2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Stable and Safe Reinforcement Learning via a Barrier-Lyapunov Actor-Critic Approach

Liqun Zhao, Konstantinos Gatsis, Antonis Papachristodoulou

Abstract— **Reinforcement Learning (RL) has demonstrated impressive performance in various areas such as video games and robotics. However, ensuring safety and stability, which are two critical properties from a control perspective, remains a significant challenge when using RL to control real-world systems. In this paper, we first provide definitions of safety and stability for the RL system, and then combine the Control Barrier Function (CBF) and Control Lyapunov Function (CLF) methods with the actor-critic method in RL to propose a Barrier-Lyapunov Actor-Critic (BLAC) framework which helps maintain the aforementioned safety and stability for the system. In this framework, CBF constraints for safety and CLF constraint for stability are constructed based on the data sampled from the replay buffer, and the augmented Lagrangian method is used to update the parameters of the RL-based controller. Furthermore, an additional backup controller is introduced in case the RL-based controller cannot provide valid control signals when safety and stability constraints cannot be satisfied simultaneously. Simulation results[1] show that this framework yields a controller that can help the system approach the desired state and cause fewer violations of safety constraints compared to baseline algorithms[2].**

I. INTRODUCTION

The remarkable success of Reinforcement Learning (RL) in solving complex sequential decision-making problems has inspired researchers to explore its potential in real-world applications. However, the trial-and-error nature of RL may lead agents to exhibit actions resulting in harmful consequences during learning. For example, it is crucial to take the challenges, such as real-world uncertainties [2], [3], into consideration when training RL agents directly in real environments where safety is fundamental. Furthermore, stability is one of the paramount properties that need to be guaranteed. Stability means the state will stay close or converge to the equilibrium, and is a pre-requisite for any performance guarantee in a control system [4].

Safety in RL has been gaining attention from researchers. However, in previous studies, safety constraints are usually defined based on the cumulative cost of an entire trajectory, rather than on the individual cost signals at each timestep along the trajectory. Consequently, safety violations at specific timesteps are wrapped by the trajectory expectation, and certain states may be permitted to be unsafe [5], [6].

Recently, there has been growing interest in combining control-theoretic methods with RL to ensure safety for the system. Concepts such as Safe Set Algorithm (SSA) [5] and Control Barrier Function (CBF) [7]–[11] have been applied as safety constraints to help RL training maintain safety. However, according to [8], the supervised learning of the CBF layer in [7] has the potential to introduce approximations that may adversely impact RL training. Also, applying a filter to aid the RL-based controller in choosing a safe action at each timestep can lead to a jerky output, or trajectory, which is often undesirable in many applications.

The Lyapunov function, a popular tool in the control community, has recently been applied to RL as well [12]. [4] applies Lyapunov functions to model-free RL to help guarantee the stability for various systems. However, considering the sample efficiency, and the fact that usually at least a nominal model is available in real applications like robotic arms, model-based RL methods can be used since they are more sample efficient than model-free RL methods.

In this paper, our focus is on helping guarantee safety and stability for systems that can be expressed by Markov decision process (MDP). Our main work and contributions can be summarized as follows:

- An RL-based controller is proposed by combining CBF and Control Lyapunov Function (CLF) with the Soft Actor-Critic (SAC) algorithm [13], and the augmented Lagrangian method is used to solve the corresponding constrained optimization problem. The RL-based controller assists to guarantee both safety and stability simultaneously with separate CBF constraints and CLF constraint, respectively, and the augmented Lagrangian method can update the controller parameters efficiently while making the hyperparameter tuning for different learning rates easier, which might be difficult for the primal-dual update widely used in previous studies.
- A backup controller is proposed to replace the RL-based controller when no feasible solution exists to satisfy both safety and stability constraints simultaneously. The inequality constraints of the backup controller are constructed based on CBFs, which help the system achieve and maintain safety. The CLF constraint is incorporated into the objective function to prevent the system from diverging too far from equilibrium while satisfying CBF constraints.
- A framework called Barrier-Lyapunov Actor-Critic (BLAC) is proposed by combining the RL-based and

*The authors are within the Department of Engineering Science, University of Oxford, Oxford, United Kingdom. E-mails: {liqun.zhao, konstantinos.gatsis,antonis}@eng.ox.ac.uk

[1]The code can be found in the GitHub repository: https://github.com/LiqunZhao/A-Barrier-Lyapunov-Actor-Critic-Reinforcement-Learning-Approach-for-Safe-and-Stable-Control

[2]For a more comprehensive version of this paper, please refer to [1] (https://arxiv.org/abs/2304.04066) which includes additional details and an extra "Simulated Cars" task.

backup controllers, and we test it on two simulation tasks. Our results show that the framework helps guarantee safety and stability of the system, and therefore achieves better results compared to baseline algorithms.

II. PROBLEM STATEMENT

A Markov decision process (MDP) with control-affine dynamics can be defined by the tuple $\mathcal{M}$, which is $(\mathcal{X}, \mathcal{U}, f, g, d, r, c, \gamma, \gamma_c)$. $\mathcal{X} \subset \mathbb{R}^n$ and $\mathcal{U} \subset \mathbb{R}^m$ are state and control signal spaces, and the state transitions for the MDP are obtained by the following control-affine system:

$$x_{t+1} = f(x_t) + g(x_t)u_t + d(x_t). \tag{1}$$

Here, $x_t \in \mathcal{X}$ is the state at timestep t, $u_t \in \mathcal{U}$ is the control signal at timestep t, and the RL-based controller π is sampled from a distribution $\pi(u_t|x_t)$. $f : \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ define the known nominal model of the system. r, c are the reward and cost, respectively, and γ and γ_c are the discount factors. $d : \mathbb{R}^n \to \mathbb{R}^n$ denotes the unknown model which is continuous with respect to the state. Similar to [7], [8], we use a Gaussian process (GP) to estimate the unknown dynamics d from data. Consequently, when constructing CBF and CLF constraints in Section III, we replace d using the mean and variance given by the GP. Due to poor scalability with a large number of data points, we cease GP updates after a certain number of episodes.

Here we present additional notation that will be used later. Based on (1), the transition probability can be denoted as $P(x_{t+1}|x_t, u_t) \triangleq I_{\{x_{t+1}=f(x_t)+g(x_t)u_t+d(x_t)\}}$ where $I_{\{x_{t+1}=f(x_t)+g(x_t)u_t+d(x_t)\}}$ is an indicator function that equals 1 if x_{t+1} satisfies (1) given x_t and u_t, and 0 otherwise. Similar to [4], the closed-loop transition probability is denoted as $P_\pi(x_{t+1}|x_t) \triangleq \int_{\mathcal{U}} \pi(u_t|x_t)P(x_{t+1}|x_t, u_t)du_t$. Moreover, the closed-loop state distribution at timestep t is denoted by $\upsilon(x_t|\rho, \pi, t)$, which can be calculated iteratively using the closed-loop transition probability: $\upsilon(x_{t+1}|\rho, \pi, t+1) = \int_{\mathcal{X}} P_\pi(x_{t+1}|x_t)\upsilon(x_t|\rho, \pi, t)dx_t$, $\forall t \in \mathbb{N}$, and $\upsilon(x_0|\rho, \pi, 0) = \rho$ is the initial state distribution.

A. Definition of Safety

Assuming there are k different safety constraints that need to be satisfied, the system is considered safe if

$$h_i(x_t) \geq 0 \quad \forall t \geq 0 \tag{2}$$

holds for each $i = 1, \ldots, k$. Each $h_i : \mathbb{R}^n \to \mathbb{R}$ is a function defined for the i-th safety constraint, and a safe set $\mathcal{C}_i \subset \mathbb{R}^n$ can be defined by the super-level set of h_i as follows:

$$\mathcal{C}_i = \{x \in \mathbb{R}^n | h_i(x) \geq 0\}. \tag{3}$$

A safe set $\mathcal{C} \subset \mathbb{R}^n$ can therefore be defined as the intersection of all $\mathcal{C}_i$: $\mathcal{C} = \bigcap_{i=1}^{k} \mathcal{C}_i = \bigcap_{i=1}^{k} \{x \in \mathbb{R}^n | h_i(x) \geq 0\}$. We require the system state to remain within this set $\mathcal{C}$, i.e., the safe set $\mathcal{C}$ should be forward invariant. Therefore, the system is required to be safe at every time-step, and thus the constraint here is stricter than that constructed by the expected return of costs, which was widely used in previous studies. A Control Barrier Function can be used to ensure forward invariance of the safe set.

Definition 1 (Discrete-time Control Barrier Function [7])**.** *Given a set $\mathcal{C}_i \subset \mathbb{R}^n$ defined by (3), the function h_i is called a discrete-time control barrier function (CBF) for system (1) if there exists $\eta \in [0, 1]$ such that*

$$\sup_{u_t \in \mathcal{U}} \big\{h_i\big(f(x_t) + g(x_t)u_t + d(x_t)\big) - h_i(x_t)\big\} \geq -\eta h_i(x_t) \tag{4}$$

holds for all $x_t \in \mathcal{C}_i$.

The existence of a CBF means the existence of a controller such that the set $\mathcal{C}_i$ is forward invariant. Consequently, safety is maintained if there exists a controller such that $\forall i \in [1, k]$, $h_i\big(f(x_t) + g(x_t)u_t + d(x_t)\big) - h_i(x_t) \geq -\eta h_i(x_t)$ holds for all $x_t \in \mathcal{C}$.

B. Definition of Stability

In a stabilization task, our goal is to find a controller that can drive the system state to the equilibrium, i.e., the desired state, eventually. To achieve this goal, given the state x_t and control signal u_t, we define the instantaneous cost signal to be $c(x_t, u_t) = \|x_{t+1} - x_{\text{desired}}\|$ where x_{t+1} is the next state according to (1), and x_{desired} denotes the desired state (i.e., equilibrium). Since we investigate the stability of a closed-loop system under a nondeterministic RL-based controller π, we combine $\pi(u_t|x_t)$, which is a Gaussian distribution in this paper, with the cost signal $c(x_t, u_t)$ to define the cost function under the controller π as

$$c_\pi(x_t) = \mathbb{E}_{u_t\sim\pi} c(x_t, u_t) = \mathbb{E}_{u_t\sim\pi}[\|x_{t+1} - x_{\text{desired}}\|]. \tag{5}$$

The cost function $c_\pi(x_t)$ represents the expected value of the norm of the difference between the next state x_{t+1} given by (1), and the desired state x_{desired}, over u_t sampled from the distribution $\pi(u_t|x_t)$. It is natural to expect that the value of $c_\pi(x_t)$ should decrease as t increases to drive the system state towards the equilibrium, and we hope that eventually $c_\pi(x_t) = 0$, which means the state reaches the equilibrium. However, as the control signal is sampled from a Gaussian distribution, the state at time-step t is also distributed, which necessitates the use of the concept of "expected value" for $c_\pi(x_t)$. Therefore, we adopt the definition of stability as presented in [4] in this paper.

Definition 2 (Stability in Mean Cost)**.** *Let $\upsilon(x_t|\rho, \pi, t)$ denote the closed-loop state distribution at timestep t. The equilibrium of a system is said to be stable in mean cost if there exists a positive constant b such that for any initial state $x_0 \in \{x_0|c_\pi(x_0) < b\}$, the condition*

$$\lim_{t\to\infty} \mathbb{E}_{x_t\sim\upsilon}\big[c_\pi(x_t)\big] = 0 \tag{6}$$

holds. If b is arbitrarily large, the equilibrium is globally stable in mean cost.

C. Definition of the Safe and Stable Control Problem

Based on the previous subsections, similar to [14], we give the formal definition of the safe and stable control problem:

Problem 1 (Safe and Stable Control Problem)**.** *Given a control-affine system $x_{t+1} = f(x_t) + g(x_t)u_t + d(x_t)$, a unique desired state (equilibrium) $x_{desired}$, a set $\mathcal{X}_b = \{x_0 | c_\pi(x_0) < b\}$ where b is an arbitrarily large positive number and x_0 denotes the initial state, a set of unsafe states $\mathcal{X}_{unsafe} \subseteq \mathcal{X}$, and a set of safe states $\mathcal{X}_{safe} \subseteq \mathcal{X}$ such that $x_{desired} \in \mathcal{X}_{safe}$ and $\mathcal{X}_{safe} \cap \mathcal{X}_b \neq \emptyset$, find a controller π generating control signal u_t such that all trajectories satisfying $x_{t+1} = f(x_t) + g(x_t)u_t + d(x_t)$ and $x_0 \in \mathcal{X}_{safe} \cap \mathcal{X}_b$ have the following properties:*

- ***Safety**: $x_t \in \mathcal{X}_{safe} \ \ \forall t \geq 0$.*
- ***The Equilibrium Is Stable in Mean Cost:** $\lim_{t\to\infty} \mathbb{E}_{x_t \sim \upsilon}\big[c_\pi(x_t)\big] = 0$ where $\upsilon(x_t|\rho, \pi, t)$ is the closed-loop state distribution at timestep t.*

Therefore $\mathcal{X}_{safe}$ is a safe set that is forward invariant, and the controller is used to help the system arrive at the desired state x_{desired} while avoiding the unsafe states.

III. Framework Design

A. Value Function of the Cost

To assist in maintaining stability for the system, inspired by the commonly-used value functions in the RL literature, we define the value function of the cost at state x_t as

$$L_\pi(x_t) = \mathbb{E}_{\tau\sim\pi}\big[\sum_{i=0}^{\infty} \gamma_c^i c_\pi(x_{t+i})\big]. \tag{7}$$

Here $\tau = \{x_t, x_{t+1}, x_{t+2}, \cdots\}$ is a trajectory under controller π starting from the initial state x_t. Based on this definition, $L_\pi(x_t)$ can also be approximated by a neural network.

To achieve stability, a natural approach is to consider imposing a condition in the algorithm that ensures that the value of $L_\pi(x_t)$ decreases along the trajectory τ. Inspired by the concept of exponentially stabilizing CLF and [4], we first make two assumptions for the MDP:

Assumption 1. *The state and control signal are sampled from compact sets, and the reward and cost values obtained at any timestep are bounded by r_{max} and c_{max}, respectively. Therefore, the value function of the reward and cost are upper bounded by $\frac{r_{max}}{1-\gamma}$ and $\frac{c_{max}}{1-\gamma_c}$, respectively.*

Assumption 2 (Ergodicity)**.** *The Markov chain induced by controller π is ergodic with a unique stationary distribution $q_\pi(x) = \lim_{t\to\infty} \upsilon(x_t = x|\rho, \pi, t)$, where $\upsilon(x_t|\rho, \pi, t)$ is the closed-loop state distribution.*

These assumptions are common; for more content on the relationship between models of control systems and Markov chains, as well as ergodicity, we refer the interested reader to [15]. We now introduce Lemma 1 invoked by [4] as follows:

Lemma 1. *Under Assumptions 1, 2, the system defined in (1) is stable in mean cost if there exist positive constants α_1, α_2, β, and a controller π, such that*

$$\alpha_1 c_\pi(x) \leq L_\pi(x) \leq \alpha_2 c_\pi(x) \tag{8}$$

$$\mathbb{E}_{x\sim\mu_\pi, x'\sim P_\pi}\big[L_\pi(x') - L_\pi(x)\big] \leq -\beta \mathbb{E}_{x\sim\mu_\pi}\big[L_\pi(x)\big] \tag{9}$$

hold for all $x \in \mathcal{X}$. Here L_π defined in (7) is the value function of the cost under the controller π, and

$$\mu_\pi(x) \triangleq \lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N} \upsilon(x_t = x|\rho, \pi, t) \tag{10}$$

is the sampling distribution, where $\upsilon(x_t|\rho, \pi, t)$ is the closed-loop state distribution at timestep t.

Proof. The proof of this lemma closely resembles that of Theorem 1 in [4], and interested readers are encouraged to refer to that paper for further details. □

According to the proof presented in [16], L_π naturally satisfies the constraints (8) under Assumption 1. In the next subsection, we present a method to obtain a controller that results in an L_π satisfying (9), and thus we can call L_π an exponentially stabilizing CLF, abbreviated as CLF hereafter.

B. Augmented Lagrangian Method for Parameter Updating

We now turn our attention to updating an RL-based controller π to meet conditions (4) and (9). A primal-dual method is commonly used to update the parameter of the RL-based controller, however, this method often requires hyperparameter tuning, which might be challenging for adjusting the learning rates used to update the parameters of different neural networks and Lagrangian multipliers. Here we adopt the augmented Lagrangian method, inspired by its effectiveness in solving constrained optimization problems and [17], to update the parameters of the RL-based controller.

Practically, we need to construct the conditions based on data. Transition pairs $(x_t, u_t, r_t, c_t, x_{t+1})$ are stored in the replay buffer, and we sample a batch of these transition pairs, denoted as $\mathcal{D}$, randomly at each time-step to construct the CBF and CLF constraints with the system model for safety and stability, respectively:

$$\begin{aligned} &h_i(\hat{x}_{t+1}) - h_i(x_t) \geq -\eta h_i(x_t) && \forall i \in [1, k], \\ &L_\pi(\hat{x}_{t+1}) - L_\pi(x_t) \leq -\beta L_\pi(x_t), \end{aligned} \tag{11}$$

for each $x_t \in \mathcal{D}$. Here $\hat{x}_{t+1} = f(x_t) + g(x_t)u + d(x_t)$ is the predicted next state, and $d(x_t)$ is estimated and replaced by the mean value given by GP. Note that u here is the control signal generated by the current controller, and thus, these constraints are functions of the controller π. To apply the augmented Lagrangian method, these inequality constraints are converted to equality constraints using ReLU [17] for each $x_t \in \mathcal{D}$:

$$\begin{aligned} &ReLU\big(h_i(x_t) - h_i(\hat{x}_{t+1}) - \eta h_i(x_t)\big) = 0 && \forall i \in [1, k], \\ &ReLU\big(L_\pi(\hat{x}_{t+1}) - L_\pi(x_t) + \beta L_\pi(x_t)\big) = 0. \end{aligned} \tag{12}$$

Then the actor-critic approach is used to learn the RL-based controller. We represent the parameters of the RL-based controller and two action-value networks by θ and $\phi_i, i = 1, 2$, respectively. Moreover, we use L_ν, which is called Lyapunov network, to approximate L_π defined in (7)

with parameters ν. Using these notations, we formulate a new constrained optimization problem as follows:

$$\begin{aligned}\min_{\theta}\ & -V^{\pi_\theta}\\ s.t.\ & \mathbb{E}_{x_t\sim\mathcal{D},u\sim\pi}\big[ReLU\big(h_i(x_t)-h_i(\hat{x}_{t+1})-\eta h_i(x_t)\big)\big]=0\\ & \qquad\qquad \forall i\in[1,k]\\ & \mathbb{E}_{x_t\sim\mathcal{D},u\sim\pi}\big[ReLU\big(L_\nu(\hat{x}_{t+1})-L_\nu(x_t)+\beta L_\nu(x_t)\big)\big]=0.\end{aligned} \tag{13}$$

Here expected values are used to construct constraints since we sample a batch of x_t from the replay buffer. The objective function $-V^{\pi_\theta}(x_t)$ is:

$$\begin{aligned}-V^{\pi_\theta}=-\mathbb{E}_{x_t\sim\mathcal{D},\xi\sim\mathcal{N}}\big[&\min_{j=1,2}Q_{\phi_j}(x_t,\tilde{u}_\theta(x_t,\xi))\\ &-\alpha\log\pi_\theta(\tilde{u}_\theta(x_t,\xi)|x_t)\big]\end{aligned} \tag{14}$$

which is the same as the commonly-used objective function in SAC [13] with only small differences in notation. Also, $\tilde{u}_\theta(x_t,\xi)=\tanh(\mu_\theta(x_t)+\sigma_\theta(x_t)\odot\xi),\xi\sim\mathcal{N}(0,I)$, where μ_θ and σ_θ denote the mean and standard deviation of the controller π, which is a Gaussian distribution, and $\odot$ represents element-wise multiplication. Additionally, loss functions of the action-value networks $Q_{\phi_i},i=1,2$, coefficient α, and Lyapunov network L_ν are:

$$\begin{aligned}J_Q(Q_{\phi_i})=\mathbb{E}_{(x_t,u_t,r_t,x_{t+1})\sim\mathcal{D},\xi\sim\mathcal{N}}\Big[\big[r_t+\gamma\big(&\min_{j=1,2}Q_{targ,\phi_j}(x_{t+1},\\ \tilde{u}_\theta(x_{t+1},\xi))-\alpha\log\pi_\theta(\tilde{u}_\theta(x_{t+1},\xi)|x_{t+1})\big)-Q_{\phi_i}(x_t,u_t)\big]^2\Big],\end{aligned} \tag{15}$$

$$J_\alpha(\alpha)=-\alpha\times\mathbb{E}_{x_t\sim\mathcal{D},\xi\sim\mathcal{N}}\big[\log\pi_\theta(\tilde{u}_\theta(x_t,\xi)|x_t)+\mathcal{H}\big], \tag{16}$$

$$J_L(L_\nu)=\mathbb{E}_{(x_t,c_t,x_{t+1})\sim\mathcal{D}}\Big[\big[c_t+\gamma_c L_{targ,\nu}(x_{t+1})-L_\nu(x_t)\big]^2\Big], \tag{17}$$

where $Q_{targ,\phi_i},i=1,2$ are the target action-value networks, and $\mathcal{H}$ is a designed threshold set to be the lower bound of the entropy of the controller π_θ. The constrained optimization problem (13) can be solved by the augmented Lagrangian method, and the augmented Lagrangian function is:

$$\begin{aligned}&\mathcal{L}_A(\theta,\lambda_i,\zeta;\rho_{\lambda_i},\rho_\zeta)=-V^{\pi_\theta}\\ &+\sum_{i=1}^{k}\lambda_i\times\mathbb{E}_{x_t\sim\mathcal{D},u\sim\pi}\big[ReLU\big(h_i(x_t)-h_i(\hat{x}_{t+1})-\eta h_i(x_t)\big)\big]\\ &+\sum_{i=1}^{k}\frac{\rho_{\lambda_i}}{2}\big[\mathbb{E}_{x_t\sim\mathcal{D},u\sim\pi}\big[ReLU\big(h_i(x_t)-h_i(\hat{x}_{t+1})-\eta h_i(x_t)\big)\big]\big]^2\\ &+\zeta\times\mathbb{E}_{x_t\sim\mathcal{D},u\sim\pi}\big[ReLU\big(L_\nu(\hat{x}_{t+1})-L_\nu(x_t)+\beta L_\nu(x_t)\big)\big]\\ &+\frac{\rho_\zeta}{2}\big[\mathbb{E}_{x_t\sim\mathcal{D},u\sim\pi}\big[ReLU\big(L_\nu(\hat{x}_{t+1})-L_\nu(x_t)+\beta L_\nu(x_t)\big)\big]\big]^2,\end{aligned} \tag{18}$$

where λ_i and ζ are the Lagrangian multipliers for CBF and CLF constraints, respectively, and ρ_{λ_i} and ρ_ζ are the corresponding coefficients for the additional quadratic terms. Since it is usually not easy to solve the problem

$$\theta_{k+1}=\arg\min_{\theta}\mathcal{L}_A(\theta,\lambda_{i,k},\zeta_k;\rho_{\lambda_{i,k}},\rho_{\zeta_k}) \tag{19}$$

directly in RL, we still apply gradient descent to update θ, and gradient ascent to update λ_i and ζ, while increasing the value of ρ_{λ_i} and ρ_ζ gradually to find a solution for the constrained optimization problem (13). The pseudocode is provided as a part of Algorithm 1.

C. Backup Controller Design

Due to the existence of several constraints, the feasibility of the constrained optimization problem (13) becomes a crucial problem during the learning process. Typically, there are two scenarios where infeasibility can cause problems:

- The CLF constraint for stability is violated, for example, the agent is required to cross an obstacle to reach the equilibrium by the stability constraint, but this is prevented by the safety constraint, and thus the agent is trapped in some specific positions near the obstacle.
- The CBF constraint for safety is violated, for example, the desired state is not fixed and can fall in a danger region at some time-steps, which means satisfying the CLF constraint can breach the safety constraint then.

The frequent invalid control signals provided by the RL-based controller due to the infeasibility may prevent the system from approaching its desired state quickly, or violate the safety constraints severely. Given that safety takes priority when safety and stability constraints cannot be satisfied simultaneously, similar to [7], [8], we propose to design a backup controller by formulating an additional constrained optimization problem that leverages CBFs as constraints to achieve and maintain safety. However, compared to previous studies, the objective function of this backup controller is designed not only to minimize the difference between the actual control signal and nominal control signal, but also to prevent the system from deviating too much from its equilibrium by incorporating the CLF constraint. We first establish a QP-based controller as follows:

$$\begin{aligned}\min_{u_{\text{modi}},\epsilon}\ &\frac{1}{2}u_{\text{modi}}^T Q\,u_{\text{modi}}+k_{\epsilon,i}\epsilon_i^2-\kappa\nabla_x L_\nu(x_t)\cdot g(x_t)u_{\text{modi}}\\ s.t.\ & h_i\big(f(x_t)+g(x_t)(u_{\text{nominal}}-u_{\text{modi}})+d(x_t)\big)\\ &\qquad -h_i(x_t)\geq-\eta h_i(x_t)-\epsilon_i\quad\forall i\in[1,k],\end{aligned} \tag{20}$$

where x_t is the current state of the system, Q is a symmetric positive semidefinite matrix, and similar to [7], [8], $d(x_t)$ is estimated and replaced by the mean and variance given by GP. ϵ_i is the slack variable introduced to enforce the feasibility of the constrained optimization problem with the coefficient $k_{\epsilon,i}$. $u_{\text{modi}}=u_{\text{nominal}}-u_{\text{actual}}$, where u_{modi} is the optimization variable, u_{nominal} is the nominal control signal, and u_{actual} is the actual control signal to perform. Additionally, $\nabla_x L_\nu(x)$ is the gradient of the Lyapunov network with respect to the state, and κ is a coefficient.

The choices of coefficients, nominal control signal, and the condition for using the backup controller depend on the task. When the CBFs are not affine with respect to the control signal, it is possible to apply local first-order linearization to the CBFs to obtain approximate affine surrogate CBFs, or other types of constrained optimization problems can be

used to construct the backup controller. In summary, the framework combining the RL-based and backup controllers can be summarized as Algorithm 1.

Algorithm 1 Barrier-Lyapunov Actor-Critic (BLAC)

Initialization: RL-based controller network π_θ, coefficient α, action-value networks $Q_{\phi_i}, i = 1, 2$, Lyapunov network L_ν, Lagrange multipliers λ_i and ζ, replay buffer $\mathcal{B}$, coefficients of quadratic terms ρ_{λ_i} and ρ_ζ, learning rates η_1, η_2, and η_3, quadratic term coefficient factor $C_\rho \in (1, \infty)$
for $k = 1, \ldots, K$ **do**
  if Backup controller should be used according to the condition specific to the task **then**
    Solve the constrained optimization problem (20)
    Apply the control signal u_{actual}
  else
    Sample and apply control signal u_t
    Store the transition pair $(x_t, u_t, r_t, c_t, x_{t+1})$ in $\mathcal{B}$
    Sample a batch of transition pairs randomly from $\mathcal{B}$, and construct CBF and CLF constraints with the system model
    Update the Lyapunov network and action-value networks by using (17) and (15) according to
$$\nu_{k+1} \leftarrow \nu_k - \eta_1 \nabla_\nu J_L(L_{\nu_k})$$
$$\phi_{i_{k+1}} \leftarrow \phi_{i_k} - \eta_1 \nabla_{\phi_i} J_Q(Q_{\phi_{i_k}})$$
    Update the controller network and coefficient α by using (18) and (16) according to
$$\theta_{k+1} \leftarrow \theta_k - \eta_2 \nabla_\theta \mathcal{L}_A(\theta_k, \lambda_{i,k}, \zeta_k; \rho_{\lambda_{i,k}}, \rho_{\zeta_k})$$
$$\alpha_{k+1} \leftarrow \alpha_k - \eta_2 \nabla_\alpha J_\alpha(\alpha_k)$$
    Update the Lagrangian multipliers using (18) according to
$$\lambda_{i,k+1} \leftarrow \lambda_{i,k} + \eta_3 \nabla_{\lambda_i} \mathcal{L}_A(\theta_{k+1}, \lambda_{i,k}, \zeta_k; \rho_{\lambda_{i,k}}, \rho_{\zeta_k})$$
$$\zeta_{k+1} \leftarrow \zeta_k + \eta_3 \nabla_\zeta \mathcal{L}_A(\theta_{k+1}, \lambda_{i,k}, \zeta_k; \rho_{\lambda_{i,k}}, \rho_{\zeta_k})$$
    Update coefficients of quadratic terms by $\rho_{\lambda_{i,k+1}} \leftarrow C_\rho \rho_{\lambda_{i,k}}$, $\rho_{\zeta_{k+1}} \leftarrow C_\rho \rho_{\zeta_k}$, and GP model
  end if
end for
return π_θ, $Q_{\phi_i}, i = 1, 2$, and L_ν.

IV. Simulations

In this section, we test the framework on two tasks to answer the following questions [3]:

- Does the BLAC framework assist in guaranteeing the stability for the system when compared to other baseline algorithms? Since in these tasks high rewards will be given when the system approaches and achieves the desired state (equilibrium), we use the cumulative reward as a measure, where a higher cumulative reward can indicate that the system can converge to the equilibrium faster with fewer deviations.
- Does the BLAC framework cause fewer violations of safety constraints compared to baseline algorithms?

We use LAC [4], CPO [18], PPO-Lagrangian and TRPO-Lagrangian [19] for comparison. Furthermore, to demonstrate whether the CLF constraint contributes to maintaining stability and therefore improves performance, we remove the CLF constraint in the BLAC framework to create an additional algorithm Barrier Actor-Critic (BAC) for comparison.

The task "Unicycle" is modified from the first environment in [8]. In this experiment, a unicycle is required to arrive at a desired location, i.e., destination, while avoiding collisions with obstacles. The model of the unicycle is given by:

$$x_{t+1} = x_t + \begin{bmatrix} \Delta T \cos(\theta_t) & 0 \\ \Delta T \sin(\theta_t) & 0 \\ 0 & \Delta T \end{bmatrix} (u_t + u_{d,t}).$$

Here $x_t = [x_{1t}, x_{2t}, \theta_t]^T$ where x_{1t} and x_{2t} are the X-coordinate and Y-coordinate of the unicycle at the timestep t, θ is the angle between the X-coordinate and the direction of the unicycle's movement at t. Also, $u_t = [v_t, \omega_t]^T$ is the control signal where v_t and ω_t are the linear and angular velocities, respectively. ΔT represents the time interval and $u_{d,t} = -0.1[\cos(\theta_t), 0]^T$ is unknown to the nominal model, and therefore is the unknown part and GPs can be used. Then, to formulate collision-free safety constraints, a point at a distance $l_p \geq 0$ ahead of the unicycle is considered, and we define the function $p : \mathbb{R}^3 \to \mathbb{R}^2$ to be

$$p(x_t) = \begin{bmatrix} x_{1t} \\ x_{2t} \end{bmatrix} + l_p \begin{bmatrix} \cos(\theta_t) \\ \sin(\theta_t) \end{bmatrix}.$$

The reward signal is defined as $-K_1(v_t - v_s)^2 + K_2 \Delta d$, where v_s is the predefined velocity, Δd is the decrease in the distance between the unicycle and destination in two consecutive timesteps, and K_1 and K_2 are coefficients set to 0.1 and 30, respectively. The cost signal is $\|p(x_{t+1}) - p(x_{\text{desired}})\|$ where $p(x_{\text{desired}}) = [x_{1\text{desired}}, x_{2\text{desired}}]^T$ denotes the position of the desired location. CBFs are defined as $h_i(x_t) = \frac{1}{2}\big((p(x_t) - p_{\text{obs}_i})^2 - \delta^2\big)$ where p_{obs_i} is the position of the i-th obstacle, and δ denotes the minimum required distance between the unicycle and obstacles. When the stability constraint is violated if safety and stability constraints cannot be satisfied simultaneously, the unicycle can get trapped near obstacles, and then the RL-based controller is replaced by the backup controller where u_{nominal} is set to be the maximum allowable control signal to encourage exploration while maintaining safety. The RL-based controller will resume when the unicycle moves away from the trapped position for a long distance, or when the predetermined time threshold for using the backup controller is exceeded.

Simulation results are shown in Figure 1. Compared to other baselines, our framework enables the system to achieve higher cumulative reward in fewer episodes, indicating that the framework helps the unicycle approach and successfully reach its destination (equilibrium) after a shorter training process. Additionally, the fluctuations of the cumulative

[3]For brevity, we provide a brief overview of the "Unicycle" task in this paper. For a more comprehensive discussion which includes additional details about the "Unicycle" task, the description of the "Simulated Cars" task and relevant experiment results, please refer to [1].

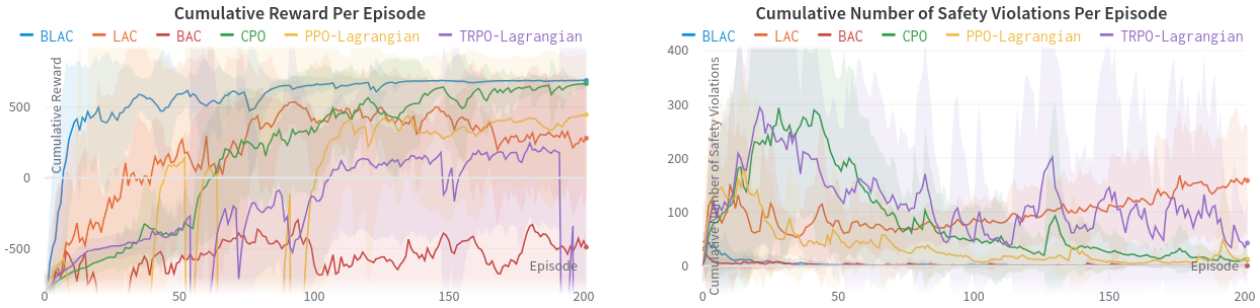

Fig. 1. The cumulative reward and cumulative number of safety violations of each episode in the unicycle environment are compared for the proposed BLAC (in blue) and other baselines. Each plot shows the mean of ten experiments using different seeds. The shading represents the standard deviation.

reward are smaller than those of any other baseline algorithm, suggesting that the system performance in obtaining high rewards can quickly recover to its original high level after deterioration. Regarding safety, as evidenced by the cumulative number of safety violations per episode, our framework results in much fewer violations compared to LAC, CPO, PPO-Lagrangian, and TRPO-Lagrangian where CBFs are not used, which means CBFs can help maintain safety. Our framework is thus suitable for safety-critical applications.

V. CONCLUSIONS

In this paper, we propose the BLAC framework, which combines separate CBF and CLF constraints with the actor-critic RL method to help to guarantee both safety and stability of the controlled system. This framework imposes safety constraints for each step in the trajectory instead of the trajectory expectation and thus imposes stricter safety constraints, which is crucial in real-world safety-critical applications. Moreover, our framework contributes to guaranteeing stability of the system, facilitating the system to approach the desired state (equilibrium) and obtain higher cumulative reward in tasks where high rewards are offered when the system gets closer to or reaches the desired state, such as navigation tasks. With the augmented Lagrangian method and backup controller, higher cumulative reward and fewer safety constraint violations are seen in experiments.

However, there are also some limitations of this framework: 1. The CBFs are predefined before the learning process, but in real-world applications, it may be nontrivial to construct valid CBFs; 2. The framework requires knowledge of the control-affine model of the system. Also, performance comparisons can be conducted between this RL-based control policy and other model-based optimal control policies [20]; 3. The framework is only tested on tasks where the relative-degree of the CBFs is 1, and therefore, more research where CBFs with high relative-degree are used should be conducted in the future. We believe that addressing these current limitations could be interesting future directions.

REFERENCES

[1] L. Zhao, K. Gatsis, and A. Papachristodoulou, "Stable and safe reinforcement learning via a barrier-lyapunov actor-critic approach," *arXiv preprint arXiv:2304.04066*, 2023.

[2] Y. Wang and D. Boyle, "Trustworthy reinforcement learning for quadrotor uav tracking control systems," *arXiv preprint arXiv:2302.11694*, 2023.

[3] Y. Wang, J. O'Keeffe, Q. Qian, and D. Boyle, "Quadue-ccm: Interpretable distributional reinforcement learning using uncertain contraction metrics for precise quadrotor trajectory tracking," in *Conference on Robot Learning*, pp. 2306–2316, PMLR, 2023.

[4] M. Han, L. Zhang, J. Wang, and W. Pan, "Actor-critic reinforcement learning for control with stability guarantee," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6217–6224, 2020.

[5] H. Ma, C. Liu, S. E. Li, S. Zheng, W. Sun, and J. Chen, "Learn zero-constraint-violation policy in model-free constrained reinforcement learning," *arXiv preprint arXiv:2111.12953*, 2021.

[6] H. Ma, C. Liu, S. E. Li, S. Zheng, and J. Chen, "Joint synthesis of safety certificate and safe control policy using constrained reinforcement learning," in *Learning for Dynamics and Control Conference*, pp. 97–109, PMLR, 2022.

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

[8] Y. Emam, P. Glotfelter, Z. Kira, and M. Egerstedt, "Safe model-based reinforcement learning using robust control barrier functions," *arXiv preprint arXiv:2110.05415*, 2021.

[9] H. Wang, K. Margellos, and A. Papachristodoulou, "Safety verification and controller synthesis for systems with input constraints," *arXiv preprint arXiv:2204.09386*, 2022.

[10] A. A. do Nascimento, A. Papachristodoulou, and K. Margellos, "A game theoretic approach for safe and distributed control of unmanned aerial vehicles," 2023.

[11] X. Tan and D. V. Dimarogonas, "On the undesired equilibria induced by control barrier function based quadratic programs," *arXiv preprint arXiv:2104.14895*, 2021.

[12] H. Cao, Y. Mao, L. Sha, and M. Caccamo, "Physical deep reinforcement learning towards safety guarantee," *arXiv preprint arXiv:2303.16860*, 2023.

[13] T. Haarnoja *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[14] C. Dawson, Z. Qin, S. Gao, and C. Fan, "Safe nonlinear control using robust neural lyapunov-barrier functions," in *Conference on Robot Learning*, pp. 1724–1735, PMLR, 2022.

[15] S. Meyn, *Control systems and reinforcement learning*. Cambridge University Press, 2022.

[16] S. Wang *et al.*, "A rl-based policy optimization method guided by adaptive stability certification," *arXiv preprint arXiv:2301.00521*, 2023.

[17] J. Li, D. Fridovich-Keil, S. Sojoudi, and C. J. Tomlin, "Augmented lagrangian method for instantaneously constrained reinforcement learning problems," in *2021 60th IEEE Conference on Decision and Control*, pp. 2982–2989, 2021.

[18] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *International conference on machine learning*, pp. 22–31, PMLR, 2017.

[19] A. Ray, J. Achiam, and D. Amodei, "Benchmarking Safe Exploration in Deep Reinforcement Learning," 2019.

[20] J. Moyalan, H. Choi, Y. Chen, and U. Vaidya, "Sum of squares based convex approach for optimal control synthesis," in *2021 29th Mediterranean Conference on Control and Automation (MED)*, pp. 1270–1275, IEEE, 2021.