

Control Barrier Functions for Flight-Critical Systems in Aerospace Applications

Max H. Cohen¹, Heather Hussain², Tamas G. Molnar³, David E. J. van Wijk⁴, Joseph Gaudio², Peter A. Fisher⁵, Marcel Menner², Anuradha Annaswamy⁵, Johannes Autenrieb⁶, Eugene Lavretsky², Kevin Wise², Ryan Ratliff², and Aaron D. Ames⁴

Abstract—We present a tutorial focused on the applications of control barrier functions (CBFs) to flight-critical aerospace systems. This tutorial is motivated by the need for mathematically justified approaches to state and input limiting on various aerospace systems, much of which is currently accomplished using heuristic techniques. Here, we illustrate how CBFs – originally motivated by applications in robotics – provide a natural framework for constrained control of aerospace systems. Our tutorial encompasses both the theoretical foundations of CBFs as well as real-world aerospace applications of CBFs.

I. INTRODUCTION

Aerospace systems, from drones to fixed-wing aircraft, are subject to flight-critical operational constraints. These constraints often manifest as hard limits on states and inputs whose satisfaction ensure these systems remain in their operational envelope using control inputs that do not exceed physical actuator limits. For an aircraft, such constraints may include, e.g., limits on the altitude, angle of attack, airspeed, and control surface deflections. In practice, satisfaction of these constraints is typically accomplished via hand-crafted modifications of existing control architectures that require extensive engineering effort to tune and do not provide formal assurances of stability or constraint satisfaction.

From a control-theoretic perspective, constrained control problems can be viewed through the lens of *set invariance* [1]. A set is (controlled) invariant if there exists a controller that ensures that all solutions that originate in this set stay inside the set. For aerospace systems, satisfaction of various state limits can be ensured by rendering the desired flight envelope controlled invariant. Over the past decade, control barrier functions (CBFs) [2]–[4], originally motivated by applications in robotics, have emerged as a mature tool for enforcing set invariance in control systems. CBF-based designs are not only backed by rigorous mathematical guarantees, but have also enjoyed practical success through their deployment on a variety of complex systems.

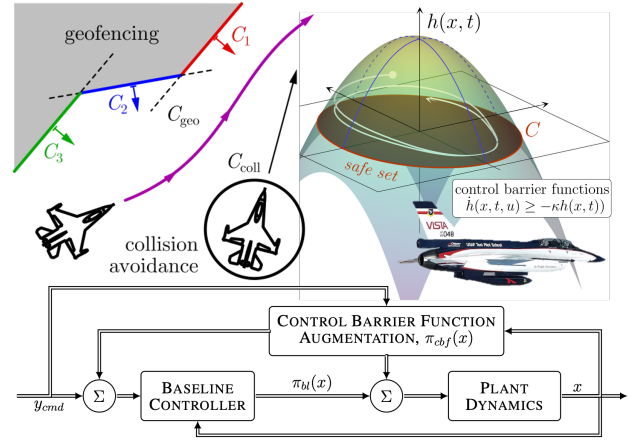


Fig. 1: Overview of CBF architecture for flight-critical systems. CBF augmentation adjusts the outer loop path, i.e. reference command modification, and/or the inner loop via control augmentation.

While CBFs have found success in robotics, their applications to other domains, such as aerospace systems, is comparatively sparse. The purpose of this paper is to illustrate how CBFs automate state and input limiting in flight-critical aerospace systems, backed by formal mathematical guarantees. Starting from the foundations of CBFs, we present techniques that facilitate the application of CBFs to relevant aerospace systems. The efficacy of these approaches are demonstrated with case studies involving real-world deployment of CBFs on systems such fixed-wing aircraft. Ultimately, we aim to provide applied researchers and practitioners in aerospace controls a new set of tools for tackling modern control problems, and to guide more theoretical researchers towards relevant applications in aerospace.

This paper is organized as follows. In Sec. I-A, we discuss background on approaches for enforcing operational limits in aerospace systems. Sec. II provides a primer on CBFs. In Sec. III, we discuss techniques for constructing CBFs for relevant classes of systems. In Sec. IV, we outline backup CBFs, which enable simultaneous state and input limiting. In Sec. V, we illustrate how CBFs integrate into layered guidance, navigation, and control architectures to perform state limiting on complex systems. In Sec. VI, we discuss how CBFs fit into more traditional servo-control approaches and specialize various CBF techniques to linear systems. Sec. VII unifies CBFs with adaptive control for systems with parametric uncertainties. Sec. VIII discusses connections

¹Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC. mhcohen2@ncsu.edu.

²The Boeing Company, Engineering & Technology Innovation, USA.

³Department of Mechanical Engineering, Wichita State University, Wichita, KS. tamas.molnar@wichita.edu.

⁴Department of Mechanical and Civil Engineering, California Institute of Technology, Pasadena, CA. {vanwijk, ames}@caltech.edu.

⁵Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA. {pafisher, aanna}@mit.edu.

⁶Department of Flight Dynamics and Simulation, Institute of Flight Systems, German Aerospace Center (DLR), Braunschweig, Germany. johannes.autenrieb@dlr.de

between CBFs and constraint forces for distributed control. Sec. IX concludes with a discussion on open problems.

A. Background and Related Works

The enforcement of state and control limits in dynamical systems has long been a central problem in aerospace control, where operationally-critical variables such as angle of attack, load factor, pitch rate, actuator position, and structural limits must remain within prescribed bounds. Early approaches relied on state limiters, command limiters, and override logic, often implemented as saturation elements (hard limits) or min-max selection architectures layered on top of nominal controllers. Although effective in practice, these approaches are often heuristic and piecewise-defined, which complicates formal analysis and can introduce undesirable interactions when multiple limits become active simultaneously.

A principled alternative to heuristic limiters for constrained control systems corresponds to Reference Governors (RGs), which has been studied extensively since the 1990s [5]–[11]. The RG, an add-on subsystem, seeks to achieve a constraint handling capability that prevents transients and large signals that may have the potential to induce constraint violation. Operating between the reference command and the input to the system (similar to the outer loop path augmentation shown in Figure 1), RGs allow control design to be separated from the task of constraint satisfaction. The latter is achieved by constructing an admissible set and performing a reachability analysis that ensures solutions stay inside such a set by suitably *governing* the *reference* commands.

Approaches leveraging the core tenets of analytical mechanics [12], with roots dating back to Lagrange, have also been employed for classes of problems with state constraints [13]–[16]. The notion of constraint forces involves geometric constraints that are enforced by augmenting the system dynamics with reaction forces derived via Lagrange multipliers. Conceptually, this framework closely parallels CBF approaches: both enforce state constraints by modifying the system dynamics along constraint gradients, with the magnitude of the modification determined by optimization principles. It should also be mentioned that other related approaches to ensuring constraint satisfaction have appeared in the literature such as [17] which employed barrier certificates and [18] which is based on non-overshooting control.

Recently, CBFs have emerged as a general framework for safety-critical control of dynamical systems [2]–[4]. CBFs encode constraint satisfaction as the forward invariance of a set defined as the zero superlevel set of a barrier function $h(x)$. By designing feedback controllers that enforce Lyapunov-like inequalities on the derivative of h , CBFs guarantee that trajectories remain within this set. This approach was formalized in [3] as a constrained optimization problem and was shown to produce inputs that ensure satisfaction of the state constraints while optimizing control objectives.

The idea behind CBFs can be traced to Nagumo’s seminal work in 1942 [19] (see [20] for a translation) on forward set invariance. The actual definition of a CBF, as employed

throughout this paper, was first introduced in [2], expanded significantly in [3], [21] and several other papers over subsequent years. We refer the reader to [4] for an excellent survey on this topic. In addition to robotics [4], [22], CBFs have been applied, more recently, to aerospace problems such as quadrotors [22]–[26] and fixed-wing aircraft [27]–[29].

Although CBFs have become a popular tool in recent safety-critical control research, earlier approaches remain relevant. Early state limiters such as projections and hard-limiters provided practical envelope protection mechanisms, and can be viewed as passive methods for ensuring constraint satisfaction. RGs and CBFs can be viewed as active methods that employ reachability analysis and set-invariance to ensure constraints. RGs continue to be attractive for applications where preserving the baseline controller is operationally preferable. However, their associated computational burden and limited applicability to rapidly varying reference commands may be disadvantages that can be overcome with CBFs. CBF-based methods provide a flexible alternative, while remaining complementary to governor-based approaches in applications where architectural modularity is paramount. More work remains to be done to understand how their combined advantages can be realized.

Lastly, throughout the literature, CBFs are commonly referred to as *safety filters* [30], [31]. That terminology is avoided here for two reasons. First, safety in aerospace systems is a broad concept that demands rigorous all-encompassing treatment. In this paper, *safety* only implies the satisfaction of operational constraints. Second, for guidance and control applications, we adopt the conventional notion of filtering as an operator-based dynamic signal-processing method to amplify, attenuate, or extract certain frequency (or temporal) components of a signal. Because the design philosophy and objectives for CBF augmentation differ, we refrain from the label *safety filter*.

The evolution of constraint-handling methods in control systems reflects a progression from heuristic and passive limit management to active and increasingly systematic formal approaches. CBFs represent a shift from supervisory reference modification to constraint enforcement at the control-input level, and is formally presented in the next section.

II. A PRIMER ON CONTROL BARRIER FUNCTIONS

Foundations of Constrained Control. Consider a controllable nonlinear control system:

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

where $x \in \mathbb{R}^n$ is the state and $u \in \mathbb{R}^m$ is the control input. The function $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is referred to as the *drift* vector field and describes the behavior of (1) without any input. The function $g : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times m}$ captures the *control directions* whose columns represent the directions along which the control acts. Note that the majority of physical systems can be written in the form of (2) as well as any linear system $\dot{x} = Ax + Bu$ with $f(x) = Ax$ and $g(x) = B$.

Throughout this tutorial, we will assume that the dynamics in (1) are at least locally Lipschitz continuous to ensure the

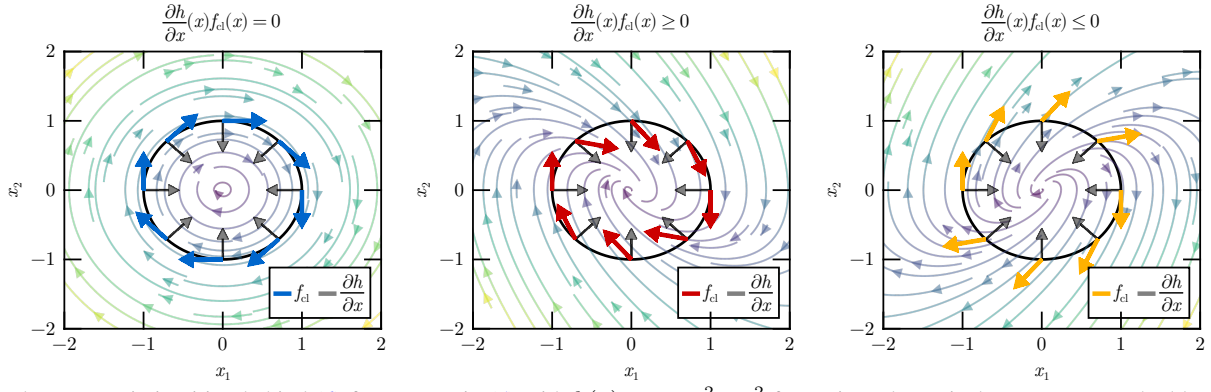


Fig. 2: The geometric intuition behind (6) for a set as in (5) with $h(x) = 1 - x_1^2 - x_2^2$ for various dynamical systems. Here the black circle denotes ∂S , the streamlines represent the dynamics f_{cl} , the colored arrows denote the vector field f_{cl} at various points on the boundary, and the black arrows denote the gradient of h . Note that the first two plots satisfy Nagumo's Theorem while the final plot does not.

existence and uniqueness of solutions. More specifically, let $\pi : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be a locally Lipschitz feedback controller; taking $u = \pi(x)$ then produces the closed-loop system:

$$\dot{x} = f_{cl}(x) = f(x) + g(x)\pi(x), \quad (2)$$

which admits a unique continuously differentiable solution $x(t)$ for each initial condition x_0 . For ease of exposition, we will assume that such solutions exist for all time.

The main objective of this tutorial is to demonstrate how CBFs can be leveraged to design controllers for classes of aerospace systems such that the corresponding closed-loop system (2) satisfies state and input limits along its evolution:

$$x(t) \in C, \quad u(t) \in U, \quad \forall t \geq 0. \quad (3)$$

Here, $C \times U \subset \mathbb{R}^n \times \mathbb{R}^m$ is an *operational constraint set* describing state/input limits that must be enforced along trajectories. To design controllers ensuring the satisfaction of (3), we must characterize the set of initial conditions from which these constraints are guaranteed to be satisfied for all time—just because a given state/input is not *currently* in violation of a constraint does not mean these signals will satisfy the constraints for all time. To design controllers meeting operational constraints, we require a richer characterization of constraint satisfaction via the notion of set invariance [1]. For clarity, in what follows we focus on state constraints C and defer a discussion on input limits to Sec. IV.

Definition 1 (Forward Invariance). A set $S \subset \mathbb{R}^n$ is said to be **forward invariant** for the closed-loop system (2) if starting in S implies solutions remain in S for all time:

$$x(0) \in S \implies x(t) \in S, \quad \forall t \geq 0. \quad (4)$$

Note the distinction between a forward invariant set S and a state constraint set C . If one wishes to formally guarantee that a closed-loop system satisfies a specified state constraint, then one must identify a forward invariant subset of the constraint set $S \subseteq C$. Starting within S then ensures that:

$x(0) \in S \implies x(t) \in S, \quad \forall t \geq 0 \implies x(t) \in C, \quad \forall t \geq 0$,
thereby guaranteeing satisfaction of the constraints for all time. Of course, it is possible that C itself is forward invariant but this typically only holds for very simple systems.

Nagumo's Theorem. We now outline conditions under which sets can be certified as forward invariant. Consider

a set S defined as the 0-superlevel set of a continuously differentiable $h : \mathbb{R}^n \rightarrow \mathbb{R}$:

$$S = \{x \in \mathbb{R}^n \mid h(x) \geq 0\}, \quad \partial S = \{x \in \mathbb{R}^n \mid h(x) = 0\}, \quad (5)$$

$$\text{Int}(S) = \{x \in \mathbb{R}^n \mid h(x) > 0\},$$

where ∂S denotes the boundary and $\text{Int}(S)$ denotes the interior of S . In Sec. VI we will generalize (5) to sets defined by multiple h 's. Before formally stating any theorems, we aim to provide some geometric intuition into the conditions under which S is forward invariant. The closed-loop system (2) is characterized by a vector field $f_{cl} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ that assigns to each state $x \in \mathbb{R}^n$ a vector $f_{cl}(x)$ describing the instantaneous rate of change of the state under the flow of the dynamics. For S to be forward invariant, this vector field must lie tangent or point “into” S for any $x \in \partial S$. For sets S as in (5), the gradient $\frac{\partial h}{\partial x}(x)$ points normal to any level set of h , including ∂S where $h(x) = 0$. Hence, when $h(x) = 0$, the inner product $\frac{\partial h}{\partial x}(x)f_{cl}(x)$, comparing the similarity between the $\frac{\partial h}{\partial x}(x)$ and $f_{cl}(x)$, characterizes if $f_{cl}(x)$ points into or out of S . Thus, so long as $\frac{\partial h}{\partial x}(x) \neq 0$ on ∂S , we have:

$$\begin{aligned} \frac{\partial h}{\partial x}(x)f_{cl}(x) &> 0 \implies f_{cl}(x) \text{ points into } S, \\ \frac{\partial h}{\partial x}(x)f_{cl}(x) &= 0 \implies f_{cl}(x) \text{ is tangent to } \partial S, \\ \frac{\partial h}{\partial x}(x)f_{cl}(x) &< 0 \implies f_{cl}(x) \text{ points out of } S. \end{aligned} \quad (6)$$

This geometric intuition is illustrated in Fig. 2 and is stated formally below as *Nagumo's Theorem*.

Theorem 1 (Nagumo's Theorem). Consider a set $S \subset \mathbb{R}^n$ as in (5) and suppose that $\frac{\partial h}{\partial x}(x) \neq 0$ whenever $h(x) = 0$. Then, S is forward invariant for (2) if and only if:

$$h(x) = 0 \implies \dot{h}(x) = \frac{\partial h}{\partial x}(x)f_{cl}(x) \geq 0. \quad (7)$$

Nagumo's Theorem¹ states that S is forward invariant if and only if f_{cl} is either tangent to or points into S on ∂S . For (7) to be meaningful, we require $\frac{\partial h}{\partial x}(x) \neq 0$ for $x \in \partial S$; else, (7) would be vacuously satisfied for any $f_{cl}(x)$. Further details on Nagumo's Theorem, originally published in [19], can be found in [1, Ch. 4] and [34, Ch. 4], with an English translation of the original paper available in [20].

¹Theorem 1 is a special case of Nagumo's Theorem [19], [20], which characterizes set invariance using tangent cones. Details on the connections between Nagumo's Theorem and Theorem 1 can be found in [32], [33].

Control Barrier Functions. Nagumo's Theorem is a complete characterization of set invariance for *closed-loop systems* but does not state how to synthesize controllers meeting such conditions. Any controller $u = \pi(x)$ rendering S forward invariant must satisfy:

$$\frac{\partial h}{\partial x}(x)(f(x) + g(x)\pi(x)) \geq 0, \quad \forall x \in \partial S. \quad (8)$$

This condition, only placed at states on ∂S , leaves open how to design such a controller at states away from the boundary. One could insist on (8) for all $x \in S$, which is sufficient for (8) to hold, but is overly restrictive. Indeed, doing so would render *every* superlevel set of h , not just the zero superlevel set, forward invariant. This observation led to the introduction of CBFs, which relax in (8) to:

$$\frac{\partial h}{\partial x}(x)(f(x) + g(x)\pi(x)) \geq -\kappa h(x), \quad \forall x \in S, \quad (9)$$

for $\kappa > 0$. When $h(x) = 0$, the condition in (9) reduces to that in (8); however, when $h(x) > 0$ the condition in (9) allows for inputs that cause h to instantaneously decrease: trajectories may approach ∂S so long as they do not cross it. Such inputs are guaranteed to exist when h is a CBF.

Definition 2 ([3]). A continuously differentiable function $h : \mathbb{R}^n \rightarrow \mathbb{R}$ defining a set S as in (5) is said to be a **CBF** for (1) if there exists $\kappa > 0$ such that for each $x \in S$:

$$\sup_{u \in \mathbb{R}^m} \left\{ \frac{\partial h}{\partial x}(x)(f(x) + g(x)u) \right\} > -\kappa h(x). \quad (10)$$

The CBF condition (10) states that for each $x \in S$ there exists a $u \in \mathbb{R}^m$ strictly² satisfying the decay condition $\dot{h} > -\kappa h$, and, thus, forward invariance of S . The parameter κ may be thought of as a *CBF gain*, with larger values permitting trajectories to approach ∂S more aggressively.

The primary utility of CBFs is their ability to augment an existing baseline controller $\pi_{\text{bl}} : \mathbb{R}^n \rightarrow \mathbb{R}^m$, e.g., a PID or LQR controller, such that the augmented input satisfies (9). This **CBF augmentation** (see Figure 1) takes the form [3], [4], [35]:

$$\pi(x) = \underbrace{\pi_{\text{bl}}(x)}_{\text{baseline controller}} + \underbrace{\pi_{\text{cbf}}(x)}_{\text{CBF augmentation}}, \quad (11)$$

where:

$$\pi_{\text{cbf}}(x) = \begin{cases} 0 & \frac{\partial h}{\partial x}(x)g(x) = 0, \\ \lambda(x)W \left(\frac{\partial h}{\partial x}(x)g(x) \right)^\top & \frac{\partial h}{\partial x}(x)g(x) \neq 0, \end{cases} \quad (12)$$

$$\lambda(x) = \max \left\{ 0, -\frac{\frac{\partial h}{\partial x}(x)(f(x) + g(x)\pi_{\text{bl}}(x)) + \kappa h(x)}{\left\| \frac{\partial h}{\partial x}(x)g(x) \right\|_W^2} \right\},$$

for a symmetric and positive definite $W \in \mathbb{R}^{m \times m}$. The intuition behind (12) is as follows. If the baseline input π_{bl} satisfies the CBF condition (9) at a state, then $\lambda(x) = 0$, implying that $u = \pi(x) = \pi_{\text{bl}}(x)$, allowing the baseline input to be applied. If the baseline input does not satisfy

²Strict satisfaction of (10) is sufficient to establish Lipschitz continuity of CBF-based controllers. The original definition of CBFs in [3] leverages a non-strict inequality, which requires additional, sometimes restrictive, assumptions to ensure continuity. Further remarks can be found in [32].

(9), then (12) performs a least-squares projection of this controller onto the set of inputs satisfying the CBF condition:

$$\mathcal{U}_{\text{cbf}}(x) = \{u \in \mathbb{R}^m \mid \frac{\partial h}{\partial x}(x)(f(x) + g(x)u) \geq -\kappa h(x)\}.$$

At such points, the total input $u = \pi(x) = \pi_{\text{bl}}(x) + \pi_{\text{cbf}}(x)$ is the closest input u to $\pi_{\text{bl}}(x)$ that satisfies $u \in \mathcal{U}_{\text{cbf}}(x)$. Finally, the case statement in λ from (12) is included to prevent division by zero. Despite this case statement, (11) can be shown to be locally Lipschitz continuous provided the strict inequality in (10) holds [36]. We summarize this discussion in the following theorem.

Theorem 2. Consider a control system (1) with locally Lipschitz dynamics, a locally Lipschitz baseline controller π_{bl} , a state constraint C , and let h be a CBF for (1) defining a set $S \subseteq C$ as in (5). If $\frac{\partial h}{\partial x}$ is locally Lipschitz, then:

- 1) The CBF augmentation in (11) is locally Lipschitz;
- 2) Taking the control as $u = \pi(x)$, with π in (11) renders S forward invariant for the closed-loop system (2);
- 3) If $x(0) \in S$, then $x(t) \in C$ for all $t \geq 0$.

Our presentation of CBF controllers is at odds with much of the literature, where controllers are computed as the solution to an optimization problem [3], [4]. While we have not framed the controller in (11) from this perspective, it is indeed the solution to the optimization problem:

$$\min_{u \in \mathbb{R}^m} \frac{1}{2} \|u - \pi_{\text{bl}}(x)\|_{W^{-1}}^2 \quad (13)$$

$$\text{subject to } \frac{\partial h}{\partial x}(x)(f(x) + g(x)u) \geq -\kappa h(x),$$

which is a strictly convex quadratic program (QP). The equivalence between (11) and (13) can be shown by noting that the solution to (13) is characterized by the *Karush–Kuhn–Tucker* (KKT) conditions [37], [38], which, for (13), states that a pair (u, λ) is optimal at a state x if and only if:

$$\begin{aligned} \frac{\partial L}{\partial u}(x, u, \lambda) &= 0 & (\text{stationarity}) \\ \frac{\partial h}{\partial x}(x)(f(x) + g(x)u) &\geq -\kappa h(x) & (\text{primal feas.}) \\ \lambda &\geq 0 & (\text{dual feas.}) \\ \lambda \left[\frac{\partial h}{\partial x}(x)(f(x) + g(x)u) + \kappa h(x) \right] &= 0 & (\text{comp. slack.}) \end{aligned}$$

where λ is the dual variable or Lagrange multiplier and:

$$L(x, u, \lambda) = \frac{1}{2} \|u - \pi_{\text{bl}}(x)\|_{W^{-1}}^2 - \lambda \left[\frac{\partial h}{\partial x}(x)(f(x) + g(x)u) + \kappa h(x) \right], \quad (14)$$

is the Lagrangian. One may verify that $u = \pi(x)$ from (11) and $\lambda = \lambda(x)$ from (12) satisfy the KKT conditions and are thus the solution to (13). These conditions will play a crucial role later in this paper in developing closed-form solutions to more complicated classes of optimization-based controllers.

III. CONSTRUCTING CONTROL BARRIER FUNCTIONS

If a CBF h defining a set S as in (5) is known, and this set is contained within the state constraint set $S \subseteq C$, then the CBF augmentation in (11) can be used to augment any locally Lipschitz baseline controller π_{bl} to enforce forward invariance of S and thus satisfaction of the state constraint C . This approach is contingent on h being a CBF – one must verify that the criteria outlined in Def. 2 is satisfied; else, the

controller in (11) may be ill-posed and fail at enforcing state constraints. In this section, we outline how to verify that a *candidate* CBF satisfies Def. 2 and present methodologies to construct CBFs for classes of systems and constraints.

A. Verifying Control Barrier Functions

We begin by providing conditions to verify that a function h is a CBF. We note that the sup condition in (10) is a concise way to write: for each $x \in S$ there exists a $u \in \mathbb{R}^m$ so that:

$$\frac{\partial h}{\partial x}(x)(f(x) + g(x)u) > -\kappa h(x). \quad (15)$$

While finding a u for each $x \in S$ may seem daunting, this process can be simplified by noting that the existence of such inputs depends heavily on the behavior of $\frac{\partial h}{\partial x}(x)g(x)$. If $\frac{\partial h}{\partial x}(x)g(x) \neq 0$ at $x \in S$, then such an input satisfying the above exists — take u as large or as small as necessary so that the left-hand-side of (15) is larger than the right-hand-side. However, if $\frac{\partial h}{\partial x}(x)g(x) = 0$ at $x \in S$, then (10) is only satisfied if the unforced dynamics, described by f , already satisfy the CBF condition. This is summarized with the following equivalent characterization of CBFs [36].

Lemma 1. *A continuously differentiable function h is a CBF if and only if there exists $\kappa > 0$ such that for each $x \in S$:*

$$\frac{\partial h}{\partial x}(x)g(x) = 0 \implies \frac{\partial h}{\partial x}(x)f(x) > -\kappa h(x). \quad (16)$$

The power of Lemma 1 is that one need only verify (10) at points where $\frac{\partial h}{\partial x}(x)g(x) = 0$. If $\frac{\partial h}{\partial x}(x)g(x) \neq 0$ for all $x \in S$, this implies that h is a CBF. Note that Lemma 1 is only valid for systems *without* hard input limits, i.e., $u \in \mathbb{R}^m$. Input saturation will be addressed later in Sec. IV.

B. Constructive CBF Control

Having outlined conditions for verifying CBFs, we now introduce procedures to *synthesize* functions that are CBFs by construction. We begin this process by assuming that:

$$C = \{x \in \mathbb{R}^n : h_c(y(x)) \geq 0\}, \quad (17)$$

where $y : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is a system *output* and $h_c : \mathbb{R}^m \rightarrow \mathbb{R}$ is an output constraint function. This output captures the components of the state whose values we wish to limit — for example, the position or angle of attack of an aircraft. For ease of exposition, we assume that this output has relative degree two with respect to the input, informally meaning that the control input appears after differentiating the output twice along system trajectories, although everything herein can be extended to higher relative degrees. Formally, we assume:

$$\frac{\partial y}{\partial x}(x)g(x) = 0, \text{ rank} \left(\frac{\partial y}{\partial x}(x)g(x) \right) = m, \forall x \in \mathbb{R}^n. \quad (18)$$

where $\dot{y}(x) = \frac{\partial y}{\partial x}(x)f(x)$. Here, $h(x) = h_c(y(x))$ is unlikely to be a CBF since:

$$\dot{h}(x) = \frac{\partial h_c}{\partial y}(y(x))\dot{y}(x) = \frac{\partial h_c}{\partial y}(y(x))\frac{\partial y}{\partial x}(x)f(x), \quad (19)$$

does not depend on u . We now present various methodologies to rectify this issue, each effectively extending h_c to a function whose time derivative does depend on the input.

High-Order CBFs. One way to use h_c to obtain a CBF is to differentiate h_c until the control input appears, and then

impose conditions on higher-order derivatives of h_c . This approach first appeared under the name of *exponential* CBFs [39] before being generalized to *high-order* CBFs [40], [41].

If (18) holds, then u appears in the second time derivative of h_c , at a state x , so long as:

$$\frac{\partial h_c}{\partial y}(y(x))\frac{\partial \dot{y}}{\partial x}(x)g(x) \neq 0. \quad (20)$$

With this, one defines the **High-Order CBF** (HOCBF):

$$\begin{aligned} h(x) &= \dot{h}_c(x) + \kappa_c h_c(y(x)) \\ &= \frac{\partial h_c}{\partial y}(y(x))\dot{y}(x) + \kappa_c h_c(y(x)), \end{aligned} \quad (21)$$

for $\kappa_c > 0$, which defines S as in (5). The motivation behind (21) is as follows. Suppose S is forward invariant so that:

$$h(x(t)) = \dot{h}_c(x(t)) + \kappa_c h_c(y(x(t))) \geq 0, \quad (22)$$

for all $t \geq 0$, which implies that h_c satisfies the closed-loop CBF condition. If additionally $x(0) \in C$, then (22) would imply that $x(t) \in C$ for all $t \geq 0$ as well. Hence, if one were to design a feedback controller π satisfying (9) with h as in (21), and $x(0) \in C \cap S$, then one would have $x(t) \in C$ for all time. This is formalized with the following result.

Theorem 3. *Consider a control system (1) with locally Lipschitz dynamics, an output $y : \mathbb{R}^n \rightarrow \mathbb{R}^m$ with relative degree two, a constraint set C as in (17), and the function $h : \mathbb{R}^n \rightarrow \mathbb{R}$ as in (21) defining a set S as in (5). Let $\pi : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be a locally Lipschitz feedback controller satisfying (9) with h as in (21). If $x(0) \in C \cap S$, then the closed-loop system (2) satisfies $x(t) \in C$ for all $t \geq 0$.*

Note that the set S is *not* a subset of the constraint set C . Rather, HOCBFs enforce the forward invariance of the intersection $C \cap S$, which ensures that $x(t) \in C$ for all time.

CBF Backstepping. We now present an alternative approach to constructing CBFs based on the concept of *backstepping* [42]. This approach is more complicated than the HOCBF approach, but is often better suited for handling outputs with a *mixed* relative degree, where different control inputs appear at different orders of differentiation. While backstepping has been around for decades, its unification with CBFs is more recent [32], [43]–[47]. These approaches are also related to the notion of “non-overshooting” control from [18] with connections to CBFs established in [48]–[50].

To illustrate CBF backstepping, let y be a relative-degree-two output defining a constraint set C as in (17). In backstepping, we pretend that y has relative degree one $\dot{y} = v$ for a virtual input v . We then ask: if we could directly control $\dot{y}$, how should we choose v ? If the goal is to render C forward invariant, the answer is to choose $v = \pi_v(y)$ such that:

$$\frac{\partial h_c}{\partial y}(y)\pi_v(y) > -\kappa h_c(y), \quad (23)$$

so that h_c satisfies the CBF condition for $\dot{y} = v$. Yet since we cannot simply set $\dot{y} = \pi_v(y)$, we must “backstep” through π_v to reach u , which appears in $\ddot{y}$. This is achieved with:

$$h(x) = h_c(y(x)) - \frac{1}{2\mu} \|\dot{y}(x) - \pi_v(y(x))\|^2, \quad (24)$$

where the first term captures the original output constraint, the second term penalizes deviations of $\dot{y}$ from the virtual controller $\pi_v(y)$, and $\mu > 0$. This function defines a set S

as in (5), which satisfies $S \subseteq C$ since $h(x) \geq 0$ implies that $h_c(y(x)) \geq 0$. So long as y has relative degree two, the input “shows up” in the time derivative of h , and can be shown to satisfy (16), implying it is a valid CBF. We formalize this backstepping procedure through the following result.

Theorem 4. Consider a control system (1) with locally Lipschitz dynamics, an output $y : \mathbb{R}^n \rightarrow \mathbb{R}^m$ with relative degree two, a constraint set C as in (17), and let $\pi_v : \mathbb{R}^m \rightarrow \mathbb{R}^m$ be a smooth function satisfying (23). Then, (24) is a CBF whose zero superlevel set S satisfies $S \subseteq C$.

An interesting requirement in Theorem 4 is that the virtual controller π_v must be smooth. Thus far, the only controller we have presented capable of satisfying (23) is that in (12), which is non-smooth due to the presence of the max function. As discussed in [32], [44], [51]; however, this max can always be replaced with a smooth over-approximation, yielding a continuously differentiable π_v .

C. CBF Construction for Fixed-wing Aircraft

We demonstrate these CBFs for a fixed-wing aircraft with collision avoidance and geofencing [27]. The dynamics are described by a *three-dimensional Dubins model*—a kinematic model that governs the evolution of the north, east, and down position coordinates, p_N , p_E , and p_D , the roll, pitch, and yaw angles, ϕ , θ , and ψ , and the speed V_T :

$$\begin{bmatrix} \dot{p}_N \\ \dot{p}_E \\ \dot{p}_D \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{V}_T \end{bmatrix} = \underbrace{\begin{bmatrix} V_T \cos \psi \cos \theta \\ V_T \sin \psi \cos \theta \\ -V_T \sin \theta \\ R \cos \phi \tan \theta \\ -R \sin \phi \\ R \frac{\cos \phi}{\cos \theta} \\ 0 \end{bmatrix}}_{f(x)} + \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & \sin \phi \tan \theta \\ 0 & 0 & \cos \phi \\ 0 & 0 & \frac{\sin \phi}{\cos \theta} \\ 1 & 0 & 0 \end{bmatrix}}_{g(x)} \underbrace{\begin{bmatrix} A_T \\ P \\ Q \end{bmatrix}}_u. \quad (25)$$

The first three rows of (25) model the velocity of the center of mass; the next three specify the Euler angle kinematics based on the angular velocities P , Q , and R about the forward, right, and down axes. The angular velocity about the down axis, describing how the aircraft turns left or right, is:

$$R = \frac{g_D}{V_T} \sin \phi \cos \theta, \quad (26)$$

with gravitational acceleration g_D . The last row of (25) relates the speed V_T to the longitudinal acceleration A_T . The model (25)-(26) has seven states and three inputs.

We design a controller for the aircraft that avoids collision with other aircraft and prevents crossing a geofence. These constraints depend on the aircraft’s position, taken as the output: $y(x) = (p_N, p_E, p_D)$. Note that the evolution of position is not directly affected by the inputs and we have $\frac{\partial y}{\partial x}(x)g(x) = 0$, necessitating the use of the HOCBFs and backstepping to construct CBFs for control design.

Collision avoidance is captured by the constraint function $h_c(y) = \|y - y_{\text{coll}}\| - \rho_{\text{coll}}$, where y_{coll} is the position of the object to be avoided, while ρ_{coll} is a radius that accounts for the combined size of the object and controlled aircraft. Note that y_{coll} depends on time if the object is moving, e.g.,

when avoiding collision with other aircraft. Geofencing is represented by the constraint $h_c(y) = n^\top(y - y_{\text{geo}}) - \rho_{\text{geo}}$, which defines a plane with normal vector n and position y_{geo} , and ρ_{geo} is the distance to be kept from the geofence.

To ensure the satisfaction of multiple constraints simultaneously, such as the combination of multiple collision avoidance and geofencing tasks, we can construct a combined constraint function h_c from individual constraints $h_{c,i}$ as:

$$h_c(y) = -\frac{1}{\eta} \ln \left(\sum_{i=1}^{N_c} e^{-\eta h_{c,i}(y)} \right). \quad (27)$$

Here, h_c is a smooth under-approximation of the minimum of $h_{c,i}$ with a smoothing parameter $\eta > 0$, so that enforcing the combined constraint ($h_c(y) \geq 0$) implies the satisfaction of all individual constraints ($h_{c,i}(y) \geq 0$ for all i) [52].

The constraint h_c is not a CBF since its derivative does not depend on u , but h_c can be used to generate a CBF via HOCBFs (21). For collision avoidance, this leads to:

$$h_{\text{ho}}(x) = \|y(x) - y_{\text{coll}}\| - \rho_{\text{coll}} + \frac{1}{\kappa} \frac{(y(x) - y_{\text{coll}})^\top}{\|y(x) - y_{\text{coll}}\|} (\dot{y}(x) - \dot{y}_{\text{coll}}),$$

while in case of geofencing we get:

$$h_{\text{ho}}(x) = n^\top(y(x) - y_{\text{geo}}) - \rho_{\text{geo}} + \frac{1}{\kappa} n^\top \dot{y}(x),$$

and their combination (27) also yields a similar h_{ho} .

Here, the CBF h_{ho} depends on the position $y(x)$ and the velocity $\dot{y}(x)$, and involves all of the states except for the roll angle ϕ . Thus, the time-derivative of h_{ho} depends on two of the control inputs, the acceleration A_T and the angular velocity Q related to pitching, but does not directly involve the third input, the angular velocity P related to rolling. This CBF could be used for control design; however, the resulting controllers would only rely on acceleration and pitching to ensure collision avoidance and geofencing, while they would not force the aircraft to roll, even when beneficial to do so.

To induce rolling, we require a CBF involving the roll angle ϕ , which we can construct using backstepping (24). We take h_{ho} and use it to design a virtual controller by viewing A_T , Q , and R as control inputs. This produces virtual acceleration $A_{T,v}(x)$, pitching $Q_v(x)$, and turning $R_v(x)$ commands for collision avoidance and geofencing. We then use the virtual turning command $R_v(x)$ to construct:

$$h(x) = h_{\text{ho}}(x) - \frac{1}{2\mu} (R - R_v(x))^2, \quad (28)$$

which allows the aircraft to accelerate, pitch, and roll.

The resulting behavior is illustrated in Fig. 3. The aircraft uses a baseline flight controller to follow a desired straight path with constant speed. The aircraft must avoid collision with another aircraft while staying on one side of the geofences. This is ensured by combining the corresponding constraints via (27), and generating the CBF through (28). The resulting CBF is used by the controller (11)-(12) to force the aircraft to decelerate, pitch, and turn, successfully avoiding collision and flying along the geofence.

IV. BACKUP CBFs FOR RUN-TIME ASSURANCE

In the preceding sections, it was assumed that (1) has sufficient control authority to enforce CBF conditions for set invariance. Yet many aerospace systems are limited by input

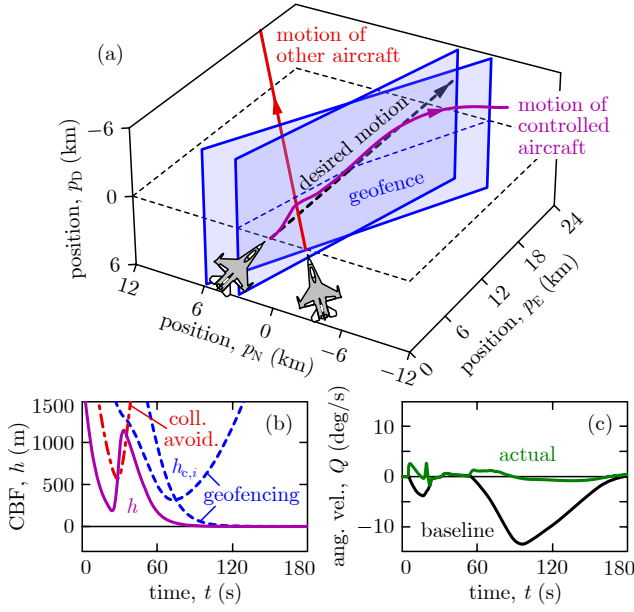


Fig. 3: Simulation of (25) for simultaneous collision avoidance and geofencing using the backstepping CBF (28). Position constraints are successfully maintained by leveraging both acceleration, pitching, and rolling. These results were originally published in [27].

constraints imposed by actuators. Failure to account for such limits could lead to actuator saturation and loss of overall control authority. Hence, CBF designs must take into account actuator limits when $u \in U \subset \mathbb{R}^m$. When input constraints are present, we use the notion of *controlled invariance*.

Definition 3 (Controlled Invariance). A set $S \subset \mathbb{R}^n$ is said to be **controlled invariant** for (1) if there exists a controller $\pi : \mathbb{R}^n \rightarrow U$, $u = \pi(x)$, that renders S forward invariant.

In this section, we present the backup CBF method, which offers a structured approach to synthesizing CBFs that adhere to input limits. This approach leverages knowledge of a backup maneuver (e.g., executing a coordinated turn) which is carefully blended with a baseline controller to ensure forward invariance. The technical details and applications discussed in this section are based on [26], [53]–[56].

To use the backup CBF method, we begin by considering a state constraint set C as in (17) that is not necessarily controlled invariant. To proceed, we consider a “backup” maneuver, characterized by a feedback controller $\pi_b : \mathbb{R}^n \rightarrow U$, along with a smaller, more conservative “backup” set:

$$S_b = \{x \in \mathbb{R}^n \mid h(x) \geq 0\}, \quad (29)$$

where $h : \mathbb{R}^n \rightarrow \mathbb{R}$ is continuously differentiable. Importantly, we assume that i) S_b is controlled invariant and π_b renders S_b forward invariant, and ii) $S_b \subset C$. One may think of S_b as a set from which the original state constraints can always be satisfied (e.g., a small envelope around steady-state flight), whereas π_b is a feedback controller that brings the system to this set of states while adhering to hard actuator limits. While this set is controlled invariant, it may be quite conservative to the point where CBFs generated from this set may interfere too much with a baseline controller, which may be detrimental to performance and robustness requirements.

The main idea behind the backup CBF method is to use S_b and π_b as an initial seed for state and input limiting, which is expanded to larger set of states by studying the flow of the system under the backup controller:

$$\frac{\partial}{\partial \tau} \varphi_b(\tau, x) = f_b(\varphi_b(\tau, x)), \quad \varphi_b(0, x) = x, \quad (30)$$

where $f_b(x) = f(x) + g(x)\pi_b(x)$. The flow $\varphi_b(\tau, x)$ is simply the evolution of our system starting at state x , using the backup controller, and evaluated at time τ . Based on (30), we define the expanded set $S_I \subseteq C$ as:

$$S_I \triangleq \left\{ x \in \mathbb{R}^n \mid \begin{array}{l} h_c(\varphi_b(\tau, x)) \geq 0, \forall \tau \in [0, T], \\ h(\varphi_b(T, x)) \geq 0 \end{array} \right\}, \quad (31)$$

which is the set of all states from which the backup flow i) remains in the state constraint set C up until time T and ii) returns to the backup set after T seconds. Note that $T > 0$ is a finite horizon that determines how far in the future the flow is evaluated. We also note that in case f_b is linear, $\varphi_b(\tau, x)$ has an analytical solution, but for general nonlinear systems this flow must be obtained via numerical techniques.

The definition (31) yields a powerful result: the expanded set S_I is controlled invariant. We omit the proof of this statement, but the interested reader is referred to [53] and [56]. Informally, this can be verified by recognizing that π_b renders S_I forward invariant, and π_b satisfies the input constraints by definition. Having formally defined the expanded set, we now present the main result of backup CBFs.

Theorem 5 ([53]). Any locally Lipschitz controller $\pi : \mathbb{R}^n \rightarrow U$, $u = \pi(x)$ satisfying

$$\dot{h}_c(\varphi_b(\tau, x), u) \geq -\kappa h_c(\varphi_b(\tau, x)), \quad (32)$$

$$\dot{h}(\varphi_b(T, x), u) \geq -\kappa h(\varphi_b(T, x)), \quad (33)$$

for $\kappa > 0$ renders the set S_I defined in (31) forward invariant for (1), and implies that the state constraints will be satisfied for all time. The backup controller π_b is one such controller.

From Theorem 5, one can form an optimization problem to solve for a control that enforces the forward invariance of S_I , whilst attempting to remain as close as possible to a baseline controller³. This problem takes the form:

$$\min_{u \in U} \frac{1}{2} \|u - \pi_{bl}(x)\|^2 \quad \text{subject to } (32), (33), \quad (34)$$

for all $\tau \in [0, T]$. We note that, similar to (13), this optimization problem is a strictly convex quadratic program when the input constraint set U is a convex polytope. Because S_I is a controlled invariant set, this optimization problem is known to be feasible, even in the presence of input constraints.

The optimization problem in (34) yields a control signal that satisfies the input and state constraints. However, for some aerospace applications it is undesirable to solve a numerical optimization problem in the control loop, e.g., when formal certification protocols do not allow for them. Hence, some works [25], [26] have explored the notion of *controller interpolation*, which involves interpolating or

³We note that in general, using a linear “CBF gain” κ in (32) and (33) may not be sufficient for safety assurance, and thus constraints (32) and (33) are typically presented using a class- $\mathcal{K}_\infty$ function $\kappa(\cdot)$ [53], [56].

mixing controllers to achieve state and input constraint satisfaction. Consider the mixed controller $\pi_m : \mathbb{R}^n \rightarrow U$ as:

$$\pi_m(x) = (1 - \mu(x))\pi_{bl}(x) + \mu(x)\pi_b(x), \quad (35)$$

where $\mu : \mathbb{R}^n \rightarrow [0, 1]$ is a blending function that regulates the weights of each controller, π_{bl} and π_b . Provided the baseline controller satisfies input bounds, the mixed controller π_m also satisfies these bounds. Under certain parameterizations of this blending function, (35) can be shown to render S_I forward invariant (see [25, Prop. 1]). One example is:

$$\mu(x) = e^{-\eta \max\{h_I(x), 0\}}, \quad (36)$$

with $\eta > 0$, where h_I is defined as:

$$h_I(x) \triangleq \min \left\{ \min_{\tau \in [0, T]} h_c(\varphi_b(\tau, x)), h(\varphi_b(T, x)) \right\}. \quad (37)$$

Since $h_I(x) = 0$ at the boundary of S_I , when using (36) we have $\mu(x) = 1$, and thus π_m reduces to π_b , which renders S_I forward invariant. Meanwhile, in the interior of S_I , a mix of the backup controller and baseline controller is used.

Blending approaches such as in (36) have illustrated success on resource-constrained robotic systems [26], but must be obtained through ad-hoc means, and may produce undesirable behavior near ∂S_I . Recent work [54] addresses this limitation by proposing a controller that optimally chooses the blending parameter μ . Though not discussed in detail here, this optimal blending parameter can be obtained as a *closed-form* solution to an optimization problem that seeks to minimize μ , subject to similar constraints as in (32) and (33). Such an approach offers the best of both worlds—a controller that ensures the forward invariance of S_I , whilst minimizing the deviation from the baseline controller, and bypassing the need for online numerical optimization.

To highlight the effectiveness of backup CBFs, we consider an example of a fixed-wing aircraft in the presence of geofencing constraints, originally published in [54]. Here, we consider a slightly different model than that in (25) as:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{p}_N \\ \dot{p}_E \\ \dot{H} \\ \dot{P} \\ \dot{N}_z \end{bmatrix} = \underbrace{\begin{bmatrix} P + \frac{N_z g_D}{V_T} \sin \phi \tan \theta \\ \frac{g_D}{V_T} (N_z \cos \phi - \cos \theta) \\ \frac{N_z g_D \sin \phi}{V_T \cos \theta} \\ V_T \cos \theta \cos \psi \\ V_T \cos \theta \sin \psi \\ V_T \sin \theta \\ -\frac{1}{\tau_p} P \\ -\frac{1}{\tau_z} N_z \end{bmatrix}}_{f(x)} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{\tau_p} & 0 \\ 0 & \frac{1}{\tau_z} \end{bmatrix}}_{g(x)} \underbrace{\begin{bmatrix} u_1 \\ u_2 \end{bmatrix}}_u, \quad (38)$$

The derivation of these dynamics can be found in [57]. In (38), the states ϕ , θ , ψ , p_N , p_E , H , and P , and N_z represent the roll, pitch, and yaw angles, the north and east position coordinates, altitude, roll rate, and normal load factor, respectively. The constants V_T , g_D , τ_p and τ_z are the true airspeed, gravitational acceleration, and the time constants for the roll mode and load factor dynamics, respectively. The bounded inputs are the commanded roll rate $u_1 \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ rad/s and commanded load factor $u_2 \in [-1, 4]$ (representing normal accelerations measured in g_D that pull the aircraft up).

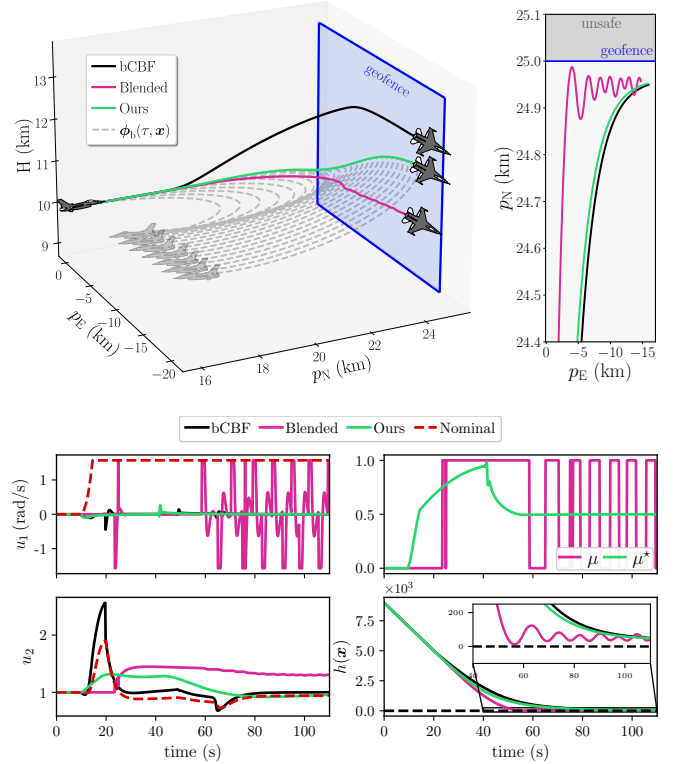


Fig. 4: Simulation of the aircraft (38) comparing (34) (black), the function-based blended controller in (35) using the blending function (36) (pink) and the optimal blending approach (green). All three methods obey the geofence constraint, though the blended controller experiences oscillations along the geofence (top right). These results were originally published in [54].

The state constraint is a geofence with location $p_g \in \mathbb{R}^2$ and normal vector $n_g \in \mathbb{R}^2$ with $\|n_g\| = 1$. With $p = (p_N, p_E)$, the state constraint set C is written as:

$$C \triangleq \{x \in \mathbb{R}^8 : h_c(x) = n_g \cdot (p - p_g) \geq 0\}. \quad (39)$$

The backup controller tracks a circular orbit in the interior of C (a circular orbit that does not cross the geofence). Thus, the backup controller renders a region around the circular orbit invariant, since it will maintain this orbit indefinitely. Details on the implementation may be found in [54].

Fig. 4 plots the simulation results comparing the approach based on numerical optimization in (34), the function-based blending approach, and the optimal blending approach. All three methods successfully maintain the geofence constraint. Further, they all satisfy the input bounds imposed by U , one of the key advantages of backup CBFs. This example also highlights the downside of the function-based blending approach. Since μ is selected in an ad-hoc manner, the blending parameter oscillates near the boundary of the geofence (see Fig. 4(b)). In contrast, the optimal interpolation approach in green induces a smooth transition near the boundary of C .

V. LAYERED CONTROL ARCHITECTURES

The control stack for complex systems, such as fixed-wing aircraft, are typically divided into multiple layers, with each layer possibly using a different model representation of the system or running at a different loop rate. The canonical

example of this is the guidance-navigation-control architecture, ubiquitous in aerospace systems. Here, the control architecture is typically divided into an inner-loop (control) that tracks reference commands while ensuring essential stability properties such as gain and phase margins, and an outer-loop (guidance) that generates reference commands for the low-level inner-loop controller. In this section, we present a comprehensive framework for using CBFs in layered control architectures (LCAs). We illustrate how implementing CBFs at higher layers of the control architecture (guidance) using reduced-order models to generate reference commands for lower-level layers (control) has led to the successful deployment of CBF-based approaches on complex systems. We will discuss the theoretical foundations for this approach as well highlight successful case studies. The technical content and applications in this section will be based on [58]–[60].

A. Mathematical Foundations of Layered Control

We focus on CBFs in LCAs consisting of two layers: a top layer generating reference commands based on a simplified, reduced-order model (ROM) and a bottom layer wherein the full-order system tracks those commands. We assume that the full-order dynamics are described by:

$$\dot{X} = F(X, U), \quad (40)$$

with state $X \in \mathbb{R}^N$ and input $U \in \mathbb{R}^M$. This could be, for example, a high-fidelity six-degrees-of-freedom model of a quadrotor or aircraft. For control design, we leverage a ROM:

$$\dot{x} = f(x) + g(x)u + d \quad (41)$$

with state $x \in \mathbb{R}^n$, input $u \in \mathbb{R}^m$, and disturbance $d \in \mathbb{R}^n$. The ROM may represent, for example, the kinematics of an aircraft or a simple single integrator model for a quadrotor.

The reduced-order state x captures variables that we wish to constrain, such as position and orientation in case of collision avoidance. These are selected from the full state X via a state projection map $x = p(X)$. Hence, the state constraints of the full system are captured by the set:

$$S = \{X \in \mathbb{R}^N \mid h(x) \geq 0\}, \quad (42)$$

with a function h that is assumed to be a CBF for the ROM. The reduced-order input u represents other state variables, such as velocity or angular velocity, that determine the evolution of the constrained states in x , and are given by a control projection map $u = q(X)$. They are viewed as inputs to the ROM as well as reference commands that will be tracked by (40). The disturbance d in (41) captures the discrepancy between the ROM and the true dynamics of the reduced-order states. We assume that d is unknown but bounded, so that there exists $\delta > 0$ such that $\|d\| \leq \delta$.

We employ a LCA that leverages both the full-order and the ROM representations. First, we equip the full system (40) with a low-level controller $\Pi : \mathbb{R}^N \rightarrow \mathbb{R}^M$, $U = \Pi(X)$, that is able to track a reference command u_{ref} for the reduced-order input u . This can be, for example, a flight controller for a quadrotor that tracks velocity or angular velocity commands. Second, we design a high-level controller $\pi : \mathbb{R}^n \rightarrow \mathbb{R}^m$, $u_{\text{ref}} = \pi(x)$, that determines the reference

command based on the reduced-order state x . This will be done using a CBF for the ROM (41) so as to guarantee the satisfaction of state constraints. For example, u_{ref} can be a CBF-based velocity at which a quadrotor must fly to avoid collision. The key idea of layered control design is that state constraints can be enforced for the full system if the low-level controller tracks reference commands sufficiently well and the high-level commands are carefully designed using CBFs. Next, we provide the underlying technical requirements.

For the low-level tracking controller, we require that it provides stable tracking of the ROM in the sense that there exists a smooth Lyapunov function $V : \mathbb{R}^n \rightarrow \mathbb{R}$ satisfying:

$$V(X) \geq \|u - \pi(x)\|^2, \quad \dot{V}(X) \leq -\gamma V(X) + \iota. \quad (43)$$

This means that the low-level controller drives the tracking error $e = u - u_{\text{ref}} = u - \pi(x)$ exponentially to a neighborhood of zero. The parameter $\gamma > 0$ represents the convergence rate and $\iota > 0$ accounts for the fact that tracking may be imperfect and the error e may not become exactly zero. The design of tracking controllers may require knowledge of the full dynamics (40). Such controllers are well established and readily available on many aerospace platforms. Note that V , γ , and ι do not necessarily need to be known for the upcoming CBF design, we just need them to exist.

For the high-level controller, we employ a CBF h designed for the ROM (41), and synthesize a controller π such that:

$$\begin{aligned} \frac{\partial h}{\partial x}(x)(f(x) + g(x)\pi(x)) &\geq -\kappa h(x) \\ &+ \frac{1}{\varepsilon} \left\| \frac{\partial h}{\partial x}(x)g(x) \right\|^2 + \frac{1}{\sigma} \left\| \frac{\partial h}{\partial x}(x) \right\|^2. \end{aligned} \quad (44)$$

The first line contains the CBF condition (9), while the two additional terms in the second line provide robustness against the tracking error e and the model uncertainty d , respectively, with tunable robustness parameters $\varepsilon, \sigma > 0$. The condition in (44) can be enforced by a modification of controller (11)–(12) where the robustness terms in the second line of (44) are subtracted from $\kappa h(x)$. The high-level controller contains three important design parameters: the CBF gain κ and the robustness gains ε, σ . As discussed below, if these parameters are chosen carefully, then the LCA ensures the satisfaction of state constraints for the full system.

The LCA can satisfy the state constraints for (40) under the following condition. The CBF gain κ of the high-level controller must be chosen to be small enough so that:

$$\gamma > \kappa. \quad (45)$$

This requires that the high-level commands do not allow the ROM to approach state limits faster (given by κ) than the low-level controller tracks reference commands (given by γ).

Under this condition, we can construct a certificate for the full system from the CBF of the high-level ROM and the Lyapunov function characterizing low-level tracking:

$$H(X) = h(x) - \frac{1}{\mu} V(X), \quad (46)$$

with $\mu = \frac{4}{\varepsilon}(\gamma - \kappa)$. Then, we define the superlevel sets of the certificate function H :

$$S_\Delta = \{X \in \mathbb{R}^N \mid H(X) \geq -\Delta\}, \quad (47)$$

with $\Delta \in \mathbb{R}$. It can be shown that S_Δ is forward invariant for a specific value of Δ [59].

Theorem 6. Consider the full-order system (40) with the ROM (41). Let a high-level controller satisfying (44) generate reference commands based on a CBF for the ROM, and a low-level controller satisfying (43) track those commands. If the CBF gain κ satisfies condition (45), then the set S_Δ in (46)-(47) is forward invariant for $\Delta = \frac{1}{\kappa}(\frac{\sigma}{4}\delta^2 + \frac{\epsilon}{\mu})$:

$$X(0) \in S_\Delta \implies X(t) \in S_\Delta, \forall t \geq 0.$$

The value of Δ determines which level set of H is forward invariant, and affects the satisfaction of state constraints. If $\Delta = 0$, then $X(t) \in S_\Delta$ implies $x(t) \in S$, hence the state constraints are satisfied by the full system. This is achieved in the ideal scenario where the ROM accurately represents the evolution of the reduced-order states ($\delta = 0$) and the low-level controller exponentially drives the tracking error to zero ($\iota = 0$). Model uncertainties ($\delta \neq 0$) and imperfect tracking ($\iota \neq 0$) may lead to slight constraint violations quantified by $\Delta > 0$ in the sense that $h(x(t)) \geq -\Delta$ rather than $h(x(t)) \geq 0$. These violations, however, can be made as small as desired by decreasing the parameters σ and ϵ .

Overall, we can achieve practical guarantees for satisfying state constraints on complex systems by using LCAs that leverage low-level tracking of carefully designed high-level commands. This requires: (i) a stable low-level controller that tracks reference commands (as in (43)), (ii) a CBF and a corresponding high-level controller for the ROM (as in (44)), and (iii) tuning the CBF gain κ to satisfy (45).

B. High-fidelity Simulation of Fixed-wing Aircraft

Next, we showcase LCAs for fixed-wing aircraft. Following the results from [27], we highlight the design of a CBF-based flight controller for the Variable-stability In-flight Simulator Test Aircraft (VISTA)—a modified F-16 jet shown in Fig. 5. In particular, we design a flight controller for executing collision avoidance and simulate it on a high-fidelity model of the VISTA. The flight controller leverages a LCA: a high-level controller provides motion commands and a low-level controller tracks those commands.

To capture the six-degrees-of-freedom dynamics of the VISTA with high fidelity, we use an implementation of its linearized, stitched model [61]. This involves a combination of linear state-space models with time delays that were fitted to flight test data at various flight conditions. The VISTA model is equipped with a low-level controller capable of tracking commands expressed in terms of power lever angle, roll rate, and normal load factor. Furthermore, the VISTA also has a neural network controller that provides these commands for achieving stable wing-level flight at given altitude, heading, and speed. The neural network controller, however, does not account for collision avoidance constraints.

To ensure that the VISTA avoids collision with other aircraft, we use CBFs to generate high-level commands that can be tracked by the low-level controller. To this end, we use the method in Section III-C, and design the high-level commands using the 3D Dubins model and backstepping. First, we

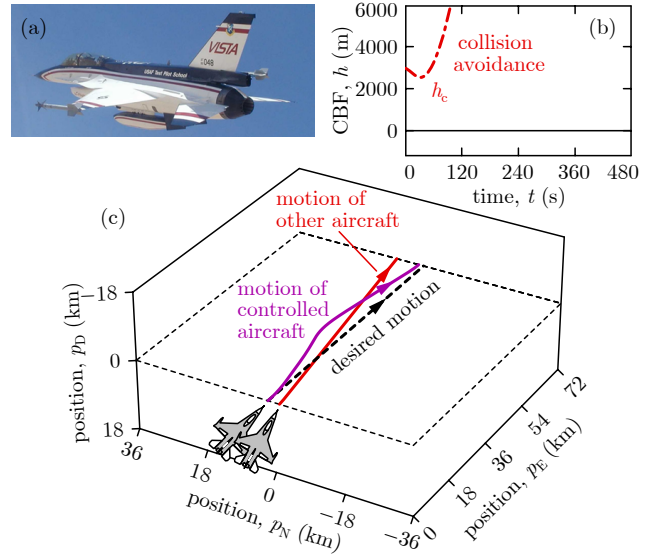


Fig. 5: High-fidelity simulation demonstrating successful collision avoidance for the VISTA, from [27]. The VISTA uses an LCA, where a low-level controller tracks acceleration and angular velocity commands that come from a high-level controller designed for the Dubins model (25)-(26) with a backstepping CBF (28).

view the neural network controller as a baseline controller. We take its commands, and convert them into acceleration and angular velocities (A_T , P , and Q). Then, we use the CBF augmentation to modify these acceleration and angular velocity commands using the controller in (11)-(12) along with the 3D Dubins model (25)-(26) and the backstepping CBF (28). Finally, the resulting CBF-augmented acceleration and angular velocity commands are converted back to power lever angle, roll rate, and normal factor, which are then tracked by the low-level controller of the VISTA.

The resulting behavior is demonstrated in Fig. 5. Thanks to its LCA, the VISTA successfully avoids a collision with another aircraft. This is made possible by stable and sufficiently fast low-level tracking of CBF-based high-level commands. When designing these commands, the CBF parameter κ needs to be selected small enough to overcome the effect of transient tracking errors. This way, the high-fidelity aircraft model avoids collision, even though the high-level commands are designed based on a simplified kinematic model.

C. Flight Tests Using Guardrails

Finally, the LCA paradigm was implemented and experimentally tested on the VISTA in collaboration with the US Air Force Test Pilot School. The underlying control system – called *Guardrails* – marks the first real-life execution of CBF-based flight control on fixed-wing aircraft. Now we briefly overview these experiments, with more details in [29]. Fourteen test scenarios were executed during five flights at the Edwards Air Force Base. During these flight tests, Guardrails was used to supervise a human pilot and enforce geofence constraints, altitude limits (ceiling and floor), g-limits (i.e., limits on the normal load factor), and the combination thereof, even in the case of adversarial pilot inputs that intended to violate these constraints.

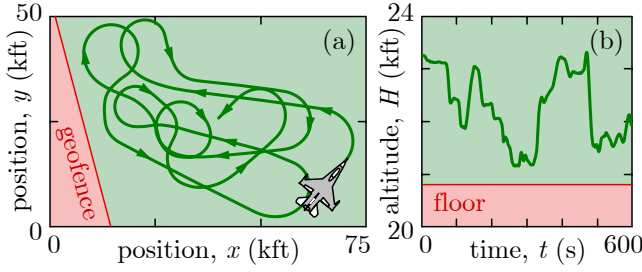


Fig. 6: Flight tests demonstrating Guardrails on the VISTA, from [29]. Guardrails modifies pilot commands to enforce a geofence constraint, an altitude floor limit, and g-limits. This is achieved using an LCA where a low-level controller (regulating the control surfaces) tracks high-level commands (normal load factor and roll rate) which were obtained based on the backup CBF-based blended controller (35) and the reduced order model (38).

In these tests, Guardrails used an instantiation of the backup CBF-based blended controller in (35) as in Sec. IV, which was designed based on the ROM in (38). The baseline controller was considered to be the human pilot's input, and the backup controller was designed to execute a coordinated turn while bringing the aircraft to a constant altitude. Guardrails modified the pilot inputs to ensure the satisfaction of both state and input limits. The resulting high-level commands (normal load factor and roll rate) were passed to a low-level tracking controller in the existing control stack of the VISTA to determine required control surface deflections.

The results of one flight test are illustrated in Fig. 6. Here, Guardrails enforced the combination of a geofence constraint, an altitude floor, and g-limits, while the pilot attempted to violate these constraints. Thanks to its LCA and careful CBF-based control design, Guardrails maintained the satisfaction of both state and input limits during the entire 10-minute experiment. Importantly, Guardrails intervened only when it was deemed necessary for enforcing constraints, else it gave back the control authority to the pilot. That is, it allowed the pilot to fly away from the geofence and the floor and execute maneuvers within the g-limits, but it intervened when the pilot tried to fly through the geofence or the floor and in case of too hard pulls on the stick. By blending pilot commands with CBF-based actions, Guardrails controlled the jet with formal guarantees of satisfying operation constraints.

VI. MIN-NORM OPTIMAL CBF SERVO-CONTROLLERS

A resounding theme of the preceding sections is that ensuring operation within a valid flight envelope while preventing actuator saturation is critical for aerospace applications. While earlier sections addressed *outer* loop constraint management, this section addresses CBF augmentations for *inner* loop flight critical control applications, such as aircraft primary flight control systems. Recent work ([62]–[65]) presents a systematic design for robust linear proportional-integral (PI) servo-controllers that enforce input and output constraints. The proposed architecture applies directly to inner loop flight control, where robust reference command tracking requirements, Anti-Windup (AW) protection, and operational constraints are simultaneously imposed.

Using Nagumo's Theorem [19] and the Comparison Lemma [66, Lem. 3.4], we derive min-norm optimal con-

trol laws—analogueous to CBFs—that ensure constraint satisfaction via an analytic continuous piecewise-linear state feedback policy. Compared to standard CBF-QP practice, the main advances are the operator-based conversion of higher-relative-degree box constraints into enforceable linear constraints [39], the closed-form min-norm solution, embedded AW for PI controllers that accommodates input limits, and tractable closed-loop dynamics that allow for classical stability-robustness analysis. This makes the method especially applicable to LTI multi-input multi-output (MIMO) applications, such as flight control, where analytic guarantees and verification of closed loop performance are necessary. Additionally, the CBF augmentation, traditionally computed as the solution to an optimization problem, can be expressed in closed-form, obviating the need to run an optimizer in the control loop. We discuss the core theoretical foundations of this approach and demonstrate applicability of the main result to the longitudinal control of open-loop unstable short-period dynamics. We refer to [62]–[65] for technical details.

A. Constrained Servo-Control Problem and Solution

Consider the control of an n_p^{th} -order LTI MIMO system whose state variables are accessible described by

$$\begin{aligned}\dot{x}_p(t) &= A_p x_p(t) + B_p u(t), \\ y_{\text{reg}}(t) &= C_{p_{\text{reg}}} x_p(t) + D_{p_{\text{reg}}} u(t), \\ z_{\text{lim}}(t) &= C_{p_{\text{lim}}} x_p(t),\end{aligned}\quad (48)$$

where $x_p \in \mathbb{R}^{n_p}$, $u \in \mathbb{R}^m$ is the control input, $y_{\text{reg}} \in \mathbb{R}^m$ represents the regulated output, and $z_{\text{lim}} \in \mathbb{R}^m$ is the limited output. The control goal is to design $u(t)$ to achieve robust tracking of an exogenous reference command y_{cmd} , while ensuring that the operational limits

$$\begin{bmatrix} u^{\min} \\ z_{\text{lim}}^{\min} \end{bmatrix} \leq \begin{bmatrix} u(t) \\ z_{\text{lim}}(t) \end{bmatrix} \leq \begin{bmatrix} u^{\max} \\ z_{\text{lim}}^{\max} \end{bmatrix}, \quad (49)$$

are satisfied. To that end, we let

$$u(t) \triangleq \overbrace{-(K_I e_{y_I}(t) + K_P x_p(t)) + w(t)}^{u_{\text{bl}}(t)}, \quad (50)$$

$$\dot{e}_{y_I}(t) \triangleq y_{\text{reg}}(t) - y_{\text{cmd}} + v(t), \quad (51)$$

which define the total control input as a baseline robust servomechanism state-feedback control law $u_{\text{bl}}(t)$ augmented with $w(t) \in \mathbb{R}^m$, and the integrated output tracking error dynamics $\dot{e}_{y_I}(t)$ augmented with $v(t) \in \mathbb{R}^m$, respectively. With a robust baseline controller established, the next aim is to derive a suitable augmentation mechanism for the components, $v(t)$ and $w(t)$, such that (49) is satisfied. Next, we reformulate the dynamical system using (48)–(51) as

$$\begin{bmatrix} \dot{e}_{y_I} \\ \dot{x}_p \end{bmatrix} = \begin{bmatrix} 0 & C_{p_{\text{reg}}} \\ 0 & A_p \end{bmatrix} \begin{bmatrix} e_{y_I} \\ x_p \end{bmatrix} + \begin{bmatrix} 0 & D_{p_{\text{reg}}} \\ 0 & B_p \end{bmatrix} \begin{bmatrix} -y_{\text{cmd}} + v \\ u_{\text{bl}} + w \end{bmatrix}, \quad (52)$$

$$y_{\text{lim}} = \begin{bmatrix} u_{\text{bl}} \\ z_{\text{lim}} \end{bmatrix} = \underbrace{\begin{bmatrix} -K_I & -K_P \\ 0 & C_{p_{\text{lim}}} \end{bmatrix}}_{C_{\text{lim}}} \begin{bmatrix} e_{y_I} \\ x_p \end{bmatrix}, \quad (53)$$

where $\tilde{u}_{bl} = [-y_{cmd}^\top u_{bl}^\top]^\top$ and $\pi_{cbf} = [v^\top w^\top]^\top$. The dynamics in (52) and (53) are shown in block diagram form in Figure 1, where the CBF augmentation is comprised of both a reference command modification (through the AW component of the integrated error state in (51)) and a baseline control augmentation.

1) *QP with Modified Operational Constraints and Control Augmentation Solution:* Prior to solving for the optimal augmentation policy using a QP as in (13), we must invoke a standard modification often performed in constrained optimization problems where the constraints show direct dependence on the control augmentation policy π_{cbf} . This modification entails the use of vector relative degree of the system limited output with respect to the control input. Thus, y_{lim} in (53) can be rewritten explicitly in terms of relative degree, $1 \leq r_i \leq n$, $\forall i \in \{1, \dots, 2m\}$ system parameters such that $(y_{lim})_i^{(r_i)} = (C_{lim})_i [A^{r_i}x + A^{r_i-1}Bu]$. The corresponding constraints in (49) are then altered to embed the forward invariance criterion from the Nagumo Theorem using stable polynomials of order r_i , with real roots $\{\lambda_{ij}\}_{j=1, \dots, r_i} \in \mathbb{C}^-$ such that $(Y_{lim})_i = \phi_i(s)[(y_{lim})_i^{(r_i)}]$ with $\phi_i(s) = \prod_{j=1}^{r_i} (s - \lambda_{ij}) = \sum_{j=0}^{r_i} c_{ij} s^j$ where s denotes the differential operator $\frac{d}{dt}$ and r_i is the relative degree of the i^{th} limited output with respect to the control augmentation. Hence, we define the system modified output Y_{lim} as

$$Y_{lim} = H_x x + H_{\tilde{u}} (\tilde{u}_{bl} + \pi_{cbf}), \quad (54)$$

and the corresponding modified constraints from (49) as

$$H(x, \tilde{u}_{bl}, \pi_{cbf}) = \begin{pmatrix} -Y_{lim} + \kappa y_{lim}^{\min} \\ Y_{lim} - \kappa y_{lim}^{\max} \end{pmatrix} = \begin{bmatrix} -\mathbb{I} \\ \mathbb{I} \end{bmatrix} H_{\tilde{u}} \pi_{cbf} + \underbrace{\begin{bmatrix} -H_x x - H_{\tilde{u}} \tilde{u}_{bl} + \kappa y_{lim}^{\min} \\ H_x x + H_{\tilde{u}} \tilde{u}_{bl} - \kappa y_{lim}^{\max} \end{bmatrix}}_{\Delta H(x, \tilde{u}_{bl}) = \begin{bmatrix} \Delta H_1(x, \tilde{u}_{bl}) \\ \Delta H_2(x, \tilde{u}_{bl}) \end{bmatrix}} \leq 0, \quad (55)$$

with system state and control sensitivity matrices defined by

$$H_x = \begin{bmatrix} (C_{lim})_1 \prod_{j=1}^{r_1} (A - \lambda_{1j} \mathbb{I}) \\ \vdots \\ (C_{lim})_{2m} \prod_{j=1}^{r_{2m}} (A - \lambda_{(2m)j} \mathbb{I}) \end{bmatrix}, \quad (56)$$

$$H_{\tilde{u}} = \begin{bmatrix} (C_{lim})_1 A^{r_1-1} \tilde{B} \\ \vdots \\ (C_{lim})_{2m} A^{r_{2m}-1} \tilde{B} \end{bmatrix}, \quad (57)$$

with $\det(H_{\tilde{u}}) \neq 0$, and the diagonal matrix $\kappa = \text{diag}(c_{10}, \dots, c_{(2m)0})$ with $c_{i0} = \prod_{j=1}^{r_i} (-\lambda_{ij}) > 0, \forall i = \{1, \dots, 2m\}$. With the modified operational constraints (55) now defined linearly with respect to the control augmentation policy π_{cbf} , we consider the following QP formulation, analogous to (13), as

$$\begin{aligned} \min_{\pi_{cbf} \in \mathbb{R}^{2m}} \quad & J(\pi_{cbf}) = (\pi_{cbf}^T R_\pi \pi_{cbf}) \\ \text{subject to} \quad & \begin{bmatrix} -\mathbb{I} \\ \mathbb{I} \end{bmatrix} H_{\tilde{u}} \pi_{cbf} + \Delta H(x, \tilde{u}_{bl}) \leq 0, \end{aligned} \quad (58)$$

which has a unique optimal solution $\pi_{cbf}^* = [v^\top w^\top]^\top$ [37]. Choosing the cost function weight such that $R_\pi =$

$H_{\tilde{u}}^\top H_{\tilde{u}} > 0$ in (58) with $H_{\tilde{u}}$ in (57), decouples the KKT conditions, yielding explicit computation of dual variables from the associated primal feasibility constraints, resulting in an piecewise linear optimal CBF augmentation solution

$$\pi_{cbf}^* = H_{\tilde{u}}^{-1} \begin{cases} (-H_x x - H_{\tilde{u}} \tilde{u}_{bl} + \kappa y_{lim}^{\min}), & \Delta H_1(x, \tilde{u}_{bl}) > 0, \\ -(H_x x + H_{\tilde{u}} \tilde{u}_{bl} - \kappa y_{lim}^{\max}), & \Delta H_2(x, \tilde{u}_{bl}) > 0, \\ 0 & \text{otherwise.} \end{cases} \quad (59)$$

2) *Core Theoretical Foundations:* This method integrates three principal elements: Nagumo's theorem, the Comparison Lemma, and a min-norm optimal control formulation. Nagumo's theorem is used to convert box constraints into differential inequality conditions on the system velocity at the boundary, ensuring forward set invariance. The Comparison Lemma enables extension to higher-relative-degree outputs by introducing stable polynomial differential operators that reshape scalar constraints into dynamically filtered outputs compatible with first-order Nagumo conditions. These modified constraints are then enforced through a min-norm QP. Owing to the quadratic cost (with R_π chosen suitably), and linear structure, the solution admits an analytic closed-form solution derived from the KKT conditions.

The next section contains examples that complement the theoretical derivations and demonstrates applicability of the main result to flight control problems.

B. Applications to Flight Control

1) *Scalar (Roll Subsidence Mode) Example:* The following illustrates two key components in the control augmentation design by deriving the modified constraints and the min-norm solution for a scalar dynamical system (approximation of an aircraft's roll dynamics) with relative degree $r_1 = 1$,

$$\dot{p} = L_p p + L_{\delta_a} (u_{bl} + \pi_{cbf}), \quad (60)$$

where p ($\frac{\text{rad}}{\text{s}}$) is the stability axis roll rate, with virtual aileron as the control input (rad), and an upperbound on the roll rate $h(p) = p - p_{lim}^{\max} \leq 0$. Since $h(p)$ cannot be manipulated directly, we use a stable polynomial operator $\phi(s) = (s + \kappa)$ to derive modified constraints with π_{cbf} appearing explicitly:

$$\begin{aligned} H(p) &= (s + \kappa)h(p) = \dot{p} + \kappa(p - p_{lim}^{\max}) \\ &= L_p p + L_{\delta_a} (u_{bl} + \pi_{cbf}) + \kappa(p - p_{lim}^{\max}) \leq 0. \end{aligned} \quad (61)$$

The key idea is to design the augmentation π_{cbf} such that (61) holds, which, in turn, guarantees $h(p) \leq 0$. Comparing to (55), it is clear that $H_u = L_{\delta_a}$ and $H_x = (L_p + \kappa)$. While any π_{cbf} satisfying (61) will render (60) forward invariant with respect to $h(p)$, we are interested in a min-norm augmentation that only becomes active to avoid violating constraints. In the scalar case, (61) can directly be used for deriving this min-norm analytical augmentation solution. Let

$$\Delta H_2(p, u_{bl}) = L_{\delta_a} u_{bl} + (L_p + \kappa)p - \kappa p_{lim}^{\max}, \quad (62)$$

then (61) is given by $L_{\delta_a} \pi_{cbf} \leq -\Delta H_2(p, u_{bl})$. Clearly, if $\Delta H_2(p, u_{bl}) < 0$, then $\pi_{cbf} = 0$ is the min-norm solution.

If $\Delta H_2(p, u_{bl}) \geq 0$, $L_{\delta_a} \pi_{cbf} = -\Delta H_2(p, u_{bl})$. Thus,

$$\pi_{cbf}^* = L_{\delta_a}^{-1} \begin{cases} -\Delta H_2(p, u_{bl}), & \Delta H_2(p, u_{bl}) \geq 0, \\ 0, & \text{otherwise.} \end{cases} \quad (63)$$

Substituting (63) with (62) into (60) yields the closed-loop dynamics as $\dot{p} = -\kappa p + \kappa p_{\lim}^{\max}$ during constraint activation, clearly demonstrating the cancellation of baseline control while maintaining closed-loop stability.

2) *Short-Period (Longitudinal) Dynamics*: In this section, we consider a second-order system derived from X-29A aircraft data (see [67], Chapter 14.7). Thus, we define the short-period longitudinal dynamics

$$\begin{bmatrix} \dot{\alpha} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} \frac{Z_{\alpha}}{V_0} & 1 + \frac{Z_q}{V_0} \\ M_{\alpha} & M_q \end{bmatrix} \begin{bmatrix} \alpha \\ q \end{bmatrix} + \begin{bmatrix} \frac{Z_{\delta_e}}{V_0} \\ M_{\delta_e} \end{bmatrix} \delta_e, \quad (64)$$

where the state $x_p \triangleq [\alpha \ q]^T$ is comprised of two components that represent the aircraft angle of attack (rad) and the body axis pitch rate ($\frac{\text{rad}}{\text{s}}$), respectively. The control input δ_e (rad) is defined as the virtual elevator input. Additionally, we define the regulated output as C^* which is a linear combination of vertical load factor $N_z = \frac{A_z}{g}$ (G) and pitch rate that follows $y_{\text{reg}} = -N_z + 12.4q$.

An open-loop linear model, trimmed to 1G wings-level flight at $V = 977 \frac{\text{ft}}{\text{s}}$, $h = 8000\text{ft}$, and $\alpha = 2.8^\circ$ is defined by

$$\dot{x}_p = \begin{bmatrix} -2.241 & 0.9897 \\ 44.74 & -0.9024 \end{bmatrix} x_p + \begin{bmatrix} -0.2331 \\ -45.93 \end{bmatrix} u, \quad (65)$$

$$y_{\text{reg}} = [66.5 \ 12.745] x_p + 8.586u, \quad (66)$$

which are unstable and non-minimum phase with respect to the regulated output. In and of itself, such a system presents a formidable challenge on the reachable levels of stability, robustness, and performance via feedback control. Designing a controller that additionally ensures satisfaction of operational limits is desirable. To that end, we proceed with designing the total control in (50).

We begin with the robust servomechanism architecture and define an LQR state feedback baseline servo-controller as $u_{bl} = -K_{\text{lqr}} x = -[K_i \ K_p] x$ with $K_i = 0.01$ and $K_p = [1.4472 \ 0.24008]$ with $\dot{e}_{y_I}$ defined in (51), and y_{cmd} is a C^* reference command to track. K_{lqr} was computed using properly selected LQR weights that yield adequate stability margins at the system input.

Output limits on pitch rate q were selected, so that $C_{p_{\lim}} = [0 \ 1]$ and $r_2 = 1$. Following the construction above, the augmentations $\pi_{cbf} = [v \ w]^T$ can be computed for the X29A dynamics in (65) and (66) using (59) with H_x in (56), $H_{\bar{u}}$ in (57) and $\kappa_v = \kappa_w = 2$. Pitch rate limits were varied from ± 5 to $\pm 15 \frac{\text{deg}}{\text{s}}$ with δ_e limited to $\pm 5^\circ$, below which there is insufficient control authority for feasibility (see Def. 3).

Figure 7 shows simulation results for a large magnitude doublet command, illustrating the reference command, regulated output, states, total control input, integrator error, and corresponding CBF augmentation components. In the bottom right subplot, pitch rate is plotted vs. the control input, with the constraint limits annotated in red. As expected, tracking performance degrades as the pitch rate becomes increasingly constrained, yet the closed-loop dynamics remain stable and

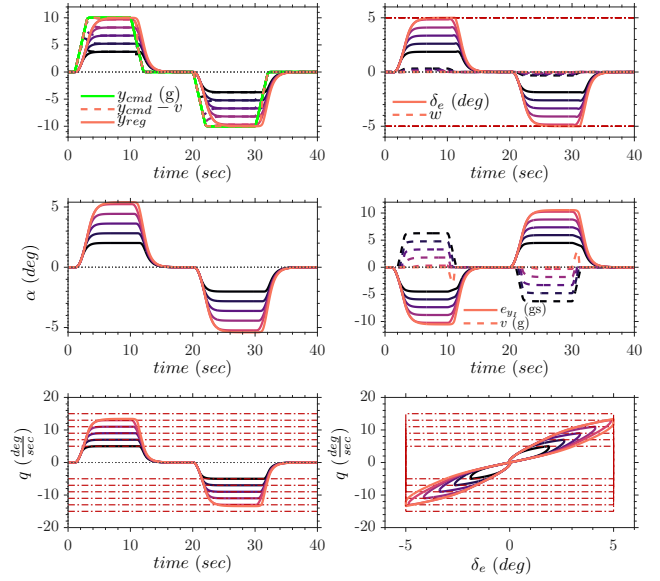


Fig. 7: Simulation results of the X29A short-period dynamics with min-norm CBF augmented controller, C^* command, pitch rate limits $\{5, 7, 9, 11, 13, 15\}$ (deg/s), and δ_e constrained to $\pm 5\text{deg}$.

bounded within the prescribed operational limits, clearly demonstrating the benefits of this method. The next section proposes a CBF architecture that fuses with adaptive control, while considering the the same X29A dynamics.

VII. A CBF APPROACH WITH ADAPTATION IN THE PRESENCE OF UNCERTAINTIES

When significant parametric uncertainties are present, adaptive control is needed both to restore command following performance and to ensure constraint satisfaction. Recent work has developed methods for designing adaptive controllers with CBF augmentations that can simultaneously realize a safety objective and a tracking objective with minimal conservatism [68]–[70]. In this section, we showcase this control design in the context of the X-29A short-period longitudinal dynamics in (64)–(66), modified with an uncertain actuator loss of effectiveness λ_* :

$$\dot{x}_p = A_p x_p + B_p \lambda_* u. \quad (67)$$

We assume that $\lambda_* \in L := [\underline{\lambda}, \bar{\lambda}]$ for known bounds $\bar{\lambda} > \underline{\lambda} > 0$.

It should be noted that control of (67) is a non-trivial problem since the parametric uncertainties make the determination of a stable and safe control input highly difficult. This is especially due to the fact that (67) is open-loop unstable and we impose constraints on states that include an unstable zero. A reduction in control effectiveness, therefore, can directly compromise the ability to simultaneously stabilize the system and satisfy constraints. Our overall objective is for the output y_{reg} in (66) to track a reference y_{cmd} stably, while requiring the states in (67) to lie in

$$C = \{x_p \in \mathbb{R}^2 : \alpha \in [\underline{\alpha}, \bar{\alpha}], q \in [\underline{q}, \bar{q}]\}. \quad (68)$$

A. Adaptive Control Design

We propose an adaptive controller

$$u = \hat{v}(u_{bl}(x) + r_{cbf}), \quad (69)$$

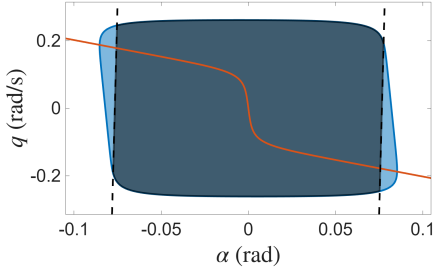


Fig. 8: The smoothed set S (blue), the curve $\frac{\partial h}{\partial x_p} B_p = 0$ (red), and the boundaries of Z (dashed black lines), and $\bar{Z}$ (black), in the original and transformed state spaces. In this example, C in (68) is characterized by $\bar{\alpha} = -\underline{\alpha} = 0.087\text{rad}$ and $\bar{q} = -\underline{q} = 0.26\text{rad}$.

where $x := [e_{y_I}^\top, x_p^\top]^\top$ [67], $u_{bl}(x) = -K_{lqr}x = -[K_i, K_p]^\top x$ as in Section VI-B.2, K_p is chosen so that $A_m := A_p - B_p K_p$ is Hurwitz, K_i is chosen so that $A_{cl} := A - B K_{lqr}$ is Hurwitz with A and B defined as in [67], $\hat{\vartheta}$ is an estimate of $1/\lambda_*$, and r_{cbf} is a CBF augmentation as in (11). The reference model, representing the desired closed-loop dynamics, is given by

$$\dot{x}_m = A_m x_m + B_p (K_i e_{y_I} + r_{cbf}). \quad (70)$$

We adjust the parameter estimate as

$$\dot{\bar{\vartheta}} = -\gamma (u_{bl}(x) + r_{cbf}) B_p^\top P e_x, \quad (71a)$$

$$\dot{\hat{\vartheta}} = \begin{cases} 0, & (\hat{\vartheta} \geq \frac{1}{\underline{\lambda}} \wedge \bar{\vartheta} \geq 0) \vee (\hat{\vartheta} \leq \frac{1}{\bar{\lambda}} \wedge \bar{\vartheta} \leq 0) \\ \bar{\vartheta}, & \text{otherwise} \end{cases} \quad (71b)$$

where $\gamma > 0$, and $P = P^\top > 0$ is the solution to

$$A_m^\top P + P A_m = -Q, \quad (72)$$

for any symmetric positive-definite Q . Global stability and tracking of x_m by x can be guaranteed using well known methods [71] for any bounded r_{cbf} .

B. CBF Augmentation Design Under Uncertainty

We now design the CBF augmentation r_{cbf} in (69) using an Error-Based Safety Filter (EBSF) proposed in [70]. This choice is given by

$$\begin{aligned} r_{cbf} &= \arg \min_{r \in \mathbb{R}} \frac{1}{2} \|r\|^2 \\ \text{subject to } \min_{\lambda \in L} \nu(x_p) &[(1 - \mu(x_p, e_x)) \bar{h}_{\text{ref}}(x, r) \\ &+ \mu(x_p, e_x) \bar{h}_{\text{robust}}(x, r, \hat{\vartheta}, \lambda)] \\ &\geq -\kappa h(x_p) + \delta, \end{aligned} \quad (73)$$

for any constants $\kappa > 0$, $d_0 \in (0, d_1)$, and $\delta \in (0, \kappa d_0)$, where $\bar{h}_{\text{ref}}(x, r)$ and $\bar{h}_{\text{robust}}(x, r, \hat{\vartheta}, \lambda)$ are given by

$$\bar{h}_{\text{ref}} = \frac{\partial h}{\partial x_p} (A_p x_p + B_p (u_{bl}(x) + r_{cbf})), \quad (74)$$

$$\bar{h}_{\text{robust}} = \frac{\partial h}{\partial x_p} (A_p x_p + B_p \lambda \hat{\vartheta} (u_{bl}(x) + r_{cbf})). \quad (75)$$

To obtain h in (74)-(75), we use the log-sum-exponential approach described in (27) as follows:

$$h(x_p) = -\frac{1}{\eta} \ln \left(\sum_{o \in \chi} \exp(-\eta h_o(x_p)) \right), \quad (76)$$

$$h_{\underline{q}}(x_p) = c_q(q - \underline{q}), \quad h_{\bar{q}}(x_p) = c_q(\bar{q} - q),$$

$$h_{\underline{\alpha}}(x_p) = \alpha - \underline{\alpha} - c_q h_{\underline{q}}(x_p), \quad h_{\bar{\alpha}}(x_p) = \bar{\alpha} - \alpha - c_q h_{\bar{q}}(x_p),$$

where $\chi = \{\underline{\alpha}, \bar{\alpha}, q, \bar{q}\}$ and $\eta, c_q > 0$. $\mu(x_p, e_x)$ represents an error-based interpolation given by

$$\mu(x_p, e_x) = \text{sat} \left(\frac{\max\{g(\|e_x\|), \frac{2\delta}{3\kappa}\} - h(x_p)}{\max\{g(\|e_x\|), \frac{2\delta}{3\kappa}\} - \frac{\delta}{3\kappa}} \right), \quad (77)$$

for any $g \in \mathcal{K}_\infty$, where $\text{sat}(z)$ is a standard saturation function with range and domain $[0, 1]$ and

$$\nu(x_p) = \text{sat} \left(\frac{d_1 - h(x_p)}{d_1 - d_0} \right), \quad (78)$$

is a relaxation term that ensures that $r_{cbf} \neq 0$ only when $h(x_p)$ is sufficiently small. μ and ν are both parameters that interpolate between constraints in the optimization problem. With r_{cbf} specified as above and the adaptive controller in Section VII-A, the main result of this section follows. In Theorem 7, we define $S := \{x_p \in \mathbb{R}^2 : h(x_p) \geq 0\} \subset C$, and Z is a subset of S that can be shown to exist [70]. Examples of S and Z are sketched in Fig. 8.

Theorem 7 ([70]). *Let $x_p(0) \in Z$ and $\frac{1}{\hat{\vartheta}(0)} \in L$. Then, the closed-loop adaptive system consisting of (67)-(78) has the following properties:*

- 1) $x_p(t) \in S \subset C \quad \forall t \geq 0$,
- 2) $\frac{1}{\hat{\vartheta}(t)} \in L \quad \forall t \geq 0$,
- 3) $x(t), x_m(t), r_{cbf}(t), e_x(t) \in \mathcal{L}_\infty$,
- 4) $\lim_{t \rightarrow \infty} \|e_x(t)\| = 0$, and
- 5) $\exists t_{\text{ebsf}} > 0$ such that $\mu(x_p(t), e_x(t)) = 0 \quad \forall t \geq t_{\text{ebsf}}$.

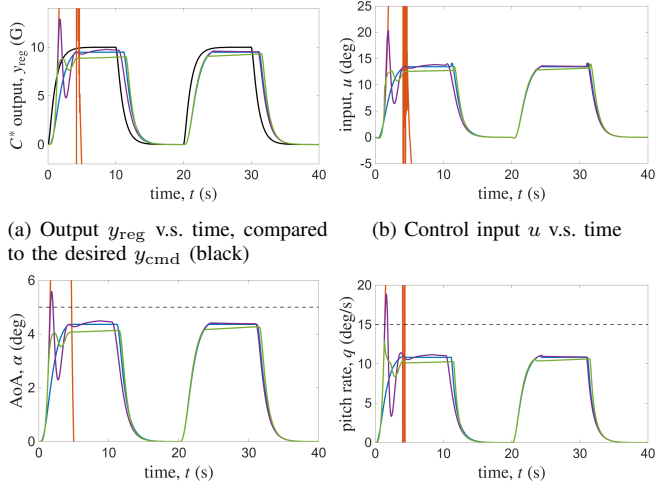
Intuition for $\bar{h}_{\text{ref}}$. Intuitively, $\bar{h}_{\text{ref}}(x, r_{cbf})$ approximates $\frac{d}{dt} h(x_p)$ for the closed-loop dynamics when there is no uncertainty, i.e. when $e_x \approx 0$. μ is designed so that, whenever $e_x \approx 0$, we have $\mu(x_p, e_x) = 0$ except when $h(x_p) \approx 0$, and thus r_{cbf} is a standard CBF augmentation based on the reference dynamics for almost all states in the interior of S .

Intuition for $\bar{h}_{\text{robust}}$. In order to ensure forward-invariance of S , we must ensure that $\frac{d}{dt} h(x_p) \geq 0 \quad \forall x_p \in \partial S$, regardless of the value of e_x . To that end, we define $\bar{h}_{\text{robust}}$ in (75) such that $\frac{d}{dt} h(x_p) = \bar{h}_{\text{robust}}(x, r, \hat{\vartheta}, \lambda_*)$. Then, whenever $\nu(x_p) = \mu(x_p, e_x) = 1$, the minimization over $\lambda \in L$ ensures that (73) produces a controller which is robust to the worst-case parametric uncertainty. Finally, (77) and (78) are designed so that $\nu(x_p) = \mu(x_p, e_x) = 1 \quad \forall x_p \in \partial S$.

Finally, we note that Theorem 7 ensures that $\bar{h}_{\text{robust}}$ brings in conservatism when adaptation is active, and the rest of the EBSF design ensures that the conservatism vanishes as adaptation ceases and does not require parameter learning or external excitation. We refer the reader to [70] where it is shown that (73) can be transformed into a QP problem.

C. Numerical Experiments

We now demonstrate the efficacy of our control design in simulation. We simulate a loss of effectiveness of $\lambda_* = 0.3$ with a priori bounds $\underline{\lambda} = 0.15$ and $\bar{\lambda} = 1$, and we choose state constraint values $\bar{q} = -\underline{q} = 15^\circ/\text{s}$ and $\bar{\alpha} = -\underline{\alpha} = 5^\circ$. The commanded output y_{cmd} is chosen as a series of smoothed step functions as in [67]. For illustrative purposes, the magnitude of y_{cmd} is chosen as $C^* = 10$ (G). The controller parameters are chosen as in [67], Chapter 14.7, with the following distinctions: $\gamma = 30$, $\kappa = 10$, $\eta = 50$,



(a) Output y_{reg} v.s. time, compared to the desired y_{cmd} (black) (b) Control input u v.s. time
(c) α v.s. time, compared to the upper constraint boundary (dashed line) (d) q v.s. time, compared to the upper constraint boundary (dashed line)
Fig. 9: Simulation results compared for four closed-loop systems: the ideal controller with λ_* known (blue), the non-adaptive baseline controller with CBF augmentation (red), an adaptive tracking controller with a standard non-adaptive CBF augmentation (purple), and the proposed adaptive controller with EBSF (green).

and $c_q = 0.02$. We obtain the interior invariant set $Z \subset S$ as illustrated in Fig. 8.

Figures 9a-9d show simulation results for the baseline controller with a non-adaptive CBF augmentation, a standard adaptive tracking controller with a non-adaptive CBF augmentation (obtained by setting $\mu = 0$ in (73)), our proposed control design, and the ideal controller which corresponds to a known λ_* . It is readily apparent from Fig. 9 that adaptation is needed for stability under the parametric uncertainty. As shown in Fig. 9a, both adaptive controllers asymptotically converge to the ideal controller. However, as shown in Figures 9c-9d, EBSF is effective in preventing state constraint violations during the transient period of adaptation, towards the start of the simulation.

VIII. DISTRIBUTED CONSTRAINT FORCES FOR GUIDANCE AND CONNECTIONS TO CBFs

This section introduces the method of Constraint Forces (CF), adapted from [13]–[16], [72], [73], for inter-agent collision avoidance, and provides connections to the foundations of CBFs introduced earlier in this tutorial. Guidance system dynamics are commonly modeled using nonlinearly-coupled double integrator dynamics (c.f. [14], [15]), where one may design a dynamic inversion-based controller that transforms the nonlinear dynamics into the form of decoupled double integrators, in three dimensions [14], [15]. We begin by assuming such a dynamic inversion controller has already been integrated, with the starting point as fully decoupled double integrator dynamics. Consider n agents indexed by i , which are modeled as a double integrator with position $x \in \mathbb{R}^3$, input acceleration command $u \in \mathbb{R}^3$, and dynamics:

$$\ddot{x}_i = u_i, \quad (79)$$

implying three-dimensional acceleration inputs are available. Note that the analytical method presented herein with closed-form solutions may be applied to other classes of nonlinear

dynamics, see for example [13]. CBF methods using numerical processing-based quadratic programming solvers have also been explored more recently in [74] and [75].

The set of n agents is denoted as $\mathcal{V} = \{1, \dots, n\}$. An interaction distance r and a minimum separation distance r_s where $0 < r_s < r$ are defined in the method of constraint forces [15], as shown in Fig. 10 in the example of two dimensions. Agents within the interaction distance r of one another are referred to as neighbors with an associated information link between agents formed. In particular, the neighbors of agent i may be expressed mathematically as $\mathcal{N}(i) = \{j \in \mathcal{V} : \|x_i - x_j\| < r, j \neq i\}$. The problem addressed in this section is the design of a constraint forces-based collision avoidance increment, which augments a baseline control input, similar to the CBF augmentation in (11). In particular, for each agent i , the total input in (79) is designed as the addition of a nominal baseline input $u_{i,bl} \in \mathbb{R}^3$ and a constraint forces-based augmentation $u_{i,cf} \in \mathbb{R}^3$ as:

$$u_i = u_{i,bl} + u_{i,cf}. \quad (80)$$

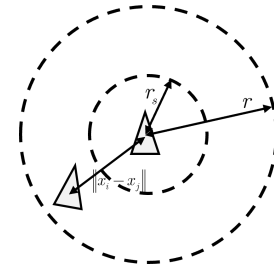


Fig. 10: Overview of distances between two agents.

The position of each two vehicle pair may be expressed by a concatenated position vector of the form:

$$x_{ij} = [x_i^\top \ x_j^\top]^\top, \quad \forall i, j \in \mathcal{V}, i \neq j. \quad (81)$$

Similar to (81), the concatenated form of the baseline input and constraint forces-based augmentation are $u_{ij,bl} = (u_{i,bl}^\top, u_{j,bl}^\top)^\top$, $u_{ij,cf} = (u_{i,cf}^\top, u_{j,cf}^\top)^\top$, $\forall i, j \in \mathcal{V}, i \neq j$. A constraint function amongst each two agent pair i, j is formed from the minimum separation distance r_s and interaction distance r of the form [15]:

$$\phi_{ij}(x_{ij}) = \frac{1}{2} \left[\min \left(0, \frac{\|x_i - x_j\| - r}{\|x_i - x_j\| - r_s} \right) \right]^2. \quad (82)$$

As shown in Fig. 11 the constraint function is positive in the buffer region between the minimum separation distance r_s and interaction distance r . It smoothly decays to zero, and thus becomes inactive beyond the interaction distance r . The constraint increases toward infinity within the buffer region, as the distance between agents tends towards the minimum separation distance r_s . The inverse of the constraint, $1/\phi_{ij}$, may be compared to the specification of the CBF function h and set S' in (5), where $1/\phi_{ij}$ is positive in the desired constraint region, beyond the minimum separation distance r_s , and decays to a value of zero at r_s .

As the dynamics (79) are of relative degree two, with an acceleration-based input with respect to position as the variable for collision avoidance (as in the constraint function in (82)), two time derivatives of the constraint function must

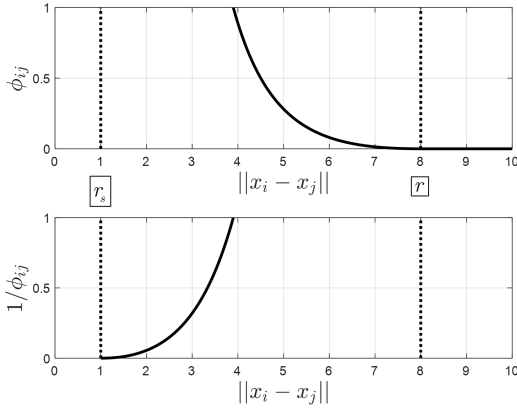


Fig. 11: The constraint function between two agents, and its inverse. be calculated, to arrive at the input. Taking a time derivative of the constraint value in (82), results in:

$$\dot{\phi}_{ij}(x_{ij}, \dot{x}_{ij}) = \frac{\partial \phi_{ij}(x_{ij})}{\partial x_{ij}} \dot{x}_{ij} \triangleq A_{ij}(x_{ij}) \dot{x}_{ij} \triangleq \begin{bmatrix} a_{ij} \\ -a_{ij} \end{bmatrix}^\top \dot{x}_{ij}, \quad (83)$$

where $A_{ij}(x_{ij}) \in \mathbb{R}^{1 \times 6}$ is the position-based constraint *Jacobian*, with available closed form elements parameterized by $a_{ij} \in \mathbb{R}^3$ [14], [15]. The time derivative of the constraint rate in (83) takes the form:

$$\ddot{\phi}_{ij}(x_{ij}, \dot{x}_{ij}, \ddot{x}_{ij}) = \dot{A}_{ij}(x_{ij}, \dot{x}_{ij}) \dot{x}_{ij} + A_{ij}(x_{ij}) \ddot{x}_{ij}, \quad (84)$$

where the concatenated agent acceleration vector $\ddot{x}_{ij}$ contains each agent's total acceleration command as in (79).

A hard constraint may be enforced by selecting the agent accelerations u_i within the concatenated agent acceleration vector $\ddot{x}_{ij}$ such that $\dot{\phi}_{ij}(t) \equiv 0$ for all $t \geq t_0$ assuming that the agent state at t_0 satisfies $\phi_{ij}(t_0) = 0$, $\dot{\phi}_{ij}(t_0) = 0$ [15]. Such a hard constraint, similar to Nagumo's condition (8), may result in an abrupt control augmentation and may not be robust to disturbances nor perturbations. Thus we introduce a feedback-based constraint augmentation similar to the CBF condition (9), in which agents have a buffer region with interaction range r , greater than the minimum separation distance r_s . This feedback-based constraint augmentation couples the dynamics of the agent acceleration and constraint acceleration within the buffer region to provide a buffer region-based augmentation, similar to (10).

In the remainder of this section, the parentheses indicating function dependencies are omitted for compactness of exposition. From (84), the constraint acceleration is designed to be coupled to the agent concatenated acceleration vector by specifying the coupling dynamics of the form [14], [15]:

$$\dot{A}_{ij} \dot{x}_{ij} + A_{ij} \ddot{x}_{ij} = \ddot{\phi}_{ij} \triangleq -k_d \dot{\phi}_{ij} + k_\lambda u_{\phi_{ij}}, \quad (85)$$

where $k_\lambda > 0$ and $k_d > 0$ are two user-defined gains and $u_{\phi_{ij}}$ is a scalar coupling input to be specified shortly.

Using the principle of virtual work, note that [14], [15]:

$$u_{ij,cf}^\top \dot{x}_{ij} = 0, \quad \forall \dot{x}_{ij} \in \{\dot{x}_{ij} \mid \dot{\phi}_{ij} = A_{ij} \dot{x}_{ij} = 0\}, \quad (86)$$

implying that the constraint input action, $u_{ij,cf}$, should be orthogonal to the velocity vectors of the agent interaction that satisfy the constraint velocity condition. Thus to satisfy (86), the input action on the constraint is selected as a scalar

multiple of the constraint Jacobian:

$$u_{ij,cf} = A_{ij}^\top \lambda_{ij}, \quad (87)$$

for a time-varying multiplier λ_{ij} which will be subsequently designed [14], [15]. Recalling that the constraint Jacobian A_{ij} encodes the constraint violation direction, the input augmentation in (87) takes the form of a scalar multiple of a vector along the constraint violation direction.

With coupled constraint dynamics in (85), agent dynamics in (79), total input design in (80), and constraint forces-based input increment in (87), the coupled constraint/agent acceleration dynamics take the form:

$$\dot{A}_{ij} \dot{x}_{ij} + A_{ij} (u_{ij,bl} + A_{ij}^\top \lambda_{ij}) = \ddot{\phi}_{ij} = -k_d \dot{\phi}_{ij} - k_\lambda \lambda_{ij}, \quad (88)$$

where the constraint coupling input from (85) was designed as $u_{\phi_{ij}} = -\lambda_{ij}$ [14], [15]. The scalar multiplier from (88) may then be solved for as:

$$\lambda_{ij} = -\frac{\dot{A}_{ij} \dot{x}_{ij} + A_{ij} u_{ij,bl} + k_d \dot{\phi}_{ij}}{k_\lambda + \|A_{ij}\|^2}. \quad (89)$$

From the parameterization of the constraint Jacobian in (83) with constraint forces-based increment in (87) and multiplier in (89), a distributed form [14], [15] of the inter-agent collision avoidance increment may be designed for each agent i as the summation of the increment over neighbors of each respective agent, i.e. agents within the interaction distance r of one another as:

$$u_{i,cf} = \sum_{j \in \mathcal{N}(i)} [I_n \quad 0_n] A_{ij}^\top \lambda_{ij} = \sum_{j \in \mathcal{N}(i)} a_{ij} \lambda_{ij}. \quad (90)$$

Note the similarity between the CF physics-based derivation and CBF derivation. In particular, both CF and CBF increments take the form of augmentations of baseline control inputs (c.f. (80) and (11)). The CF increment in (87), respectively CBF increment in (12), furthermore both take the form of a scalar multiple of a Jacobian function. The Jacobian multipliers for both CF in (89) and CBF in (12) also share comparable forms, where the CF multiplier includes a scalar regularization, k_λ , in the multiplier denominator. This scalar multiplier will be shown to form the key interconnection between plant and constraint dynamics in the stability analysis outline presented in the remainder of this section.

We now outline a stability analysis for the setting of two agents to elucidate the interconnection between an example baseline tracking input design and the constraint forces-based augmentation. The presented analysis follows from [14], [15]. Given unique static desired positions for each agent $x_{i,des}$, $i \in \{1, 2\}$, a common form of the baseline input to promote agent convergence to the respective desired positions may be designed as:

$$u_{i,bl} = -k_P (x_i - x_{i,des}) - k_D \dot{x}_i, \quad (91)$$

where $k_P > 0$ and $k_D > 0$ are user-designed gains. The baseline input in (91) takes the form of a weighted position tracking error, with velocity-based damping. With concatenated desired position vector $x_{ij,des} = [x_{i,des}^\top \ x_{j,des}^\top]^\top$, $i = 1, j = 2$, a candidate function may be formed as:

$$V = \frac{k_P}{2} \sum_{i=1}^2 \|x_i - x_{i,des}\|^2 + \frac{1}{2} \sum_{i=1}^2 \|\dot{x}_i\|^2 + \frac{1}{2k_\lambda} \dot{\phi}_{ij}^2 \\ = \frac{k_P}{2} \|x_{ij} - x_{ij,des}\|^2 + \frac{1}{2} \|\dot{x}_{ij}\|^2 + \frac{1}{2k_\lambda} \dot{\phi}_{ij}^2. \quad (92)$$

The time derivative of V in (92) may be expressed as:

$$\dot{V} = k_P(x_{ij} - x_{ij,des})^\top \dot{x}_{ij} + \dot{x}_{ij}^\top \ddot{x}_{ij} + \frac{1}{k_\lambda} \dot{\phi}_{ij} \ddot{\phi}_{ij}. \quad (93)$$

With agent dynamics in (79), total input design in (80), Jacobian defined in (83), with baseline input in (91) and constraint forces-based increment in (87), resulting in the two-agent coupled constraint acceleration in (88), the time derivative $\dot{V}$ in (93) may be simplified as:

$$\dot{V} = -k_D \|\dot{x}_{ij}\|^2 - \frac{k_d}{k_\lambda} \dot{\phi}_{ij}^2 \leq 0. \quad (94)$$

Note that $\dot{V}$ in (94) is non-positive, which provides the core foundation for stable design. It can additionally be noted that the analysis outline with V in (92) and $\dot{V}$ in (94) is expressed fully with equality signs, thus resulting in an exact/non-conservative analysis outline. The reader is referred to [14], [15] for a further description of signal convergence, which builds upon the outline introduced in here.

IX. DISCUSSION AND CONCLUSIONS

This paper presented a tutorial on techniques that leverage CBFs for enforcing flight-critical operational limits. The methods covered in this tutorial represent early efforts in the development of CBF-based techniques for enforcing operational constraints on flight-critical aerospace systems, with many promising directions for future development.

At the guidance level, one of the outstanding problems in the CBF literature is simultaneous state and input limiting. The approaches discussed in Sec. II and III provide systematic approaches to constructing CBFs and their associated controllers for state limits, but do not address input limits. The backup CBF framework in Sec. IV addresses simultaneous state and input limits, but requires significant online computation. Hence, despite the broad range of methods developed in the CBF literature, computationally efficient state and input limiting remains an open problem.

While CBF-based controllers at the outer loop (e.g. guidance) are well-studied—and have been deployed on real aircraft systems—comparatively less attention has been given to inner-loop CBF flight controllers, where ensuring stability and robustness properties such as classical gain, phase, and time-delay margins is paramount. The servo-control architecture outlined in Sec. VI makes substantive progress in this direction, but further development and maturation is required. One fundamental open question is mutual feasibility of multiple operational constraints imposed on inner-loop controllers, i.e., when there exist control inputs satisfying all constraints simultaneously. The methods in Sec. VI illustrate that when the limited output has a vector relative degree, then the corresponding box constraints are mutually feasible. Yet, determining feasibility for systems without a valid relative degree, such as when there are fewer inputs than outputs, remains an open challenge. These challenges are exacerbated in the presence of input constraints, wherein the mutual feasibility of multiple state limits must be jointly verified alongside input limits, requiring one to construct a control invariant set as discussed in Sec. IV. Hence, the efficient and systematic construction of control invariant sets for aircraft dynamics is also highly relevant.

Ultimately, we believe these open problems present a rich opportunity at the intersection of aerospace control and CBFs, requiring contributions from both applied researchers in aerospace and academic researchers in the control community to develop mathematically rigorous but practically applicable solutions.

REFERENCES

- [1] F. Blanchini and S. Miani, *Set-theoretic methods in control*. Springer, 2008.
- [2] A. D. Ames, J. W. Grizzle, and P. Tabuada, “Control barrier function based quadratic programs with application to adaptive cruise control,” in *Proc. Conf. Decis. Control*, pp. 6271–6278, 2014.
- [3] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, “Control barrier function based quadratic programs for safety critical systems,” *IEEE Trans. Autom. Control*, vol. 62, no. 8, pp. 3861–3876, 2017.
- [4] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, “Control barrier functions: theory and applications,” in *Proc. Eur. Control Conf.*, pp. 3420–3431, 2019.
- [5] E. G. Gilbert, I. Kolmanovsky, and K. T. Tan, “Discrete-time reference governors and the nonlinear control of systems with state and control constraints,” *International Journal of Robust and Nonlinear Control*, vol. 5, no. 5, pp. 487–504, 1995.
- [6] E. Gilbert and I. Kolmanovsky, “Set-point control of nonlinear systems with state and control constraints: a lyapunov-function, reference-governor approach,” in *Proceedings of the 38th IEEE Conference on Decision and Control*, vol. 3, pp. 2507–2512 vol.3, 1999.
- [7] A. Bemporad, “Reference management for constrained nonlinear systems,” *IEEE Trans. Autom. Control*, vol. 43, no. 3, pp. 415–419, 1998.
- [8] E. Gilbert and I. Kolmanovsky, “Nonlinear tracking control in the presence of state and control constraints: a generalized reference governor,” *Automatica*, vol. 38, no. 12, pp. 2063–2073, 2002.
- [9] I. Kolmanovsky, E. Garone, and S. Di Cairano, “Reference and command governors: A tutorial on their theory and automotive applications,” in *2014 American Control Conference*, pp. 226–241, 2014.
- [10] E. Garone, S. Di Cairano, and I. Kolmanovsky, “Reference and command governors for systems with constraints: A survey on theory and applications,” *Automatica*, vol. 75, pp. 306–328, 2017.
- [11] Y. Liu, J. Osorio, and H. Ossareh, “Decoupled reference governors for multi-input multi-output systems,” in *2018 IEEE Conference on Decision and Control (CDC)*, pp. 1839–1846, 2018.
- [12] C. Lanczos, *The Variational Principles of Mechanics*. 1949.
- [13] I.-A. F. Ihle, J. Jouffroy, and T. I. Fossen, “Formation control of marine surface craft: A lagrangian approach,” *IEEE J. Ocean. Eng.*, 2006.
- [14] Y. Zou, P. R. Pagilla, and R. T. Ratliff, “Distributed formation flight control using constraint forces,” *J. Guid. Ctrl. Dyn.*, vol. 32, 2009.
- [15] Y. Zou, *Distributed Control of Multiple Vehicle Systems Using Constraint Forces*. PhD thesis, Oklahoma State University, 2008.
- [16] Y. Zou, P. R. Pagilla, and E. Misawa, “Formation of a group of vehicles with full information using constraint forces,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 129, pp. 654–661, 2007.
- [17] S. Prajna and A. Jadbabaie, “Safety verification of hybrid systems using barrier certificates,” in *Proceedings of the International Workshop on Hybrid Systems: Computation and Control*, pp. 477–492, 2004.
- [18] M. Krstić and M. Bement, “Nonovershooting control of strict-feedback nonlinear systems,” *IEEE Trans. Autom. Control*, vol. 51, 2006.
- [19] M. Nagumo, “Über die lage der integralkurven gewöhnlicher differentialgleichungen,” *Proceedings of the Physico-Mathematical Society of Japan. 3rd Series*, vol. 24, pp. 551–559, 1942.
- [20] M. Menner and E. Lavretsky, “Translation of nagumo’s foundational work on barrier functions: On the location of integral curves of ordinary differential equations,” *arXiv preprint arXiv:2406.18614*, 2024.
- [21] X. Xu, P. Tabuada, J. W. Grizzle, and A. D. Ames, “Robustness of control barrier functions for safety critical control,” in *Proc. IFAC Conf. on Analysis and Design of Hybrid Syst.*, pp. 54–61, 2015.
- [22] L. Wang, A. D. Ames, and M. Egerstedt, “Safe certificate-based maneuvers for teams of quadrotors using differential flatness,” in *Proc. Int. Conf. Robot. and Autom.*, pp. 3293–3298, 2017.
- [23] L. Wang, E. A. Theodorou, and M. Egerstedt, “Safe learning of quadrotor dynamics using barrier certificates,” in *Proc. Int. Conf. Robot. and Autom.*, pp. 2460–2465, 2018.

- [24] G. Wu and K. Sreenath, "Safety-critical control of a planar quadrotor," in *Proc. Amer. Control Conf.*, pp. 2252–2258, 2016.
- [25] A. Singletary, T. Gurriet, P. Nilsson, and A. D. Ames, "Safety-critical rapid aerial exploration of unknown environments," in *Proc. Int. Conf. Robot. and Autom.*, pp. 10270–10276, 2020.
- [26] A. Singletary, A. Swann, Y. Chen, and A. D. Ames, "Onboard safety guarantees for racing drones: High-speed geofencing with control barrier functions," *IEEE Robot. Aut. Lett.*, vol. 7, no. 2, 2022.
- [27] T. G. Molnar, S. K. Kannan, K. Dunlap, J. Cunningham, K. L. Hobbs, and A. D. Ames, "Collision avoidance and geofencing by fixed-wing aircraft using control barrier functions," *IEEE Trans. Contr. Syst. Tech.*, vol. 33, no. 5, pp. 1493–1508, 2025.
- [28] E. Squires, P. Pierpaoli, and M. Egerstedt, "Constructive barrier certificates with applications to fixed-wing aircraft collision avoidance," in *IEEE Conf. Ctrl. Tech. App.*, pp. 1656–1661, 2018.
- [29] A. W. Singletary, M. H. Cohen, T. G. Molnar, and A. D. Ames, "Safety guardrails in the sky: Realizing control barrier functions on the VISTA F-16 jet," *arXiv preprint arXiv:2603.27912*, 2026.
- [30] K. P. Wabersich, A. J. Taylor, J. J. Choi, K. Sreenath, C. J. Tomlin, A. D. Ames, and M. N. Zeilinger, "Data-driven safety filters: Hamilton-Jacobi reachability, control barrier functions, and predictive methods for uncertain systems," *IEEE Control Systems Magazine*, vol. 43, no. 5, pp. 137–177, 2023.
- [31] K. C. Hsu, H. Hu, and J. F. Fisac, "The safety filter: A unified view of safety-critical control in autonomous systems," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 7, 2023.
- [32] M. H. Cohen, T. G. Molnar, and A. D. Ames, "Safety-critical control for autonomous systems: Control barrier functions via reduced order models," *Annual Reviews in Control*, vol. 57, p. 100947, 2024.
- [33] R. Konda, A. D. Ames, and S. Coogan, "Characterizing safety: Minimal control barrier functions from scalar comparison systems," *IEEE Contr. Syst. Lett.*, vol. 5, no. 2, pp. 523–528, 2021.
- [34] R. Abraham, J. E. Marsden, and T. Ratiu, *Manifolds, tensor analysis, and applications*. Addison-Wesley, 1983.
- [35] R. A. Freeman and P. V. Kokotovic, "Inverse optimality in robust stabilization," *SIAM J. Ctrl. Opt.*, vol. 34, no. 4, pp. 1365–1391, 1996.
- [36] M. Jankovic, "Robust control barrier functions for constrained stabilization of nonlinear systems," *Automatica*, vol. 96, 2018.
- [37] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.
- [38] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 2016.
- [39] Q. Nguyen and K. Sreenath, "Exponential control barrier functions for enforcing high relative-degree safety-critical constraints," in *Proc. Amer. Control Conf.*, pp. 322–328, 2016.
- [40] W. Xiao and C. Belta, "High order control barrier functions," *IEEE Trans. Autom. Control*, vol. 67, no. 7, pp. 3655–3662, 2022.
- [41] X. Tan, W. S. Cortez, and D. V. Dimarogonas, "High-order barrier functions: robustness, safety and performance-critical control," *IEEE Trans. Autom. Control*, vol. 67, no. 6, pp. 3021–3028, 2022.
- [42] M. Krstić, I. Kanellakopoulos, and P. Kokotović, *Nonlinear and Adaptive Control Design*. Wiley, 1995.
- [43] A. J. Taylor, P. Ong, T. G. Molnar, and A. D. Ames, "Safe backstepping with control barrier functions," in *Proc. Conf. Decis. Control*, 2022.
- [44] M. H. Cohen, R. K. Cosner, and A. D. Ames, "Constructive safety-critical control: Synthesizing control barrier functions for partially feedback linearizable systems," *IEEE Contr. Syst. Lett.*, 2024.
- [45] L. Gacsi, M. H. Cohen, and T. G. Molnar, "Activated backstepping with control barrier functions for the safe navigation of automated vehicles," *IEEE Contr. Syst. Lett.*, vol. 9, pp. 721–726, 2025.
- [46] G. Bahati, R. K. Cosner, M. H. Cohen, R. M. Bena, and A. D. Ames, "Control barrier function synthesis for nonlinear systems with dual relative degree," in *Proc. Conf. Decis. Control*, 2025.
- [47] M. H. Cohen, P. Ong, and A. D. Ames, "Input-to-state safe backstepping: Robust safety-critical control with unmatched uncertainties," in *Proc. Amer. Control Conf.*, 2026.
- [48] M. Krstic, "Inverse optimal safety filters," *IEEE Trans. Autom. Control*, vol. 69, no. 1, pp. 16–31, 2024.
- [49] K. H. Kim, M. Diagne, and M. Krstić, "Robust control barrier function design for high relative degree systems: Application to unknown moving obstacle collision avoidance," in *Proc. Amer. Control Conf.*, 2025.
- [50] I. Abel, D. Steeves, M. Krstić, and M. Janković, "Prescribed-time safety design for strict-feedback nonlinear systems," *IEEE Trans. Autom. Control*, vol. 69, no. 3, pp. 1464–1479, 2024.
- [51] M. H. Cohen, P. Ong, G. Bahati, and A. D. Ames, "Characterizing smooth safety filters via the implicit function theorem," *IEEE Contr. Syst. Lett.*, vol. 7, pp. 3890–3895, 2023.
- [52] T. G. Molnar and A. D. Ames, "Composing control barrier functions for complex safety specifications," *IEEE Contr. Syst. Lett.*, 2023.
- [53] T. Gurriet, M. Mote, A. Singletary, P. Nilsson, E. Feron, and A. D. Ames, "A scalable safety critical control framework for nonlinear systems," *IEEE Access*, vol. 8, pp. 187249–187275, 2020.
- [54] D. E. J. van Wijk, E. Daş, T. G. Molnar, A. D. Ames, and J. W. Burdick, "Safety-critical control with bounded inputs: A closed-form solution for backup control barrier functions," *arXiv preprint arXiv:2510.05436*, 2025.
- [55] D. E. J. van Wijk, S. Coogan, T. G. Molnar, M. Majji, and K. L. Hobbs, "Disturbance-robust backup control barrier functions: Safety under uncertain dynamics," *IEEE Contr. Syst. Lett.*, vol. 8, 2024.
- [56] T. G. Molnar and A. D. Ames, "Safety-critical control with bounded inputs via reduced order models," in *Proc. Amer. Control Conf.*, pp. 1414–1421, 2023.
- [57] S. Stephens, *A Real-time Algorithm to Achieve Precise Coordinated Arrival Times in a Time-variant Environment*. PhD thesis, Air Force Institute of Technology, 2021.
- [58] T. G. Molnar, R. K. Cosner, A. W. Singletary, W. Ubellacker, and A. D. Ames, "Model-free safety-critical control for robotic systems," *IEEE Robot. Aut. Lett.*, vol. 7, no. 2, pp. 944–951, 2022.
- [59] M. H. Cohen, N. Csomay-Shanklin, W. D. Compton, T. G. Molnar, and A. D. Ames, "Safety-critical controller synthesis with reduced-order models," in *Proc. Amer. Control Conf.*, pp. 5216–5221, 2025.
- [60] W. D. Compton, M. H. Cohen, and A. D. Ames, "Learning for layered safety-critical control with predictive control barrier functions," in *Proc. Learning for Dynamics & Control Conf.*, 2025.
- [61] M. E. Knapp, T. Berger, M. Tischler, and M. C. Cotting, "Development of a full envelope flight identified F-16 simulation model," in *AIAA SciTech Forum*, 2018.
- [62] E. Lavretsky and M. Menner, "Servo-controllers for linear time-invariant systems with operational constraints," in *Proc. Amer. Control Conf.*, pp. 4909–4916, 2025.
- [63] M. Menner and E. Lavretsky, "Robust linear design for flight control systems with operational constraints," in *AIAA SCITECH Forum*, 2026.
- [64] E. Lavretsky, "Servo-controllers with operational constraints," *arXiv preprint arXiv:2504.12208*, 2025.
- [65] E. Lavretsky, "State feedback controllers with operational constraints," *arXiv preprint arXiv:2511.19683*, 2025.
- [66] H. K. Khalil, *Nonlinear Systems*. Prentice Hall, 3 ed., 2002.
- [67] E. Lavretsky and K. Wise, *Robust and Adaptive Control: With Aerospace Applications*. Springer, 2024.
- [68] J. Autenrieb and A. Annaswamy, "Safe and stable adaptive control for a class of dynamic systems," in *Proc. Conf. Decis. Control*, 2023.
- [69] J. A. Solano-Castellanos, P. A. Fisher, and A. M. Annaswamy, "Safe and stable formation control with autonomous multi-agents using adaptive control," *arXiv preprint arXiv:2403.15674*, 2025.
- [70] P. A. Fisher, J. Autenrieb, and A. M. Annaswamy, "An error-based safety buffer for safe adaptive control," *arXiv preprint arXiv:2510.23491*, 2025.
- [71] K. S. Narendra and A. M. Annaswamy, *Stable Adaptive Systems*. Dover, 2005.
- [72] J. Baumgarte, "Stabilization of constraints and integrals of motion in dynamical systems," *Comp. Meth. Appl. Mech. Eng.*, 1972.
- [73] L. R. Petzold, Y. Ren, and T. Maly, "Regularization of higher-index differential-algebraic equations with rank-deficient constraints," *SIAM Journal on Scientific Computing*, 1997.
- [74] L. Wang, A. D. Ames, and M. Egerstedt, "Safety barrier certificates for collisions-free multirobot systems," *IEEE Trans. Robot.*, 2017.
- [75] P. Glotfelter, J. Cortes, and M. Egerstedt, "Nonsmooth barrier functions with applications to multi-robot systems," *IEEE Contr. Syst. Lett.*, 2017.