting shifted CLBF surface is shown in Fig. 2. The local-scale view in Fig. 2(a) shows the behavior of $W_c^{\mathrm{sh}}(x)$ near the nominal steady state, while the full-scale view in Fig. 2(b) highlights the high CLBF values near D.

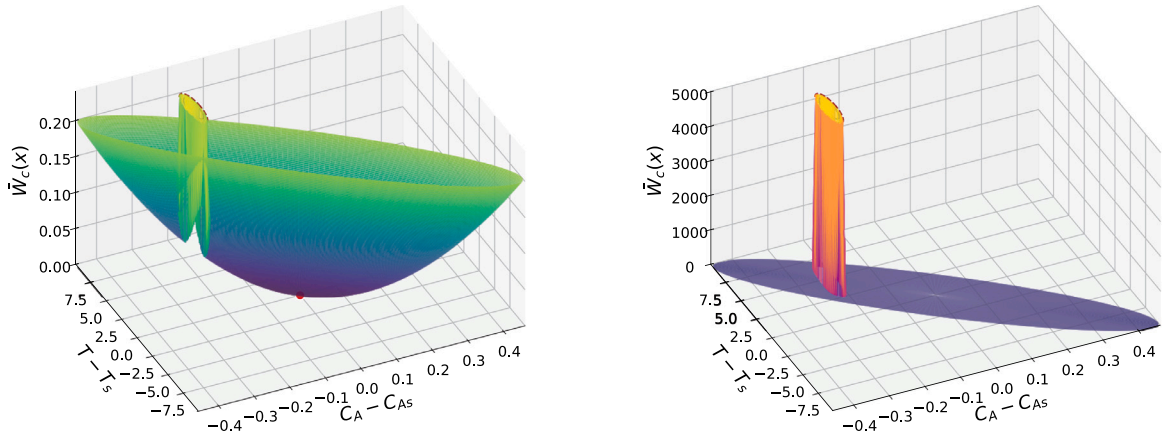
The immediate cost used for controller design and RL training is chosen as:

$$L\left(x(t_k), u(t_k)\right) = x(t_k)^\top Q_L x(t_k) + u(t_k)^\top R_L u(t_k) \tag{77}$$

where:

$$Q_L = \begin{bmatrix} 1{,}000 & 0 \\ 0 & 10 \end{bmatrix}, \qquad R_L = \begin{bmatrix} 1 & 0 \\ 0 & 100 \end{bmatrix} \tag{78}$$

The same stage cost is used as the negative reward in the RL formulation. In the closed-loop simulations, the process model is integrated using the explicit Euler method with integration time step $h_c = 1 \times 10^{-5}$ min, and the sampling period is $\Delta = 2 \times 10^{-3}$ min.



(a) Local-scale view of $W_c^{\mathrm{sh}}(x)$. (b) Full-scale view of $W_c^{\mathrm{sh}}(x)$.

Fig. 2. Visualization of the shifted CLBF $W_c^{\mathrm{sh}}(x)$ over the CSTR state space.

Table 2
Fixed hyperparameters used in CLBF-RL training.

Hyperparameter	Value	Hyperparameter	Value
Total environment steps	300,000	Warm-up steps	10,000
Evaluation interval steps	3,000	Mini-batch size	128
Replay buffer capacity	1,000,000	Discount factor	0.99
Target update coefficient	0.005	Policy update delay	2
Actor learning rate	1×10^{-3}	Critic learning rate	1×10^{-3}
Hidden units per layer	256	Target policy noise	0.08
Exploration noise	0.15 to 0.02	Noise clipping bound	0.20

4.3. CLBF-RL

4.3.1. CLBF-RL training

The numerical CLBF-RL implementation used the TD3 update described in Section 3.3, with the CSTR model, sampling period, and stage cost specified in Section 4.2. Training initial states were sampled uniformly from the Lyapunov ellipse $V(x) \leq 0.2$ and then accepted only if $V(x) > 2 \times 10^{-6}$, $F_D(x) > 2 \times 10^{-4}$, and $W_c(x) \leq \rho_c + 1 \times 10^{-9}$. Each training episode lasted 3.0 min, corresponding to 1,500 sampling periods with $\Delta = 2 \times 10^{-3}$ min. The standard TD3 hyperparameters were kept fixed and are summarized in Table 2. The two design parameters introduced by the proposed constraint-informed policy structure, namely the correction-penalty weight λ_c in Eq. (32) and the number of gated refinement steps N_r in Eq. (25), were studied separately.

We first examined the effect of λ_c with $N_r = 1$. Four values were tested: $\lambda_c \in \{0, 0.01, 0.1, 1\}$. The corresponding actor and critic losses are shown in Fig. 3. The actor loss curves exhibit broadly similar trends for all tested values of λ_c, while the critic loss for $\lambda_c = 1$ is visibly larger than those for the other three cases; however, the loss curves alone do not determine whether the learned policy gives better closed-loop performance. In particular, a larger critic loss may indicate that the critic target is harder to fit, but it is not by itself sufficient to conclude whether the resulting controller is better or worse. Therefore, the trained policies were further compared using the average cumulative closed-loop cost.

The average cumulative closed-loop cost was computed from periodic evaluations during training. Specifically, the policy was evaluated every 3,000 environment steps on a fixed evaluation test set. At an evaluation checkpoint s, the current policy is denoted by π_s. For each initial condition x_0 in the test set, a deterministic closed-loop trajectory was simulated under π_s without exploration noise, and the cumulative cost $J_{\pi_s}(x_0)$ was computed as the sum of the stage costs over the simulation horizon. Thus, for a fixed policy π_s and a fixed initial condition x_0, the cumulative cost $J_{\pi_s}(x_0)$ is determined by the resulting closed-loop trajectory. The reported average cumulative cost at checkpoint s is the arithmetic mean of these cumulative trajectory costs over the fixed evaluation test set. Let $\mathcal{D}_{\mathrm{eval}}$ denote the evaluation test set. Then the average cumulative cost is defined as

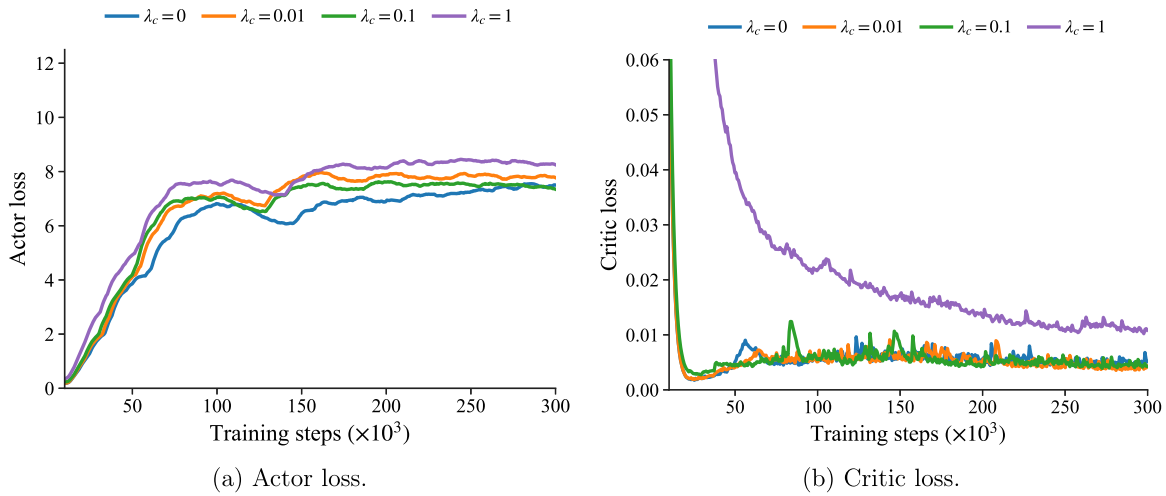
$$J_{\mathrm{avg}}(\pi_s) = \frac{1}{|\mathcal{D}_{\mathrm{eval}}|} \sum_{x_0 \in \mathcal{D}_{\mathrm{eval}}} J_{\pi_s}(x_0). \tag{79}$$

In the following comparisons, the best cost is the minimum value of $J_{\mathrm{avg}}(\pi_s)$ over all evaluation checkpoints, whereas the final cost is the value of $J_{\mathrm{avg}}(\pi_s)$ at the last evaluation checkpoint.

The performance comparison is shown in Fig. 4. The best average cumulative costs for $\lambda_c = 0$, $\lambda_c = 0.01$, $\lambda_c = 0.1$, and $\lambda_c = 1$ are 63.89, 65.26, 63.38, and 66.29, respectively. The corresponding final costs are 71.84, 71.84, 67.62, and 69.87. Among the tested values, $\lambda_c = 0.1$ gives both the lowest best cost and the lowest final cost. Therefore, $\lambda_c = 0.1$ was selected for the refinement-step study and the subsequent closed-loop tests.

We then examined the effect of the number of gated refinement steps with $\lambda_c = 0.1$. Three values were tested: $N_r \in \{1, 3, 5\}$. As shown in Fig. 5, increasing N_r does not lead to a clear improvement in either actor or critic loss. The resulting closed-loop cost comparison is shown in Fig. 6. The best average cumulative costs for $N_r = 1$, $N_r = 3$, and $N_r = 5$ are 63.38, 63.99, and 64.15, respectively, while the corresponding final costs are 67.62, 70.34, and 71.84. Thus, using more refinement steps does not improve the closed-loop cost for this example.

The computational effect of N_r is reported in Fig. 7. The one-time controller evaluation time increases from 0.75 ms for $N_r = 1$ to 1.72 ms for $N_r = 3$, and to 2.73 ms for $N_r = 5$. The total training time also increases from 682 min to 1,124 min and 1,216 min, respectively. Therefore, larger N_r increases both online evaluation time and offline training time without improving the closed-loop cost. Based on these results, the final CLBF-RL controller used in the following closed-loop tests was trained with $\lambda_c = 0.1$ and $N_r = 1$.



(a) Actor loss. (b) Critic loss.

Fig. 3. Training losses of CLBF-RL with different correction-penalty weights and $N_r = 1$. The curves are smoothed for visualization.

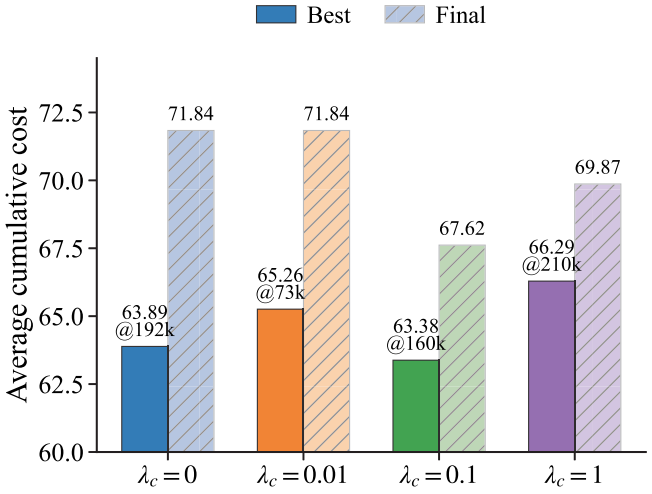


Fig. 4. Best and final average cumulative costs of CLBF-RL with different correction-penalty weights and $N_r = 1$. The average cumulative cost is computed from periodic evaluations every 3,000 environment steps by averaging the cumulative closed-loop costs over the fixed evaluation test set. The label above each best-cost bar gives the training step at which the best policy was obtained.

Remark 7. RL training can be affected by hyperparameters, random seeds, reward design, and exploration noise. In this work, the standard TD3 hyperparameters in Table 2 were selected from commonly used ranges and checked in preliminary runs. Moderate changes in standard settings, such as the learning rate, mini-batch size, and total number of training steps, did not change the main qualitative closed-loop conclusions for the CSTR case, so these parameters were fixed in the reported study. The reported sensitivity analysis therefore focuses on the two hyperparameters introduced by the proposed CLBF-RL structure, namely the correction-penalty weight λ_c and the number of gated refinement steps N_r. A full grid search over all TD3 hyperparameters is outside the scope of this work.

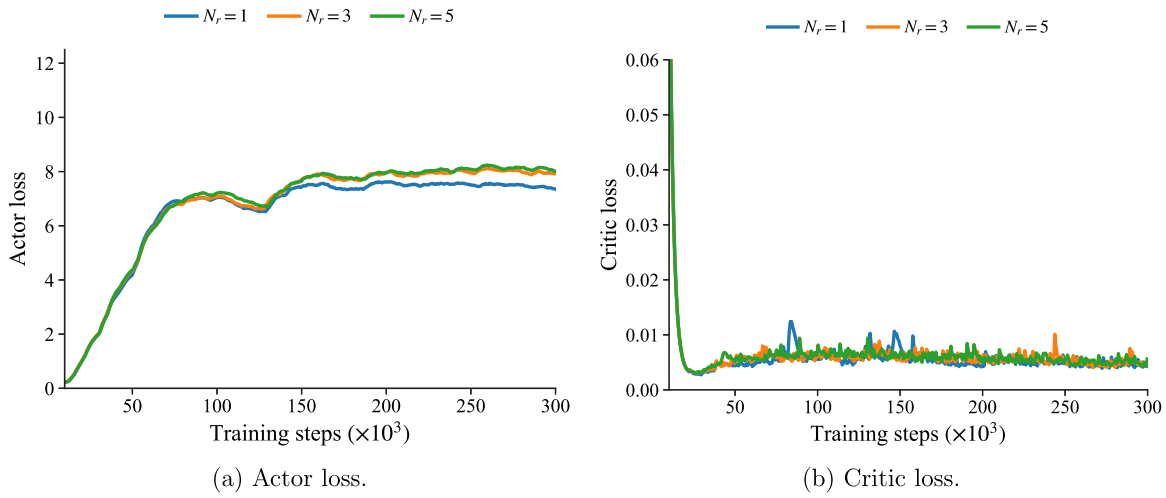
4.3.2. CLBF-RL closed-loop performance

The CLBF-RL controller selected from the training study, with $\lambda_c = 0.1$ and $N_r = 1$, was first evaluated on the nominal CSTR model. The test set contained 1,000 initial states. To generate these initial states, points were sampled uniformly from the ellipse $V(x) = x^\top P x \le 0.2$ using rejection sampling. The candidate point was accepted only if $V(x) > 2\times10^{-6}$, $F_D(x) > 2\times10^{-4}$, and $W_c(x) \le \rho_c + 1\times10^{-9}$. Therefore, the test initial states were inside the sampled Lyapunov ellipse, outside the small target region, outside the unsafe region, and inside the admissible CLBF level set. No additional oversampling near the unsafe-region boundary, the auxiliary set boundary, or the stationary point x_e was used.

The nominal closed-loop state trajectories are shown in Fig. 8. Both state components are driven toward the nominal steady state over the simulation horizon. The concentration deviation $C_A - C_{A,s}$ approaches zero from both positive and negative initial values, and the temperature deviation $T - T_s$ also converges to a small neighborhood of zero. The corresponding state-space trajectories are shown in Fig. 9. The gray points denote the initial states, the dashed black curve denotes $V(x) = 0.2$, the red dashed curve denotes H, the red solid curve denotes D, and the black star denotes the setpoint. The trajectories move toward the setpoint while avoiding the unsafe region, which shows that the trained CLBF-RL controller achieves the desired stabilizing and safety behavior on the nominal CSTR model. We also recorded the activation frequency of the final outer constraint check during deterministic testing. Across the 1,000 closed-loop test trajectories and all evaluated sampling instants, the internally gated actor output passed the final constraint check in 99.55% of the cases. Thus, the fallback input was used in only 0.45% of the evaluated sampling instants. This shows that the closed-loop performance of CLBF-RL is mainly generated by the learned policy, while the fallback controller is used only rarely as a final safety-verification mechanism.

The same trained CLBF-RL controller was then evaluated under process disturbances using the same 1,000 initial states. The disturbance was non-Gaussian and was sampled from an empirical density-point distribution extracted from industrial data (Luo, 2023). The disturbance bounds were $|w_1| \le 0.7709695313$ and $|w_2| \le 2.443973414$, giving $W_{\max} = 2.5627$. Over the 1,000 disturbed trajectories, the realized ranges were $w_1 \in [-0.770940740, 0.770964849]$ and $w_2 \in [-2.443815261, 2.443878735]$, with empirical variances 0.0757053 and 0.761233, respectively. The maximum realized disturbance norm was 2.5601, equal to 99.90% of $W_{\max}$. During this test, the disturbance was applied only to the real process simulation, while the CLBF gate, CLBF constraint checks, reference controller, and backup logic continued to use the nominal CSTR model. Therefore, this test evaluates whether the trained controller can maintain closed-loop performance when the plant response differs from the nominal model used by the controller.

The disturbed closed-loop state trajectories are shown in Fig. 10. Compared with the nominal case, the disturbance produces a slightly wider spread of transient trajectories and small deviations near the steady state. Nevertheless, both state components remain bounded and



(a) Actor loss. (b) Critic loss.

Fig. 5. Training losses of CLBF-RL with different numbers of gated refinement steps and $\lambda_c = 0.1$. The curves are smoothed for visualization.

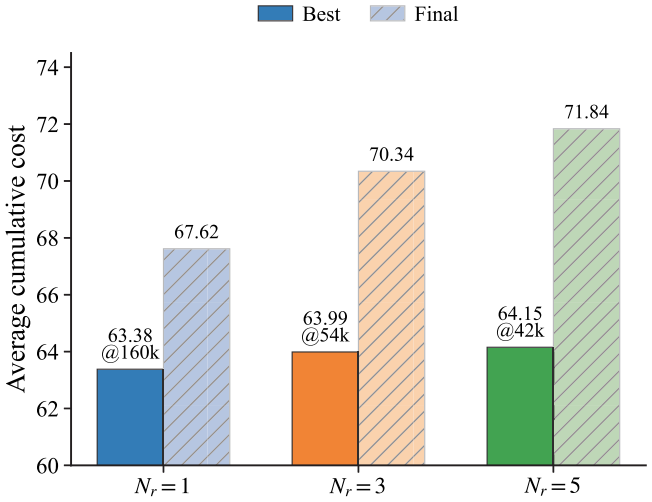


Fig. 6. Best and final average cumulative costs of CLBF-RL with different numbers of gated refinement steps and $\lambda_c = 0.1$. The average cumulative cost is computed from periodic evaluations every 3,000 environment steps by averaging the cumulative closed-loop costs over the fixed evaluation test set. The label above each best-cost bar gives the training step at which the best policy was obtained.

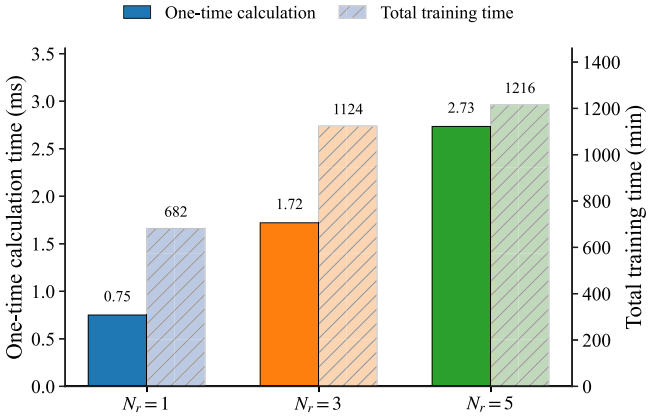


Fig. 7. One-time controller evaluation time and total training time of CLBF-RL with different numbers of gated refinement steps and $\lambda_c = 0.1$.

are driven toward the setpoint region. The state-space trajectories in Fig. 11 show that the closed-loop trajectories still avoid the unsafe region D under the tested disturbances. These results indicate that the CLBF-RL controller maintains the desired safety behavior and remains robust to the tested non-Gaussian process disturbance. Across the 1,116,403 evaluated sampling instants, the minimum observed unsafe-region function value was $F_D(x) = 2.68519 \times 10^{-4}$, corresponding to a minimum margin of 6.85192×10^{-5} above the unsafe boundary $F_D(x) = 2 \times 10^{-4}$. The implemented control actions had zero CLBF constraint-margin violations. The final safety check rejected 3,593 candidate actor actions (0.3218%) and replaced them with the fallback input.

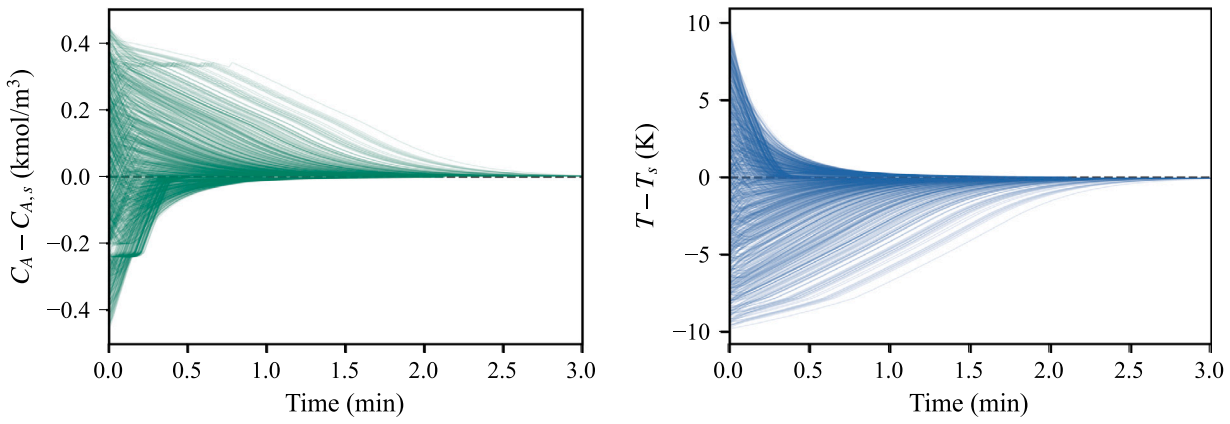
4.4. Benchmark comparison with different controllers

4.4.1. CLBF-MPC design

The CLBF-MPC controller is used as the optimization-based benchmark for evaluating the proposed CLBF-RL controller. It is implemented by applying the general CLBF-MPC formulation in Eq. (6) to the CSTR system described in Section 4.2, following the design of Wu et al. (2019). The controller uses the nominal deviation-variable CSTR model, the input constraints, the CLBF $W_c(x)$, and the stage cost $L(x, u)$ defined above. The prediction horizon is selected as $P_N = 10$.

At each sampling time, the measured state initializes the nominal prediction, and the CLBF-MPC problem is solved in a receding-horizon manner. The active CLBF constraint is selected according to the same three modes described in Eq. (6): the CLBF decrease condition is used when the state is outside the inner stabilizing region and away from the stationary point, the inner-level invariance condition is used after the state enters the inner region, and the stationary-point escape condition is used when the state is near x_e. In the numerical implementation, the inner region is represented by the small Lyapunov level set used in the simulations, and the strict decrease requirement near x_e is enforced using a small numerical margin.

The nonlinear programming problem is implemented in CasADi and solved using IPOPT. After the optimization is solved, only the first input of the optimal input sequence is applied to the process over the next sampling period. If the solver does not return a feasible solution, the implementation applies $\Phi(x)$ as the backup input in the outer and inner modes. Near the stationary point, a bounded one-step search is used to compute an auxiliary input that moves the nominal state to a lower CLBF level before falling back to $\Phi(x)$ if needed. This CLBF-MPC implementation serves as the benchmark controller in the subsequent closed-loop comparisons.



(a) $C_A - C_{A,s}$. (b) $T - T_s$.

Fig. 8. Nominal closed-loop state trajectories under the CLBF-RL controller from different initial states.

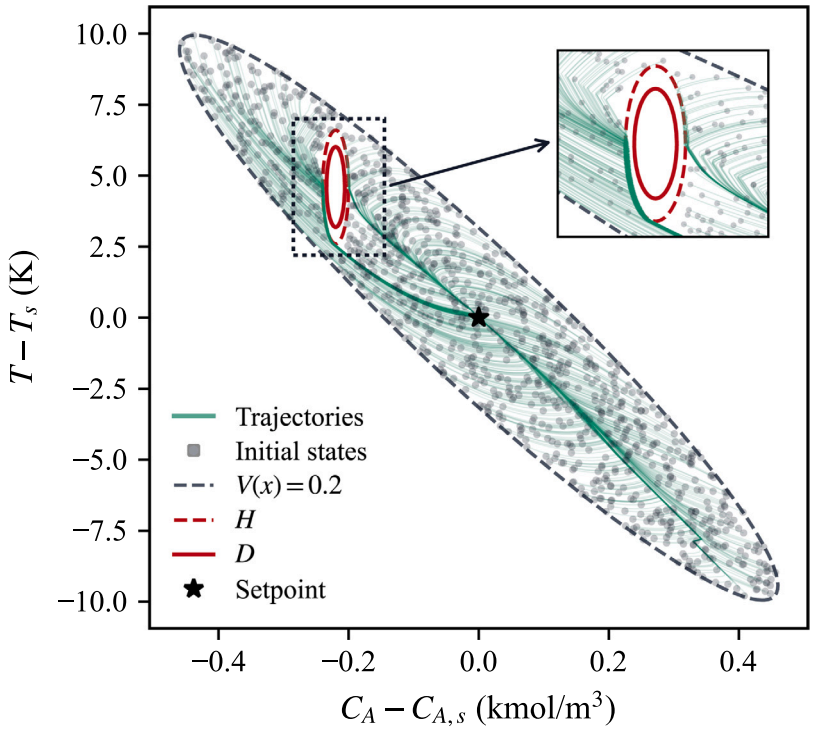


Fig. 9. Nominal closed-loop state-space trajectories under the CLBF-RL controller.

4.4.2. Imitation learning from CLBF-MPC

Another way to reduce the online computational cost of CLBF-MPC is to replace the online optimizer with a neural network trained from CLBF-MPC data (W. Wang et al., 2024). To study this idea, an imitation learning controller was developed by training a feedforward neural network (FNN) to reproduce the input actions generated by the CLBF-MPC controller. This FNN controller is different from the proposed CLBF-RL controller because it does not include the CLBF constraints in its policy structure. Instead, it learns the state-to-input map produced by CLBF-MPC and is used as a fast neural-network baseline.

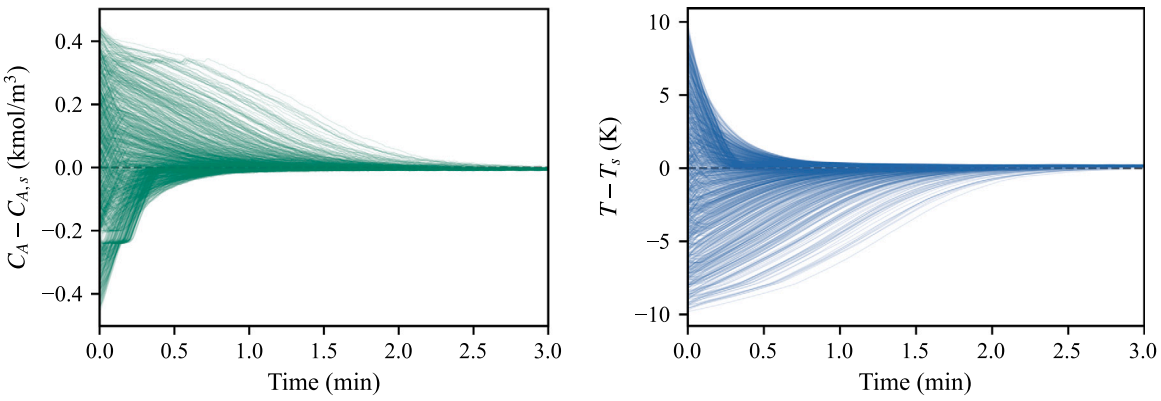
Expert closed-loop trajectories were generated by applying the CLBF-MPC controller described in Section 4.4.1 to the nominal CSTR model. Initial states were sampled uniformly from the Lyapunov ellipse $V(x) \le 0.2$. A sampled initial state was accepted only if it was outside the unsafe region, outside a small neighborhood of the origin, and inside the CLBF level set used for safe operation. Specifically, the accepted initial states satisfied $V(x) > 2 \times 10^{-6}$, $F_D(x) > 2 \times 10^{-4}$, and $W_c(x) \le \rho_c + 10^{-9}$. No process disturbance was added during expert-data generation. Each expert rollout was simulated for up to 3.0 min, corresponding to 1,500 sampling periods, and was stopped early if the state entered the small Lyapunov level set $V(x) < 2 \times 10^{-6}$.

At each sampling instant, the expert dataset stores the current state, the first input computed by CLBF-MPC, the next state, the stage cost, the reward, the active CLBF-MPC mode, the solver status, and the controller computation time. For imitation learning, only the state–action pairs are used. The supervised learning dataset is written as:

$$\mathcal{D}_{\mathrm{IL}} = \left\{ \left(x_k^{(i)}, u_{\mathrm{MPC},k}^{(i)} \right) \right\}_{i=1}^{N_{\mathrm{IL}}} \tag{80}$$

where $x_k^{(i)} = [C_A - C_{A,s},\ T - T_s]^\top$ and $u_{\mathrm{MPC},k}^{(i)} = [\Delta C_{A0},\ \Delta \dot{Q}]^\top$. The state and input data are normalized before training as:

$$x_{\mathrm{norm}} = \begin{bmatrix} x_1/0.5 \\ x_2/10 \end{bmatrix}, \qquad u_{\mathrm{norm}} = \begin{bmatrix} u_1/1 \\ u_2/0.0167 \end{bmatrix} \tag{81}$$



(a) $C_A - C_{A,s}$. (b) $T - T_s$.

Fig. 10. Disturbed closed-loop state trajectories under the CLBF-RL controller from different initial states.

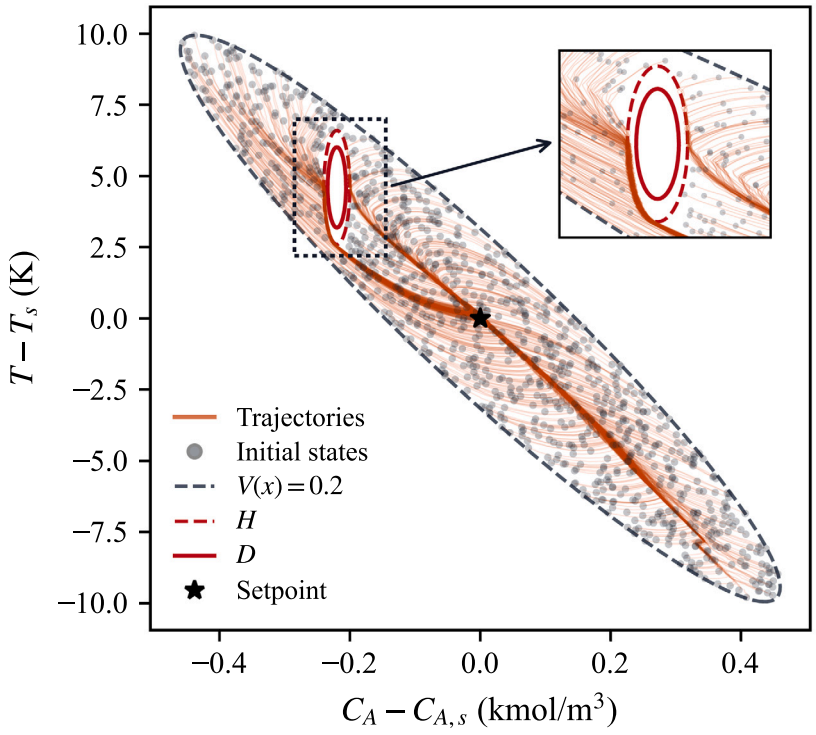


Fig. 11. Disturbed closed-loop state-space trajectories under the CLBF-RL controller.

so that the two state components and two input components have comparable numerical scales. Only samples for which the CLBF-MPC optimization returned a feasible MPC input are used as labels in the pure imitation learning dataset. Samples generated by backup actions are not used for training the FNN.

The imitation model is a bounded FNN actor. Given the normalized state x_{norm}, the network predicts the normalized input through a final tanh activation:

$$\hat{u}_{\text{norm}} = \tanh\big(f_\theta(x_{\text{norm}})\big) \tag{82}$$

The physical input is obtained by rescaling $\hat{u}_{\text{norm}}$ using the same input scales in Eq. (81). Because of the final tanh layer, the FNN output remains within the normalized input bounds.

The FNN is trained by minimizing the mean-squared error between the normalized CLBF-MPC labels and the normalized FNN predictions:

$$J_{\text{BC}}(\theta) = \frac{1}{N_b}\sum_{i=1}^{N_b}\left\|\pi_\theta(x^{(i)}_{\text{norm}}) - u^{(i)}_{\text{MPC,norm}}\right\|^2 \tag{83}$$

where N_b is the mini-batch size. The data are split at the trajectory level into training, validation, and test sets. This avoids placing samples from the same rollout in different splits. The validation set is used to select the network hyperparameters, and the test set is used only for the final evaluation.

The hyperparameters are selected using Bayesian optimization. The search includes the number of hidden layers, the hidden width, the activation function, the mini-batch size, the learning rate, the weight decay, and the dropout rate. In each trial, the FNN is trained using AdamW, and early stopping is applied based on the validation mean-squared error. The corresponding hyperparameters are summarized in Table 3.

The selected FNN is evaluated on the test trajectories. The resulting prediction error distributions for the two manipulated inputs are shown in Fig. 12. This FNN controller is then used as an additional baseline in the closed-loop comparison.

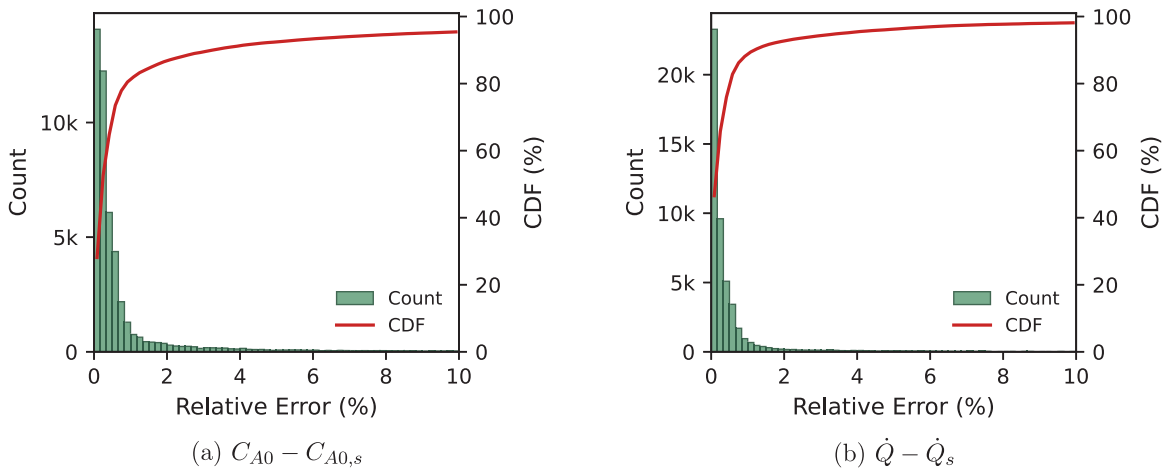


Fig. 12. Prediction error distributions and cumulative distributions for the FNN approximation of the CLBF-MPC deviation-input actions $C_{A0} - C_{A0,s}$ and $\dot{Q} - \dot{Q}_s$.

Table 3
Selected hyperparameters for the FNN imitation controller.

Hyperparameter	Value	Hyperparameter	Value
Hidden layers	3	Hidden width	256
Activation function	ReLU	Mini-batch size	1,024
Learning rate	3.08×10^{-4}	Weight decay	4.04×10^{-8}
Dropout rate	0.0412	Optimizer	AdamW

4.4.3. Comparison results

The closed-loop performance of CLBF-MPC, FNN from imitation learning (IL-NN), and CLBF-RL was compared using the same set of 1,000 initial conditions. Fig. 13 summarizes the cumulative cost and the number of steps required to reach the target set ($\mathcal{X}_{\text{tar}} = \{x \in \mathbb{R}^2 \mid V(x) \le 2 \times 10^{-6}\}$) as functions of the initial Lyapunov value. In general, both metrics increase as the initial Lyapunov value becomes larger, because trajectories starting farther from the equilibrium require more control effort and more time to reach the target set.

As shown in Fig. 13(a), CLBF-RL gives the lowest cumulative cost over most initial-condition bins. Over all test trajectories, the median cumulative cost is 31.55 for CLBF-RL, compared with 39.10 for CLBF-MPC and 38.74 for IL-NN. This corresponds to cost reductions of approximately 19.3% relative to CLBF-MPC and 18.6% relative to IL-NN. The CLBF-MPC and IL-NN curves are close to each other, which is expected because IL-NN is trained to imitate the CLBF-MPC input map. In contrast, CLBF-RL is trained directly to reduce the cumulative closed-loop cost while using the CLBF-based gate to enforce admissible inputs.

A local peak in the cumulative cost appears for initial Lyapunov values around 0.10–0.15. This peak is consistent with the geometry of the safety constraint. In this range, a subset of the trajectories passes close to the auxiliary set H and the unsafe region D. The controller must then avoid the unsafe region while still driving the state toward the setpoint, which can lead to longer paths and more conservative inputs. Therefore, the local increase in cost does not contradict the overall trend with respect to the initial Lyapunov value; it reflects that states with similar Lyapunov values can have different difficulty depending on their location relative to the unsafe region.

Fig. 13(b) shows the number of sampling steps needed to reach the target set. The same qualitative behavior is observed: the required number of steps generally increases with the initial Lyapunov value, and a local increase also appears near the same Lyapunov range as in the cost plot. CLBF-RL reaches the target set faster than the other two controllers for most bins. The median number of steps is 891 for CLBF-RL, while CLBF-MPC and IL-NN require 1,079.5 and 1,036.5 steps, respectively. Thus, CLBF-RL reduces the median convergence time by approximately 17.5% relative to CLBF-MPC and 14.0% relative to IL-NN. The similar behavior of CLBF-MPC and IL-NN again indicates that the imitation-learning controller reproduces the main closed-loop behavior of the MPC expert, while CLBF-RL achieves better performance through direct reinforcement learning of the constrained policy.

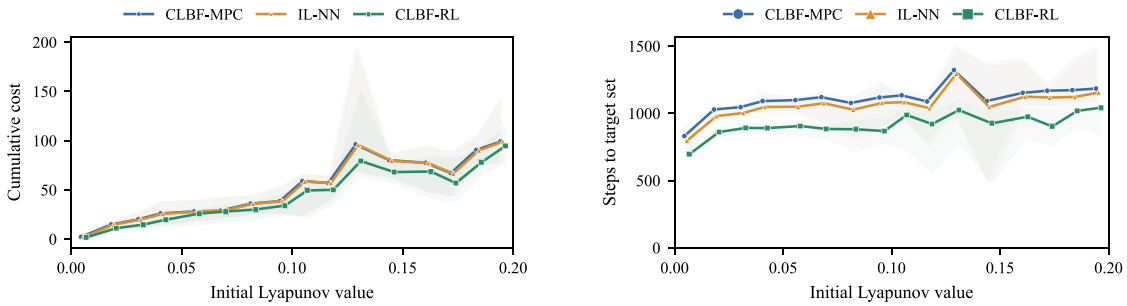
The one-step controller computation times are compared in Fig. 14. CLBF-MPC has the largest computation time because it solves a nonlinear optimization problem at each sampling instant. Its mean one-step computation time is 49.20 ms, and its 99th percentile is 93.12 ms. In contrast, IL-NN and CLBF-RL have much lower computation times, with mean values of 1.23 ms and 1.07 ms, respectively. The similar computation times of IL-NN and CLBF-RL are expected because both controllers are implemented mainly through neural-network evaluation, rather than online nonlinear programming. The CLBF-RL controller also includes the CLBF gate and constraint checks, but these operations are much less expensive than solving the CLBF-MPC optimization problem.

The vertical dashed line in Fig. 14 denotes the sampling time of 120 ms. Most controller evaluations are completed within one sampling interval. The fractions of one-step computations exceeding 120 ms are 0.0701%, 0.0009%, and 0.0005% for CLBF-MPC, IL-NN, and CLBF-RL, respectively. Therefore, CLBF-RL achieves the best closed-loop performance among the three controllers while retaining a computation time comparable to that of the neural-network imitation controller. Overall, these results show that CLBF-RL provides the best trade-off between cumulative cost, convergence speed, and online computational efficiency in this CSTR example.

Although the same sampling period is used for all controllers in the comparison to ensure a fair evaluation under the same update frequency, the computation-time results suggest an additional practical advantage of CLBF-RL. CLBF-MPC solves a nonlinear constrained optimization problem at each sampling instant, so the sampling period must be large enough to accommodate the online solve time. In contrast, CLBF-RL mainly requires a neural-network forward pass and the evaluation of the active CLBF constraints. Therefore, CLBF-RL can be implemented with a smaller sampling period when the hardware and measurement system allow it. This is useful not only from a computational viewpoint, but also from the theoretical viewpoint in Remark 6: a smaller Δ reduces the sample-and-hold and nominal-to-actual mismatch terms in the feasibility and practical-stability analysis, which can make the sufficient conditions less conservative and reduce the size of the robust inner level reached by the disturbed closed-loop system.

5. Conclusion

This work developed a CLBF-RL control framework for nonlinear systems with input constraints and unsafe operating regions. The proposed controller uses a constraint-informed actor network with an



(a) Cumulative cost versus initial Lyapunov value.

(b) Steps to the target set versus initial Lyapunov value.

Fig. 13. Closed-loop performance comparison over 1,000 trajectories generated from the same initial-condition set. The curves show binned median values, and the shaded regions indicate variability across trajectories within each bin.

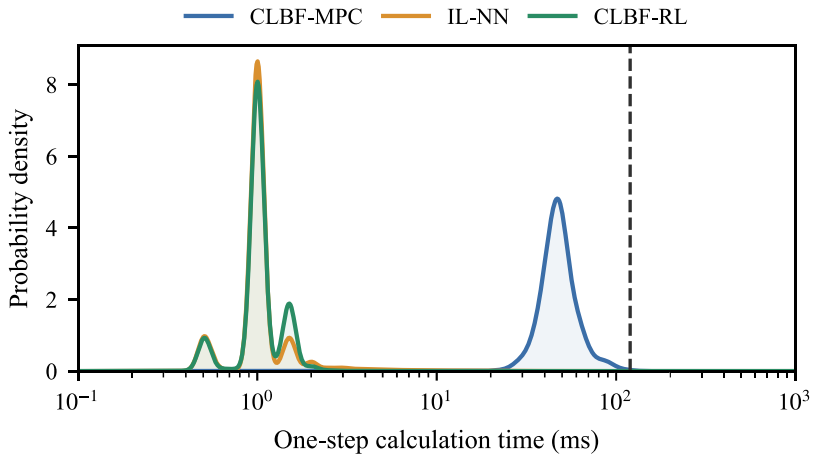


Fig. 14. Distribution of one-step controller computation time. The vertical dashed line indicates the sampling time of 120 ms.

internal CLBF gate, together with an outer constraint check during RL training and implementation. In this structure, the actor is trained to improve closed-loop performance while the implemented input satisfies the active CLBF-based constraint. A correction penalty is introduced during training to discourage reliance on fallback actions when exploration noise leads to inadmissible submitted actions. Theoretical analysis established that, under bounded disturbances and sample-and-hold implementation, the proposed CLBF-RL controller is feasible at every sampling instant, keeps the closed-loop state inside the admissible CLBF level set, avoids the unsafe region, and drives the state to a robust inner CLBF level set.

The proposed method was demonstrated on a non-isothermal CSTR with an embedded unsafe region. The training study showed that $\lambda_c = 0.1$ and $N_r = 1$ provided the best trade-off between closed-loop cost and computational effort for the case study considered. Closed-loop simulations under both nominal and non-Gaussian disturbed conditions showed that the trained CLBF-RL controller drives the state toward the nominal steady state while avoiding the unsafe region. Compared with CLBF-MPC and an imitation-learning neural-network controller, CLBF-RL achieved lower median cumulative cost and faster convergence to the target set, while requiring an online computation time comparable to the neural-network imitation controller and much smaller than CLBF-MPC. These results show that the proposed CLBF-RL framework can combine the online efficiency and performance-learning capability of RL with CLBF-based safety and stability guarantees for nonlinear process control.

CRediT authorship contribution statement

Xiaodong Cui: Writing – original draft, Software, Methodology, Investigation, Conceptualization. **Arthur Khodaverdian:** Writing – original draft, Methodology, Investigation, Conceptualization. **Panagiotis D. Christofides:** Writing – review & editing, Methodology, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

Financial support from the National Science Foundation, CBET-2140506, is gratefully acknowledged.

Data availability

Data will be made available on request.

References

Albalawi, F., Durand, H., Christofides, P.D., 2017. Process operational safety using model predictive control based on a process safeness index. Comput. Chem. Eng. 104, 76–88.

Ames, A.D., Coogan, S., Egerstedt, M., Notomista, G., Sreenath, K., Tabuada, P., 2019. Control barrier functions: Theory and applications. In: Proceedings of European Control Conference. Naples, Italy, pp. 3420–3431.

Berkenkamp, F., Turchetta, M., Schoellig, A., Krause, A., 2017. Safe model-based reinforcement learning with stability guarantees. In: Proceedings of 31st Conference on Neural Information Processing Systems. Long Beach, CA, USA.

Chen, W.-H., Ballance, D.J., O'Reilly, J., 2000. Model predictive control of nonlinear systems: Computational burden and stability. IEE Proc., Control Theory Appl. 147, 387–394.

Cheng, R., Orosz, G., Murray, R.M., Burdick, J.W., 2019. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 3387–3395.

Choi, J., Castaneda, F., Tomlin, C.J., Sreenath, K., 2020. Reinforcement learning for safety-critical control under model uncertainty, using control Lyapunov functions and control barrier functions. In: Proceedings of Robotics: Science and Systems. Corvallis, Oregon, USA.

Crowl, D.A., Louvar, J.F., 2011. Chemical Process Safety: Fundamentals with Applications, third ed. Prentice Hall, Upper Saddle River, NJ.

Cui, X., Khodaverdian, A., Christofides, P.D., 2026. Robust reinforcement learning for nonlinear process control with stability guarantees. Digit. Chem. Eng. 18, 100291.

Cui, X., Peters, D., Wang, Y., Çıtmacı, B., Richard, D., Morales-Guio, C.G., Christofides, P.D., 2024. Modeling and control of a protonic membrane steam methane reformer. Chem. Eng. Res. Des. 212, 493–519.

Elmaz, F., Di Caprio, U., Wu, M., Wouters, Y., Van Der Vorst, G., Vandervoort, N., Anwar, A., Leblebici, M.E., Hellinckx, P., Mercelis, S., 2023. Reinforcement learning-based approach for optimizing solvent-switch processes. Comput. Chem. Eng. 176, 108310.

Emam, Y., Notomista, G., Glotfelter, P., Kira, Z., Egerstedt, M., 2022. Safe reinforcement learning using robust control barrier functions. IEEE Robot. Autom. Lett. 10, 2886–2893.

Faria, R.d., Capron, B.D.O., Secchi, A.R., de Souza, Jr., M.B., 2022. Where reinforcement learning meets process control: Review and guidelines. Processes 10, 2311.

Fujimoto, S., Hoof, H., Meger, D., 2018. Addressing function approximation error in actor-critic methods. In: Proceedings of International Conference on Machine Learning. pp. 1587–1596.

Garcıa, J., Fernández, F., 2015. A comprehensive survey on safe reinforcement learning. J. Mach. Learn. Res. 16, 1437–1480.

Haarnoja, T., Zhou, A., Abbeel, P., Levine, S., 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: Proceedings of International Conference on Machine Learning. pp. 1861–1870.

Kaelbling, L.P., Littman, M.L., Moore, A.W., 1996. Reinforcement learning: A survey. J. Artificial Intelligence Res. 4, 237–285.

Kasaura, K., Miura, S., Kozuno, T., Yonetani, R., Hoshino, K., Hosoe, Y., 2023. Benchmarking actor-critic deep reinforcement learning algorithms for robotics control with action constraints. IEEE Robot. Autom. Lett. 8 (8), 4449–4456.

Khodaverdian, A., Cui, X., Christofides, P.D., 2025. Utilizing reinforcement learning in feedback control of nonlinear processes with stability guarantees. Digit. Chem. Eng. 17, 100277.

Lees, F., 2012. Lees' Loss prevention in the process industries: Hazard identification, assessment and control. Butterworth-Heinemann.

Levine, S., Kumar, A., Tucker, G., Fu, J., 2020. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. arXiv preprint arXiv:2005.01643.

Lin, Y., Sontag, E.D., 1991. A universal formula for stabilization with bounded controls. Systems Control Lett. 16, 393–397.

Liu, S., Liu, J., 2016. Economic model predictive control with extended horizon. Automatica 73, 180–192.

Löfberg, J., 2012. Oops! I cannot do it again: Testing for recursive feasibility in MPC. Automatica 48, 550–555.

Luo, J., 2023. Machine Learning Modeling for Process Control and Electrochemical Reactor Operation. University of California, Los Angeles.

Mhaskar, P., El-Farra, N.H., Christofides, P.D., 2006. Stabilization of nonlinear systems with state and control constraints using Lyapunov-based predictive control. Syst. Control. Lett. 55, 650–659.

Osinenko, P., Dobriborsci, D., Aumer, W., 2022. Reinforcement learning with guarantees: a review. IFAC-PapersOnLine 55, 123–128.

Peters, D., Cui, X., Wang, Y., Morales-Guio, C.G., Christofides, P.D., 2026. Model predictive control of an experimental protonic membrane steam methane reforming system. AIChE J. 72 (2), e70105.

Qin, S.J., Badgwell, T.A., 2003. A survey of industrial model predictive control technology. Control Eng. Pract. 11, 733–764.

Rawlings, J.B., Mayne, D.Q., Diehl, M.M., 2017. Model Predictive Control: Theory, Computation, and Design, second ed. Nob Hill Publishing, Madison, WI.

Stauffer, T., Chastain-Knight, D., 2021. Do not let your safe operating limits leave you S-O-L (out of luck). Process. Saf. Prog. 40, e12163.

Sutton, R.S., Barto, A.G., 2018. Reinforcement Learning: An Introduction, second ed. MIT Press, Cambridge, MA.

Wang, Y., Wu, Z., 2024. Control Lyapunov-barrier function-based safe reinforcement learning for nonlinear optimal control. AIChE J. 70, e18306.

Wang, Y., Xiao, M., Wu, Z., 2024. Safe transfer-reinforcement-learning-based optimal control of nonlinear systems. IEEE Trans. Cybern. 54, 7272–7284.

Wang, W., Zhang, H., Wang, Y., Tian, Y., Wu, Z., 2024. Fast explicit machine learning-based model predictive control of nonlinear processes using input convex neural networks. Ind. Eng. Chem. Res. 63 (40), 17279–17293.

Wu, Z., Albalawi, F., Zhang, Z., Zhang, J., Durand, H., Christofides, P.D., 2019. Control Lyapunov-barrier function-based model predictive control of nonlinear systems. Automatica 109, 108508.

Wu, Z., Christofides, P.D., 2019. Handling bounded and unbounded unsafe sets in control Lyapunov-barrier function-based model predictive control of nonlinear processes. Chem. Eng. Res. Des. 143, 140–149.

Wu, Z., Christofides, P.D., 2020. Control Lyapunov-barrier function-based predictive control of nonlinear processes using machine learning modeling. Comput. Chem. Eng. 134, 106706.

Zeng, J., Zhang, B., Sreenath, K., 2021. Safety-critical model predictive control with discrete-time control barrier function. In: Proceedings of American Control Conference. New Orleans, LA, USA, pp. 3882–3889.