# Safety-Critical Controller Synthesis with Reduced-Order Models

Max H. Cohen[1], Noel Csomay-Shanklin[1], William D. Compton[1], Tamas G. Molnar[2], and Aaron D. Ames[1]

***Abstract*— Reduced-order models (ROMs) provide lower dimensional representations of complex systems, capturing their salient features while simplifying control design. Building on previous work, this paper presents an overarching framework for the integration of ROMs and control barrier functions, enabling the use of simplified models to construct safety-critical controllers while providing safety guarantees for complex full-order models. To achieve this, we formalize the connection between full and ROMs by defining projection mappings that relate the states and inputs of these models and leverage simulation functions to establish conditions under which safety guarantees may be transferred from a ROM to its corresponding full-order model. The efficacy of our framework is illustrated through simulation results on a drone and hardware demonstrations on ARCHER, a 3D hopping robot.**

## I. Introduction

Control barrier functions (CBFs) [1] have proven successful in designing safety-critical controllers for nonlinear systems. Despite this, developing general procedures for constructing CBFs for high-dimensional complex systems has remained elusive [2]. More recently, the authors have attempted to leverage reduced-order models (ROMs) to construct CBFs for simple models that may be refined to ensure the safety of more complex, full-order systems [2]–[4]. This paradigm may be traced back to [3] wherein simple kinematic models were used to generate safe velocity commands to be tracked by more complicated robotic systems. Such ideas were expanded on in [4] to address ROMs with bounded inputs and in [2] where this ROM paradigm is related to nonlinear control techniques such as backstepping [5]. In this paper, we unify and generalize previous developments [2]–[4] to provide a formal framework for leveraging ROMs in the context of safety-critical control with CBFs.

The paradigm of safety-critical control with CBFs and ROMs is closely related to planner-tracker frameworks in which reduced-order planning models are used to generate trajectories that are tracked by full-order tracking models. The majority of planner-tracker methods focus on the construction of a tracking controller and associated tracking error bound using methods such as Hamilton-Jacobi reachability [6], sum of squares programming [7], model predictive control [8], or contraction theory [9]. Within the context of planner-tracker frameworks, we address a problem converse to those above: rather than designing a tracking controller for a given planning model, we focus on designing safe reduced-order reference commands for a full-order system with a fixed tracking controller. This is motivated by the observation that many systems of interest are equipped with high-performance, but "black-box," tracking controllers that are not easily modified. The canonical example is robotic systems, where one can often send "joystick" commands, without knowledge of the underlying tracking controller.

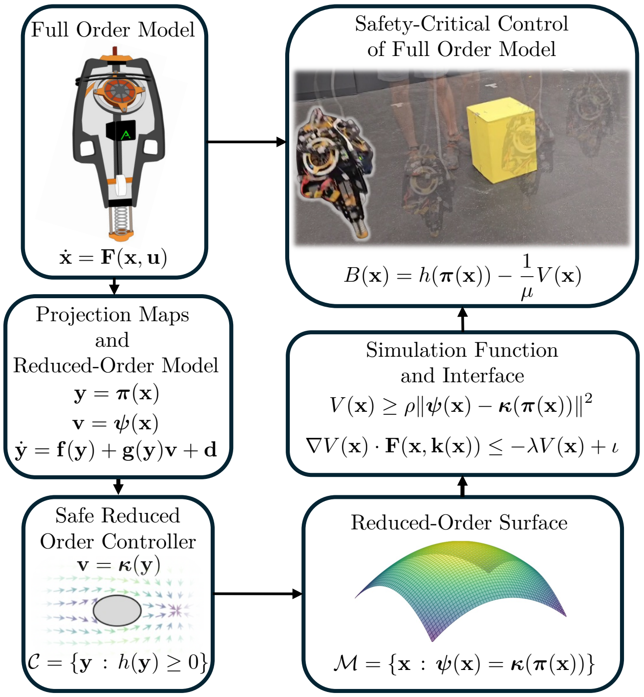
Full Order Model
$\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, \mathbf{u})$
Safety-Critical Control of Full Order Model
$B(\mathbf{x}) = h(\boldsymbol{\pi}(\mathbf{x})) - \frac{1}{\mu}V(\mathbf{x})$
Projection Maps and Reduced-Order Model
$\mathbf{y} = \boldsymbol{\pi}(\mathbf{x})$
$\mathbf{v} = \boldsymbol{\psi}(\mathbf{x})$
$\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y}) + \mathbf{g}(\mathbf{y})\mathbf{v} + \mathbf{d}$
Simulation Function and Interface
$V(\mathbf{x}) \geq \rho\|\boldsymbol{\psi}(\mathbf{x}) - \boldsymbol{\kappa}(\boldsymbol{\pi}(\mathbf{x}))\|^2$
$\nabla V(\mathbf{x}) \cdot \mathbf{F}(\mathbf{x}, \mathbf{k}(\mathbf{x})) \leq -\lambda V(\mathbf{x}) + \iota$
Safe Reduced Order Controller
$\mathbf{v} = \boldsymbol{\kappa}(\mathbf{y})$
$\mathcal{C} = \{\mathbf{y} : h(\mathbf{y}) \geq 0\}$
Reduced-Order Surface
$\mathcal{M} = \{\mathbf{x} : \boldsymbol{\psi}(\mathbf{x}) = \boldsymbol{\kappa}(\boldsymbol{\pi}(\mathbf{x}))\}$

Fig. 1. **Overview:** We project high-dimensional control systems onto reduced-order spaces, design safety-critical controllers for a reduced-order representation of the original system, and then relate the inputs of this reduced-order system back to the full-order system. Our theoretical developments are illustrated through their application to ARCHER, a 3D hopping robot, a video of which is available at `https://vimeo.com/1010060590?share=copy`.

The perspective taken herein also has roots in classical works from the hybrid systems community on abstraction-based control [10], [11]. In particular, our ROM paradigm is inspired by the notion of $\phi$-related control systems [12], [13] and we leverage approximate simulation relations [14] to formalize the connection between full and reduced-order models. While these works offer a powerful framework to formally reason about system abstractions, their application has often been limited to simple, low-dimensional systems. Here, we expand the applicability of such ideas to more complex systems by illustrating how concepts such as simulation relations integrate with CBFs, leading to practical constructions of safety-critical controllers.

[1] The authors are with the Department of Mechanical and Civil Engineering, California Institute of Technology, Pasadena, CA {maxcohen,noelcs,wcompton,ames}@caltech.edu.

[2] The author is with the Department of Mechanical Engineering, Wichita State University, Wichita, KS {tamas.molnar}@wichita.edu.

This research was supported by Boeing and NSF CPS Award #1932091.

In this paper, we present a unified framework for ROMs in the context of safety-critical control with CBFs, generalizing the ideas in [2]–[4]. Specifically, we introduce projection mappings that relate the states and inputs of full-order and reduced-order models, which allow for transferring properties of a ROM back to its corresponding full-order model. A key assumption enabling our approach is the existence of a simulation function and corresponding interface [14] that refines reduced-order inputs to those for the full-order model. At a high level, this assumption formalizes the capability of a full-order model to track commands generated by a suitable ROM. While this may seem restrictive, we argue that for many practical systems, it is not: drones and other unmanned aerial vehicles often come equipped with well-designed tracking controllers [15], [16], whereas state-of-the-art methods in robotic locomotion rely on converting velocity references into joint torques using model predictive control [17] or reinforcement learning [18]. Rather than redesigning the control architecture of such systems to incorporate safety considerations, the perspective taken herein is to leverage these existing architectures by passing them suitably designed reduced-order inputs, which, as we will demonstrate experimentally, leads to practical safety guarantees.

Our approach generalizes previous results on CBFs and ROMs [2]–[4] as follows. We consider a larger class of FOMs and ROMs than those in [2], [3], which focused on fully actuated robotic systems and strict-feedback systems, respectively. We also provide a Lyapunov-like characterization of tracking controllers via simulation functions [14], which, compared to [4], yields time-invariant safe sets rather than time-varying safe sets and relaxes other conditions such as only using CBFs with bounded gradients. Furthermore, we showcase the efficacy of our framework on ARCHER [19]–[21], a highly underactuated 3D hopping robot, advancing the practical applications of ROMs within a CBF framework.

## II. Preliminaries and Problem Formulation

**Notation:** A continuous function $\alpha : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is said to be a class $\mathcal{K}$ function ($\alpha \in \mathcal{K}$) if $\alpha(0) = 0$ and $\alpha$ is strictly increasing. For a set $\mathcal{S}$ we use $\partial\mathcal{S}$ to denote its boundary. A real number $a \in \mathbb{R}$ is said to be a regular value of a scalar function $h : \mathbb{R}^n \to \mathbb{R}$ if $h(\mathbf{x}) = a$ implies $\nabla h(\mathbf{x}) \neq \mathbf{0}$.

**Safety:** Consider a system with state $\mathbf{x} \in \mathbb{R}^n$ and dynamics:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \tag{1}$$

where $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ is a locally Lipschitz vector field. Under this Lipschitz assumption, for each initial condition $\mathbf{x}_0 \in \mathbb{R}^n$ the dynamics (1) generate a unique continuously differentiable trajectory $\mathbf{x} : I(\mathbf{x}_0) \to \mathbb{R}^n$ satisfying (1) on some maximal interval of existence $I(\mathbf{x}_0) \subseteq \mathbb{R}_{\geq 0}$. A set $\mathcal{S} \subset \mathbb{R}^n$ is said to be forward invariant for (1) if for each initial condition $\mathbf{x}_0 \in \mathcal{S}$, the resulting trajectory $t \mapsto \mathbf{x}(t)$ satisfies $\mathbf{x}(t) \in \mathcal{S}$ for all $t \in I(\mathbf{x}_0)$. The following result, known as Nagumo's Theorem, provides necessary and sufficient conditions for set invariance.

**Theorem 1.** *A closed set $\mathcal{S} \subset \mathbb{R}^n$ is forward invariant for* (1) *if and only if $\mathbf{f}(\mathbf{x}) \in \mathcal{T}_{\mathcal{S}}(\mathbf{x})$ for all $\mathbf{x} \in \partial\mathcal{S}$, where $\mathcal{T}_{\mathcal{S}}(\mathbf{x})$ denotes the contingent cone*[1] *to $\mathcal{S}$ at $\mathbf{x}$.*

Informally, Nagumo's Theorem states that a set is forward invariant if and only if, for each $\mathbf{x} \in \partial\mathcal{S}$, the vector field characterizing the dynamics points into $\mathcal{S}$. Further details of Nagumo's Theorem can be found in [22, Ch. 4] and [23, Ch. 4]. In this paper, we associate the concept of set invariance with that of safety: a system is safe if it remains in a desirable subset of the state space. In what follows, our main objective is to design safety-critical controllers for complex high-dimensional systems using relatively simple, or, reduced-order, models within the framework of CBFs [2].

## III. Reduced-Order Models

Reduced-order models (ROMs) provide lower-dimensional representations of systems, capturing the salient features of more complex models while simplifying controller design. To formalize this idea, consider a control system:

$$\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, \mathbf{u}), \tag{2}$$

with state $\mathbf{x} \in \mathbb{R}^N$, input $\mathbf{u} \in \mathbb{R}^M$ and locally Lipschitz dynamics $\mathbf{F} : \mathbb{R}^N \times \mathbb{R}^M \to \mathbb{R}^N$, which we refer to as the full-order model (FOM). To obtain a reduced-order representation of (2), we define a differentiable *state projection map* $\boldsymbol{\pi} : \mathbb{R}^N \to \mathbb{R}^n$ and a *control projection map* $\boldsymbol{\psi} : \mathbb{R}^N \to \mathbb{R}^m$, which map the full-order state $\mathbf{x} \in \mathbb{R}^N$ to a reduced-order state $\mathbf{y} \in \mathbb{R}^n$ and input $\mathbf{v} \in \mathbb{R}^m$ as:

$$\mathbf{y} \coloneqq \boldsymbol{\pi}(\mathbf{x}), \quad \mathbf{v} \coloneqq \boldsymbol{\psi}(\mathbf{x}). \tag{3}$$

This state projection map $\boldsymbol{\pi}$ allows for defining the dynamics of the FOM (2) projected onto the reduced-order space:

$$\dot{\mathbf{y}} = \frac{\partial \boldsymbol{\pi}}{\partial \mathbf{x}}(\mathbf{x})\mathbf{F}(\mathbf{x}, \mathbf{u}). \tag{4}$$

While (4) provides a reduced-order representation of (2), it depends on the full-order states and inputs, complicating the design of a reduced-order controller. We resolve this by defining *idealized* reduced-order dynamics, characterized by locally Lipschitz functions $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ and $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^{n \times m}$ that are used to rewrite (4) as:

$$\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y}) + \mathbf{g}(\mathbf{y})\mathbf{v} + \mathbf{d}, \tag{5}$$

where $\mathbf{d} \coloneqq \frac{\partial \boldsymbol{\pi}}{\partial \mathbf{x}}(\mathbf{x})\mathbf{F}(\mathbf{x}, \mathbf{u}) - \mathbf{f}(\mathbf{y}) - \mathbf{g}(\mathbf{y})\mathbf{v}$ captures the discrepancy between (4) and the idealized reduced-order dynamics. Ideally, one would choose $\boldsymbol{\pi}$ and $\boldsymbol{\psi}$ such that:

$$\frac{\partial \boldsymbol{\pi}}{\partial \mathbf{x}}(\mathbf{x})\mathbf{F}(\mathbf{x}, \mathbf{u}) = \mathbf{f}(\boldsymbol{\pi}(\mathbf{x})) + \mathbf{g}(\boldsymbol{\pi}(\mathbf{x}))\boldsymbol{\psi}(\mathbf{x}),$$

for all $(\mathbf{x}, \mathbf{u}) \in \mathbb{R}^N \times \mathbb{R}^M$ so that $\mathbf{d} \equiv \mathbf{0}$, although we will not impose this as a strict requirement. Hereafter, we refer to (5) as a ROM of (2). To make the preceding developments more concrete, we introduce the following running example.

**Example 1.** Consider a full-order model of a quadrotor from [24] with state $\mathbf{x} = (\mathbf{p}, \mathbf{q}, \dot{\mathbf{p}}) \in \mathbb{R}^3 \times \mathbb{S}^3 \times \mathbb{R}^3 = \mathcal{X}$, where

[1] See [22, Ch. 4] for a precise definition of the contigent cone.

$\mathbf{p} = (x, y, z) \in \mathbb{R}^3$ is the position, $\mathbf{q} \in \mathbb{S}^3$ is the orientation represented as a unit quaternion, and $\dot{\mathbf{p}} = (\dot{x}, \dot{y}, \dot{z}) \in \mathbb{R}^3$ is the velocity. The dynamics of the quadrotor are given by:

$$\underbrace{\begin{bmatrix} \dot{\mathbf{p}} \\ \dot{\mathbf{q}} \\ \ddot{\mathbf{p}} \end{bmatrix}}_{\dot{\mathbf{x}}} = \underbrace{\begin{bmatrix} \dot{\mathbf{p}} \\ \boldsymbol{\omega} \\ -\mathbf{e}_z g + \frac{1}{m}\mathbf{R}(\mathbf{q})\mathbf{e}_z \tau \end{bmatrix}}_{\mathbf{F}(\mathbf{x},\mathbf{u})}, \tag{6}$$

where the full-order input $\mathbf{u} = (\boldsymbol{\omega}, \tau) \in \mathfrak{s}^3 \times \mathbb{R}$ is the angular velocity $\boldsymbol{\omega}$ and thrust $\tau$. Our objective is to control the quadrotor as if it were a two-dimensional single integrator evolving in the plane. To this end, we define $\mathbf{y} = \boldsymbol{\pi}(\mathbf{x}) := (x, y) \in \mathbb{R}^2$, $\mathbf{v} = \boldsymbol{\psi}(\mathbf{x}) = (\dot{x}, \dot{y}) \in \mathbb{R}^2$ noting the dynamics of the FOM projected onto the reduced-order space are:

$$\underbrace{\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}}_{\dot{\mathbf{y}}} = \underbrace{\begin{bmatrix} 0 \\ 0 \end{bmatrix}}_{\mathbf{f}(\mathbf{y})} + \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{\mathbf{g}(\mathbf{y})} \underbrace{\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}}_{\mathbf{v}},$$

which matches the idealized single integrator dynamics (i.e., these projections produce a ROM (5) with $\mathbf{d} \equiv \mathbf{0}$).

Our main objective in this paper is to design safety-critical controllers $\boldsymbol{\kappa} : \mathbb{R}^n \to \mathbb{R}^m$ for the ROM (5), which are then refined to ensure safety of the FOM (2). To relate the properties of a reduced-order controller $\boldsymbol{\kappa}$ to (2), we define the *reduced-order surface*:

$$\mathcal{M} := \{\mathbf{x} \in \mathbb{R}^N : \boldsymbol{\psi}(\mathbf{x}) - \boldsymbol{\kappa}(\boldsymbol{\pi}(\mathbf{x})) = \mathbf{0}\}. \tag{7}$$

Constraining (2) to $\mathcal{M}$ along a given trajectory $t \mapsto \mathbf{x}(t)$ ensures that $\boldsymbol{\psi}(\mathbf{x}(t)) = \boldsymbol{\kappa}(\boldsymbol{\pi}(\mathbf{x}(t)))$ for all $t \geq 0$ so that properties of the reduced-order controller $\boldsymbol{\kappa}$ may be transferred back to states of the FOM. While we will not impose the rather strict assumption that $\mathcal{M}$ be rendered forward invariant, we will assume that it is possible to drive the FOM to a neighborhood of $\mathcal{M}$, which is captured by the notion of a simulation function [14], slightly modified to suite the context of this paper.

**Definition 1.** A continuously differentiable function $V : \mathcal{D} \subseteq \mathbb{R}^N \to \mathbb{R}_{\geq 0}$ is said to be a *simulation function* from the ROM (5) to the FOM (2) with an associated locally Lipschitz interface $\mathbf{k} : \mathbb{R}^N \to \mathbb{R}^M$ if there exists $\rho > 0$ such that:

$$V(\mathbf{x}) \geq \rho \|\boldsymbol{\psi}(\mathbf{x}) - \boldsymbol{\kappa}(\boldsymbol{\pi}(\mathbf{x}))\|^2, \tag{8}$$

$$\nabla V(\mathbf{x}) \cdot \mathbf{F}(\mathbf{x}, \mathbf{k}(\mathbf{x})) \leq -\lambda V(\mathbf{x}) + \iota, \tag{9}$$

for all $\mathbf{x} \in \mathcal{D}$, where $\lambda > 0$ and $\iota \geq 0$ satisfy $\beta > \iota/\lambda$ with:

$$\Omega_\beta := \{\mathbf{x} \in \mathcal{D} : V(\mathbf{x}) \leq \beta\},$$

the largest sublevel set of $V$ contained within $\mathcal{D}$.

**Lemma 1.** *Let $V : \mathcal{D} \to \mathbb{R}_{\geq 0}$ be a simulation function from* (5) *to* (2) *with an associated interface $\mathbf{k} : \mathbb{R}^N \to \mathbb{R}^M$. Then, for any initial condition $\mathbf{x}_0 \in \Omega_\beta$, trajectories $t \mapsto \mathbf{x}(t)$ of closed-loop full-order system $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, \mathbf{k}(\mathbf{x}))$ satisfy:*

$$\|\boldsymbol{\psi}(\mathbf{x}(t)) - \boldsymbol{\kappa}(\boldsymbol{\pi}(\mathbf{x}(t)))\|^2 \leq \frac{V(\mathbf{x}_0)}{\rho} e^{-\lambda t} + \frac{\iota}{\lambda \rho}, \ \forall t \in I(\mathbf{x}_0). \tag{10}$$

The proof of Lemma 1 is a direct consequence of the Comparison Lemma [25, Lemma 3.4]. The following result will be useful when discussing set invariance properties.

**Lemma 2.** *If $V : \mathbb{R}^N \to \mathbb{R}_{\geq 0}$ is a simulation function from* (5) *to* (2) *with interface* $\mathbf{k}$*, then $\beta$ is a regular value of $V$.*

*Proof.* For the sake of contradiction assume $\beta$ is not a regular value of $V$. Then, when $V(\mathbf{x}) = \beta$ we have $\nabla V(\mathbf{x}) = \mathbf{0}$ and $\nabla V(\mathbf{x}) \cdot \mathbf{F}(\mathbf{x}, \mathbf{k}(\mathbf{x})) = 0$, which, by (9), implies that:

$$0 \leq -\lambda V(\mathbf{x}) + \iota = -\lambda \beta + \iota < -\lambda \frac{\iota}{\lambda} + \iota = 0,$$

where the final inequality follows from the fact that $\beta > \iota/\lambda$. As we cannot have $0 < 0$, the above contradicts the initial claim that $\beta$ is not a regular value of $V$. Hence, by contradiction, $\beta$ must be a regular value of $V$. □

The existence of a simulation function and associated interface allows for refining inputs for a ROM to those for the FOM. When implemented on the FOM, this interface ensures that states within the control projection map $\boldsymbol{\psi}(\mathbf{x})$ remain close to the inputs generated by the ROM controller $\boldsymbol{\kappa}(\boldsymbol{\pi}(\mathbf{x}))$, allowing to relate properties of the reduced-order controller $\boldsymbol{\kappa}$ to the full-order state $\mathbf{x}$.

## IV. Safety-Critical Control with ROMs

We now discuss how the preceding framework may be used to design safety-critical controllers for complex systems based on their corresponding ROMs. To this end, consider a state constraint set for a FOM:

$$\mathcal{C} := \{\mathbf{x} \in \mathbb{R}^N : h(\boldsymbol{\pi}(\mathbf{x})) \geq 0\}, \tag{11}$$

where $h : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable. While $\mathcal{C}$ exists in the full-order space, it only depends on the states related to the ROM. The derivative of $h$ along the full-order dynamics is given by:

$$\begin{aligned} \dot{h} =& \nabla h(\boldsymbol{\pi}(\mathbf{x})) \cdot \frac{\partial \boldsymbol{\pi}}{\partial \mathbf{x}}(\mathbf{x}) \mathbf{F}(\mathbf{x}, \mathbf{u}) \\ =& L_{\mathbf{f}} h(\mathbf{y}) + L_{\mathbf{g}} h(\mathbf{y}) \mathbf{v} + \nabla h(\mathbf{y}) \cdot \mathbf{d}. \end{aligned} \tag{12}$$

Similar to backstepping [5], we view $\mathbf{v} = \boldsymbol{\psi}(\mathbf{x})$, the input to the ROM, as a "virtual" control input and design a controller $\boldsymbol{\kappa} : \mathbb{R}^n \to \mathbb{R}^m$ for the ROM that would enforce forward invariance of $\mathcal{C}$, provided its dynamics were directly controllable. Since the inputs to the ROM are not the same as those of the FOM, we then relate the inputs of the ROM $\mathbf{v} = \boldsymbol{\kappa}(\mathbf{y})$ to those of the FOM (2) via a simulation function $V$ and interface $\mathbf{k}$. To design the ROM controller, we leverage CBFs: suppose there exists a locally Lipschitz controller $\boldsymbol{\kappa} : \mathbb{R}^n \to \mathbb{R}^m$ satisfying:

$$\begin{aligned} L_{\mathbf{f}} h(\mathbf{y}) + L_{\mathbf{g}} h(\mathbf{y}) \boldsymbol{\kappa}(\mathbf{y}) >& -\alpha h(\mathbf{y}) + \frac{1}{\varepsilon} \|L_{\mathbf{g}} h(\mathbf{y})\|^2 \\ &+ \frac{1}{\sigma} \|\nabla h(\mathbf{y})\|^2 \end{aligned} \tag{13}$$

for all $\mathbf{y} \in \mathbb{R}^n$, where $\alpha, \varepsilon, \sigma > 0$. The above condition effectively requires $h$ to be an *input-to-state safe* CBF [26] for the ROM (5), where the last two terms in (13) are included to compensate for transient tracking errors $\|\boldsymbol{\psi}(\mathbf{x}) - \boldsymbol{\kappa}(\boldsymbol{\pi}(\mathbf{x}))\|$

of the interface $\mathbf{k}$ from Def. 1 and to compensate for $\mathbf{d}$ from (5). We now combine $h$ with a simulation function $V$ to form the barrier function candidate for the FOM:

$$B(\mathbf{x}) = h(\boldsymbol{\pi}(\mathbf{x})) - \frac{1}{\mu}V(\mathbf{x}), \tag{14}$$

where $\mu \in \mathbb{R}_{>0}$, which is used to define a candidate safe set:

$$\mathcal{S} := \{\mathbf{x} \in \mathbb{R}^N \,:\, B(\mathbf{x}) \geq 0\}, \tag{15}$$

for the FOM (2). Since $V(\mathbf{x}) \geq 0$, we have $B(\mathbf{x}) \geq 0 \implies h(\boldsymbol{\pi}(\mathbf{x})) \geq 0$ so that rendering $\mathcal{S}$ forward invariant leads to satisfaction of the state constraint in (11). Before proceeding, we note that with $\mathbf{u} = \mathbf{k}(\mathbf{x})$, $\mathbf{d}$ may be written as:

$$\mathbf{d}(\mathbf{x}) = \frac{\partial \boldsymbol{\pi}}{\partial \mathbf{x}}(\mathbf{x})\mathbf{F}(\mathbf{x}, \mathbf{k}(\mathbf{x})) - \mathbf{f}(\boldsymbol{\pi}(\mathbf{x})) - \mathbf{g}(\boldsymbol{\pi}(\mathbf{x}))\boldsymbol{\psi}(\mathbf{x}). \tag{16}$$

The following theorem constitutes the main result of this paper and illustrates that when there exists a simulation function from the ROM to the FOM, then one may refine reduced-order controllers to ensure safety of the FOM.

**Theorem 2.** *Consider the FOM* (2)*, the ROM* (5)*, the set $\mathcal{C} \subset \mathbb{R}^n$ as in* (11)*, and suppose there exists a simulation function $V : \mathcal{D} \to \mathbb{R}_{\geq 0}$ from* (5) *to* (2) *with an associated locally Lipschitz interface $\mathbf{k} : \mathbb{R}^N \to \mathbb{R}^M$. Define:*

$$\mathcal{S}_\delta := \left\{\mathbf{x} \in \mathbb{R}^N \,:\, B(\mathbf{x}) + \frac{1}{\alpha}\left(\frac{\sigma}{4}\delta^2 + \frac{\iota}{\mu}\right) \geq 0\right\}, \tag{17}$$

*where $B$ is defined as in* (14) *and $\delta := \sup_{\mathbf{x}\in\Omega_\beta} \mathbf{d}(\mathbf{x})$ with $\mathbf{d}$ as in* (16)*. Provided that:*

$$\lambda \geq \alpha + \frac{\varepsilon\mu}{4\rho}, \tag{18}$$

*then $\mathcal{W} := \mathcal{S}_\delta \cap \Omega_\beta$ is forward invariant for the closed-loop full-order system* (2) *with $\mathbf{u} = \mathbf{k}(\mathbf{x})$.*

Note that if $\mathbf{d} \equiv \mathbf{0}$ and $\iota = 0$ then $\mathcal{S} \cap \Omega_\beta$ is forward invariant and that for a fixed interface $\mathbf{k}$ satisfying (9), it is always possible to satisfy (18) by decreasing both $\alpha$ and $\varepsilon$. Theorem 2 states that the intersection of $\Omega_\beta$ and an *inflated* safe set $\mathcal{S}_\delta \supseteq \mathcal{S}$ is rendered forward invariant, where the inflation is proportional to $\delta$, the bound on $\mathbf{d}$ from (16), and $\iota$, which characterizes the bound on $\|\boldsymbol{\psi}(\mathbf{x}) - \boldsymbol{\kappa}(\boldsymbol{\pi}(\mathbf{x}))\|$ in Lemma 1. The size of this inflation can be shrunk by decreasing $\sigma$, which is decoupled from (18), and by increasing $\alpha$ and $\mu$, which are coupled to (18). Importantly, while there may exist points along a given trajectory such that $B(\mathbf{x}(t)) \leq 0$, this does not necessarily imply violation of the state constraint from (11) since $h(\boldsymbol{\pi}(\mathbf{x})) \geq B(\mathbf{x})$.

*Proof.* Define:

$$B_\delta(\mathbf{x}) := B(\mathbf{x}) + \frac{1}{\alpha}\left(\frac{\sigma}{4}\delta^2 + \frac{\iota}{\mu}\right), \tag{19}$$

noting that $\mathcal{S}_\delta$ from (15) is the zero superlevel set of $B_\delta$. Computing the derivative of $B_\delta$ along the closed-loop full-order dynamics yields:

$$\begin{aligned}\dot{B}_\delta(\mathbf{x}) =& L_{\mathbf{f}}h(\mathbf{y}) + L_{\mathbf{g}}h(\mathbf{y})\mathbf{v} + \nabla h(\mathbf{y}) \cdot \mathbf{d} - \frac{1}{\mu}\dot{V}(\mathbf{x})\\ =& L_{\mathbf{f}}h(\mathbf{y}) + L_{\mathbf{g}}h(\mathbf{y})\boldsymbol{\kappa}(\mathbf{y}) + \nabla h(\mathbf{y}) \cdot \mathbf{d}\\ &+ L_{\mathbf{g}}h(\mathbf{y})(\mathbf{v} - \boldsymbol{\kappa}(\mathbf{y})) - \frac{1}{\mu}\nabla V(\mathbf{x}) \cdot \mathbf{F}(\mathbf{x}, \mathbf{k}(\mathbf{x})),\end{aligned}$$

where the first equality follows from (14) and (12) and the second from adding zero. For notational brevity, $\mathbf{y}$ and $\mathbf{v}$ are viewed as functions of $\mathbf{x}$ via $\mathbf{y} = \boldsymbol{\pi}(\mathbf{x})$ and $\mathbf{v} = \boldsymbol{\psi}(\mathbf{x})$. Lower bounding the above on $\mathcal{W}$ using (13) and (9) yields:

$$\begin{aligned}\dot{B}_\delta(\mathbf{x}) >& -\alpha h(\mathbf{y}) + \frac{1}{\varepsilon}\|L_{\mathbf{g}}h(\mathbf{y})\|^2 + \frac{1}{\sigma}\|\nabla h(\mathbf{y})\|^2\\ &- \|\nabla h(\mathbf{y})\|\|\mathbf{d}\| - \|L_{\mathbf{g}}h(\mathbf{y})\|\|\mathbf{v} - \boldsymbol{\kappa}(\mathbf{y})\|\\ &+ \frac{\lambda}{\mu}V(\mathbf{x}) - \frac{\iota}{\mu}.\end{aligned}$$

By completing squares and lower bounding, we obtain:

$$\begin{aligned}\dot{B}_\delta(\mathbf{x}) >& -\alpha h(\mathbf{y}) - \frac{\varepsilon}{4}\|\mathbf{v} - \boldsymbol{\kappa}(\mathbf{y})\|^2 - \frac{\sigma}{4}\delta^2 + \frac{\lambda}{\mu}V(\mathbf{x}) - \frac{\iota}{\mu}\\ =& -\alpha B_\delta(\mathbf{x}) + \frac{1}{\mu}\left(\lambda - \alpha\right)V(\mathbf{x}) - \frac{\varepsilon}{4}\|\mathbf{v} - \boldsymbol{\kappa}(\mathbf{y})\|^2,\end{aligned}$$

where the equality follows from (14) and (19). Using (8):

$$\dot{B}_\delta(\mathbf{x}) > -\alpha B_\delta(\mathbf{x}) + \frac{1}{\mu}\left(\lambda - \alpha - \frac{\varepsilon\mu}{4\rho}\right)V(\mathbf{x}).$$

Hence, provided (18) holds, we have:

$$\dot{B}_\delta(\mathbf{x}) > -\alpha B_\delta(\mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{W}. \tag{20}$$

Now define $B_\beta(\mathbf{x}) := \beta - V(\mathbf{x})$. This function satisfies:

$$\begin{aligned}\dot{B}_\beta(\mathbf{x}) =& -\nabla V(\mathbf{x}) \cdot \mathbf{F}(\mathbf{x}, \mathbf{k}(\mathbf{x}))\\ \geq& \lambda V(\mathbf{x}) - \iota = -\lambda B_\beta(\mathbf{x}) + \lambda\beta - \iota > -\lambda B_\beta(\mathbf{x}),\end{aligned}$$

$\forall \mathbf{x} \in \mathcal{W}$, where the first inequality follows from (9) and the second from $\beta > \iota/\lambda$. With $B_\delta$ and $B_\beta$ we have that:

$$\mathcal{W} = \{\mathbf{x} \in \mathbb{R}^N \,:\, B_\delta(\mathbf{x}) \geq 0 \wedge B_\beta(\mathbf{x}) \geq 0\}.$$

Since (20) holds with strict inequality, we have $\nabla B_\delta(\mathbf{x}) \neq \mathbf{0}$ whenever $B_\delta(\mathbf{x}) = 0$ implying that zero is a regular value of $B_\delta$ (this can be shown using a similar argument to Lemma 2). Furthermore, since $V$ is a simulation function $\beta$ is a regular value of $V$ by Lemma 2, which implies that 0 is a regular value of $B_\beta$. Let $\mathrm{Act}(\mathbf{x}) := \{i \in \{\beta, \delta\} : B_i(\mathbf{x}) = 0\}$ denote the set of active constraints of $\mathcal{W}$ and note that since zero is a regular value of $B_\beta$ and $B_\delta$, we have [22, Ch. 4]:

$$\mathcal{T}_{\mathcal{W}}(\mathbf{x}) = \{\mathbf{z} \in \mathbb{R}^N \,:\, \nabla B_i(\mathbf{x}) \cdot \mathbf{z} \geq 0, \quad \forall i \in \mathrm{Act}(\mathbf{x})\}.$$

When $B_i(\mathbf{x}) = 0$ we have:

$$\dot{B}_i(\mathbf{x}) = \nabla B_i(\mathbf{x}) \cdot \mathbf{F}(\mathbf{x}, \mathbf{k}(\mathbf{x})) > 0,$$

$\forall i \in \mathrm{Act}(\mathbf{x})$. Hence, for each $\mathbf{x} \in \partial\mathcal{W}$ we have $\mathbf{F}(\mathbf{x}, \mathbf{k}(\mathbf{x})) \in \mathcal{T}_{\mathcal{W}}(\mathbf{x})$, which, by Theorem 1, implies the forward invariance of $\mathcal{W}$, as desired. □

**Remark 1.** Constructing a ROM, simulation function, and corresponding interface is a challenging problem in general; however, various methods exist for certain classes of systems. For linear FOMs, [12], [14] provide systematic methods for constructing linear ROMs and corresponding simulation functions with linear interfaces [14]. When the FOM (2) is control affine, [13] provides a method to construct control affine ROMs, and methods based on sum-of-squares programming [7], backstepping [27], feedback linearization [28], and differential flatness [29] may be used to construct simulation functions and corresponding interfaces. As demonstrated in the following section, the recently developed notion of zero dynamics policies [30] may also be leveraged to construct simulation functions and interfaces for highly underactuated systems. Any of these methods may be integrated with our ROM framework, although we emphasize that explicit expressions for both $V$ and $\mathbf{k}$ are not necessary for *implementation* of our approach (bounds on $V$ are required for *verification* of the conditions in Theorem 2). When such conditions cannot be easily verified (e.g., when $V$ is unknown and $\mathbf{k}$ is a black-box component in an existing control architecture), the conditions of Theorem 2 may still be satisfied by initializing $\alpha$ and $\varepsilon$ very small to ensure safety and then increasing such parameters until adequate performance is achieved.

**Example 2.** Continuing Example 1, suppose our objective is to drive the quadrotor to a goal while avoiding a cylindrical obstacle centered at $(x,y)=(x_o,y_o)\in\mathbb{R}^2$ with a radius of $r_o\in\mathbb{R}_{>0}$. This requirement leads to the state constraint $h(\boldsymbol{\pi}(\mathbf{x})) := \|\boldsymbol{\pi}(\mathbf{x})-(x_o,y_o)\|^2 - r_o^2$ for the FOM, which defines a state constraint set $\mathcal{C}$ as in (11). This state constraint is a valid input-to-state safe CBF [26] for the single integrator ROM, which may be used to synthesize a controller $\boldsymbol{\kappa}$ satisfying (13) using a quadratic program [1] or a smooth safety filter [31]. This reduced-order controller is then refined to produce inputs for the FOM using an off-the-shelf tracking controller (e.g., that in [15]) as an interface between the ROM and FOM. Example trajectories of the quadrotor using this approach are illustrated in Fig. 2, where the left plot displays the position of the quadrotor, which attempts to track velocities from the ROM for different choices of $\alpha$, and the right plot illustrates the difference between the reduced-order controller and the velocity of the quadrotor. As indicated by condition (18) of Theorem 2, picking $\alpha$ too large may lead to safety violations (blue curve), whereas decreasing $\alpha$ allows such conditions to be satisfied for a fixed tracking controller and ensures safety (red and green curves).

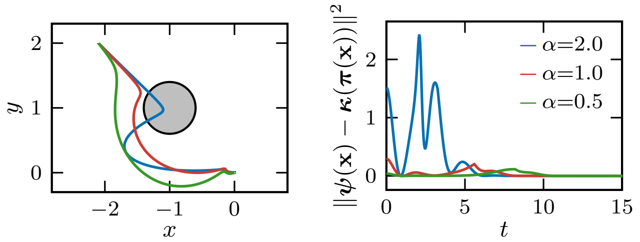

Fig. 2. (**Left**) Evolution of the quadrotor's position for different choices of $\alpha$ in (13). (**Right**) Difference between quadrotor's planar velocity $\mathbf{v}=\boldsymbol{\psi}(\mathbf{x})$ and desired velocity $\boldsymbol{\kappa}(\boldsymbol{\pi}(\mathbf{x}))$ generated by a single integrator. The velocity error remains bounded per Lemma 1, although the error is too large when $\alpha=2$, leading to safety violations. For each of these simulations we took $\varepsilon=20$ and omitted the $\sigma$ term since $\nabla h = L_{\mathbf{g}}h$. Varying $\varepsilon$ had minimal effect on trajectories.

## V. Hardware Demonstrations

We now illustrate our developments through their application[2] to safety-critical control of ARCHER, a 3D hopping robot [19]–[21]. The state of ARCHER is described by $\mathbf{x}=(\mathbf{p},\mathbf{q},\dot{\mathbf{p}},\boldsymbol{\omega})\in\mathbb{R}^3\times\mathbb{S}^3\times\mathbb{R}^3\times\mathfrak{s}^3 =: \mathcal{X}$, where $\mathbf{p}$ denotes the position of the robot, $\mathbf{q}$ a unit quaternion representing its orientation, and $\boldsymbol{\omega}$ its angular rates. Overall, ARCHER is a high-dimensional, hybrid, underactuated system, which prohibits the use of traditional CBF synthesis methods. To overcome this, we leverage a ROM paradigm by designing a safety-critical controller for an abstracted version of ARCHER, the inputs of which are then refined for the full-order system using an existing high-performance interface [21], [30]. To construct a ROM of ARCHER, we define our state and control projection maps as $\boldsymbol{\pi}(\mathbf{x}) := (x,y)\in\mathbb{R}^2$ and $\boldsymbol{\psi}(\mathbf{x}) := (\dot{x},\dot{y})\in\mathbb{R}^2$, where $\mathbf{p}=(x,y,z)$, so that the idealized reduced-order dynamics are:

$$\underbrace{\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix}}_{\dot{\mathbf{y}}} = \underbrace{\begin{bmatrix}0\\ 0\end{bmatrix}}_{\mathbf{f}(\mathbf{y})} + \underbrace{\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}}_{\mathbf{g}(\mathbf{y})}\underbrace{\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix}}_{\mathbf{v}} + \mathbf{d},$$

which corresponds to a disturbed single integrator evolving in the $(x,y)$ plane. The inputs of this ROM are related back to those of ARCHER using a zero dynamics policy [21], [30] as an interface while this policy's associated Lyapunov function is used as a simulation function to certify adequate tracking of the ROM. Similar to Example 2, our objective is to design a controller that avoids a collection of planar obstacles, each of which is captured by a safe set $\mathcal{C}_i$ and CBF $h_i$ as in Example 2. We synthesize a safety-critical controller for this ROM using the safety filter:

$$\begin{aligned}\boldsymbol{\kappa}(\mathbf{y}) := \quad & \underset{\mathbf{v}\in\mathbb{R}^2}{\arg\min} \quad \tfrac{1}{2}\|\mathbf{v}-\boldsymbol{\kappa}_{\mathrm{d}}(\mathbf{y})\|^2 \\ & \text{s.t.} \quad \nabla h(\mathbf{y})\cdot\mathbf{v} \geq -\alpha h(\mathbf{y}) + \frac{1}{\varepsilon}\|\nabla h(\mathbf{y})\|^2,\end{aligned}$$

where $\boldsymbol{\kappa}_{\mathrm{d}}:\mathbb{R}^2\to\mathbb{R}^2$ is a desired reduced-order input, and $h$ combines each individual CBF into a single CBF [32]. For our demonstration, $\boldsymbol{\kappa}_{\mathrm{d}}$ corresponds to desired velocity commands given via joystick that attempt to drive the hopper from one location to another without accounting for safety. This unsafe velocity is then passed to the above safety filter, which outputs a safe velocity command for ARCHER that is tracked by the corresponding interface $\mathbf{k}$. The results of applying this interface to ARCHER are illustrated in Fig. 3 and Fig. 4. As shown in Fig. 3, the resulting full-order trajectory is safe – it avoids obstacles at all times as indicated by the positivity of $h(\boldsymbol{\pi}(\mathbf{x}(t)))$ for all $t\geq 0$. The safe velocity commands generated by the ROM and the velocities achieved by the full-order system are illustrated in Fig. 4, where the difference between the true velocities $\boldsymbol{\psi}(\mathbf{x})$ and

[2] `https://vimeo.com/1010060590?share=copy`

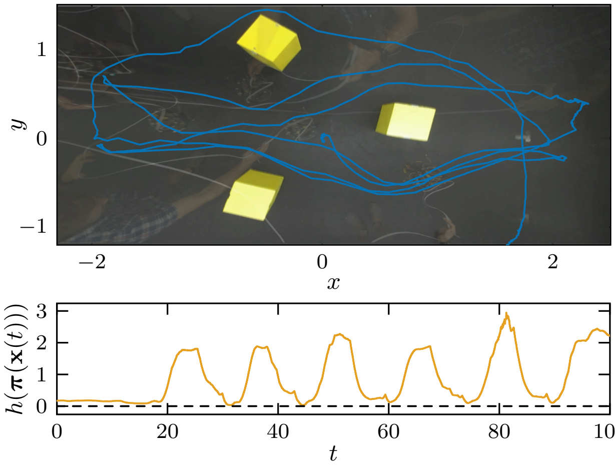

Fig. 3. (**Top**) Evolution of ARCHER's position where the yellow cubes denote the obstacles. (**Bottom**) Evolution of the ROM's CBF $h$ along the trajectory of the full-order system, which remains positive for all time.

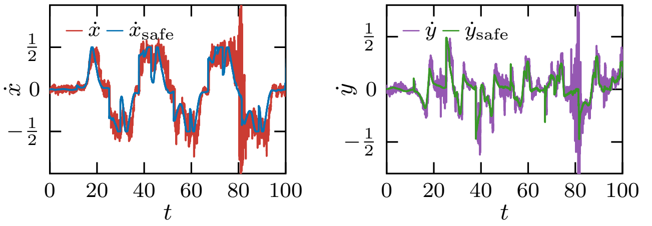

Fig. 4. Commanded velocities output by the reduced-order safety filter $\boldsymbol{\kappa}(\boldsymbol{\pi}(\mathbf{x})) = (\dot{x}_{\rm safe}, \dot{y}_{\rm safe})$ and the velocities of the full-order system $(\dot{x}, \dot{y})$.

desired velocities $\boldsymbol{\kappa}(\boldsymbol{\pi}(\mathbf{x}))$ are bounded per Lemma 1. The fact that the full-order model cannot track the commanded velocities exactly is compensated for by using a relatively small $\alpha = 0.4$ to satisfy the conditions of Theorem 2.

## VI. Conclusions

We presented a framework for safety-critical control with CBFs and ROMs in which the relation between full and reduced-order models is characterized using simulation functions. This paradigm facilitates the use of highly simplified models for safety-critical control design while still providing safety guarantees for the original full-order model. Our theoretical developments were illustrated in both simulation and via hardware demonstrations on a hopping robot.

## References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] M. Krstić, I. Kanellakopoulus, and P. Kokotović, *Nonlinear and Adaptive Control Design*. Wiley, 1995.

[6] M. Chen, S. L. Herbert, H. Hu, Y. Pu, J. F. Fisac, S. Bansal, S. Han, and C. J. Tomlin, "Fastrack: A modular framework for real-time motion planning and guaranteed safe tracking," *IEEE Trans. Autom. Control*, vol. 66, no. 12, pp. 5861–5876, 2021.

[7] K. S. Schweidel, H. Yin, S. W. Smith, and M. Arcak, "Safe-by-design planner–tracker synthesis with a hierarchy of system models," *Annual Reviews in Control*, vol. 53, pp. 138–146, 2022.

[8] D. Benders, J. Köhlher, T. Niesten, R. Babŭska, J. Alonso-Mora, and L. Ferranti, "Embedded hierarchical mpc for autonomous navigation," *arXiv preprint arXiv:2406.11506*, 2024.

[9] S. Singh, B. Landry, A. Majumdar, J. J. Slotine, and M. Pavone, "Robust feedback motion planning via contraction theory," *The International Journal of Robotics Research*, vol. 42, no. 9, pp. 655–688, 2023.

[10] P. Tabuada, *Verification and control of hybrid systems: a symbolic approach*. Spring Science & Business Media, 2009.

[11] C. Belta, B. Yordanov, and E. A. Gol, *Formal methods for discrete-time dynamical systems*. Springer, 2017.

[12] G. J. Pappas, G. Lafferriere, and S. Sastry, "Hierarchically consistent control systems," *IEEE Trans. Autom. Control*, vol. 45, no. 6, pp. 1144–1160, 2000.

[13] G. J. Pappas and S. Simić, "Consistent abstractions of affine control systems," *IEEE Trans. Autom. Control*, vol. 47, no. 5, pp. 745–756, 2002.

[14] A. Girard and G. J. Pappas, "Hierarchical control system design using approximate simulation," *Automatica*, vol. 45, pp. 566–571, 2009.

[15] T. Lee, M. Leoky, and N. H. McClamroch, "Geometric tracking control of a quadrotor UAV on SE(3)," in *Proc. Conf. Decis. Control*, pp. 5420–5425, 2010.

[16] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. Int. Conf. Robot. and Autom.*, pp. 2520–2525, 2011.

[17] P. M. Wensing, M. Posa, Y. Hu, A. Escande, N. Mansard, and A. D. Prete, "Optimization-based control for dynamic legged robots," *IEEE Trans. Robot*, vol. 40, 2024.

[18] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, 2019.

[19] E. Ambrose, *Creating ARCHER: A 3D Hopping Robot with Flywheels for Attitude Control*. PhD thesis, California Institute of Technology, 2022.

[20] N. Csomay-Shanklin, V. D. Dorobantu, and A. D. Ames, "Nonlinear model predictive control of a 3D hopping robot: Leveraging Lie group integrators for dynamically stable behaviors," in *Proc. Int. Conf. Robot. and Autom.*, pp. 12106–12112, 2023.

[21] N. Csomay-Shanklin, W. Compton, I. D. J. Rodriguez, E. R. Ambrose, Y. Yue, and A. D. Ames, "Robust agility via learned zero dynamics policies," *arXiv preprint arXiv:2409.06125*, 2024.

[22] F. Blanchini and S. Miani, *Set-theoretic methods in control*. Springer, 2008.

[23] R. Abraham, J. E. Marsden, and T. Ratiu, *Manifolds, tensor analysis, and applications*. Addison-Wesley, 1983.

[24] R. K. Cosner, I. Sadalski, J. K. Woo, P. Culbertson, and A. D. Ames, "Generative modeling of residuals for real-time risk-sensitive safety with discrete-time control barrier functions," in *Proc. Int. Conf. Robot. and Autom.*, pp. 9960–9967, 2024.

[25] H. K. Khalil, *Nonlinear Systems*. Prentice Hall, 3 ed., 2002.

[26] A. Alan, A. J. Taylor, C. R. He, A. D. Ames, and G. Orosz, "Control barrier functions and input-to-state safety with application to automated vehicles," *IEEE Trans. Contr. Syst. Tech.*, vol. 31, no. 6, pp. 2744–2759, 2023.

[27] K. S. Schweidel, *Robust Hierarchical Control with Connected Layers*. PhD thesis, University of California, Berkeley, 2023.

[28] J. Fu, S. Shah, and H. G. Tanner, "Hierarchical control via approximate simulation and feedback linearization," in *Proc. Amer. Control Conf.*, pp. 1816–1821, 2013.

[29] A. Colombo and A. Girard, "An approximate abstraction approach to safety control of differentially flat systems," in *Proc. Eur. Control Conf.*, pp. 4226–4231, 2013.

[30] W. Compton, I. D. J. Rodriguez, N. Csomay-Shanklin, Y. Yue, and A. D. Ames, "Constructive nonlinear control of underactuated systems via zero dynamics policies," *arXiv preprint arXiv:2408.14749*, 2024.

[31] 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.

[32] T. G. Molnar and A. D. Ames, "Composing control barrier functions for complex safety specifications," *IEEE Contr. Syst. Lett.*, vol. 7, pp. 3615–3620, 2023.