Learning Control Barrier Functions with High Relative Degree for Safety-Critical Control

Chuanzheng Wang, Yiming Meng, Yinan Li, Stephen L. Smith, Jun Liu

Abstract— **Control barrier functions have shown great success in addressing control problems with safety guarantees. These methods usually find the next safe control input by solving an online quadratic programming problem. However, model uncertainty is a big challenge in synthesizing controllers. This may lead to the generation of unsafe control actions, resulting in severe consequences. In this paper, we develop a learning framework to deal with system uncertainty. Our method mainly focuses on learning the dynamics of the control barrier function, especially for high relative degree with respect to a system. We show that for each order, the time derivative of the control barrier function can be separated into the time derivative of the nominal control barrier function and a remainder. This implies that we can use a neural network to learn the remainder so that we can approximate the dynamics of the real control barrier function. We show by simulation that our method can generate safe trajectories under parametric uncertainty using a differential drive robot model.**

I. INTRODUCTION

A. Background and Literature Review

In many applications, one must solve a control problem that requires not only achieving control objectives, but also providing control actions with guaranteed safety [10]. In practice, this is of great importance and it is necessary to incorporate safety criteria while designing controllers. For example, industrial robotics, medical robots as well as self-driving vehicles are all areas where safe controllers are critical. The notion of safety control was first proposed in [13] in the form of correctness and was then formalized in [1], in which the authors stated that a safety property stipulates that some "bad thing" does not happen during execution.

More recently, control barrier functions (CBFs) are widely used to deal with safety control [2]. Barrier functions are Lyapunov-like functions which were initially used in optimization problems [5]. CBFs are combined with control Lyapunov functions as constraints of quadratic programming (QP) problems in [3] and the authors show that safety criteria can be converted into a linear constraint of the QP problem for control inputs. By solving the QP problems, we can find the next action so that safety is guaranteed during execution. It is shown in [17] that finding safe control inputs by solving QP problems can be extended to an arbitrary number of constraints and any nominal control law. As a result, CBFs are widely used in safety control such as lane keeping [4] and obstacle avoidance [6]. However, using CBFs in the QP problems means that the first order derivative of the CBFs should depend on the control input and as a result, this usually violates with many robot systems such as bipedal or car-like robots [11]. Consequently, CBFs are extended to handle position-based constraints for relative degree of two [23]. The authors in [16] propose a way of designing exponential control barrier functions (ECBFs) using input-output linearization to handle CBFs with higher relative degree. Safe control actions are calculated for quadrotors using ECBF in [21]. Furthermore, a more general form of higher-order control barrier functions (HOCBFs) is introduced in [24].

In practice models used to design controllers are imperfect because of disturbance or parametric uncertainty. This uncertainty may lead to unsafe or even dangerous behavior, and thus it is of great importance that we synthesize controllers to handle model uncertainty. Learning-based approaches have shown great promise in controlling systems with uncertainty [12]. Several methods using data-driven approaches have been utilized in this area. Training data is collected to learn the real dynamics for the design of more accurate controllers. In [26], the HOCBF under external disturbance is proposed and imitation learning is used to obtain a feedback controller. Gaussian process (GP) is used to approximate the model as in [9]. The authors in [7] also use Gaussian process to estimate the model but focusing on the safety during the training process. A reinforcement learning (RL) based method to learn the model uncertainty compensation for input-output linearization control is introduced in [22] and a RL-based framework for policy improvement is proposed in [8] as well. However, both methods do not rely on a nominal controller and using nominal controllers are more flexible because they can be replaced by any other reliable controllers in practice. Our work is mostly close to [19], in which the dynamics of the CBF of real model is learned based on the dynamics of the CBF for the nominal model. However, the main difference between our work and [19] is that we focus on learning CBFs with higher relative degree with respect to more complex systems. Besides, we also provide sufficient conditions on controllers via CBFs with high relative degree for set invariance.

B. Contribution

In this paper, we propose a learning framework for CBFs with high relative degree. We consider a machine learning method to reduce model uncertainty using supervised regres-

Chuanzheng Wang, Yiming Meng, Yinan Li, and Jun Liu are with the Department of Applied Mathematics, University of Waterloo, Waterloo, Ontario, Canada, {cz.wang, yiming.meng, yinan.li, j.liu}@uwaterloo.ca

Stephen L. Smith is with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario, Canada, stephen.smith@uwaterloo.ca

sion. Safe trajectories are generated using the learned CBFs. It is shown that the dynamics of the real CBF can be learned based on the nominal CBF. As a result, the main contribution of our work is summarized as below:

1 We propose a learning framework for CBFs with high relative degree for safety-critical control.
2 We provide sufficient conditions on controllers via CBFs with high relative degree for set invariance.
3 We show theoretically that for each order of time derivative, the dynamics of the real CBF can be separated into two terms: the time derivative of the nominal CBF and a remainder that is independent of the control input.
4 We use supervised regression to learn the remainder so that the dynamics of the real CBF can be accurately approximated.
5 We validate our method in simulation using a differential drive model under system uncertainty with static obstacles, multiple obstacles and moving obstacles.

II. Preliminary and Problem Definition

A. Model and Uncertainty

Throughout the paper, we consider a SISO nonlinear control affine model

$$\begin{aligned} \dot{x} &= f(x) + g(x)u, \\ y &= h(x), \end{aligned} \tag{1}$$

such that $f : \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^n$ are locally Lipschitz, $x \in \mathbb{R}^n$ is the state and $u \in \mathbb{R}$ is the control and $h : \mathbb{R}^n \to \mathbb{R}$ is a r^{th}-order continuously differentiable function for some integer r. A solution of system (1) from an initial condition $x_0 \in \mathbb{R}^n$ is denoted by $x(t, x_0)$.

We also consider parametric uncertainty for the model, and as a result, we have a nominal model that estimates the dynamics of Eq (1) as

$$\begin{aligned} \dot{\hat{x}} &= \hat{f}(x) + \hat{g}(x)u, \\ \hat{y} &= \hat{h}(x), \end{aligned} \tag{2}$$

where $\hat{f} : \mathbb{R}^n \to \mathbb{R}^n$ and $\hat{g} : \mathbb{R}^n \to \mathbb{R}^n$ are locally Lipschitz continuous and $\hat{h} : \mathbb{R}^n \to \mathbb{R}$ is an r^{th}-order continuously differentiable function as well.

B. Control Barrier Function

We consider a set $\mathcal{C}$ defined as a superlevel set of a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$ such that

$$\begin{aligned} \mathcal{C} &= \{x \in \mathbb{R}^n : h(x) \geq 0\}, \\ \partial\mathcal{C} &= \{x \in \mathbb{R}^n : h(x) = 0\}, \\ \text{Int}(\mathcal{C}) &= \{x \in \mathbb{R}^n : h(x) > 0\}. \end{aligned} \tag{3}$$

We refer $\mathcal{C}$ as the safe set and safety can be framed in the context of enforcing invariance of $\mathcal{C}$. Due to the local Lipschitz assumption of f and g, for any initial condition x_0, there exists a maximum interval of existence $I(x_0) = [0, \tau_{\text{max}})$ such that $x(t, x_0)$ is the unique solution to (1) on $I(x_0)$. As a result, we can define a set to be forward invariant as below.

Definition 1: Let $h : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function and $\mathcal{C} \subset \mathbb{R}^n$ be a superlevel set of h as defined in Eq (3). The set $\mathcal{C}$ is forward invariant if for every $x_0 \in \mathcal{C}$, $x(t) \in \mathcal{C}$ for all $t \in I(x_0)$, where $x(t)$ is the solution to Eq (1) with $x(0) = x_0$. The system (1) is safe with respect to $\mathcal{C}$ if $\mathcal{C}$ is forward invariant.

We note that an extended $\mathcal{K}_\infty$ function is a function $\alpha : \mathbb{R} \to \mathbb{R}$ that is strictly increasing and $\alpha(0) = 0$. Based on this, we can define the control barrier function as follows.

Definition 2: Let $h : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function and $\mathcal{C} \subset \mathcal{D} \subset \mathbb{R}^n$ be a superlevel set of h as defined in Eq (3). Then h is a control barrier function (CBF) if there exists an extended $\mathcal{K}_\infty$ function α such that for the control system Eq (1),

$$\sup_{u \in \mathbb{R}} [L_f h(x) + L_g h(x) u] \geq -\alpha(h(x))$$

for all $x \in \mathcal{D}$, where $L_f h(x) = f \cdot \frac{\partial h}{\partial x}$ and $L_g h(x) = g \cdot \frac{\partial h}{\partial x}$.

We can then consider the set consisting of all control values that render $\mathcal{C}$ to be safe [2]:

$$K_{\text{cbf}} = \{u(x) \in \mathbb{R} : L_f h(x) + L_g h(x) u + \alpha(h(x)) \geq 0\}.$$

C. Safety-Critical Control

Suppose we are given a feedback controller $u = k(x)$ for the system (1) and we wish to control the system while guaranteeing safety. It may be the case that sometimes the feedback controller $u = k(x)$ is not safe, i.e., there exists some x such that $u(x) \notin K_{\text{cbf}} = \{u(x) \in \mathbb{R} : L_f h(x) + L_g h(x) u + \alpha(h(x)) \geq 0\}$. We can use the following quadratic programming to find the safe control with minimum perturbation [4]:

$$\begin{aligned} u(x) = \underset{u \in \mathbb{R}}{\arg\min}\ & \frac{1}{2}||u - k(x)||^2 \\ \text{s.t.} \quad & L_f h(x) + L_g h(x) u + \alpha(h(x)) \geq 0. \end{aligned} \tag{CBF-QP}$$

D. Relative Degree and Exponential Control Barrier Function

The relative degree of a continuous differentiable function h on a set with respect to a system as in Eq (1) is the number of times we need to differentiate h along the dynamics of the system before the control input u explicitly appears. The formal definition of relative degree is as below.

Definition 3: Given an r^{th}-order continuously differentiable function h, a set D and a system as defined in Eq (1), we say h has a relative degree of r with respect to system Eq (1) on D if $L_g L_f^{r-1} h(x) \neq 0$ and $L_g L_f h(x) = L_g L_f^2 h(x) = \cdots = L_g L_f^{r-2} h(x) = 0$ for $x \in D$, where $L_f^r h(x) = L_f L_f^{r-1} h(x)$.

Remark 1: In this paper, we assume that h has a well-defined relative degree of r with respect to system Eq (1) on a domain D of interest, similar to [25], where the author assumed $D = \mathbb{R}^n$.

The r^{th}-order time-derivative of $h(x)$ is

$$h^r(x) = L_f^r h(x) + L_g L_f^{r-1} h(x) u$$

and $h^r(x)$ is dependent on the control input u. The system is input-output linearizable if $L_g L_f^{r-1} h(x)$ is invertible. For a given control $\mu \in \mathbb{R}$, u can be chosen such that $L_f^r h(x) + L_g L_f^{r-1} h(x) u = \mu$. The control input u renders the input-output dynamics of the system linear. Defining a system with state

$$\eta(x) := \begin{bmatrix} h(x) \\ \dot{h}(x) \\ \vdots \\ h^{r-1}(x) \end{bmatrix} = \begin{bmatrix} h(x) \\ L_f h(x) \\ \vdots \\ L_f^{r-1} h(x) \end{bmatrix},$$

we can then construct a state-transformed linear system

$$\begin{aligned} \dot{\eta}(x) &= F\eta(x) + G\mu, \\ h(x) &= C\eta(x), \end{aligned} \tag{4}$$

where

$$F = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}, \quad G = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix},$$
$$C = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \end{bmatrix}.$$

The exponential control barrier function is defined below as in [16].

Definition 4: Given a r^{th}-order continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$ and a superlevel set $\mathcal{C}$ of h as defined in Eq (3), then h is an exponential control barrier function (ECBF) if there exists a row vector $K = [k_0, k_1, \dots, k_{r-1}]$ such that

$$\sup_{u \in \mathbb{R}} [L_f^r h(x) + L_g L_f^{r-1} h(x) u] \geq -K\eta(x).$$

for any $x \in \mathcal{C}$, where K is chosen such that the transformed system Eq (4) is stable.

Remark 2: It is explained in [16] that the ECBF with $r = 1$ is the same as the CBF as in Definition 2. The design of the ECBF, i.e, the selection of $k_0, k_1, \dots, k_{r-1}$ in K is also explained in [16] using state feedback control and pole placement.

As a result, given an ECBF and a nominal controller $u = k(x)$, we can consider the following quadratic programming problem to enforce the condition in Definition 4 with minimum perturbation

$$\begin{aligned} u(x) = \arg\min_{u \in \mathbb{R}} \frac{1}{2} ||u - k(x)||^2 \qquad & \text{(ECBF-QP)} \\ \text{s.t.} \quad L_f^r h(x) + L_g L_f^{r-1} h(x) u \geq -K\eta(x). & \end{aligned}$$

E. High Order Control Barrier Function and Controlled Set Invariance

Exponential control barrier functions (ECBF) can be seen as a special case of higher order control barrier functions (HOCBF) defined in [24]. In this section, we present some sufficient conditions on using HOCBF for enforcing set invariance. We first define a series of continuously differentiable function $b_0, b_j : \mathbb{R}^n \to \mathbb{R}$ for each $j = 1, 2, \dots, r$ and corresponding superlevel sets $\mathcal{C}_j$ as

$$\begin{aligned} b_0(x) &= h(x), \\ b_j(x) &= \dot{b}_{j-1}(x) + c_j \alpha_j(b_{j-1}(x)), \end{aligned} \tag{5}$$

and

$$\mathcal{C}_j = \{x \in \mathbb{R}^n : b_{j-1}(x) \geq 0\}, \tag{6}$$

where $c_j > 0$ are constants and $\alpha_j(\cdot)$ are differentiable extended class $\mathcal{K}$ functions. We further assume that the interiors of the sets $\mathcal{C}_i$ are given by

$$\text{Int}(\mathcal{C}_i) = \{x \in \mathbb{R}^n : b_{j-1}(x) > 0\}.$$

Definition 5: A continuously differentiable function h is an r^{th}-order control barrier function (HOCBF) for system (1), if there exists extended differentiable class $\mathcal{K}$ functions $\alpha_j(\cdot)$ for $j = 1, 2, ..., r$, such that for $b_j(x)$ defined in Eq (5) with any arbitrary $c_j > 0$ and the corresponding superlevel sets $\mathcal{C}_j$ defined as in Eq (6), the following

$$\sup_{u \in \mathbb{R}} [L_f^r h(x) + L_g L_f^{r-1} h(x) u + \mathcal{O}(h)] \geq -c_r \alpha_r(b_{r-1}(x)) \tag{7}$$

holds for all $x \in \bigcap_{j=1}^r \mathcal{C}_j$, where $\mathcal{O}(h)$ denotes the Lie derivatives of h along f with degree up to $r-1$.

Remark 3: Note that $\mathcal{C}_1$ is uniquely defined, whereas $\mathcal{C}_2, \mathcal{C}_3, ..., \mathcal{C}_r$ is defined based on the choice of $c_1, c_2, ..., c_{r-1}$.

Proposition 1: Consider r^{th}-order HOCBF $h : \mathbb{R}^n \to \mathbb{R}$ with the associated α_j and sets $\mathcal{C}_i$ for $j \in \{1, 2, ..., r\}$. Suppose that h has relative degree r with respect to system (1) on a set D containing $\bigcap_{j=1}^r \mathcal{C}_j$. Then any Lipschitz continuous controller $u(x)$ that satisfies

$$L_f^r h(x) + L_g L_f^{r-1} h(x) u(x) + \mathcal{O}(h) \geq -c_r \alpha_r(b_{r-1}(x)) \tag{8}$$

for all $x \in \bigcap_{j=1}^r \text{Int}(\mathcal{C}_j)$ renders the set $\bigcap_{j=1}^r \text{Int}(\mathcal{C}_j)$ forward invariant. Furthermore, given any functions α_j, $j \in \{1, 2, ..., r\}$, and any compact initial set $X_0 \subset \text{Int}(\mathcal{C}_1)$, there exist appropriate choices of $c_j > 0$ such that $X_0 \subset \bigcap_{j=1}^r \text{Int}(\mathcal{C}_j)$.

Due to space limitations, we omit the proof of this result, but it can be found in [20]. However, the key step in its proof is to establish the following lemma, whose proof is also contained in [20].

Lemma 1: Consider a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$ and dynamics on $\mathbb{R}^n$

$$\dot{x} = f(x) \tag{9}$$

such that $f : \mathbb{R}^n \to \mathbb{R}$ is locally Lipschitz. If the Lie derivative of h along the trajectories of x satisfies

$$\dot{h}(x) \geq -\alpha(h(x)), \ \forall x \in \mathcal{C} \tag{10}$$

where α is a locally Lipschitz extended class $\mathcal{K}$ function, then the set $\text{Int}(\mathcal{C})$ is forward invariant.

Remark 4: Note that the result cannot be extended to the invariance of the set $\mathcal{C}$, despite that it is widely stated so in the literature. A simple counterexample is when $\text{Int}(\mathcal{C}) = \emptyset$, we can define $h(x) = -x^2$ and therefore $\mathcal{C} = \{0\}$. Then for $\dot{x} = c \neq 0$, even though we have a satisfaction of (10) on $\mathcal{C} = \{0\}$, it is not invariant under the flow.

Now assume $\text{Int}(\mathcal{C}) \neq \emptyset$, we also need to necessarily assume the locally Lipschitz continuity of α. As for a counter example, let $\dot{x} = -1$ and $h(x) = \frac{2\sqrt{2}}{3\sqrt{3}} x^{3/2}$ for $x \geq 0$. Then

$\dot{B}(x) = -B^{1/3}(x)$ satisfies the condition on $\mathcal{C}$. However, the point 0 loses asymptotic behavior and $h(x)$ will reach 0 within finite time for any $x_0 > 0$.

Remark 5: Sufficient conditions for enforcing set invariance using ECBF or HOCBF can be found in [16] and [24], respectively (see also [2]). The first part of Proposition 1 recaptures the results in [2], [16], [24], but we spell out the importance of the local Lipschitz condition on α_j's and the fact that the set $\bigcap_{j=1}^{r} \mathcal{C}_j$ itself may not be controlled invariant under the well-known (zeroing) CBF condition (see Remark 4 above) even for the case $r = 1$ without further assumptions. The second part of Proposition 1 is in case that by a given HOCBF h, the initial point $x_0 \notin \bigcap_{j=1}^{r} \mathcal{C}_j$. However, one can always rescale the existing α_j with proper choices of c_j to provide invariance conditions, such that controllers adjusted to the conditions will lead the trajectories starting from any compact initial set $X_0 \subset \text{Int}(\mathcal{C}_1)$ invariant within $\mathcal{C}_1$.

F. Problem Formulation

The objective of this paper is to control a nonlinear system (1) with unknown parameters to reach a given target set while ensuring safety, i.e., staying inside a safe set. We assume that the nominal model (2) is known and there exists a nominal feedback controller such that the closed-loop system can safely reach the target set. Then the problem is formally formulated as below.

Problem 1: Given system as in Eq (1), a goal region $\mathcal{X}_{\text{goal}} \subset \mathbb{R}^n$, a safe set $\mathcal{X}_{\text{safe}} \subset \mathbb{R}^n$, a nominal controller $k(x)$, and an initial state x_{init}, design a feedback controller $u = \tilde{k}(x)$, where $x \in \mathcal{X}_{\text{safe}}$ and $\tilde{k} : \mathcal{X}_{\text{safe}} \to \mathbb{R}$, such that the solution of the closed-loop system satisfies that $x(T, x_{\text{init}}) \in \mathcal{X}_{\text{goal}}$ for some $T > 0$ and $x(t, x_{\text{init}}) \in \mathcal{X}_{\text{safe}}$ for all $t \geq 0$.

III. Model Uncertainty and Learning Framework

In this section, we discuss how we deal with model uncertainty and learn the real model. As defined in Section II, we consider a real model as in Eq (1), where f and g are not known precisely in practice and a nominal model as in Eq (2) that estimates the true dynamics of the system is available. Then we can rewrite the real model using parametric uncertainty as

$$\dot{x} = \hat{f}(x) + \hat{g}(x)u + b(x) + A(x)u, \tag{11}$$

where $b(x) = f(x) - \hat{f}(x)$ and $A(x) = g(x) - \hat{g}(x)$.

Proposition 2: Given a nominal model and a real model as in Eq (2) and Eq (11), respectively, and the corresponding control barrier functions $\hat{h}$ and h of the same relative degree r with respect to the nominal model, real model and uncertainty on a set D, we have

$$h^m(x) = \hat{h}^m(x) + \Delta_m(x), \quad x \in D, \tag{12}$$

for $m = 1, 2, \ldots, r-1$, where $\Delta_m(x)$ is a remainder term that is independent of the control input u.

We omit the proof of this proposition due to space limit. The proof can be found in [20]. The above proposition shows that for $m = 1, 2, \cdots, r-1$, we can always separate the time derivative of the CBF for the real system into the time derivative of the CBF for the nominal system and a remainder. As a result, for $m = r$:

$$\begin{aligned} h^r(x) &= \frac{\partial (L_{\hat{f}}^{r-1} h(x) + \Delta_{r-1}(x))}{\partial x} \cdot (\hat{f}(x) + \hat{g}(x)u \\ &\quad + b(x) + A(x)u) \\ &= L_{\hat{f}}^{r} h(x) + L_{\hat{g}} L_{\hat{f}}^{r-1} h(x) + L_b L_{\hat{f}}^{r-1} h(x) \\ &\quad + L_A L_{\hat{f}}^{r-1} h(x) u + \frac{\partial \Delta_{r-1}(x)}{\partial x} \cdot (\hat{f}(x) + b(x)) \\ &\quad + \frac{\partial \Delta_{r-1}}{\partial x} \cdot (\hat{g}(x) + A(x))u \\ &= \hat{h}^r + \Delta_r + \Sigma_r u, \end{aligned}$$

where $\Delta_r(x) = \frac{\partial \Delta_{r-1}(x)}{\partial x} \cdot (\hat{f}(x) + b(x)) + L_b L_{\hat{f}}^{r-1} h(x)$ and $\Sigma_r(x) = L_A L_{\hat{f}}^{r-1} h(x) + \frac{\partial \Delta_{r-1}(x)}{\partial x} \cdot (\hat{g}(x) + A(x))$. According to the above conclusion, we know that the higher order time derivative of the real CBF h^r can be separated into the higher order time derivative of the nominal CBF $\hat{h}^r$ and a remainder $\Delta_r + \Sigma_r u$. This implies that we can use neural networks to approximate $\Delta_r(x)$ and $\Sigma_r(x)$ via supervised regression. We can sample initial states and let the system evolve according to the given nominal controller. At each time step, we can store transition information into a buffer $\mathcal{B} = \{(x_i, u_i), h_i^r\}_i^N$, where N is the length of the buffer. The term h_i^r is calculated using numerical differentiation and this is the true value of r^{th}-order time derivative of CBF. Then we can construct an estimator to learn this true value using

$$\hat{\hat{E}}(x) = \hat{h}^r + \Delta(x) + \Sigma(x)u.$$

Specifying a loss function $\mathcal{L}$ using minimum square error (MSE), the regression task is to find the estimator such that the loss function $\frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(\hat{\hat{E}}(x), h^r(x))$ is minimized. Meanwhile, a very important property of learning process is that the data has to be independently and identically distributed (i.i.d). Since the data generated along the trajectories violate this assumption, we use a buffer to store memory along trajectories as in [15]. We first sample an initial point within working space and roll out according to the nominal controller. The control input executed during the transition is calculated by solving the quadratic programming problem

$$\begin{aligned} u(x) = \arg\min_{u \in \mathbb{R}} \ & \frac{1}{2} ||u - k(x)||^2 \qquad \text{(ECBF-QP)} \\ \text{s.t.} \quad & \hat{\hat{E}}(x) \geq -K\eta(x), \end{aligned} \tag{13}$$

as in [4] but using $\hat{\hat{E}}(x)$ as the estimation of $h^r(x)$. This quadratic programming problem helps to find a safe control that is nearest to the nominal control $k(x)$. The estimator is also improved along the sampling trajectories and is updated at each time step. At each time step, we sample data from buffer $\mathcal{B}$ and update neural networks such that the loss function is minimized. The algorithm of learning CBF with high relative degree is shown in Algorithm 1. The algorithm will finally provide an estimator that is accurate enough to mimic the dynamics of the r^{th}-order time derivative of CBF for the real model and safe trajectories can be generated using the learned CBF.

Algorithm 1 Learning algorithm for CBFs with high relative degree

---

Require: A working space, a safe set, a nominal CBF $\hat{h}$, Dataset $\mathcal{B}$, nominal control policy $k(x)$, maximum step n in each trajectory, initial neural network, number of trajectory sampled $\mathcal{N}$, batch size M, loss function $\mathcal{L}$.

Initialize neural network and buffer $\mathcal{B}$
for i in $\mathcal{N}$ **do**
  Sample an initial point x_0
  for j in $1, 2, \cdots, n$ **do**
    Calculate control u_j by solving QP problem in Eq (13)
    Get x_{j+1} from x_j and u_j
    $\mathcal{B} \leftarrow ((x_j, u_j), h_j^r)$
    Sample batch from $\mathcal{B}$
    Update neural network by minimizing the loss function $\mathcal{L}$
  end for
end for

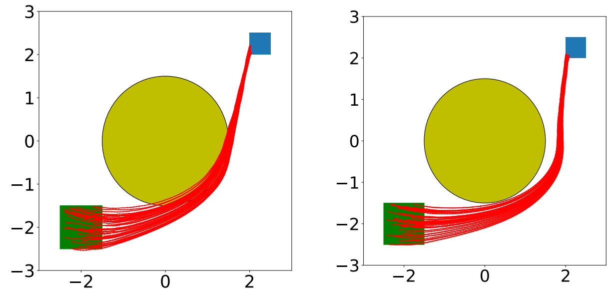
IV. Simulation Result

In this section, we test our algorithm using a differential drive model as in [14]: $\dot{x} = v\cos\vartheta$, $\dot{y} = v\sin\vartheta$, $\dot{\vartheta} = \omega$, where x and y are the planar positions of the center of the vehicle, ϑ is its orientation, v is the forward velocity and ω is the control input of the system. The uncertainty of the systems comes from the forward velocity of v.

A. Experiment 1

In the first experiment, we test our algorithm for single static obstacle avoidance. The working space is $[-3, 3] \times [-3, 3] \times [-\pi, \pi]$. The center of the obstacle is at the origin $(0, 0)$ with radius $r_O = 1.5$. We use the CBF $h(x, y, \vartheta) = x^2 + y^2 - r_O^2$. The CBF has a relative degree $r = 2$ with respect to the system as there is no orientation ϑ in it. The nominal policy is calculated using TRPO [18] with 2 millions training steps in the working space without any obstacles. The forward velocity of the nominal model is 1 while the forward velocity of the real system is 0.7. We use a neural network with 2 hidden layers and 200 nodes in each layer to learn the barrier function.

We show the result by testing the safe rate for the trajectories with 50 initial points between using the nominal CBF and the learned CBF. Since the uncertainty will most likely make trajectories that are close to the obstacle unsafe, we only sample initial points from the green areas as in Fig 1a and Fig 1b. The result are shown in TABLE I. We can see that all the trajectories are safe by using the learned CBF while for the nominal CBF, the safe rate is only 28%.



(a) (b)

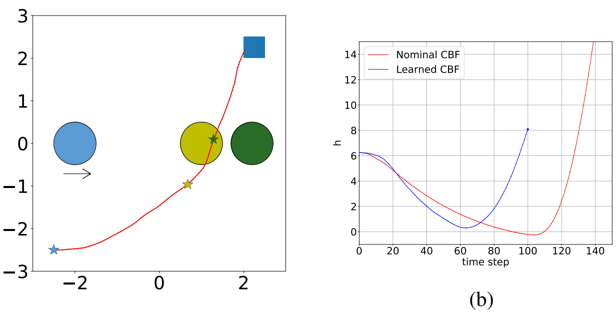
Fig. 1. Safe rate comparison between using nominal CBF and the learned CBF with 50 trajectories with uncertainty in forward velocity. The velocity of the nominal system is 1 while the velocity of the real system is 0.7. All the initial points are sampled from green areas (a): 50 trajectories using the nominal CBF. (b): 50 trajectories using the learned CBF.

	Nominal CBF	Learned CBF
Number of samples	50	50
Number of unsafe trajectories	36	0
Safe rate	28%	100%

TABLE I: Safe rate between the nominal CBF and the learned CBF for Experiment 1.

B. Experiment 2

We test our algorithm using a dynamic obstacle in the second experiment. As is shown in Figure 2a, the initial position of the robot is marked as the blue star. A moving obstacle moves along the x-axis to the right from $(-2, 0)$ with a speed of $0.6/s$. The radius of the obstacle is $r_O = 0.5$ and the goal is marked as the blue square. We use the control barrier function $h = (x - x_O)^2 + (y - y_O)^2 - r_O^2$, where x_O and y_O are x and y coordinate of the obstacle. The parameters for the nominal model are the same as in the first experiment and we use the same nominal controller as well. The real system has the uncertainty that $v = 0.7$. We also use the same structure of the neural network as in Experiment 1 and sample 40 trajectories for training. The trajectory using the learned CBF is shown in Figure 2a. The yellow circle and star are the position of the obstacle and the robot at time step $n = 50$ and the green circle and star are those for time step $n = 70$. We also plot the value of h during the simulation in Figure 2b for using the learned CBF and the nominal CBF. We see that the value of h is always positive using the learned CBF while the value drops below 0 for using the nominal CBF. This implies that the robot avoids the moving obstacle successfully when we use the learned CBF to solve the QP problems while it collides with the obstacle when we use the nominal CBF. Besides, we can see that the blue curve terminates earlier in Figure 2b than the red curve. This is because we terminate plotting h when the robot reach the goal region. We also test the safe rate of our method. We sample 50 initial points to test the result and compare the safe rate using the nominal CBF and the learned CBF. As in Experiment 1, all the initial points are sample within in the region $[-2.5,-1.5]\times[-2.5,-1.5]$. The result is presented in TABLE II. From the table, we can see that our method guarantees a 100% safe rate while using the nominal CBF for the real system, the success rate is only 36%.



(a)

(b)

Fig. 2. Simulation result for Experiment 2: (a): The initial position is at $(-2.5,-2.5,0)$ marked as the blue star. The obstacle is marked with the blue circle at $(-2,0)$ and moves right with a speed of $0.6/s$. The yellow star and circle are the positions of the robot and obstacle at time step $n=50$. The green star and circle are the positions of the robot and obstacle at time step $n=70$. The blue square is the goal region. The trajectory calculated using the learned CBF is the red curve. (b): The value of h during simulation.

	Nominal CBF	Learned CBF
Number of samples	50	50
Number of unsafe trajectories	32	0
Safe rate	36%	100%

TABLE II: Safe rate between the nominal CBF and the learned CBF for Experiment 2.

V. Conclusion

In this paper, we present a framework for learning the CBFs with high relative degree for systems with uncertainty. We first provide sufficient conditions on controllers via CBFs with high relative degree for set invariance. We also show that the dynamics of the real CBF can be learned from that of the nominal CBF and a remainder by using neural networks. We show in simulation that our method can handle model uncertainty using a differential driving robot model. Since we need to calculate high order derivative of the control barrier functions during training using numerical differentiation, the error in high order derivative will affect the performance of the networks. As a result, we will study the impact of numerical differentiation for the learning process.

References

[1] Bowen Alpern and Fred B Schneider. Defining liveness. *Information Processing Letters*, 21(4):181–185, 1985.

[2] Aaron D Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *Proc. of ECC*, pages 3420–3431. IEEE, 2019.

[3] Aaron D Ames, Jessy W Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs with application to adaptive cruise control. In *Proc. of CDC*, pages 6271–6278. IEEE, 2014.

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

[5] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[6] Yuxiao Chen, Huei Peng, and Jessy Grizzle. Obstacle avoidance for low-speed autonomous vehicles with barrier function. *IEEE Transactions on Control Systems Technology*, 26(1):194–206, 2017.

[7] Richard Cheng, Gábor Orosz, Richard M Murray, and Joel W Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *Proc. of AAAI*, volume 33, pages 3387–3395, 2019.

[8] Jason Choi, Fernando Castañeda, Claire J Tomlin, and Koushil Sreenath. Reinforcement learning for safety-critical control under model uncertainty, using control lyapunov functions and control barrier functions. *arXiv preprint arXiv:2004.07584*, 2020.

[9] Girish Chowdhary, Hassan A Kingravi, Jonathan P How, and Patricio A Vela. Bayesian nonparametric adaptive control using gaussian processes. *IEEE Transactions on Neural Networks and Learning Systems*, 26(3):537–550, 2014.

[10] Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

[11] Shao-Chen Hsu, Xiangru Xu, and Aaron D Ames. Control barrier function based quadratic programs with application to bipedal robotic walking. In *Proc. of ACC*, pages 4542–4548. IEEE, 2015.

[12] S Mohammad Khansari-Zadeh and Aude Billard. Learning control lyapunov function to ensure stability of dynamical system-based robot reaching motions. *Robotics and Autonomous Systems*, 62(6):752–765, 2014.

[13] Leslie Lamport. Proving the correctness of multiprocess programs. *IEEE Transactions on Software Engineering*, (2):125–143, 1977.

[14] Steven M LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[16] Quan Nguyen and Koushil Sreenath. Exponential control barrier functions for enforcing high relative-degree safety-critical constraints. In *Proc. of ACC*, pages 322–328. IEEE, 2016.

[17] Manuel Rauscher, Melanie Kimmel, and Sandra Hirche. Constrained robot control using control barrier functions. In *Proc. of IROS*, pages 279–285. IEEE, 2016.

[18] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *Proc. of ICML*, pages 1889–1897, 2015.

[19] Andrew Taylor, Andrew Singletary, Yisong Yue, and Aaron Ames. Learning for safety-critical control with control barrier functions. In *Learning for Dynamics and Control*, pages 708–717. PMLR, 2020.

[20] Chuanzheng Wang, Yinan Li, Yiming Meng, Stephen L Smith, and Jun Liu. Learning control barrier functions with high relative degree for safety-critical control. *arXiv preprint arXiv:2011.10721*, 2020.

[21] Li Wang, Aaron D Ames, and Magnus Egerstedt. Safe certificate-based maneuvers for teams of quadrotors using differential flatness. In *Proc. of ICRA*, pages 3293–3298. IEEE, 2017.

[22] Tyler Westenbroek, David Fridovich-Keil, Eric Mazumdar, Shreyas Arora, Valmik Prabhu, S Shankar Sastry, and Claire J Tomlin. Feedback linearization for unknown systems via reinforcement learning. *arXiv preprint arXiv:1910.13272*, 2019.

[23] Guofan Wu and Koushil Sreenath. Safety-critical and constrained geometric control synthesis using control lyapunov and control barrier functions for systems evolving on manifolds. In *Proc. of ACC*, pages 2038–2044. IEEE, 2015.

[24] Wei Xiao and Calin Belta. Control barrier functions for systems with high relative degree. In *Proc. of CDC*, pages 474–479. IEEE, 2019.

[25] Xiangru Xu. Constrained control of input–output linearizable systems using control sharing barrier functions. *Automatica*, 87:195–201, 2018.

[26] Shakiba Yaghoubi, Georgios Fainekos, and Sriram Sankaranarayanan. Training neural network controllers using control barrier functions in the presence of disturbances. *arXiv preprint arXiv:2001.08088*, 2020.