Reinforcement Learning-based Receding Horizon Control using Adaptive Control Barrier Functions for Safety-Critical Systems*

Ehsan Sabouni[†,1], H.M. Sabbir Ahmad[†,1], Vittorio Giammarino[†,1],
Christos G. Cassandras[2], Ioannis Ch. Paschalidis[3], Wenchao Li[2]

***Abstract*— Optimal control methods provide solutions to safety-critical problems but easily become intractable. Control Barrier Functions (CBFs) have emerged as a popular technique that facilitates their solution by provably guaranteeing safety, through their forward invariance property, at the expense of some performance loss. This approach involves defining a performance objective alongside CBF-based safety constraints that must always be enforced with both performance and solution feasibility significantly impacted by two key factors: (i) the selection of the cost function and associated parameters, and (ii) the calibration of parameters within the CBF-based constraints, which capture the trade-off between performance and conservativeness. To address these challenges, we propose a Reinforcement Learning (RL)-based Receding Horizon Control (RHC) approach leveraging Model Predictive Control (MPC) with CBFs (MPC-CBF). In particular, we parameterize our controller and use bilevel optimization, where RL is used to learn the optimal parameters while MPC computes the optimal control input. We validate our method by applying it to the challenging automated merging control problem for Connected and Automated Vehicles (CAVs) at conflicting roadways. Results demonstrate improved performance and a significant reduction in the number of infeasible cases compared to traditional heuristic approaches used for tuning CBF-based controllers, showcasing the effectiveness of the proposed method. In order to guarantee reproducibility, our code is provided here: *link*[1].**

I. Introduction

In recent years, CBFs have been extensively applied in safety-critical control using a solution approach based on a sequence of Quadratic Programs (QPs) [1], referred to as QP-CBF. This method is often subject to infeasibility issues due to (i) the lack of any prediction ability due to the myopic nature of the QP-CBF method, and (ii) inter-sampling control errors due to time discretization. Additionally, CBF constraints require choosing a class $\mathcal{K}$ function that provides a trade-off between safety and performance.

To address this issue, [2] proposed an event-triggered control method to tackle primarily the inter-sampling error problem. The authors in [3], [4] proposed MPC-based control using CBFs which can combat the myopic limitations of QPs through a look-ahead prediction ability. However, these works lack systematic means to tune the parameters. Usually, this choice is done heuristically, which can result in sub-optimal controller response and, in the worst case, safety violation. In order to tackle this issue, adaptive CBFs (AdaCBF) are proposed in [5] to address the conservativeness of CBFs. However, this method becomes challenging to implement in practice because it requires defining "penalty terms" in the CBF constraints and their corresponding dynamics. Additionally, this method provides no guidance on tuning the CBF constraints to balance the trade-off for optimizing the system response while ensuring safety.

In the context of AdaCBFs, [6] proposed an end-to-end neural network-based controller design using differential QPs, where the parameters are tuned by means of behavioral cloning [7]; this performs poorly on out-of-distribution data due to a quadratic compounding of errors over time [8]. In [9], [10] RL is used to learn optimal parameters of an MPC controller by using Q-learning and actor-critic algorithms. However, these methods require computing gradients by solving KKT conditions and backpropagating through the MPC (which can be nonlinear), respectively, which makes them computationally expensive and generally intractable.

The QP-CBF method has been applied to safety-critical control problems involving Connected and Automated Vehicles (CAVs) at various conflict points of transportation networks [11]. Aside from the issues addressed above, the solutions to these problems have also been largely limited to the longitudinal dynamics of vehicles, decoupling these from their lateral dynamics. In this context, we make the following contributions:

- We propose a parameterized MPC controller with CBFs in order to address the infeasibility issues encountered in the QP-CBF methods. Moreover, we use RL to learn the optimal parameters in the MPC objective function, as well as the parameters in the CBF-based constraints, thus balancing the trade-off between safety and performance.
- Our proposed approach does not require backpropagating through the MPC-CBF controller, which can be computationally expensive and intractable. As a result, our approach is always computationally efficient.
- We tackle the problem of CAV control in merging

*Research partially supported by the NSF under grants ECCS-1931600, DMS-1664644, CNS-1645681, ECCS-2317079, CCF-2200052, CCF-2340776, CPS-1932162, and IIS-1914792, the DOE under grants DE-AC02-05CH11231 and DE-EE0009696, the ONR under grant N00014-19-1-2571 and the ARPA-E under grant DEAR0001282.

† These authors contributed equally to this paper.

[1] Ehsan Sabouni, H.M. Sabbir Ahmad and Vittorio Giammarino are with Division of Systems Engineering, Boston University, Boston, MA 02215, USA {esabouni, sabbir92, vgiammar}@bu.edu

[2] Christos G. Cassandras and Wenchao Li are with the Department of Electrical and Computer Engineering and the Division of Systems Engineering, Boston University, Boston, MA 02215, USA {cgc, wenchao}@bu.edu

[3] Ioannis Ch. Paschalidis is with the Department of Electrical and Computer Engineering, Division of Systems Engineering, and Department of Biomedical Engineering, Boston University, Boston, MA 02215, USA yannisp@bu.edu

[1] github.com/EhsanSabouni/CDC2024_RL_adpative_MPC_CBF/tree/main

roadways by considering both longitudinal and lateral motions. Besides validating our approach, the problem setting allows us to highlight the generalizability of the learnt controller. We achieve this by training the controller parameters for a single CAV and using them across a set of homogeneous CAVs during deployment.

II. Preliminaries

Consider a control affine system

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + g(\boldsymbol{x})\boldsymbol{u}, \tag{1}$$

where $f:\mathbb{R}^n \to \mathbb{R}^n$ and $g:\mathbb{R}^n \to \mathbb{R}^{n\times q}$ are locally Lipschitz, $\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^n$ denotes the state vector and $\mathbf{u} \in \mathcal{U} \subset \mathbb{R}^q$ the input vector with $\mathcal{U}$ the control input constraint set defined as:

$$\mathcal{U} := \{\boldsymbol{u} \in \mathbb{R}^q : \boldsymbol{u}_{\min} \leq \boldsymbol{u} \leq \boldsymbol{u}_{\max}\}. \tag{2}$$

with $\boldsymbol{u}_{\min}, \boldsymbol{u}_{\max} \in \mathbb{R}^q$ and the inequalities are interpreted component-wise . It is assumed that the solution of (1) is forward complete.

A. Control Barrier Functions

Definition 1 (Class $\mathcal{K}$ function): A continuous function $\alpha : [0,a) \to [0,\infty], a > 0$ is said to belong to class $\mathcal{K}$ if it is strictly increasing and $\alpha(0) = 0$.

Definition 2: A set C is forward invariant for system (1) if for every $\boldsymbol{x}(0) \in C$, we have $\boldsymbol{x}(t) \in C$, for all $t \geq 0$.

Definition 3 (Control barrier function [12]): Given a continuously differentiable function $b : \mathbb{R}^n \to \mathbb{R}$ and the set $C := \{\boldsymbol{x} \in \mathbb{R}^n : b(\boldsymbol{x}) \geq 0\}$, $b(\boldsymbol{x})$ is a candidate control barrier function (CBF) for the system (1) if there exists a class $\mathcal{K}$ function α such that

$$\sup_{\boldsymbol{u}\in\mathcal{U}} \left[L_f b(\boldsymbol{x}) + L_g b(\boldsymbol{x})\boldsymbol{u} + \alpha(b(\boldsymbol{x}))\right] \geq 0, \tag{3}$$

for all $\boldsymbol{x} \in C$, where L_f, L_g denote the Lie derivatives along f and g, respectively.

Definition 4 (Relative degree): The relative degree of a (sufficiently many times) differentiable function $b:\mathbb{R}^n \to \mathbb{R}$ with respect to system (1) is the number of times it needs to be differentiated along its dynamics until the control $\boldsymbol{u}$ explicitly appears in the corresponding derivative.

For a constraint $b(\boldsymbol{x}) \geq 0$ with relative degree $m,\ b:\mathbb{R}^n \to \mathbb{R}$, and $\zeta_0(\boldsymbol{x}) := b(\boldsymbol{x})$, we define a sequence of functions $\zeta_i : \mathbb{R}^n \to \mathbb{R}, i \in \{1,\ldots,m\}$:

$$\zeta_i(\boldsymbol{x}) := \dot{\zeta}_{i-1}(\boldsymbol{x}) + \alpha_i\left(\zeta_{i-1}(\boldsymbol{x})\right),\ i \in \{1,\ldots,m\}, \tag{4}$$

where $\alpha_i(\cdot),\ i \in \{1,\ldots,m\}$ denotes a $(m-i)^{\text{th}}$ order differentiable class $\mathcal{K}$ function. We further define a sequence of sets $C_i,\ i \in \{1,...,m\}$ associated with (4) which take the following form,

$$C_i := \{\boldsymbol{x} \in \mathbb{R}^n : \zeta_{i-1}(\boldsymbol{x}) \geq 0\},\ i \in \{1,...,m\}. \tag{5}$$

Definition 5 (High Order CBF (HOCBF) [13]): Let $C_1,...,C_m$ be defined by (5) and $\zeta_1(\boldsymbol{x}),...,\zeta_m(\boldsymbol{x})$ be defined by (4). A function $b : \mathbb{R}^n \to \mathbb{R}$ is a High Order Control Barrier Function (HOCBF) of relative degree m for system (1) if there exists $(m-i)^{th}$ order differentiable class $\mathcal{K}$ functions $\alpha_i,\ i \in \{1,...,m-1\}$ and a class $\mathcal{K}$ function α_m such that

$$\sup_{\boldsymbol{u}\in\mathcal{U}}[L_f^m b(\boldsymbol{x}) + L_g L_f^{m-1} b(\boldsymbol{x})\boldsymbol{u} + S(b(\boldsymbol{x})) + \alpha_m(\zeta_{m-1}(\boldsymbol{x}))] \geq 0 \tag{6}$$

for all $\boldsymbol{x} \in \bigcap_{i=1}^m C_i$. In (6), L_f^m and L_g denotes derivative along f and g m times and one time respectively, and $S(\cdot)$ denotes the remaining Lie derivative along f with degree less than or equal to $m-1$ (ommited for simplicty, see [13]).

The following theorem on HOCBFs implies the forward invariance property of the CBFs and the original safety set. The proof is omitted (see [13] for the proof).

Theorem 1 ([12]): Given a constraint $b(\boldsymbol{x}(t))$ with the associated sets C_i's as defined in (5), any Lipschitz continuous controller $\boldsymbol{u}(t)$, that satisfies (6) $\forall t \geq t_0$ renders the sets C_i (including the set corresponding to the actual safety constraint C_1) forward invariant for control system (1).

Definition 6 (Control Lyapunov function (CLF)[14]): A continuously differentiable function $V : \mathbb{R}^n \to \mathbb{R}$ is a globally and exponentially stabilizing CLF for (1) if there exists constants $c_i \in \mathbb{R}_{>0},\ i = 1,2,$ such that $c_1||x||^2 \leq V(x) \leq c_2||x||^2$, and the following inequality holds

$$\inf_{\boldsymbol{u}\in\mathcal{U}}[L_f V(\boldsymbol{x}) + L_g V(\boldsymbol{x})u + \eta(\boldsymbol{x})] \leq e, \tag{7}$$

where e makes this a soft constraint.

III. Problem Formulation

We consider the control of a safety-critical system. We assume that the system has the affine dynamics in (1) with control bounds as in (2). The system has a certain high-level performance specification associated with it, as reflected in the cost function $l(\boldsymbol{x},\boldsymbol{u})$. In addition to the cost, we have a set of safety constraints $b_j(\boldsymbol{x}) \geq 0,\ j \in \Lambda_S$ (where b_j is continuously differentiable, Λ_S is a constraint set). We define the stage cost associated to the system in (1) as:

$$L(\boldsymbol{x},\boldsymbol{u}) = l(\boldsymbol{x},\boldsymbol{u}) + \sum_{j\in\Lambda_S} \mathcal{I}_\infty(b_j(\boldsymbol{x})), \tag{8}$$

where we use the indicator function

$$\mathcal{I}_\infty(x) = \begin{cases} \infty, & \text{if } x < 0 \text{ for some } i, \\ 0, & \text{otherwise.} \end{cases}$$

to ensure the satisfaction of each $b_j(\boldsymbol{x}) \geq 0$. Finally, we consider a parameterized state feedback controller $\boldsymbol{u}(\boldsymbol{x}|\boldsymbol{\theta})$; our goal is to learn the parameters $\boldsymbol{\theta}$ that achieve:

- Minimize the expected sum of discounted costs:

$$\boldsymbol{\theta}^*_{0,\ldots,k,\ldots} = \underset{\boldsymbol{\theta}_{0,\ldots,k,\ldots}}{\text{argmin}}\ \mathbb{E}_{\tau_{\boldsymbol{u}}}\Big[\sum_{k=0}^{\infty} \gamma^k L(\boldsymbol{x}_k, \boldsymbol{u}_k(\boldsymbol{x}_k|\boldsymbol{\theta}_k))\Big], \tag{9}$$

  where, $\tau_{\boldsymbol{u}}$ are trajectories sampled based on the dynamics in (1) given an initial state distribution at $k = 0$. Here, γ is the discount factor which yields a well-posed problem, i.e., the state and state-action value functions, are well posed and finite over some sets.
- Guarantee the satisfaction of the constraints $b_j(\boldsymbol{x}_k),\ \forall k,\ j \in \Lambda_S$, and control bounds in (2).

Given the objective in (9), we will first present the formulation of the parameterized state feedback controller, which in our case is obtained using MPC-CBF, followed by the determination of the parameters in (9) learnt using RL.

IV. Parameterized MPC-CBF control Design

In this section, we present the formulation of the parameterized controller $\boldsymbol{u}(\boldsymbol{x}|\boldsymbol{\theta})$ (cf. Section III). Note that we exclude the temporal dependency to streamline the notation.

MPC objective: We adopt a Receding Horizon Control (RHC) approach using MPC due to its look-ahead prediction ability. The stage cost in (8) has two parts: a cost term and a penalty term associated with safety constraints. We encode the safety constraints as hard constraints in the MPC formulation. The cost $l(\boldsymbol{x}, \boldsymbol{u})$ in (8) can be directly incorporated into the objective function. Note that the cost $l(\boldsymbol{x}, \boldsymbol{u})$ is usually selected arbitrarily to reflect some high-level system specification/objective. Consequently, it can be highly nonlinear and/or not differentiable, making it computationally intractable. Hence, we select a continuous quadratic function of states $\boldsymbol{x}$ and control input $\boldsymbol{u}$ as the stage cost in our MPC objective and parameterize it to optimize (9). We express the parameterized MPC stage cost as follows:

$$J(\boldsymbol{x}, \boldsymbol{u}|\boldsymbol{\theta}_o) = \begin{bmatrix} \boldsymbol{x}^T & \boldsymbol{u}^T \end{bmatrix} A(\boldsymbol{\theta}_o) \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{u} \end{bmatrix} + B(\boldsymbol{\theta}_o)^T \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{u} \end{bmatrix}, \quad (10)$$

where $\boldsymbol{\theta}_o$ is the vector of the learnable parameters of the objective. Without loss of generality, $\boldsymbol{\theta}_o$ may consist of weights for state and control components in (10)).

MPC Constraints: We deal with the second term in (8) by enforcing hard HOCBF constraints in the MPC. As mentioned in Section II, HOCBFs, because of their forward invariance property, provide a sufficient condition for the satisfaction of the original constraints. In addition, as seen in (6), the transformed HOCBF constraints include class $\mathcal{K}$ functions which can be parameterized to quantify the trade-off between safety and performance. In particular, if the slope of the function is too small, then the controller may be overly conservative resulting in suboptimal performance. Conversely, a steep class $\mathcal{K}$ function in (6) will allow for optimization of the system performance to a greater extent as the set of feasible inputs is larger and trajectories are allowed to approach the boundary of the safety set at a higher rate. However, such a HOCBF constraint (6) becomes active only near the unsafe set boundary, requiring a large control input effort which may cause infeasibility due to the control bound in (2) (see also [6]). To balance this trade-off, we make the HOCBFs adaptive by parameterizing the class $\mathcal{K}$ function in the HOCBF constraints. Thus, the HOCBF in (6) becomes:

$$\begin{aligned} &L_f^m b_j(\boldsymbol{x}) + L_g L_f^{m-1} b_j(\boldsymbol{x})\boldsymbol{u} + S\left(b_j(\boldsymbol{x}) \mid \boldsymbol{\theta}_c\right) \\ &+ \alpha_m\left(\zeta_{m-1}(\boldsymbol{x}) \mid \boldsymbol{\theta}_c\right) \geq 0, \quad \forall j \in \Lambda_S, \end{aligned} \quad (11)$$

where $\boldsymbol{\theta}_c$ are the learnable parameters of the HOCBF constraint. For example, a linear parametric class $\mathcal{K}$ function is of the form: $\theta_c^{(1)} b_j(\boldsymbol{x})$, where $\theta_c^{(1)} \in \mathbb{R}_{>0}$ is the parameter to be learnt. It is important to note that parameterizing HOCBF constraints does not affect the forward invariance property

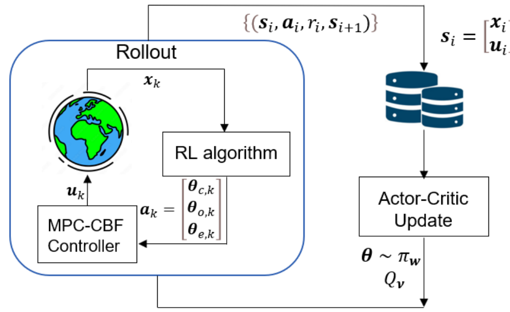


Fig. 1. RL training pipeline for parameterized MPC-CBF. The RL agent learns the parameters $[\boldsymbol{\theta}_{c,k} \ \boldsymbol{\theta}_{o,k} \ \boldsymbol{\theta}_{e,k}]^T$ where $\boldsymbol{\theta}_o$ is the vector of the learnable parameters of the objective, $\boldsymbol{\theta}_c$ are learnable parameters of the CLF constraint and $\boldsymbol{\theta}_e$ is the vector of weights of the penalty terms associated with the relaxation parameters of the CLF constraints. These parameters are then used in the MPC-CBF problem in (13) which is optimized to compute the optimal control input.

of the HOCBFs. In order to include penalty terms related to reference/set-point tracking, we define Lyapunov functions and include the corresponding CLF constraints in the MPC, as in prior works (e.g., [5]). Additionally, we parameterize the CLF constraints to also make them adaptive based on the stage cost in (8) to optimize (9). A parameterized CLF constraint is expressed as follows:

$$L_f V(\boldsymbol{x}) + L_g V(\boldsymbol{x}) u + \eta(\boldsymbol{x}|\boldsymbol{\theta}_c) \leq e, \quad (12)$$

where $\boldsymbol{\theta}_c$ are learnable parameters of the CLF constraint, and e is a relaxation parameter for the CLF constraint which is included to make the constraint soft as it is not a safety (critical) constraint.

Parameterized MPC-CBF approach: Finally, the parametrized MPC-CBF problem can be written as follows:

$$\begin{aligned} \min_{\substack{\boldsymbol{x}_{0:N|k} \\ \boldsymbol{u}_{0:N-1|k}}} \ & \sum_{h=0}^{N-1} \left(J(\boldsymbol{x}_{h|k}, \boldsymbol{u}_{h|k}|\boldsymbol{\theta}_{o,k}) + \boldsymbol{\theta}_{e,k}^T \boldsymbol{e}_{h|k}^2 \right) \\ & + V_N(\boldsymbol{x}_{N|k}|\boldsymbol{\theta}_{o,k}), \end{aligned} \quad (13)$$

$$\begin{aligned} \text{subject to} \quad & \boldsymbol{x}_{h+1|k} = f(\boldsymbol{x}_{h|k}, \boldsymbol{u}_{h|k}), && h = 0, \ldots, N-1, \\ & (2), (11), (12), && h = 0, \ldots, N-1, \\ & \boldsymbol{x}_{h|k} \in \mathcal{X}, x_N \in \mathcal{X}_f. \end{aligned}$$

where $\boldsymbol{\theta}_e$ is the vector of weights of the penalty terms associated with the relaxation parameters of the CLF constraints. The CBF (or HOCBF) constraints correspond to the safety constraints $b_j(\boldsymbol{x})$, $\forall j \in \Lambda_S$.

RL algorithm: We denote the learnable parameters of the MPC-CBF as $\boldsymbol{\theta}_k = [\boldsymbol{\theta}_{c,k} \ \boldsymbol{\theta}_{o,k} \ \boldsymbol{\theta}_{e,k}]^T$, where $\boldsymbol{\theta}_k \in \mathbb{R}_{>0}^{|\boldsymbol{\theta}|}$. To learn the optimal parameters corresponding to (9), we set the state space $\mathcal{S} = \mathcal{X} \times \mathcal{U}$ as defined in Section II and the action space $\mathcal{A} = \mathbb{R}_{>0}^{|\boldsymbol{\theta}|}$. We define the reward $\mathcal{R}(\boldsymbol{s}_k) = -L(\boldsymbol{s}_k)$, with $L(\boldsymbol{s}_k)$ in (8). Finally, we use RL to learn the optimal parameters $\boldsymbol{\theta}_k(\boldsymbol{s}_k)$ which change at every time step k as a function of the state $\boldsymbol{s}_k$. Specifically, we parameterize the optimal controller parameter policy using a neural network with parameters $\boldsymbol{w}$ and train it in order to maximize the expected discounted reward $\mathcal{R}(\boldsymbol{s}_k)$ (i.e., solve (9)). This step can be implemented using any actor-critic RL algorithms

[15], [16]. The optimal parameters $\boldsymbol{\theta}_k$ are then sampled from the learnt policy, i.e., $\boldsymbol{\theta}_k \sim \pi_{\boldsymbol{w}}(\boldsymbol{s}_k)$. The overall pipeline of RL with MPC-CBF is shown in Fig. 1. Note that modern Deep RL actor-critic algorithms parameterize both the actor $\pi_{\boldsymbol{w}}$ and the critic $Q_{\boldsymbol{\nu}}$ ($\boldsymbol{\nu}$ is a vector of the parameters of the critic network, which can be implemented using a neural network). The critic is updated regressing the Bellman equation as in [17] and the actor is then updated by means of policy gradient which exploits the reparameterization trick [15], [16].

V. Multi-Agent Control of CAVs

In this section, we review the setting for CAVs whose motions are coordinated at conflict areas of a traffic network. We define a Control Zone (CZ) to be an area within which CAVs can communicate with each other or with a coordinator (e.g., a Road-Side Unit (RSU)) which is responsible for facilitating the exchange of information (but not control individual vehicles) inside this CZ. As an example, Fig. 2 shows a conflict area due to vehicles merging from two

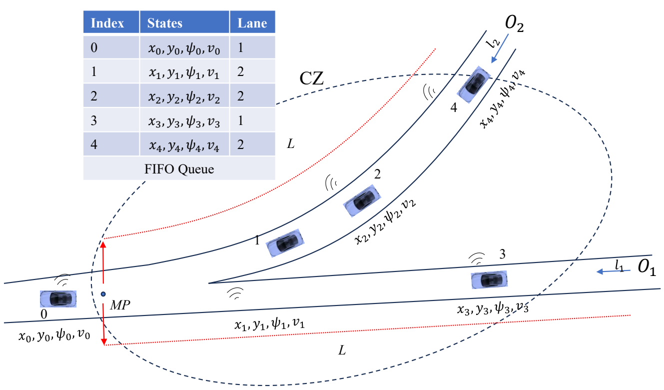


Fig. 2. The merging control problem for CAVs

Let $F(t)$ be the set of indices of all CAVs located in the CZ at time t. A CAV enters the CZ at one of several origins (e.g., O_1 and O_2 in Fig. 2) and leaves at one of possibly several exit points (e.g., MP in Fig. 2). The index 0 is used to denote a CAV that has just left the CZ. Let $S(t)$ be the cardinality of $F(t)$. Thus, if a CAV arrives at time t, it is assigned the index $S(t)+1$. All CAV indices in $F(t)$ decrease by one when a CAV passes over the MP and the vehicle whose index is -1 is dropped.

The vehicle dynamics for each CAV $i \in F(t)$ along its lane in a given CZ are assumed to be of the form

$$\underbrace{\begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\psi}_i \\ \dot{v}_i \end{bmatrix}}_{\dot{\boldsymbol{x}}_i(t)} = \underbrace{\begin{bmatrix} v_i \cos\psi_i \\ v_i \sin\psi_i \\ 0 \\ 0 \end{bmatrix}}_{f(\boldsymbol{x}_i(t))} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & v_i/(l_f+l_r) \\ 1 & 0 \end{bmatrix}}_{g(\boldsymbol{x}_i(t))} \underbrace{\begin{bmatrix} u_i \\ \phi_i \end{bmatrix}}_{\boldsymbol{u}_i(t)}, \quad (14)$$

where $x_i(t), y_i(t), \psi_i(t), v_i(t)$ represent the current longitudinal position, lateral position, heading angle, and speed, respectively. $u_i(t)$ and $\phi_i(t)$ are the acceleration and steering angle (controls) of vehicle i at time t, respectively, $g(x_i(t)) = [g_u(\boldsymbol{x}_i(t)), g_\phi(\boldsymbol{x}_i(t))]$. There are three objectives for each CAV, which are minimization of the travel time, energy consumption and center lane deviation respectively, further details can be found in [18].

Constraint 1 (Safety constraint): Let i_p denote the index of the CAV which physically immediately precedes i in the CZ (if one is present). We require that the distance between vehicles i and i_p remains safe by defining a speed dependent ellipsoidal safe region $b_1(\boldsymbol{x}_i, \boldsymbol{x}_{i_p})$ as follows:

$$b_1(\boldsymbol{x}_i, \boldsymbol{x}_{i_p}) := \frac{(x_i(t) - x_{i_p}(t))^2}{(a_i v_i(t))^2} + \frac{(y_i(t) - y_{i_p}(t))^2}{(b_i v_i(t))^2} - 1 \geq 0, \quad (15)$$

where a_i, b_i are weights adjusting the length of the major and minor axes of the ellipse.

Constraint 2 (Safe merging): Each CAV i has to maintain safe distancing relative to the CAV that merges immediately ahead of it, denoted by i_c. This is ensured by enforcing a constraint on CAV i denoted by $b_2(\boldsymbol{x}_i, \boldsymbol{x}_{i_c})$ at the merging point m at time t_i^m when it crosses the MP and it is defined as follows:

$$b_2 = r_i(t)\mu_i(t) - r_{i_c}(t)\mu_{i_c}(t) - \Phi(x_i(t), y_i(t))\, v_i(t) - \delta \geq 0, \quad (16)$$

where, $r_i = \sqrt{x_i^2 + y_i^2}$, $\mu_i = \tan^{-1}\left(\frac{y_i}{x_i}\right)$; similarly, $r_{i_c} = \sqrt{x_{i_c}^2 + y_{i_c}^2}, \mu_{i_c} = \tan^{-1}\left(\frac{y_{i_c}}{x_{i_c}}\right)$. Here $\Phi : \mathbb{R}^2 \to \mathbb{R}$ may be any continuously differentiable function as long as it is strictly increasing and satisfies the boundary conditions $\Phi(x_i(t_i^0), y_i(t_i^0)) = 0$ and $\Phi(x_i(t_i^m), y_i(t_i^m)) = \varphi$ (see [19] for details). Here φ denotes the reaction time and δ is a given minimum safe distance.

The determination of CAV i_c depends on the policy adopted for sequencing CAVs through the CZ, which are detailed in [18].

Constraint 3 (Vehicle limitations): Constraints on the speed and control for each $i \in F(t)$ are detailed in [18].

Constraint 4 (Road boundaries): To ensure the vehicle stays within the road boundaries we impose two constraints defined as b_{rc} and b_{lc} for the right and left boundary of the road respectively, as follows:

$$b_3 = (x_i(t) - x_{i,lc}(t))^2 + (y_i(t) - y_{i,lc}(t))^2 - r_{lc}^2 \geq 0, \quad (17)$$

$$b_4 = (x_i(t) - x_{i,rc}(t))^2 + (y_i(t) - y_{i,rc}(t))^2 - r_{rc}^2 \leq 0, \quad (18)$$

where $x_{i,lc}(t)$ and $x_{i,rc}(t)$ are the longitudinal coordinates and $y_{i,lc}(t)$ and $y_{i,rc}(t)$ are the lateral coordinates of the left and right boundary respectively for vehicle i at time t and $r_{rc} \in \mathbb{R}_{>0}$, $r_{lc} \in \mathbb{R}_{>0}$ are constants. The derivation of the (HO)CBF constrains can be found in [18]. The optimization problem details can be found in [18].

VI. Simulation Results

The details of the implementation can be found in [18].

MPC-CBF and model parameters: We have considered the merging problem shown in Fig. 2 where CAVs arrive at

TABLE I

CAVs METRIC COMPARISON UNDER THE PROPOSED FRAMEWORK AND THE BASELINE. THE MEAN AND STANDARD DEVIATION WERE CALCULATED FROM 10 SIMULATIONS WITH DIFFERENT RANDOM INITIAL STATES AND ARRIVAL TIMES

Item	MPC-CBF w/o RL				MPC-CBF w/ RL
	Conservative	Moderately conservative	Moderately aggressive	Aggressive	-
Ave. travel time	11.98 ±0.1	11.87 ±0.11	11.82 ±0.1	11.78 ±0.1	**10.94** ± 0.11
Ave. $\frac{1}{2}u^2$	12.19 ±1.23	11.68 ±1.29	12.55±1.62	12.83 ±1.75	**8.70** ± 1.16
Ave. fuel consumption	10.52 ±0.33	10.84 ±0.29	11.35 ±0.3	11.14 ±0.31	**7.20** ±0.27
Total infeasibility	259	185	199	209	**71**

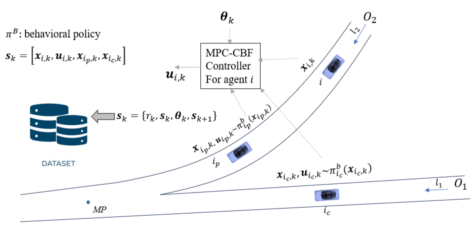


Fig. 3. Illustration of the scenario used to generate rollouts during RL training.

the predefined CZ according to Poisson arrival processes with a given arrival rate. The parameters used in the implementation cam be found in [18]. In our simulations, we included the computation of a more realistic energy consumption model [20].

RL algorithm and hyperparameters: Each CAV/agent at most interacts with two other CAVs (i_p and i_c) given in (15), (16). Therefore, we train an agent i in the presence of two other agents (i_p and i_c) and sample their control input from two behavior policies $\pi^B_{i_c}$ and $\pi^B_{i_p}$ respectively to ensure sufficient exploration and generalization across all CAVs. Fig. 3 illustrates the scenario used to generate the rollouts during training, further details can be found in [18].

Furthermore, this allows us to define a reward function that depends on CAV i only as follows:

$$\mathcal{R}_i = -L_i = -(\beta_1 ||u_i||_2^2 + \beta_2 ||\phi_i||^2 + \beta_3 ||v_i - v_{des}||_2^2 + \beta_4 ||\psi_i - \psi_{des}||_2^2 + \beta_5 f_v + \mathcal{I}_\infty), \quad (19)$$

where $\beta_q \in [0,1]$, $\forall q$ and $\mathcal{I}_\infty$, denotes the infeasiblity penalty which is set to be much higher than other terms in the reward function. The actor network output is fed into the optimization problem formulated in (13) which then outputs the control inputs for the agents. We train our agent using a soft actor-critic [16].

Numerical Results: We present simulation results for 50 random initialized vehicles with two sets of parameters θ_k: the first is our proposed method with a time-varying class $\mathcal{K}$ function and objective weights, while the second is a fixed class $\mathcal{K}$ function (linear function) and fixed objective weights heuristically (commonly used in literature) as the baseline. To ensure we pick both conservative and aggressive parameter values in the baseline, the whole parameter range is divided into four sets (i.e., conservative to aggressive).

The performance of both methods in terms of average travel time, ℓ^2-norm of the acceleration, average fuel consumption and total number of infeasible cases (i.e., the number of times there is no solution for the minimization problem formulated in (13)) are compared and illustrated in Table I. According to the results, the proposed method not only improves the desired metrics defined in the reward function (19), but also reduces the infeasible instances by almost 65%. It is worth noting that an "infeasible" case *does not necessarily imply a constraint violation*, since CBF constraints are only sufficient, but not necessary, conditions.

A similar scenario as depicted in Fig. 2 is used in two experiments, one in which we assign fixed values to the set of parameters, θ_k, and another using our proposed method whose results are shown in Figs. 4 and 5, respectively. As it can be seen from Fig. 4a, the CAV 4 trajectory encounters an infeasiblity (i.e., the obtained control input violated its bounds as shown in Fig. 4b). This infeasiblity happens due to the conservativeness of the fixed class $\mathcal{K}$ function which then violates the HOCBF constraints defined in (18) as shown in Fig. 4c. Similarly, CAV 3 encounters an infeasiblity and fails to satisfy CBF constraint of safe merging defined in (16) as shown in Fig. 4c. However, with our proposed method, this issue is eliminated as shown in Fig. 5a.

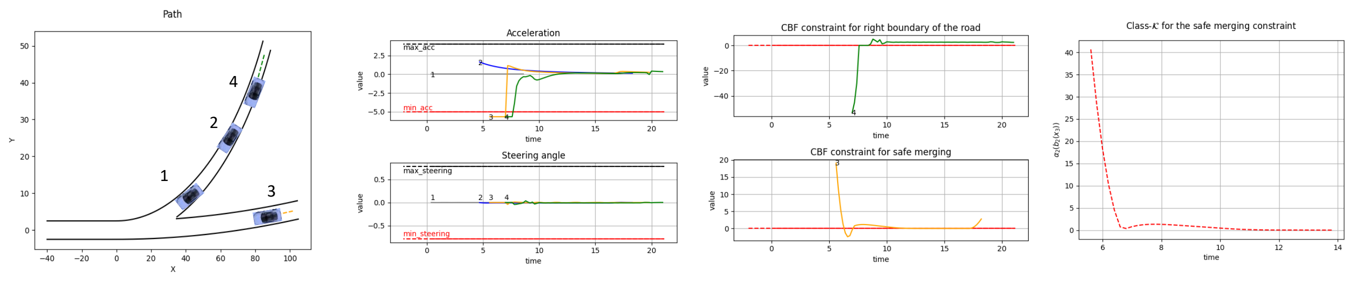
VII. CONCLUSION

We have proposed a control method based on RHC using MPC with CBFs which can provably guarantee safety in safety-critical control systems. In order to tackle the issues of performance as well as feasibility, we use RL to learn the parameters of our MPC-CBF controller by optimizing system performance encapsulated in properly defined rewards/costs. We have validated our approach on a multi-agent CAV merging control problem at conflicting roadways. As the proposed approach involves a single-agent, we use RL to train the controller for one CAV and use it across other agents assuming they are homogeneous to evaluate the generalizeability of the proposed approach. Experimental results show that our proposed approach performs better compared to the baseline controller with heuristically selected parameters and also generalizes to unseen scenarios.

REFERENCES

[1] A. D Ames, J. W Grizzle, and P. Tabuada. Control barrier function based quadratic programs with application to adaptive cruise control. In *53rd IEEE Conf. on Decision and Control*, pages 6271–6278. IEEE, 2014.

[2] E. Sabouni, H.M. S. Ahmad, W. Xiao, C. G. Cassandras, and W. Li. Optimal control of connected automated vehicles with event-triggered control barrier functions: a test bed for safe optimal merging. In *2023 IEEE Conf. on Control Technology and Applications (CCTA)*, pages 321–326, 2023.



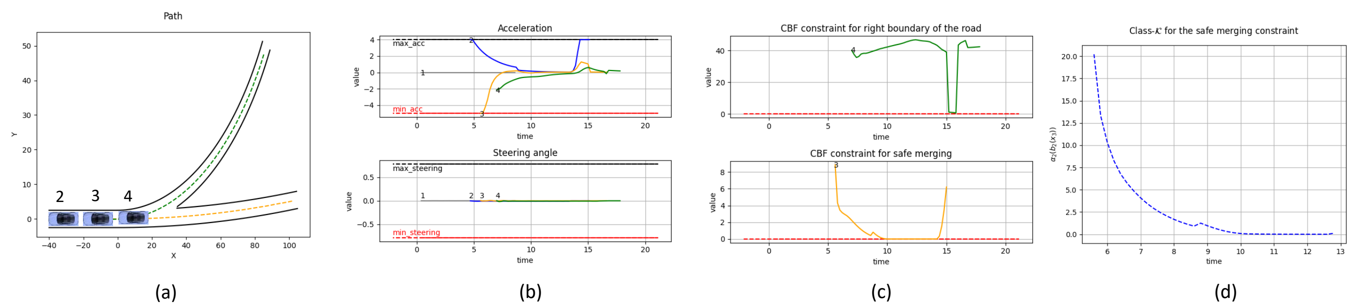


Fig. 5. Simulation results of the scenario depicted in Fig 2 with our proposed approach. **(a)**: A screenshot of the simulation at the point where vehicles 3 and 4 successfully finished their paths without encountering any infeasiblity, indicated by yellow and green dashes. **(b)**: All vehicle acceleration and steering angles profiles are within the bounds. **(c)**: CBF constraint values of right boundary of the road and safe merging constraints of vehicles 4 and 3, respectively. In contrast to Fig 4, the values in this plot are all above zero. **(d)**: Evolution of the class $\mathcal{K}$ function $\alpha_2(b_2(\boldsymbol{x}))$ values for the safe merging constraint of CAV 3.

[3] J. Zeng, B. Zhang, and K. Sreenath. Safety-critical model predictive control with discrete-time control barrier function. In *2021 American Control Conference (ACC)*, pages 3882–3889, 2021.

[4] J. Zeng, Z. Li, and K. Sreenath. Enhancing feasibility and safety of nonlinear model predictive control with discrete-time control barrier functions. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 6137–6144, 2021.

[5] W. Xiao, C. Belta, and C. G. Cassandras. Adaptive control barrier functions. *IEEE Trans. on Automatic Control*, 67(5):2267–2281, 2022.

[6] W. Xiao, T. Wang, R. Hasani, M. Chahine, A. Amini, X. Li, and D. Rus. Barriernet: Differentiable control barrier functions for learning of safe robot control. *IEEE Transactions on Robotics*, 39(3):2289–2307, 2023.

[7] D. A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 1. Morgan-Kaufmann, 1988.

[8] S. Ross and D. Bagnell. Efficient reductions for imitation learning. In Yee Whye Teh and Mike Titterington, editors, *Proc. of the Thirteenth International Conf. on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 661–668, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR.

[9] M. Zanon and S. Gros. Safe reinforcement learning using robust mpc. *IEEE Transactions on Automatic Control*, 66(8):3638–3652, 2021.

[10] A. Romero, Y. Song, and D. Scaramuzza. Actor-critic model predictive control, 2024.

[11] H.M. Sabbir Ahmad, Ehsan Sabouni, Wei Xiao, Christos G. Cassandras, and Wenchao Li. Trust-aware resilient control and coordination of connected and automated vehicles. In *2023 ITSC*, pages 4309–4314, 2023.

[12] Aaron D. Ames, Kevin Galloway, and J. W. Grizzle. Control lyapunov functions and hybrid zero dynamics. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 6837–6842, 2012.

[13] W. Xiao and C. Belta. High-order control barrier functions. *IEEE Transactions on Automatic Control*, 67(7):3655–3662, 2022.

[14] A. D Ames, K. Galloway, and J. W Grizzle. Control lyapunov functions and hybrid zero dynamics. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 6837–6842. IEEE, 2012.

[15] T. P Lillicrap, J. J Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *US Patent*, 15(217,758), 2020.

[16] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[17] V. Mnih, D. Kavukcuoglu, K.and Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[18] Ehsan Sabouni, H. M. Sabbir Ahmad, Vittorio Giammarino, Christos G. Cassandras, Ioannis Ch. Paschalidis, and Wenchao Li. Reinforcement learning-based receding horizon control using adaptive control barrier functions for safety-critical systems. https://doi.org/10.48550/arXiv.2403.17338, 2024.

[19] Ehsan Sabouni, H. M. Sabbir Ahmad, Christos G. Cassandras, and Wenchao Li. Merging control in mixed traffic with safety guarantees: A safe sequencing policy with optimal motion control. In *2023 ITSC*, pages 4260–4265, 2023.

[20] M. A. S. Kamal, M. Mukai, J. Murata, and T. Kawabe. Model predictive control of vehicles on urban roads for improved fuel economy. *IEEE Trans. on Control Systems Technology*, 21(3):831–841, 2012.