Model-Free Barrier Functions via Implicit Evading Maneuvers

Eric Squires, Rohit Konda, Samuel Coogan, and Magnus Egerstedt†

***Abstract*— This paper demonstrates that the safety override arising from the use of a barrier function can in some cases be needlessly restrictive. In particular, we examine the case of fixed-wing collision avoidance and show that when using a barrier function, there are cases where two fixed-wing aircraft can come closer to colliding than if there were no barrier function at all. In addition, we construct cases where the barrier function labels the system as unsafe even when the vehicles start arbitrarily far apart. In other words, the barrier function ensures safety but with unnecessary costs to performance. We therefore introduce model-free barrier functions which take a data driven approach to creating a barrier function. We demonstrate the effectiveness of model-free barrier functions in a collision avoidance simulation of two fixed-wing aircraft.**

I. INTRODUCTION

Barrier functions [1], a function of the state whose derivative is bounded, can be used to maximize performance while ensuring safety. However, if the safety constraint from the barrier function is overly restrictive then performance can be diminished. For example, in adaptive cruise control, safety designers can choose a minimum inter-vehicle distance that the vehicle must satisfy. Setting this distance too high will result in excessive inter-vehicle distances where speed setpoints are difficult to achieve. In other words, the performance goal (speed) is negatively impacted by an overly conservative constraint (inter-vehicle distances).

In this paper we show a general solution to this problem and apply it to fixed-wing unmanned aerial vehicle (FW-UAV) collision avoidance. We first consider the case where the barrier function ensures each vehicle can maintain a straight trajectory without collisions. In this case even when the vehicles are arbitrarily far apart the barrier function can indicate the vehicles are unsafe, resulting in performance degradation. For instance, a vehicle located far away could orient itself in a way that makes the barrier function imply an override is needed. This can make the system unpredictable as non-local factors (e.g. vehicles far away) can have an impact on control choices. This could even be exploited by malevolent actors who choose to orient their own aircraft in a way that forces the aircraft to adjust in suboptimal ways.

Another case is a barrier function that ensures vehicles can employ a turning maneuver. We construct a scenario where using a nominal controller designed for performance but not safety would result in vehicle distances far greater than the threshold but a barrier function results in a significant alteration that causes them to barely exceed the safety distance. This reduces performance, increases safety risks from unmodelled noise, and reduces trust as observers see the safety override causing the vehicles to fly needlessly close.

Prior work has relaxed the override while ensuring safety by constructing a barrier function that accounts for the nominal controller. In [2] the authors maximize the set of safe states that are compatible with a region of attraction to maximize performance. Similarly, a nominal controller and barrier function are learned simultaneously in [3]. Imitation learning was used in [4] where a barrier function is constructed from expert trajectories where the expert can consider performance and safety factors. Barrier functions have also been used to guide exploration in [5] via off policy reinforcement learning (RL). Similarly, [6] introduces a barrier function to constrain the policy update in RL.

Rather than training both the nominal controller and safety override, we maximize the set of available safe controls that could be applied to any nominal policy. This separates concerns to simplify controller design [7]. In particular, we show that maximizing the set of safe states is not enough to ensure that an override is not restrictive. In other words, given a state that is safe for two different barrier functions, it may be that the set of controls to keep the system safe is larger for a barrier function with a smaller overall safe set.

We also construct a barrier function without requiring a dynamics model which differs from prior work on barrier functions with uncertainty [8], [9]. This can reduce model mismatch that can lead to real-world performance degradation. Further, model-free approaches can often outperform model-based systems [10] as they are less restricted in fitting to data. Finally, the model-free approach of this paper enables a general solution that can be applied across a large class of problems with different dynamics and safety constraints without having to manually re-derive a barrier function. For instance, while we demonstrate the algorithm using FW-UAV collision avoidance, the same algorithm could equally be applied to quadrotors.

Thus, we propose model-free barrier functions (MFBFs), which are learned from interactions with the environment, to reduce how much the system is overridden. This approach differs from, for instance, model-free RL as it allows introspection of safety characteristics to identify why safety override selections are made, whereas introspection in model-free RL is difficult. Contributions are the following. First, we motivate MFBFs with examples from FW-UAV collision avoidance [11] that demonstrate a model-based approach induces unnecessary overrides. Second, we derive MFBFs. Third, we demonstrate the approach in simulation.

†Eric Squires is with the Georgia Tech Research Institute. Rohit Konda is with the University of California Santa Barbara. Samuel Coogan is with the School of Electrical and Computer Engineering, Georgia Institute of Technology. Magnus Egerstedt is with the Samueli School of Engineering University of California, Irvine. This work was enabled by NSF Grant No. 1836932 and the University System of Georgia Tuition Assistance Program.

A video of the behavior is available [12]. This paper is organized as follows. Section II introduces the background for barrier functions. Section III derives MFBFs. Section IV demonstrates the algorithm in simulation. Contents of this paper have previously appeared in the thesis [13].

II. Background

In this paper we motivate model-based and model-free barrier functions with FW-UAV collision avoidance. Given two FW-UAVs indexed by i ($i \in \{1,2\}$), vehicle i state and control inputs are $x_{k,i} = \begin{bmatrix} p_{k,i,x} & p_{k,i,y} & \theta_{k,i} & p_{k,i,z} \end{bmatrix}^T$ and $u_{k,i} = \begin{bmatrix} v_{k,i} & \omega_{k,i} & \zeta_{k,i} \end{bmatrix}^T$, where $p_{k,i,x}$, $p_{k,i,y}$, and $p_{k,i,z}$ are the x, y, and z position while $v_{k,i}$, $\omega_{k,i}$, and $\zeta_{k,i}$ are the translational, rotational, and vertical velocities with $v_{min} \leq v_{k,i} \leq v_{max}$, $v_{min} > 0$, $|\omega_{k,i}| \leq \omega_{max}$, and $|\zeta_{k,i}| \leq \zeta_{max}$. The discrete time dynamics for vehicle i are

$$x_{k+1,i} = \begin{bmatrix} p_{k,i,x} + v_{k,i}\cos\theta_{k,i}\Delta t \\ p_{k,i,y} + v_{k,i}\sin\theta_{k,i}\Delta t \\ \theta_{k,i} + \omega_{k,i}\Delta t \\ p_{k,i,z} + \zeta_{k,i}\Delta t \end{bmatrix}.$$

The two FW-UAV system has state $x_k = \begin{bmatrix} x_{k,1}^T & x_{k,2}^T \end{bmatrix}^T$ with dynamics of the form

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

where $x_k \in \mathbb{R}^n$, $u_k \in U \subset \mathbb{R}^m$, and U is the set of available controls for the system. In the system above of two FW-UAVs, $n = 8$ and $m = 6$. We briefly summarize [14], which develops barrier functions for discrete time with dynamics in (1). Let $h : \mathbb{R}^n \to \mathbb{R}$ be an output function of the state and define the safe set $\mathcal{C}$ as a superlevel set of h so that

$$\mathcal{C} = \{x_k \in \mathbb{R}^n \ : \ h(x_k) \geq 0\}. \tag{2}$$

Let $\Delta h(x_k, u_k) = h(x_{k+1}) - h(x_k)$. The following definition is an adaptation from Definition 4 of [14] using terminology similar to [1].

Definition 1. A map $h : \mathbb{R}^n \to \mathbb{R}$ is a *Discrete-Time Exponential Control Barrier Function (DT-ECBF)* on a set $\mathcal{D}$ where $\mathcal{C} \subseteq \mathcal{D}$ if there is a $u_k \in \mathbb{R}^m$ and λ such that $\Delta h(x_k, u_k) + \lambda h(x_k) \geq 0$ and $0 \leq \lambda \leq 1$ for all $x_k \in \mathcal{D}$.

The following is an adaptation from Proposition 4 of [14] using the admissible control space [1] defined as

$$K(x_k) = \{u_k \in U \ : \ \Delta h(x_k, u_k) + \lambda h(x_k) \geq 0\}. \tag{3}$$

Proposition 1. *Given a set $\mathcal{C} \subset \mathbb{R}^n$ defined in (2) for an output function h, let h be a DT-ECBF on $\mathcal{D}$ and $u : \mathbb{R}^n \to U$ be such that $u(x_k) \in K(x_k)$ for all $x_k \in D$. If $x_0 \in \mathcal{C}$ then $x_k \in \mathcal{C}$ for all $k > 0$.*

If the system has a nominal controller $\hat{u}_k$ that does not necessarily ensure safety, an optimization can select a control value u_k^* as close as possible to $\hat{u}_k$ while ensuring safety:

$$\begin{aligned} u_k^* \ = \ & \arg\min_{u_k \in U} \frac{1}{2}\|u_k - \hat{u}_k\|^2 \\ \text{s.t. } & u_k \in U \\ & u_k \in K(x_k). \end{aligned} \tag{4}$$

Equation (4) is nonconvex [14] and we resolve this by assuming U is a discrete set.

III. Generating a Model-Free Barrier Function via Evasive Maneuvers

A. Constructing Barrier Functions For Discrete Time

In [11] the authors demonstrate how to construct a barrier function for continuous time systems so we first adapt that method to discrete time. A similar approach is [15] although [11] does not require a backup set. Let $\rho : \mathbb{R}^n \to \mathbb{R}$ be a safety function that must be nonnegative at all times for the system to be safe. Let $\gamma : \mathbb{R}^n \to U$ be an evasive maneuver. Note that γ is not the safety override but instead constructs a barrier function. A candidate DT-ECBF is the worst case safety value after forward propagating the state using γ. Let

$$h(x_0) = \inf_{k \geq 0} \rho(\hat{x}_k) \tag{5}$$

where $\hat{x}_0 = x_0$ and $\hat{x}_{k+1} = f(\hat{x}_k, \gamma(\hat{x}_k))$ for $k > 0$. In forming a MFBF, we treat f as a black box simulation model.

Theorem 1. *Given a dynamical system (1) and a function h defined in (5) with a safety function ρ and an evasive maneuver γ, h is a DT-ECBF on the set $\mathcal{C}$.*

Proof. Suppose $x_0 \in \mathcal{C}$ so that $h(x_0) \geq 0$. Then $\Delta h(x_0, \gamma(x_0)) = \inf_{k \geq 1} \rho(\hat{x}_k) - \inf_{k \geq 0} \rho(\hat{x}_k)$. The right hand side is nonnegative because it is the subtraction of the infimum of the same function on different intervals where the first interval is a subset of the second interval. Then $\Delta h(x_0, \gamma(x_0)) \geq 0$. Recalling as well that $x_0 \in \mathcal{C}$ means that $h(x_0) \geq 0$, this implies that $\Delta h(x_0, \gamma(x_0)) + \lambda h(x_0) \geq 0$. Then $\gamma(x_0) \in K(x_0)$ so h is a DT-ECBF. □

Remark 1. This theorem and proof are similar to Theorem 2 of [11] but for discrete time. Although in Definition 1 $\mathcal{D}$ can be larger than $\mathcal{C}$, Theorem 1 is only valid for $\mathcal{C} = \mathcal{D}$. See [11] for conditions for $\mathcal{C} \subset \mathcal{D}$ in continuous time.

B. The Effect of The Evasive Maneuver on Safe Sets

While Theorem 1 shows that h in (5) is a DT-ECBF and can be used to guarantee safety, different choices of γ can result in drastically different safe sets. Consider the two examples given in [11] where $\rho(x_k) = d_{1,2}(x_k) - D_s$, $d_{1,2}$ is the distance between the vehicles, and D_s is the safety threshold. An evasive maneuver where two vehicles turn at the same rate but have possibly different speeds is given by $\gamma_{turn} = \begin{bmatrix} \eta v & \omega & 0 & v & \omega & 0 \end{bmatrix}^T$ where $0 < \eta \leq 1$. A second evasive maneuver where two vehicles that stay straight for all time is given by $\gamma_{straight} = \begin{bmatrix} v_1 & 0 & \zeta_1 & v_2 & 0 & \zeta_2 \end{bmatrix}^T$. We denote h_{turn} and $h_{straight}$ as the h in (5) constructed from γ_{turn} and $\gamma_{straight}$, respectively. These evasive maneuvers are considered in [11] because they enable a closed form solution to (5) so that the barrier function can be calculated in real-time. We consider some examples where the safe set implied by h_{turn} and $h_{straight}$ results in either an unnecessary override or labeling states as unsafe that have ample room to avoid a collision.

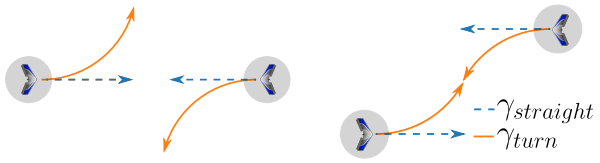

Fig. 1: Vehicles are not safe (left) facing each other with $h_{straight}$, (right) passing on the left with h_{turn}.

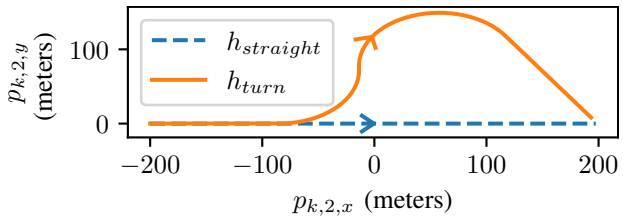

Fig. 2: Given the Fig 1 (right) setup, h_{turn} significantly alters the vehicle 2 trajectory but $h_{straight}$ does not.

A graphical view of these scenarios is in Fig 1. The path traversed by the vehicles for Example 2 is in Fig 2.

Example 1. *States Are Labelled Unsafe Where Collisions Can Be Avoided.* For h_{turn} consider an initial condition where the two vehicles are at the same altitude with orientations pointing at each other. Then no matter how far apart the vehicles start, (5) yields $h = -D_s$, implying the initial conditions are unsafe. This is because $\gamma_{straight}$ implies a future collision. As the vehicles are placed arbitrarily far apart, there is time to turn to avoid a collision. Nevertheless, according to $h_{straight}$, this configuration is outside of the safe set. Note that this scenario has been previously discussed in [16] where it was shown that there does not exist a finite range sensor to ensure safety given $h_{straight}$.

Example 2. *An Unnecessary Invasive Override.* While h_{turn} does not have the issue in Example 1, there are other initial conditions that lead to an unnecessary override with h_{turn}. Suppose the vehicles pass on the left with a lateral separation of more than the safety distance but less than four turn radii. Then if the vehicles continue straight the vehicles will eventually approach an unsafe condition according to h_{turn} and the overriding safety controller will induce a large path correction so that each vehicle can pass on the others' right.

We also plot the set of unsafe states for a variety of configurations (Fig 4) to demonstrate that even when the vehicles are not pointing at each other, the vehicles can be spaced far apart and be in an unsafe state with $h_{straight}$. Further, Fig 4 (top left) shows that the vehicles are unsafe even when they have flown past each other with h_{turn}. These examples demonstrate cases where a barrier function results in restrictive overrides. This paper resolves these issues by fitting a barrier function whose safe set iteratively grows as well as increases the admissible control space. The method is not specific to FW-UAV collision avoidance.

C. An Initial Model-Free Barrier Function

The issues in Figures 1 and 4 result because the γ used to calculate h are constant. While more complicated γ may be preferable, it makes (5) difficult to solve in closed form. To resolve this, we propose a data driven approach. To do so, we start the state at some $x_0 \in \mathbb{R}^n$ and apply some evasive maneuver γ.[1] If an evasive maneuver has not been specified, let $\gamma = \hat{u}$. Given an evasive maneuver, we create a sequence $\{x_k\}_{k=0}^T$ where T is some horizon over which safety is evaluated. In the case of FW-UAV collision avoidance, T may represent battery life of the vehicles after which collisions will not occur.

Note that the sequence $\{x_k\}_{k=0}^T$ is the enumeration of states on the right hand side of (5). Thus, given a starting state x_0, $\rho_{min} = \min_{k\geq 0} \rho(x_k)$ is a sample $h(x_0)$. Suppose this process is repeated N times to form a dataset $D = \{(x_0^j, \rho_{min}^j)\}_{j=1}^N$. Then we can fit a function $\hat{h}$ to approximate the mapping (5) with the dataset D. In the perfect case without error we are left with a function that directly calculates (5) without having to do the integration because the integration is implicit in the fitting of the data.

However, when fitting $\hat{h}$ there will be errors. Errors where the learned $\hat{h}$ is less than the true h leads to conservative behavior by considering states to be unsafe that are actually safe. However, when $\hat{h}$ over predicts, it can imply the state is safe when it is not. A conservative approach is to bias the learned $\hat{h}$ downward to reflect uncertainty. This can be done by biasing the loss function [17] or alternatively with a Bayesian approach (e.g., Gaussian Processes were used for barrier functions in [18]. Bayesian neural networks [19], [20] can also output an uncertainty) by subtracting a desired number of standard deviations (denoted σ) from the model output. We note though that while this method reduces the chances that the fitted $\hat{h}$ will over predict the true h, because it cannot be guaranteed this type of error does not occur, the strict safety guarantee arising from Theorem 1 is lost.

D. Iteratively Expanding the Admissible Control Space

Consider the output of $\hat{h}$ when applied to FW-UAV collision avoidance with a waypoint following nominal controller. Position two vehicles arbitrarily far apart with waypoints located at the starting position of the other vehicle, and orientations pointing at their respective waypoint. This configuration will be unsafe for $\hat{h}$ for the same reason as described in Example 1. We now show how to improve on this initial estimated $\hat{h}$ with an iterative algorithm.

We examine the case where a barrier function h is available and generate a new barrier function h^1 with a larger safe set than h. Given $x_0 \in \mathbb{R}^n$ and $\hat{u}_k \in \mathbb{R}^m$, let $\gamma^1 : \mathbb{R}^n \to U$ be the output[2] of (4). Then γ^1 can be used as an evasive maneuver since it is a function that maps to the action space

[1] x_0 is sampled from $\mathbb{R}^n$ rather than $\mathcal{D}$ during the data-generation phase. Otherwise the data would have a bias toward safe prediction.

[2] Note that because x_0 is sampled from $\mathbb{R}^n$ rather than $\mathcal{D}$ it is not guaranteed that the optimization program has a solution when $x_0 \notin \mathcal{D}$. This can be resolved for instance by adding a slack variable.

Given a barrier function h, we find optimal controls u_k for all x_0, and then map a function gamma to u_k, using which we generate a new dataset for rho_min, to which we fit a new function h_1

as required by Theorem 1. Thus, we form a new barrier function h^1 via (5) with safe set $\mathcal{C}^1$ such that

$$h^1(x_0) = \min_{k\geq 0} \rho(\hat{x}_k), \tag{6}$$

$$\hat{x}_{k+1} = f(\hat{x}_k, \gamma^1(\hat{x}_k)). \tag{7}$$

Theorem 2. *Given a dynamical system (1) let h be defined in (5) with safety function ρ and evasive maneuver γ. Let h^1 be defined in (6) with safety function ρ and evasive maneuver γ^1 defined as the output of (4). Then $\mathcal{C} \subseteq \mathcal{C}^1$.*

Proof. Let $x_0 \in \mathcal{C}$. From Proposition 1, because γ^1 maps to values in $K(x_k)$ for all $x_k \in \mathcal{D}$, $\rho(\hat{x}_k) \geq 0$ for $k \geq 0$ where $\hat{x}_k$ is defined in (7). Then $h^1(x_0) \geq 0$. Then $x_0 \in \mathcal{C}^1$. □

Theorem 2 says that by using γ^1 rather than γ as the evasive maneuver, the safe set does not get smaller. We now show a case where $\mathcal{C}$ is a strict subset of $\mathcal{C}^1$.

Example 3. Consider a discrete double integrator system

$$x_{k+1} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} 0 \\ \Delta t \end{bmatrix} u_k, \tag{8}$$

where $x_{k,1}, x_{k,2}$ are the position and velocity, respectively. Let $\rho(x_k) = x_{k,1}$ so the system is point wise safe when the position is nonnegative, $\gamma(x_k) = 1$, $\Delta t = 0.1$, and $x_0 = \begin{bmatrix} 0.5 & -1 \end{bmatrix}^T$. Then $h(x_0) = -0.05$ so $x_0 \notin \mathcal{C}$. In the case where $\hat{u}_k = 2 \in U$, the result of (4) is $\gamma^1(x_k) = 2$. Then using γ^1 to construct h^1 via (6), $h^1(x_0) = 0.2$ so $x_0 \in \mathcal{C}^1$.

The point of Example 3 is that γ^1 can in some cases do a better job at avoiding unsafe conditions and as a result the safety set is enlarged. However, as discussed in Section III-C, to apply Theorem 2, one needs to forward propagate the dynamics (7) for all future time where the controller at every future timestep is the result of a nonconvex program (4) and return the minimum $\rho(x_k)$ for the resulting sequence $\{\hat{x}_k\}_{k=0}^T$. For online safety overrides, this is not computationally feasible. Thus, we pursue the data driven approach discussed in Section III-C. See Algorithm 1.

Given Theorem 2, if there are no errors in fitting $\hat{h}^1$, we expect that $\mathcal{C}^1$ will be a superset of $\mathcal{C}$. However, we can continue this process to form γ^2 with the property that $\gamma^2(x_k) \in K^1(x_k)$ for all $x_k \in \mathcal{C}^1$ where $K^1(x_k) = \{u \in U : \Delta\hat{h}^1(x_k, u_k) + \lambda\hat{h}^1(x_k) \geq 0\}$. See Algorithm 2. For a barrier function h^i we denote the admissible control space by K^i and the safe set by $\mathcal{C}^i$. However, the next example shows that for $i > j$, $\mathcal{C}^j \subseteq \mathcal{C}^i$ does not always imply $K^j(x_k) \subseteq K^i(x_k)$ for all $x_k \in C^j$.

Example 4. Consider the system in Example 3. Let $x_0 = \begin{bmatrix} 2 & -1 \end{bmatrix}$, $\gamma(x_k) = 0.5$, and $\lambda = 0.9$. Then a numerical calculation shows that $K(x_0) = \{u_0 : u_0 \geq -3.77\}$. Let $\hat{u}(x_k) = 1$ if $x_{k,0} = 2$ and $x_{k,1} = -1$, and $\hat{u}(x_k) = 0.5$ otherwise. Then $K^1(x_0) = \{u_0 : u_0 \geq -3.67\}$. Thus, although Theorem 2 shows that $\mathcal{C} \subseteq \mathcal{C}^1$, $K(x_0) \not\subseteq K^1(x_0)$.

Example 4 shows that even though the safe set is enlarged when using Algorithm 2, the set of controls available to keep the system safe may be reduced. This means that there may be a more aggressive safety override when using h^1 rather than h. Thus, we use the maximum of the barrier functions h^j for $j \leq i$ in Algorithm 2. Note that maximums for boolean composition of barrier functions for continuous time systems was analyzed in [21]. Here we additionally show that a maximum of barrier functions is a barrier function.

Algorithm 1: Initial algorithm for learning a MFBF.

input : h (barrier function), N (number of samples), $\hat{u}$ (nominal controller), T (safety horizon)
output: $\hat{h}^1$

Function `ExpandSafeSet` (h, ρ, N, T):
  $D = \{\}$;
  repeat
    select a random x_0;
    $x \leftarrow x_0$;
    $\rho_{min} \leftarrow \rho(x)$;
    repeat
      $\gamma^1 \leftarrow$ from equation (4) using x, h, and $\hat{u}$;
      $x \leftarrow f(x, \gamma^1)$;
      $\rho_{min} \leftarrow \min(\rho_{min}, \rho(x))$;
    until *repeated T times*;
    append $\{x_0, \rho_{min}\}$ to D;
  until *repeated N times*;
  $\hat{h}^1 \leftarrow$fit to D;
  return $\hat{h}^1$;

Algorithm 2: Iteratively Expanding the Safe Set

input : h (barrier function), N (number of samples), $\hat{u}$ (nominal controller), T (safety horizon), L (number of expansions)
output: $\hat{h}^L$

$\hat{h}^0 \leftarrow h$;
for $i \leftarrow 1$ **to** L **do**
  $\hat{h}^i \leftarrow$ExpandSafeSet($\hat{h}^{i-1}$, ρ, N, $\hat{u}$, T);
end

Theorem 3. *Given a dynamical system (1) and DT-ECBFs h^1 and h^2, the function h^3 defined by $h^3(x_k) = \max(h^1(x_k), h^2(x_k))$ is a DT-ECBF on $\mathcal{C}^1 \cup \mathcal{C}^2$. Further, if $x_0 \in C^1 \cup C^2$, $K^1(x_0) \subseteq K^3(x_0)$ or $K^2(x_0) \subseteq K^3(x_0)$.*

Proof. We first prove that h^3 is a DT-ECBF on $\mathcal{C}^1 \cup \mathcal{C}^2$. Suppose $x_0 \in \mathcal{C}^1 \cup \mathcal{C}^2$ and without loss of generality, assume $h^1(x_0) \geq h^2(x_0)$ so $h^3(x_0) = h^1(x_0)$. Suppose $u_0 \in U$ satisfies $\Delta h^1(x_0, u_0) + \lambda h^1(x_0) \geq 0$ and let $x_1 = f(x_0, u_0)$. Such a u_0 exists because h^1 is a DT-ECBF. Then

$$\begin{aligned} &\Delta h^3(x_0, u_0) + \lambda h^3(x_0) \\ &\quad = [\max(h^1(x_1), h^2(x_1)) - \max(h^1(x_0), h^2(x_0))] \\ &\qquad + \lambda \max(h^1(x_0), h^2(x_0)) \\ &\quad = \max(h^1(x_1), h^2(x_1)) - h^1(x_0) + \lambda h^1(x_0). \end{aligned} \tag{9}$$

Case 1: If $h^1(x_1) \geq h^2(x_1)$ then (9) becomes

$\Delta h^3(x_0,u_0)+\lambda h^3(x_0)=\Delta h^1(x_0,u_0)+\lambda h^1(x_0)\geq 0$.
Case 2: If $h^1(x_1)<h^2(x_1)$ then (9) becomes

$$\begin{aligned}\Delta h^3(x_0,u_0)+\lambda h^3(x_0) &= h^2(x_1)-h^1(x_0)+\lambda h^1(x_0)\\ &\geq h^1(x_1)-h^1(x_0)+\lambda h^1(x_0)\\ &= \Delta h^1(x_0,u_0)+\lambda h^1(x_0)\geq 0.\end{aligned}$$

Then h^3 is a DT-ECBF. This also establishes $K^1(x_0)\subseteq K^3(x_0)$ on $\mathcal{C}^1\cup\mathcal{C}^2$ if $h^1(x_0)\geq h^2(x_0)$. By the same logic, for $h^2(x_0)\geq h^1(x_0)$ with $x_0\in\mathcal{C}^1\cup\mathcal{C}^2$, $K^2(x_0)\subseteq K^3(x_0)$. □

Remark 2. The optimization (4) is non-convex so finding an online solution may infeasible. A direct solution to this is to assume U is a small finite set so (4) can be solved with an exhaustive search. However, when h^1 is defined via (5) for some γ, Theorem 1 demonstrates that γ is always a feasible solution of (4) provided $h^1(x_k)\geq 0$ (and similarly for an evasive maneuver used to construct h^2 for $x_k\in\mathcal{C}^2$). Because $K^1(x_k)\subseteq K^3(x_k)$ for all $x_k\in\mathcal{C}^1$, this means that γ is a feasible solution for (4) when using h^3 and $x_k\in\mathcal{C}^1$.

The proof of Theorem 3 showed that for $h^1(x_0)\geq h^2(x_0)$, $K^1(x_0)\subseteq K^3(x_0)$ and we now show an example where the set inclusion is strict. In other words, by taking the maximum of two barrier functions, we can not only expand the safe set but also expand the admissible control space.

Example 5. Consider again the system in Example 3 with the given ρ, $\lambda=0.9$, and $x_0=\begin{bmatrix}2 & -1\end{bmatrix}^T$. Let h^1 and h^2 be as defined in (5) where γ^1 is defined by $\gamma^1(x_k)=1$ and γ^2 is defined by $\gamma^2(x_k)=5$ if $x_{k,0}\leq 0.5$ and 0 otherwise. Let h^3 be defined by $h^3(x_0)=\max(h^1(x_0),h^2(x_0))$. Then a numerical calculation shows $h^1(x_0)=0.45$, $h^2(x_0)=0.25$, $K^1(x_0)=\{u:u\geq -2.56\}$, $K^2(x_0)=\{u:u\geq -6.78\}$, and $K^3(x_0)=\{u:u\geq -6.48\}$. In other words, $h^1(x_0)>h^2(x_0)$ and $K^1(x_0)\subset K^3(x_0)$.

E. Practical Algorithm

Here we discuss two updates to Algorithms 1 and 2 to enable computationally and memory efficient model-free overrides. First, while h^L in Algorithm 2 is model-free, a model is still required to use h^L to compute an override. This is because computing a solution to (4) requires a calculation of $\Delta h(x_k,u_k)$ which necessitates a model for the dynamics. Thus, to make the final result of Algorithm 2 model-free we must also create a learned function $\Delta\hat{h}$ in Algorithm 1. To do so, record the minimum $\rho_{min,1}=\rho(x_k)$ for $x_k=1,\ldots,T$ in Algorithm 1 and train $\Delta\hat{h}$ to predict $\rho_{min,1}-\rho_{min}$ given x_0 and u_0. When Algorithm 1 outputs these two functions, $\hat{h}$ and $\Delta\hat{h}$, a model-free override can be computed in (4).

Second, the result of Algorithm 2 is a set of L barrier functions. Theorem 3 says we can take the maximum of these L barrier functions to iteratively enlarge both the safe set and admissible control space. However, this implies that L barrier functions must be maintained, which implies memory growth and reduces online computation capability because L models must be queried at every step. Thus, to avoid memory growth and improve online computation, we can instead adjust the dataset of Algorithm 1 in line 12 as follows:

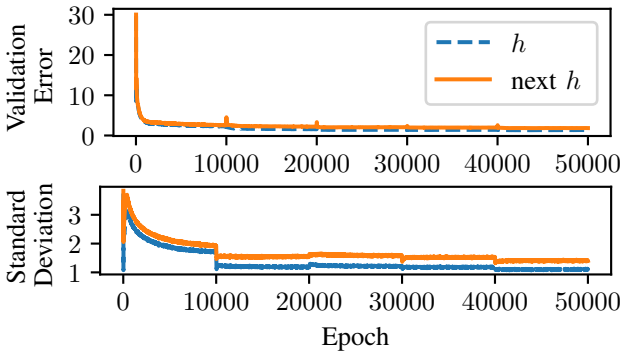

Fig. 3: MFBF validation error (top) and σ (bottom).

12 append $\{x_0,\max(h(x_0),\rho_{min})\}$ to D;

IV. SIMULATION EXPERIMENTS

We now validate the approach of Algorithm 1. We restrict the action space of both vehicles to $[-12,0,12]$ degrees per second for ω while holding velocity fixed at 15 m/s and altitude rate at 0. The initial state for each vehicle is between $\begin{bmatrix}-200 & -200 & -\pi & 0\end{bmatrix}^T$ and $\begin{bmatrix}200 & 200 & \pi & 0\end{bmatrix}^T$. Let $\rho(x)=\max(50,d_{1,2}(x)-D_s)$ where $d_{1,2}$ is the distance between the vehicles and the max simplifies data normalization. Note that this clipping does not change $\mathcal{C}$. We let $D_s=25$, used a learning rate of $1e-4$, 10000 epochs per iteration, 50% dropout rate, 50 samples to calculate σ, and had 4 layers of 1024 nodes with relu activation. We trained the network with a mean squared error loss. To form an initial h, we ran 50,000 episodes using a waypoint following controller without a barrier function and fit a mapping of the initial state to closest vehicle distance for each episode. Training statistics are in Fig 3. During training, the percent of cases where the output minus 3σ is above the true value in the validation set is between 1 and 2.5 percent.

Fig 4 shows the unsafe set for the mean value of the MFBF and when 3σ is subtracted. The latter results in a larger unsafe set. Fig 5 plots how the unsafe set is enlarged as the algorithm proceeds. For iterations 1 to 5, we start each episode so that the barrier function is nonnegative. The system with a nominal controller alone had a collision rate of $(8.9,8.8,8.9,8.8,9.0)$ percent vs the collision percentages of the system with the MFBF of $(0.0,0.5,0.8,0.4,0.5)$ percent so the number of collisions when using a MFBF is less than 10% of the nominal controller. Additionally note that there are not zero collisions when using a MFBF as there is noise in fitting to the data. Nevertheless, safety is significantly improved over using the nominal controller alone.

V. CONCLUSION

In this paper we discussed a few issues with model-based barrier functions: they may label safe states as unsafe (Example 1), cause unnecessary overrides that cause the state to get closer to the boundary of the safe set than without an override (Example 2), be difficult to solve for a barrier function in closed form for complex systems (h_{turn} and

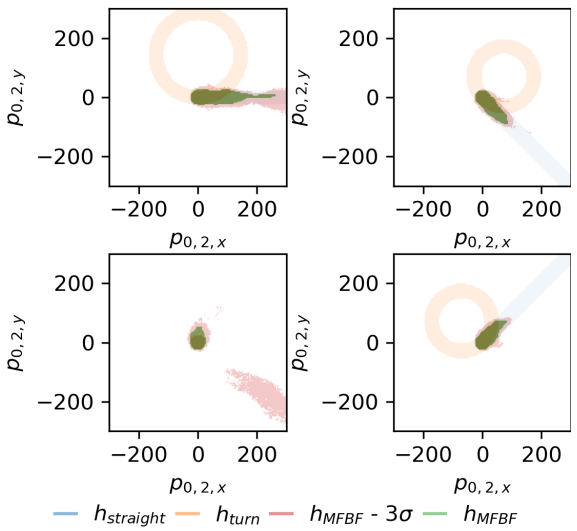

Fig. 4: Points where $h(x) < 0$ given $x_1 = [0 \quad 0 \quad 0 \quad 0]^T$ (vehicle 1 is at the origin pointing right) and vehicle 2 positions vary. Vehicle 2 orientation is left (top left), up (top right), right (bottom left), down (bottom right). Training data was sampled from horizontal positions $(-200, -200)$ to $(200, 200)$ so out-of-sample points have higher uncertainty causing more unsafe states.

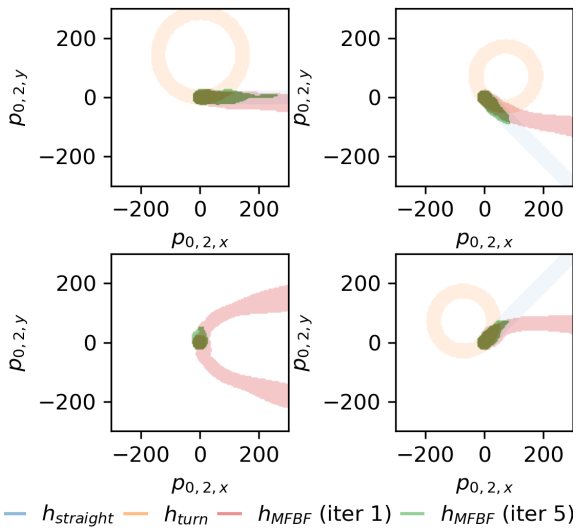

Fig. 5: The same setup as Fig 4 but showing how the unsafe set changes during training. As predicted by Theorem 3, the MFBF unsafe set is smaller at iteration 5 than iteration 1.

$h_{straight}$ exist due to closed form solutions but lead to large unsafe sets, see Fig. 4), and be numerically infeasible to solve for a barrier function when there is a long horizon (eq. (5)). Thus, we introduced MFBFs which take a data-driven approach to developing a barrier function. The tradeoff is that because the barrier function cannot perfectly fit to the data, safety guarantees are lost but the benefit is that the safety set may be significantly enlarged (Fig. 4). We demonstrated the efficacy of the approach in a FW-UAV collision avoidance scenario where, because of the MFBF, the safety of the system is significantly improved over using a nominal controller alone.

References

[1] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, 2017.

[2] L. Wang, D. Han, and M. Egerstedt, "Permissive barrier certificates for safe stabilization using sum-of-squares," in *2018 Annual American Control Conference (ACC)*, 2018, pp. 585–590.

[3] Z. Qin, K. Zhang, Y. Chen, J. Chen, and C. Fan, "Learning safe multi-agent control with decentralized neural barrier certificates," *arXiv preprint arXiv:2101.05436*, 2021.

[4] A. Robey, H. Hu, L. Lindemann, H. Zhang, D. V. Dimarogonas, S. Tu, and N. Matni, "Learning control barrier functions from expert demonstrations," in *2020 59th IEEE Conference on Decision and Control (CDC)*, 2020, pp. 3717–3724.

[5] 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," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 3387–3395, Jul. 2019.

[6] H. Ma, J. Chen, S. E. Li, Z. Lin, Y. Guan, Y. Ren, and S. Zheng, "Model-based constrained reinforcement learning using generalized control barrier function," *arXiv preprint arXiv:2103.01556*, 2021.

[7] U. Borrmann, L. Wang, A. D. Ames, and M. Egerstedt, "Control barrier certificates for safe swarm behavior," *IFAC-PapersOnLine*, vol. 48, no. 27, pp. 68–73, 2015.

[8] J. Choi, F. Castaneda, C. J. Tomlin, and K. 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] A. Robey, L. Lindemann, S. Tu, and N. Matni, "Learning robust hybrid control barrier functions for uncertain systems," *arXiv preprint arXiv:2101.06492*, 2021.

[10] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine, "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 7559–7566.

[11] E. Squires, P. Pierpaoli, R. Konda, S. Coogan, and M. Egerstedt, "Composition of multiple safety constraints with applications to decentralized fixed-wing collision avoidance," *AIAA Journal of Decision, Guidance, and Control (to appear)*, 2022.

[12] E. Squires, "Model free barrier functions via implicit evading maneuvers," https://youtu.be/QNbKrhUxPjk, 2021, accessed: 2022-01-26.

[13] E. G. Squires, "Barrier functions and model free safety with applications to fixed wing collision avoidance," Ph.D. dissertation, Georgia Institute of Technology, 2021.

[14] A. Agrawal and K. Sreenath, "Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation," in *Proceedings of Robotics: Science and Systems*, Cambridge, Massachusetts, July 2017, pp. 73–82.

[15] T. Gurriet, M. Mote, A. D. Ames, and E. Feron, "An online approach to active set invariance," in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 3592–3599.

[16] E. Squires, R. Konda, P. Pierpaoli, S. Coogan, and M. Egerstedt, "Safety with limited range sensing constraints for fixed wing aircraft," in *International Conference on Robotics and Automation*. IEEE, 2021.

[17] M. Srinivasan, A. Dabholkar, S. Coogan, and P. A. Vela, "Synthesis of control barrier functions using a supervised machine learning approach," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 7139–7145.

[18] L. Wang, E. A. Theodorou, and M. Egerstedt, "Safe learning of quadrotor dynamics using barrier certificates," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[19] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 1613–1622.

[20] Y. Gal, "Uncertainty in deep learning," *University of Cambridge*, vol. 1, no. 3, 2016.

[21] P. Glotfelter, J. Cortés, and M. Egerstedt, "Nonsmooth barrier functions with applications to multi-robot systems," *IEEE Control Systems Letters*, vol. 1, no. 2, pp. 310–315, 2017.