Data-Driven Safety-Critical Control: Synthesizing Control Barrier Functions With Koopman Operators

Carl Folkestad, *Student Member, IEEE*, Yuxiao Chen, *Member, IEEE*, Aaron D. Ames, *Fellow, IEEE*, and Joel W. Burdick, *Member, IEEE*

***Abstract*—Control barrier functions (CBFs) are a powerful tool to guarantee safety of autonomous systems, yet they rely on the computation of control invariant sets, which is notoriously difficult. A backup strategy employs an implicit control invariant set computed by forward integrating the system dynamics. However, this integration is prohibitively expensive for high dimensional systems, and inaccurate in the presence of unmodelled dynamics. We propose to learn discrete-time Koopman operators of the closed-loop dynamics under a backup strategy. This approach replaces forward integration by a simple matrix multiplication, which can mostly be computed offline. We also derive an error bound on the unmodeled dynamics in order to robustify the CBF controller. Our approach extends to multi-agent systems, and we demonstrate the method on collision avoidance for wheeled robots and quadrotors.**

***Index Terms*—Robotics, computational methods, supervisory control.**

I. Introduction

THE FIELD of safety-critical control has received increasing attention since safety is a primary requirement for important autonomous systems, such as autonomous cars and robots. While Control Barrier Functions (CBFs) [1], [2] can provide safety guarantees for autonomous systems, the feasibility of this approach in the presence of input bounds relies on control invariant sets, which may be difficult to compute [3]–[5]. In [6], the authors proposed a CBF approach that uses an implicitly defined control invariant set based on a *backup strategy*, computed by forward integrating the dynamics. The approach was extended to multi-agent systems in a fully decentralized manner in [7]. However, online evaluation of the CBF and its Lie derivative requires forward integration of the system dynamics and a sensitivity matrix, which can be a bottleneck for nonlinear and high dimensional systems. Moreover, when parts of the dynamics are unmodelled, the inaccurate integration can lead to safety violations.

Manuscript received September 14, 2020; revised November 21, 2020; accepted December 11, 2020. Date of publication December 21, 2020; date of current version March 22, 2021. This work was supported in part by Raytheon Technologies. The work of Carl Folkestad was supported by the Aker Scholarship Foundation. Recommended by Senior Editor F. Dabbene. *(Corresponding author: Carl Folkestad.)*

Carl Folkestad is with the Department of Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA 91106 USA (e-mail: cfolkest@caltech.edu).

Yuxiao Chen, Aaron D. Ames and Joel W. Burdick are with the Department of Mechanical and Civil Engineering, California Institute of Technology, Pasadena, CA 91106 USA (e-mail: chenyx@caltech.edu; ames@caltech.edu).

Digital Object Identifier 10.1109/LCSYS.2020.3046159

We propose to learn an approximate Koopman operator for the closed-loop dynamics under the backup strategy. This replaces the expensive forward integration of the dynamics and sensitivity matrix with matrix multiplication. We also develop an error bound on the learned model that supports a robust version of the supervisory controller: it guarantees safety when the learned model forward propagates the backup trajectory. Additionally, a Koopman operator trained on real system data produces more accurate backup trajectories, especially in the presence of unmodeled dynamics, which improves system safety guarantees and, potentially, the system's mobility.

Koopman inspired modelling and identification techniques have received substantial attention [8], [9]. In particular, the Dynamic Mode Decomposition (DMD) and extended DMD (EDMD) methods efficiently identify finite dimensional approximations of the Koopman operator [10], [11]. However, as most prior work focused on learning unknown dynamics, utilizing Koopman-based learning to improve the computation efficiency and dynamic model uncertainty for safety-critical applications has not been previously explored.

Building upon [6], we introduce a data-driven approach that combines Koopman-based learning and CBFs to achieve a safety-critical control system that

1) guarantees safety under limited actuation and errors in the learned Koopman model.
2) requires little online computation to implement.
3) can learn backup trajectories from data, improving the applicability of the approach in real-world scenarios where accurate models may be unavailable.

Our experiments and simulations show that the method can be incorporated in the decentralized framework of [7], further expanding the impact of our efficiency improvements.

Section II reviews CBFs, the backup approach to craft CBFs, and Koopman theory. Section III describes how to learn

Koopman operators of closed-loop dynamics under a backup strategy. Section IV introduces a CBF controller using the Koopman model. Section V presents experimental results.

II. Preliminaries

A. Control Barrier Functions

Using CBFs [1], [12], a supervisory controller can be designed to maintain system safety with minimum intervention. Specifically, consider the control-affine dynamic system:

$$\dot{x} = f(x,u) = a(x) + b(x)u, \quad x \in \mathcal{X} \subseteq \mathbb{R}^n, u \in \mathcal{U} \subseteq \mathbb{R}^m, \tag{1}$$

where $a : \mathcal{X} \to \mathbb{R}^n, b : \mathcal{X} \to \mathbb{R}^{n\times m}$. Suppose we can encode the safety criterion as a CBF, $h : \mathbb{R}^n \to \mathbb{R}$, that satisfies

$$\begin{aligned} &\forall\, x \in \mathcal{X}_0,\ \ h(x) \geq 0 \\ &\forall\, x \in \mathcal{X}_d,\ \ h(x) < 0 \\ &\forall\, x \in \mathcal{X},\ \ \exists\, u \in \mathcal{U} \text{ s.t. } \dot{h} + \alpha(h) \geq 0, \end{aligned} \tag{2}$$

where $\mathcal{X}_0$ is the set of initial states and $\mathcal{X}_d$ are the states to avoid. $\alpha(\cdot)$ is an extended class-$\mathcal{K}$ function, i.e., $\alpha(\cdot)$ is strictly increasing and satisfies $\alpha(0) = 0$. For any legacy controller, the supervisory CBF controller constraints the state inside $\{x \mid h(x) \geq 0\}$ via the following quadratic program:

$$\begin{aligned} u^\star &= \arg\min_{u\in\mathcal{U}} \left\| u - u^0(x) \right\|^2 \\ &\text{s.t. } \nabla h \cdot f(x,u) + \alpha(h(x)) \geq 0, \end{aligned} \tag{3}$$

where $u^0(x)$ is the input of the legacy controller. To ensure that (3) is always feasible when $h(x) \geq 0$, $\{x|h(x) \geq 0\}$ need to be a control invariant set, which is defined as follows.

Definition 1: A set $\mathcal{S} \subseteq \mathbb{R}^n$ is a control invariant set if there exists a control law $\pi : \mathbb{R}^n \to \mathcal{U}$ such that for all initial conditions $x(0) \in \mathcal{S}$ $\forall t \geq 0$, $\Phi^t_{f_\pi}(x(0)) \in \mathcal{S}$.

Here, f_π denotes the closed-loop dynamics under controller π, and $\Phi^t_{f_\pi}(x)$ denotes the system flow map: $\Phi^t_{f_\pi}(x)$ is the state at t, following f_π, starting at x.

B. Control Barrier Function With Backup Controller

This section reviews a backup strategy for control invariant set generation [6]. Given system (1), suppose the following constraint must hold for all $t \geq 0$ to ensure system safety:

$$x(t) \in \mathcal{C} \doteq \{x \in \mathcal{X} | h^{\mathcal{C}}(x) \geq 0\},$$

where the smooth function $h^{\mathcal{C}} : \mathbb{R}^n \to \mathbb{R}$ defines the safe set $\mathcal{C}$. If a control invariant set $\mathcal{S} \subseteq \mathcal{C}$ is known, a CBF can be constructed and the supervisory controller (3) guarantees that the state will remain in $\mathcal{S}$, and thus in $\mathcal{C}$, for any initial condition $x(0) \in \mathcal{S}$. However, computing $\mathcal{S}$ is often difficult. To bypass this problem, the backup strategy approach was introduced in [6] and [13]. Suppose we find a small initial control invariant set, e.g., by linearizing the dynamics, $\mathcal{S}_0 \doteq \{x|h^{\mathcal{S}}(x) \geq 0\} \subseteq \mathcal{C}$, where $h^{\mathcal{S}}$ is smooth. Any equilibrium point of (1) inside $\mathcal{C}$ satisfies the requirement. Then, for a backup strategy $\pi : \mathbb{R}^n \to \mathcal{U}$ and horizon T, define

$$\mathcal{S} \doteq \{x \in \mathcal{X} | \Phi^T_{f_\pi}(x) \in \mathcal{S}_0 \wedge \quad \forall t \in [0,T], \Phi^t_{f_\pi}(x) \in \mathcal{C}\}, \tag{4}$$

which is the set of all initial conditions from which the backup strategy would bring the state to $\mathcal{S}_0$ at $t = T$ while satisfying the constraint $\forall t \in [0,T], x(t) \in \mathcal{C}$.

Since $\mathcal{S}_0$ is a control invariant set, by Definition 1, there exists a control law π_0 that keeps any state starting inside $\mathcal{S}_0$ within $\mathcal{S}_0$. Therefore, we fix $\pi|_{\mathcal{S}_0} = \pi_0|_{\mathcal{S}_0}$ such that any state reaching $\mathcal{S}_0$ will remain within $\mathcal{S}_0$ under π. As a result, $\mathcal{S}$ contains the initial condition that can reach $\mathcal{S}_0$ in $[0,T]$ (instead of exactly at T) while satisfying the state constraint. This result and how to construct a CBF from $\mathcal{S}$ are summarized in the two following lemmas.

Lemma 1 [7]: $\mathcal{S}$ is a control invariant set and $\mathcal{S}_0 \subseteq \mathcal{S} \subseteq \mathcal{C}$.

Lemma 2 [7]: $\mathcal{S}$ is the 0-level set of the following function

$$h(x) = \min\{\min_{t\in[0,T]} h^{\mathcal{C}}(\Phi^t_{f_\pi}(x)), h^{\mathcal{S}}(\Phi^T_{f_\pi}(x))\}. \tag{5}$$

Using Lemma 2, a modified constraint set of the controller (3) enforces the system to satisfy both $h^{\mathcal{C}}$ and $h^{\mathcal{S}}$:

$$\begin{aligned} &u^* = \text{argmin}_{u\in\mathcal{U}} ||u - u^0||^2 \\ &\text{s.t.} \quad \forall t \in [0,T], \frac{dh^{\mathcal{C}}(\Phi^t_{f_\pi}(x))}{dt}(x,u) + \alpha(h^{\mathcal{C}}(\Phi^t_{f_\pi}(x))) \geq 0 \\ &\frac{dh^{\mathcal{S}}(\Phi^T_{f_\pi}(x))}{dt}(x,u) + \alpha(h^{\mathcal{S}}(\Phi^T_{f_\pi}(x))) \geq 0, \end{aligned} \tag{6}$$

where for any $t \in [0,T]$, the total derivative of $h^{\mathcal{C}}(\Phi^t_{f_\pi}(x))$ is computed as

$$\frac{dh^{\mathcal{C}}(\Phi^t_{f_\pi}(x))}{dt} = \nabla h^{\mathcal{C}} \frac{d\Phi^t_{f_\pi}}{dt} = \nabla h^{\mathcal{C}}(\nabla\Phi^t_{f_\pi}(x) f(x,u) - \frac{\partial \Phi^t_{f_\pi}}{\partial t}),$$

where $\nabla\Phi^t_{f_\pi}$ is the sensitivity matrix and $-\frac{\partial\Phi^t_{f_\pi}}{\partial t}$ accounts for the nominal flow under the backup strategy. For fixed dynamics, f_π, and initial condition x, denote the sensitivity matrix at time t as $Q_{x,f_\pi}(t) = \nabla\Phi^t_{f_\pi}$. Then it follows that

$$Q_{x,f_\pi}(0) = I, \dot{Q}_{x,f_\pi} = \nabla f_\pi Q_{x,f_\pi}(t). \tag{7}$$

Proposition 1 [7]: For all $x \in \{x|h(x) \geq 0\}$, a solution to (6) is always feasible and Eq. (6) implies $\dot{h}(x,u) + \alpha(h(x)) \geq 0$. Moreover, h is a CBF that satisfies

- $\forall x \notin \mathcal{C}, h(x) < 0$,
- $\forall x \in \{x \mid h(x) \geq 0\}, \exists u \in \mathcal{U}$, s.t. $\dot{h}(x,u) + \alpha(h(x)) \geq 0$.

When implementing the supervisory controller (6), the first constraint is not implementable because there are uncountably many t in $[0,T]$. Consequently, the continuous interval $[0,T]$ is replaced with a finite sequence, $0 = t_0 < t_1 < \cdots < t_K = T$, $t_{i+1} - t_i = \Delta t$, and the constraint enforced at these points instead. This discretization calls for additional robustness of the control strategy, as is discussed in [14].

Assumption 1: Δt is sufficiently small such that robustly satisfying the constraints of (6) at each t_k (by adding an error buffer) implies constraint satisfaction for all $\Phi^t_{f_\pi}(x)|^{t=t_k+\Delta t}_{t=t_k}$.

C. Koopman Spectral Theory

As the constraint of (6) must be evaluated at discrete points, we will approximate the flow of the system under the backup controller at these points using an approximated discrete-time Koopman operator. We first introduce the underlying theory of discrete-time Koopman operators, and then a procedure to learn an approximation (Section III). Consider the discrete-time autonomous dynamical system

$$x[k+1] = f(x[k]) \tag{8}$$

with state $\mathcal{X} \subseteq \mathbb{R}^n$ and $f(\cdot)$ Lipschitz continuous on $\mathcal{X}$. Define a real-valued observable function $\gamma : \mathcal{X} \to \mathbb{R}$. Then, the Koopman operator is defined as

$$\mathcal{K}\gamma = \gamma \circ f \tag{9}$$

where $\circ$ denotes function composition such that $\mathcal{K}\gamma(x[k]) = \gamma(f(x[k])) = \gamma(x[k+1])$. Crucially, $\mathcal{K}$ is a *linear* operator. Practically, a finite dimensional approximation of the Koopman operator can be estimated using regression techniques. This results in a *lifted* state space model of the form

$$z_{k+1} = Az_k, \qquad x_k = C_x z_k. \tag{10}$$

where A, C_x are learned from data, and $z_0 = \phi(x_0)$, where $\phi : \mathbb{R}^n \to \mathbb{R}^D$ is a dictionary of nonlinear transformations.

III. Koopman Operator for Backup Trajectories

A. Motivating the Use of Koopman Operators

For high dimensional systems and/or long time spans T, the forward integration of the sensitivity matrix in (6) may be prohibitively expensive. Besides, in the presence of unmodelled dynamics, the online integration can be inaccurate. Additionally, the trajectory under the backup strategy typically evolves for a finite time, which allows us to bound the prediction error under the Koopman operator, making it suitable for safety-critical applications. Specifically, we identify a finite dimensional Koopman approximation as in (10) which enables estimating the sensitivity matrix in (6) as

$$\nabla \Phi^{t_k}_{f_\pi}(x_0) \approx \nabla\Big(C_x A^k \phi(x_0)\Big) = C_x A^k \nabla\phi(x_0), \tag{11}$$

which only requires matrix multiplication. Moreover, $C_x A^k$ can be precomputed for $k = 0, \dots, K$, significantly reducing the real-time computational cost of the forward integration of the sensitivity matrix, improving the method's scalability.

B. Learning Koopman Operators for Backup Trajectories

To learn an approximate Koopman operator associated with the closed-loop backup controller dynamics, M trajectories of length T are simulated from initial conditions $x_0^j \in \Omega^i, j = 1, \dots, M$ under backup control. The set Ω is chosen to cover the operating region of interest. States are sampled at fixed time intervals, Δt, resulting in the data sets

$$\mathcal{D} = \left(\left(x_k^j\right)_{k=0}^{\hat{K}}\right)_{j=1}^{M}, \quad i = 1, \dots, N, \quad \hat{K} = T/\Delta t. \tag{12}$$

Define a dictionary of D functions $z = \phi(x)$, $\phi : \mathbb{R}^n \to \mathbb{R}^D$. The choice of basis can be based on system knowledge (i.e., feature engineering) or they can be a generic basis of functions such as monomials of the state up to a certain degree. Selecting the type and number of dictionary functions is an open question in Koopman-based learning. Promising efforts in this direction use data-driven Koopman eigenfunctions [15], [16]. Because we aim to learn approximate Koopman operators of the closed-loop dynamics, which is autonomous, the learning of data-driven Koopman eigenfunctions is simplified. In particular, the method presented by the authors in [15] can be applied with minor modifications.

The data set $\mathcal{D}$ is organized into matrices X, X' as described in (13). Then, the lifted state data matrices are constructed by applying $\phi(x)$ to each column of X and X', denoted $Z = \phi(X), Z' = \phi(X')$ by slight abuse of notation. Utilizing the constructed data matrices, the best fit Koopman operator approximation is inferred by solving a linear least squares regression problem (13a). In addition, regularization can be added to the regression formulation.

$$\min_{A \in \mathbb{R}^{D\times D}} ||AZ - Z'||^2 \tag{13a}$$

$$\min_{C_x \in \mathbb{R}^{n\times D}} ||C^x Z - X||^2 \tag{13b}$$

$$X = [x_0^1 \dots x_{(\hat{K}-1)}^1 \dots x_0^M \dots x_{(\hat{K}-1)}^M], \; Z = \phi(X)$$

$$X' = [x_1^1 \dots x_{\hat{K}}^1 \dots x_0^M \dots x_{\hat{K}}^M], \; Z' = \phi(X')$$

The second regression problem (13b) is formulated to learn the matrix projecting the lifted state back to the original state. Depending on the chosen dictionary of functions $\phi(x)$, C_x may be computed analytically rendering the regression problem obsolete. For example, a monomial basis will typically include the state itself, making the computation trivial.

C. Quantifying the Error of the Koopman Approximation

For safety-critical application, bounding the prediction error of the Koopman operator is essential. We are interested in the system flow at a finite number of sample points $t_0, t_1, \dots, t_K$, where $t_k = k\Delta t$. We denote the true flow of the closed-loop dynamics under the backup controller at these sampling points as $\Phi^{t_k}_{f_\pi}(x)$. Similarly, denote the estimate of the same flow using the learned Koopman operator as $\hat{\Phi}^{t_k}_{f_\pi}(x) = C_x A^k \phi(x)$. Then, the true system evolves as:

$$\Phi^{t_k}_{f_\pi}(x) = \hat{\Phi}^{t_k}_{f_\pi}(x) + \underbrace{\Phi^{t_k}_{f_\pi}(x) - \hat{\Phi}^{t_k}_{f_\pi}(x)}_{\Xi_k(x)} \tag{14}$$

To guarantee safety under the approximation error of the Koopman operator, $\Xi_k(x)$, we first introduce definitions and assumptions that are later used in Propostion 2 to derive an error bound. A key concept to bound the prediction error is the incremental stability of discrete-time systems [17].

Definition 2: The dynamical system (1) is *incrementally stable* in $\mathcal{X} \subseteq \mathbb{R}^n$ if $\forall t \in \mathbb{N}$ $\forall$ $x_1, x_2 \in \mathcal{X}$ and control sequence $u(\cdot) : \mathbb{N} \to \mathbb{R}^m$ such that the closed-loop evolution $\Phi^t_{f_{u(\cdot)}}(x_1)$ and $\Phi^t_{f_{u(\cdot)}}(x_2)$ remain in $\mathcal{X}$, the state evolution satisfies $\|\Phi^t_{f_{u(\cdot)}}(x_1) - \Phi^t_{f_{u(\cdot)}}(x_2)\| \leq \beta(\|x_1 - x_2\|, t)$, where $\beta : \mathbb{R} \times \mathbb{N} \to \mathbb{R}$ is nonincreasing in t and $\beta(\cdot, t)$ is a class-$\mathcal{K}$ function, i.e., β is a class-$\mathcal{KL}$ function.

Remark 1: Since the backup strategy is designed before implementing the CBF, techniques exist to synthesize backup strategies that renders the closed-loop system incrementally stable, even for open-loop unstable systems, e.g., the LMI approach presented in [14, Sec. 3.2].

Proposition 2: For system (1), assume that the closed-loop dynamics under the backup controller $\dot{x} = f_\pi(x)$ is incrementally stable, and that projection matrix C_x is exact, i.e., $x = C_x z, z = \phi(x)$. Further assume that $\phi(x)$ is Lipschitz continuous on $\mathcal{X}$ with Lipschitz constant L and that the distance between points in $\mathcal{X}$ and the closest point in $\mathcal{D}$ is bounded by μ, where $\mu = \max_{x\in\mathcal{X}} \min_{\hat{x}\in\mathcal{D}} ||y - \hat{x}|| < \infty$. Then, error $\Xi_k(x)$ defined in (14) can be upper bounded as

$$||\Xi_k(x)|| \le ||C_x A^k|| L\mu + ||\varepsilon^{max}|| \textstyle\sum_{j=1}^{k-1} ||C_x A^j|| + \mu$$

where the norm of the maximum residual error is given by

$$||\varepsilon^{max}|| = \max_{x_k^j \in \mathcal{D}\setminus x_K^j} ||A\phi(x_k^j) - \phi(x_{k+1}^j)||.$$

Proof: Let $x \in \mathcal{X}, \hat{x} \in \mathcal{D}$, and add and subtract $\Phi_{f_\pi}^{t_k}(\hat{x})$

$$||\Xi_k(x)|| = ||\Phi_{f_\pi}^{t_k}(x) - \hat{\Phi}_{f_\pi}^{t_k}(x) + \Phi_{f_\pi}^{t_k}(\hat{x}) - \Phi_{f_\pi}^{t_k}(\hat{x})||.$$

Define the global error of the Koopman approximation on the training data as $E_k = \phi(x_k) - A^k\phi(x_0)$. Then, $\Phi_{f_\pi}^{t_k}(\hat{x}) = \hat{\Phi}_{f_\pi}^{t_k}(\hat{x}) + C_x E_k$ and separating each term of the norm yields

$$||\Xi_k(x)|| \le ||\hat{\Phi}_{f_\pi}^{t_k}(\hat{x}) - \hat{\Phi}_{f_\pi}^{t_k}(x)|| + ||C_x E_k|| + ||\Phi_{f_\pi}^{t_k}(x) - \Phi_{f_\pi}^{t_k}(\hat{x})||$$

The first term follows from the definition of $\hat{\Phi}_{f_\pi}^{t_k}$, Lipschitz continuity of $\phi(x)$, and a bound on the distance from $\mathcal{X}$ to $\mathcal{D}$:

$$||\hat{\Phi}_{f_\pi}^{t_k}(\hat{x}) - \hat{\Phi}_{f_\pi}^{t_k}(x)|| \le ||C_x A^k(\phi(x) - \phi(\hat{x}))|| \le ||C_x A^k|| L\mu$$

To bound the second term, consider that the error can be expressed as $E_k = \sum_{j=1}^{k-1} A^j \varepsilon_{j-1}$, where $\varepsilon_j = \phi(x_j) - A\phi(x_{j-1})$. The term can then be bounded as follows [18]

$$||C_x E_k|| \le ||C_x \sum_{j=1}^{k-1} A^j \varepsilon_{j-1}|| \le ||\varepsilon^{max}|| \sum_{j=1}^{k-1} ||C_x A^j||.$$

By incremental stability, the last term $||\Phi_{f_\pi}^{t_k}(x) - \Phi_{f_\pi}^{t_k}(\hat{x})|| \le ||x - \hat{x}|| \le \mu$. This results in the desired bound. ∎

Remark 2: Following Section III, the system state is usually included in the function dictionary, making the projection C_x exact. Else, the state can be added, $\bar{\phi}(x) = [x^T \phi(x)^T]^T$.

Remark 3: The maximum distance between a new data point x to the nearest point in data set $\mathcal{D}$ needs only consider to the non-cyclic states of the system. Furthermore, the closed-loop system under the backup controller is stable to the initial control invariant set $\mathcal{S}_0$, and thus the Koopman operator associated with the dynamics is often at least marginally stable. This can be enforced in the learning process if needed, see for example [18]. Consequently, the terms $||C_x A^k||$ in the bound are non-increasing w.r.t. k.

Remark 4: If $h^{\mathcal{C}}, h^{\mathcal{S}}$ depend on a subset of states, the bound in Proposition 2 can be tightened by replacing C_x by C_h, which projects the state subset. If computation permits, $L\mu$ can be replaced with $||C_x A^k(\phi(x) - \phi(\hat{x}))||$ and μ by the exact distance from the current state x to the closest point in $\mathcal{D}$.

IV. CBF With Koopman Predicted State Flow

Using the learned Koopman operators for the closed-loop dynamics under the backup controller, the supervisory controller (6) can be reformulated as an optimization problem:

$$\begin{aligned} u^* = &\operatorname{argmin}_{u\in\mathcal{U}} ||u - u^0||^2 \\ \text{s.t.}\ &\nabla h^{\mathcal{C}}(C_x A^k \nabla_x \phi(x)(f(x,u) - f_\pi(x)) \\ &\quad + \alpha(h^{\mathcal{C}}(C_x A^k \phi(x))) \ge 0, \quad k = 1, \ldots, K \\ &\nabla h^{\mathcal{S}}(C_x A^K \nabla_x \phi(x)(f(x,u) - f_\pi(x))) \\ &\quad + \alpha(h^{\mathcal{S}}(C_x A^K \phi(x))) \ge 0, \end{aligned} \tag{15}$$

where the sensitivity matrix for every t_k is replaced as (11).

By using the generalization bound derived in Propostion 2, the supervisory controller can be made robust to the approximation error of the flow estimated by the Koopman operator.

Theorem 1: Consider the control system (1) and an associated backup controller with backup trajectories approximated by a learned Koopman operator satisfying the assumptions of Propostion 2. If the constraints (15) are satisfied for $C_x A^k \phi(x) + \Delta_k(x)$, with $\Delta_k(x) = \{\delta : ||\delta - x|| \le ||\Xi_k(x)||\}$, then the true system flow satisfies

$$\Phi_{f_\pi}^{t_k}(x) \in C_x A^k \phi(x) + \Delta_k(x).$$

Under Assumption 1, $\Phi_{f_\pi}^{t}(x) \in \mathcal{C}\ \forall t \in [0, T]$, i.e., the true system evolution under π satisfies the safety constraint.

Proof: The error bound on the evolution of the true system follows directly from Proposition 2. The robustness of the controller follows from [14], Theorem 1. ∎

Remark 5: For an incrementally stable system, the constraints of (15) can be enforced for all $\Phi_{f_\pi}^{t_k}(x) \in C_x A^k \phi(x) + \Delta_k(x)$ by calculating the flow over the backup trajectory for the nominal value of x, and evaluating the constraints over the set. This greatly simplifies the computation and enables the use of interval arithmetic libraries such as `INTLAB` [19] and `libaffa` [20], see [14] for details.

V. Simulation and Experimental Results

A. Ground Robot Obstacle Avoidance

We first demonstrate our method performing single-agent obstacle avoidance. A legacy controller drives the robot in a straight line to a distant point, and then returns. The supervisory controller maintains safety by utilizing a backup strategy applying maximum brake and turn. The robot is modeled as a Dubin's car with dynamics:

$$\dot{x} = \begin{bmatrix} \dot{X} & \dot{Y} & \dot{v} & \dot{\theta} \end{bmatrix}^\mathsf{T} = \begin{bmatrix} v\cos(\theta) & v\sin(\theta) & a & r \end{bmatrix}^\mathsf{T}, \tag{16}$$

where X, Y, v, θ denote its Cartesian coordinates, velocity, and heading angle. The acceleration a and yaw rate r inputs are bounded by $a^{\max}$ and $r^{\max}$. We conduct simulated and physical experiments using Georgia Tech's Robotarium [21].

We learn a Koopman operator for the closed-loop system by collecting 200 data points sampled in the operating region of interest while the system operates under backup control.[1],[2] Based on knowledge of the system, 21 dictionary functions are chosen, $\phi(x) =$

[1]Code available at https://github.com/Cafolkes/koopman-cbf.

[2]Video of all the experiments available at https://youtu.be/IfBUbtKP53c.

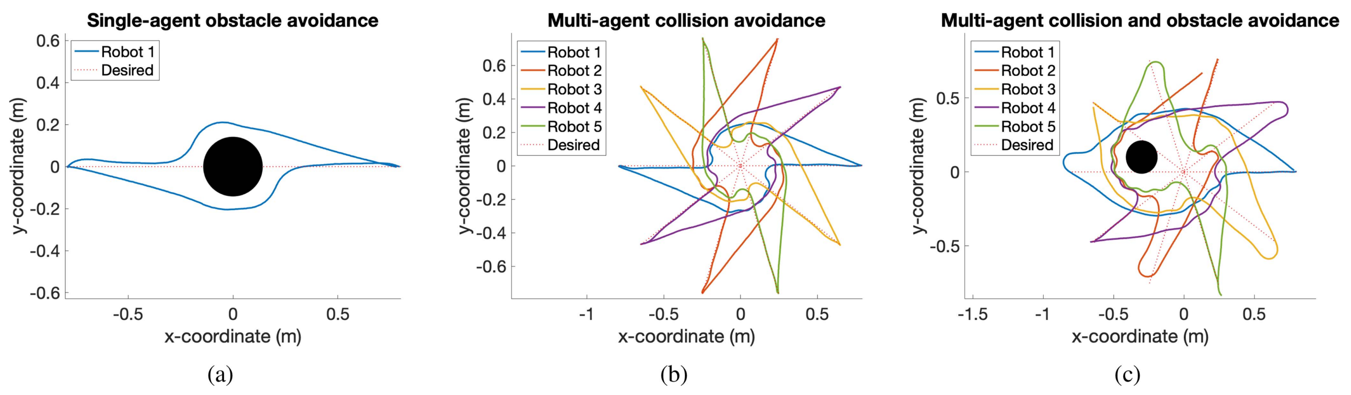

Fig. 1. Robotarium experiment traces, the Koopman CBF controller guarantees zero collisions. Each robot visits a point and returns to its initial position while (a) avoiding obstacle, (b) avoiding collision, and (c) avoiding collision and obstacle.

TABLE I
STATE AND SENSITIVITY MATRIX FORWARD INTEGRATION COMPUTATION TIMES PER AGENT

	`ODE45` (ms)		`CasADi` (ms)		Koopman (ms)	
	mean	std	mean	std	mean	std
Dubin's car	21.5	12.4	1.1	0.5	0.2	0.1
Quadrotor	28.0	1.5	4.5	0.2	0.2	0.1

$[1\, X\, Y\, v\, \theta\, v^2, \dots, v^5\, cos(\theta)\, sin(\theta)\, vcos(\theta), \dots, v^5 sin(\theta)]^T$. The Lipschitz constant of the function dictionary used in Proposition 2 is calculated by maximizing the symbolic Jacobian over the operating region of interest. The robust supervisory controller that enforces the conditions of Theorem 1 is implemented with `INTLAB` [19]. Fig. 1(a) shows the robot's path. Table I reports computation times for forward integration of the sensitivity matrix and dynamics using `ode45`, `CasADi`, and the learned Koopman operator. Our approach reduced computation time by $\sim 80\%$.

B. Multi-Agent Ground Robot Collision Avoidance

We extend the single-agent supervisory controller (15) to a safe decentralized multi-agent controller, following [7]. Each agent has a backup strategy that brings it to a stable equilibrium. All agents' backup strategies are known a priori to every agent as part of a centralized design. Then, each agent measures the adjacent agents' states, and ensures that if other agents execute a backup strategy, its own backup strategy avoids the danger set.

The initial positions of 5 robots are equally spaced on a circle. A greedy legacy controller drives each robot to the opposite point on the circle, and then back. The supervisory controller avoids collision between agents and with a fixed obstacle. Fig. 1(b) shows the robot's motion traces as they execute the task without a fixed obstacle. As the robots near the circle center, they circle around each other and/or the obstacle to avoid collision. The seemingly coordinated behavior is the result of the decentralized supervisory controllers. Similarly, Fig. 1(c) shows the result when a fixed obstacle is added. The results demonstrate that the CBF utilizing the learned Koopman operator can maintain system safety.

C. Multi-Agent Quadrotor Collision Avoidance and Landing

To showcase the method's scalability, we consider 3 quadrotors trying to land on the same landing pad while avoiding collisions and hard ground impacts. The 16-dimensional quadrotor state is $x = [\mathbf{r}, \mathbf{v}, \boldsymbol{\theta}, \mathbf{w}, \Omega]^\intercal$ where $\mathbf{r}$ is the position, $\mathbf{v}$ is the velocity, $\boldsymbol{\theta}$ are the Euler angles, $\mathbf{w}$ is the angular velocity, and Ω is the vector of propeller rotation rates. The motor drive voltages are the control inputs, $u = [V_1, V_2, V_3, V_4]^\intercal$. The dynamics are derived from a force-balance in a rotating frame, and a first order motor model. This high dimensional system makes the sensitivity matrix expensive to forward integrate with previous methods, especially in the multi-agent case. To highlight the benefits of learning the backup-controlled dynamics from data, we introduce an external forcing that models the *ground effect*: a thrust amplification model (see [22, Sec. 2.2.]) acts when a quadrotor nears the ground. This unmodeled effect is not captured by the nominal dynamics model, but is captured when learning the Koopman operator.

The legacy PD controller is designed to drive each agent from its initial position to a setpoint $\mathbf{r}_d = [0, 0, 0]^T$ with velocity $\mathbf{v}_d = [0, 0, -2]^T$, chosen such that the legacy controller lands at a velocity that may cause damage. The backup policy is a PD controller that aims to quickly decelerate the quadrotor to hover. The corresponding backup set is a small ball around zero linear velocity, pitch, and roll angles.

We compare the Koopman-based supervisory controller with controllers using forward integration (see Section II-B). We consider 3 scenarios to highlight some benefits of the method; (1) landing with the supervisory controller only enforcing collision avoidance, (2) landing with the supervisory controller enforcing collision avoidance and avoiding crashing into the ground, and, finally, (3) landing with the supervisory controller enforcing collision avoidance and avoiding crashing into the ground with a Koopman operator trained on data that captures the ground effect. The collision avoidance barrier function $h_c(\mathbf{r}_i, \mathbf{r}_j)$ is defined pairwise for all $i, j = 1, 2, 3, i \neq j$ as a ball of radius r around each agent, $h_c(\mathbf{r}_i, \mathbf{r}_j) = \mathbf{r}_i^T \mathbf{r}_j - r^2$. The ground avoidance barrier function $h_g(\mathbf{r}_i)$ seeks to keep each agent above a paraboloid placed at the center of the landing pad $h_g(\mathbf{r}_i) = z - \mathbf{r}_i^T \mathbf{r}_i$. Landing is defined as reaching an altitude less than 1 cm.

The data set is collected by simulating the system under the backup controller from 250 initial points sampled from the operating region of interest. The data set includes data points close to the ground, thereby capturing trajectories influenced by the ground effect. Based on knowledge of the system, 47 dictionary functions are chosen consisting of the state,

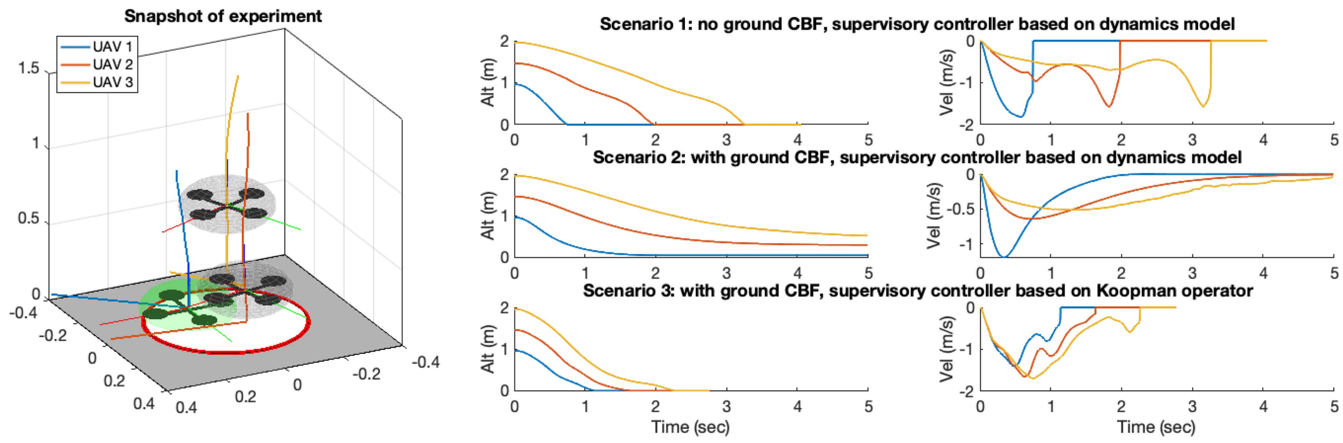

Fig. 2. (left) Snapshot of scenario 3 simulation and traces of full trajectory for each UAV. (right) Altitude and velocity of each agent. The Koopman CBF controller guarantees zero collisions while maintaining sufficient mobility to achieve landing.

products of the angular rates and trigonometric functions of the Euler angles, rotation matrix terms, angular and linear acceleration terms, and a simple ground effect nonlinearity, $\frac{\Omega_i^2}{1-(R/(4z)^2}$ for each $\Omega_i, i = 1, \ldots, 4$, where R is the propeller radius. Then, the Koopman operator is learned and the controller for scenario 3 constructed.

A snapshot of scenario 3, and the altitude and velocities for all scenarios are shown in Fig. 2. When not enforcing the ground avoidance CBF (scenario 1), the quadrotors may hit the ground at speeds up to 1.2 m/s. Enforcing the ground avoidance CBF based on the dynamics model, but without ground effects, causes the supervisory controller to be too conservative, thereby prohibiting the quadrotor from reaching the ground. Finally, when the ground avoidance is enforced with a Koopman operator trained on data capturing the ground effect, smooth landing is achieved with impact speeds less than 0.3 m/s. Furthermore, the forward integration computation time is reduced by approximately 95% when using the learned Koopman operator (see Table I).

VI. Conclusion

We present a method using learned Koopman operators to improve the computational efficiency of a control barrier function-based supervisory controller and show how to maintain theoretical guarantees even though backup trajectories are learned from data. Through physical experiments on wheeled robots and simulated experiments on quadrotors, we show that our method can be incorporated in a decentralized supervisory controller design that can take full advantage of the computational benefits of Koopman-based forward integration. Furthermore, we exemplify how learning the backup trajectories from data can improve the mobility of the system, thereby improving overall control performance.

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, Aug. 2017.

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

[3] S. Rakovic, A. Teel, D. Mayne, and A. Astolfi, "Simple robust control invariant tubes for some classes of nonlinear discrete time systems," in *Proc. IEEE Conf. Decis. Control*, 2006, pp. 6397–6402.

[4] F. Blanchini, "Set invariance in control," *Automatica*, vol. 35, no. 11, pp. 1747–1767, 1999.

[5] Y. Chen, H. Peng, J. Grizzle, and N. Ozay, "Data-driven computation of minimal robust control invariant set," in *Proc. IEEE Conf. Decis. Control*, 2018, pp. 4052–4058.

[6] T. Gurriet, M. Mote, A. D. Ames, and E. Feron, "An Online approach to active set invariance," in *Proc. IEEE Conf. Decis. Control*, 2019, pp. 3592–3599.

[7] Y. Chen, A. Singletary, and A. D. Ames, "Guaranteed obstacle avoidance for multi-robot operations with limited actuation: A control barrier function approach," *IEEE Control Syst. Lett.*, vol. 5, no. 1, pp. 127–132, Jan. 2021.

[8] C. W. Rowley, I. Mezi, S. Bagheri, P. Schlatter, and D. S. Henningson, "Spectral analysis of nonlinear flows," *J. Fluid Mech.*, vol. 641, pp. 115–127, Dec. 2009.

[9] M. Budišić, R. Mohr, and I. Mezić, "Applied Koopmanism," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 22, no. 4, 2012. Art. no. 047510.

[10] P. J. Schmid, "Dynamic mode decomposition of numerical and experimental data," *J. Fluid Mech.*, vol. 656, pp. 5–28, Aug. 2010.

[11] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, "A data–Driven approximation of the Koopman operator: Extending dynamic mode decomposition," *J. Nonlinear Sci.*, vol. 25, no. 6, pp. 1307–1346, 2015.

[12] A. D. Ames, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs with application to adaptive cruise control," in *Proc. IEEE Conf. Decis. Control*, 2014, pp. 6271–6278.

[13] A. Singletary, P. Nilsson, T. Gurriet, and A. D. Ames, "Online Active Safety for Robotic Manipulators," in *Proc. IEEE Conf. Intell. Robots Syst.*, 2019, pp. 173–178.

[14] A. Singletary, Y. Chen, and A. D. Ames, "Control barrier functions for sampled-data systems with input delays," 2020. [Online]. Available: arXiv:2005.06418.

[15] C. Folkestad, D. Pastor, I. Mezic, R. Mohr, M. Fonoberova, and J. Burdick, "Extended dynamic mode decomposition with learned Koopman Eigenfunctions for prediction and control," in *Proc. Amer. Control Conf.*, 2019, pp. 3906–3913.

[16] M. Korda and I. Mezic, "Optimal construction of Koopman eigenfunctions for prediction and control," *IEEE Trans. Autom. Control*, vol. 65, no. 12, pp. 5114–5129, Dec. 2020.

[17] D. N. Tran, B. S. R´uffer, and C. M. Kellett, "Incremental stability properties for discrete-time systems," in *Proc. IEEE Conf. Decis. Control*, 2016, pp. 477–482.

[18] G. Mamakoukas, I. Abraham, and T. D. Murphey, "Learning data-driven stable Koopman operators," 2020. [Online]. Available: arXiv:2005.04291.

[19] S. M. Rump, "INTLAB- INTerval LABoratory," in *Developments in Reliable Computing*. Dordrecht, The Netherlands: Kluwer Acad. Publ., 1999, pp. 77–104.

[20] O. Gay, D. Coeurjolly, and N. Hurst, "Libaffa-c++ affine arithmetic library for gnu/linux," 2006.

[21] D. Pickem *et al.*, "The Robotarium: A remotely accessible swarm robotics research testbed," in *Proc. IEEE Conf. Robot. Autom.*, 2017, pp. 1699–1706.

[22] P. Sanchez-Cuevas, G. Heredia, and A. Ollero, "Characterization of the aerodynamic ground effect and its influence in multirotor control," *Int. J. Aerosp. Eng.*, vol. 2017, Aug. 2017, Art. no. 1823056.