

# Neural Koopman Control Barrier Functions for Safety-Critical Control of Unknown Nonlinear Systems

Vrushabh Zinage

Efstathios Bakolas

**Abstract**—We consider the problem of synthesis of safe controllers for nonlinear systems with unknown dynamics using Control Barrier Functions (CBF). We utilize Koopman operator theory (KOT) to associate the (unknown) nonlinear system with a higher dimensional bilinear system and propose a data-driven learning framework that uses a learner and a falsifier to simultaneously learn the Koopman operator based bilinear system and a corresponding CBF. We prove that the learned CBF for the latter bilinear system is also a CBF for the unknown nonlinear system by characterizing the  $\ell^2$ -norm error bound between these two systems. We show that this error can be partially tuned by using the Lipschitz constant of the Koopman based observables. The CBF is then used to formulate a quadratic program to compute inputs that guarantee safety of the unknown nonlinear system. Numerical simulations are presented to validate our approach.

**Index Terms**—Computational methods, Modeling, Robotics

## I. INTRODUCTION

In this paper, we consider the problem of synthesizing controllers that render a nonlinear system with unknown dynamics safe, that is, controllers that guarantee that a certain set will be forward invariant relative to the (unknown) nonlinear system (we refer to the latter set as the *safe set*). In particular, we consider the problems of 1) representing the unknown nonlinear system as a lifted Koopman based bilinear system and 2) learning a valid Control Barrier Function (CBF) for the unknown system that can induce a controller that can guarantee safety.

**Literature review:** In recent years, Koopman operator theory (KOT) has emerged as a popular tool to analyze and control nonlinear systems in applications in various fields, including robotics and aerospace engineering [1]–[5]. KOT enables the transformation of a controlled nonlinear system to an infinite-dimensional bilinear system. However, the control design problem based on the infinite-dimensional bilinear system poses practical and computational challenges. For this reason, methods such as the Extended Dynamic Mode Decomposition (EDMD) are utilized to approximate the infinite-dimensional linear system corresponding to a control-free nonlinear dynamical system to a finite-dimensional linear system. The lifted state (that is, the state of the lifted model) is determined by the so-called Koopman-based observables which are functions of the original states of the nonlinear system. However, the computation of these observables is a major challenge in general. Recent methods either 1) guess the observables by identifying certain terms in the nonlinear dynamics [6], 2) derive the set of observables for particular classes of nonlinear systems such as those describing attitude dynamics and rigid body motion [1], [2], [7] or 3) use machine learning tools to learn the observables [8]. Moreover, for a selected set of observables there is no guarantee that the lifted linear model constructed by using the EDMD algorithm can approximate the controlled

nonlinear dynamics accurately. To that end, [9] provides an approach for the selection of the observables from a given dictionary of “guessed” observables. [10] constructs a lifted linear system corresponding to a nonlinear system, such a quadrotor, from guessed observables using the EDMD algorithm and design an LQR type controller based on the lifted system representation of the nonlinear system. Authors of [3] consider modelling and control of soft robots using KOT based methods. Further, [11] constructs a Koopman based lifted linear system by characterizing the observables as higher derivatives of the underlying nonlinear dynamics.

Two major requirements for a controlled nonlinear system is closed-loop stability and safety which can be guaranteed by designing controllers that rely on the notions of Control Lyapunov Functions (CLF’s) and Control Barrier Functions (CBF’s), respectively. Safety is of prime importance for many engineering applications. To this end, forward invariance of a set relative to a system can be thought as the dual property for safety of the latter system. The notion of CBF, which was first introduced in [12], can be used to design a feedback control law that would render the safety set forward invariant relative to the closed-loop system. Control design methods based on Control Barrier functions (CBF’s) have proved to be effective tools to guarantee safety for nonlinear systems. Although there are many methods to synthesize CBF’s, most of them assume that either an exact or an approximate model of the underlying nonlinear dynamics is known a priori. Authors of [13] learn a CBF from safe trajectories generated by an expert but assumes that the control affine nonlinear dynamics is known a priori. Next, [14] provide a learning framework for synthesis of CBF’s from sensor data and uses a vector classifier function to characterize the barrier function. Recently, the authors of [15] have used KOT to propagate the nonlinear system via a lifted linear system for faster computation of invariant safety sets. However, [15] assumes that the CBF and the Koopman based observables are known. Further, [16]–[21] assume that a CBF is known. Synthesizing CBF’s for unknown nonlinear system using Gaussian processes, under the assumption that the control vector field term of the control affine system is known has been considered in [22]. In addition, in [22], the CBF is characterized by a polynomial function with unknown coefficients that need to be learned. Recently, there has been a plethora of results that leverage neural networks and Sequential Modulo Theories (SMT) solvers to learn certificate functions such as CLF using a learner and a falsifier [23]–[25]. Our approach is distinct from [25] in two important ways. First, we characterize the  $\ell_2$  norm of the difference between the time derivatives of the CBF along the trajectories of the Koopman based bilinear system and the unknown nonlinear system and prove that the learned CBF for the bilinear system also acts as a CBF for the unknown system. Second, we show that the  $\ell_2$  norm of this difference can be adjusted by tuning the Lipschitz continuity of the CBF via spectral normalization of neural networks.

**Main contributions:** The main contribution of the paper is two-fold. First, we simultaneously learn a Koopman based bilinear model and a corresponding CBF by using a learner

This research has been supported in part by NSF award CMMI-1937957. Vrushabh Zinage (graduate student) and Efstathios Bakolas (Associate Professor) are with the Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, Austin, Texas 78712-1221, USA, Emails: vrushabh.zinage@utexas.edu; bakolas@austin.utexas.edu

and a falsifier. This is in contrast with most approaches in the field that assume that a CBF and / or the dynamics of the nonlinear system are known. Second, we show that the learned CBF for the learned bilinear system also acts as a CBF for the unknown nonlinear system. Further, we use the learned bilinear model and the corresponding CBF to compute control inputs guaranteeing safety for the unknown nonlinear system via the solution of a quadratic program (QP). We verify our approach for a collision avoidance problem with a differential drive robot.

*Structure of the paper:* Section II discusses preliminaries and Section III describes the problem statement. Section IV discusses the Koopman based CBF and Section V presents the learning framework to simultaneously learn the Koopman based bilinear system and a corresponding CBF. Section VI discusses the numerical simulations followed by concluding remarks in Section VII.

## II. PRELIMINARIES

Consider a nonlinear control-affine system:

$$\dot{\mathbf{x}} = F(\mathbf{x}, \mathbf{u}) = f(\mathbf{x}) + g(\mathbf{x})\mathbf{u}, \quad \mathbf{x}(0) = \mathbf{x}_{\text{init}}, \quad (1)$$

where  $f$  and  $g$  are Lipschitz continuous functions,  $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^n$  is the state and  $\mathbf{u} \in \mathcal{U} \subset \mathbb{R}^m$  is the control input. Although we do not consider hard input or state constraints, we will assume that the sets  $\mathcal{X}$  and  $\mathcal{U}$  are compact (the latter assumption is needed to avoid some practical issues that can arise when solving the learning problem over an unbounded domain). In addition, we assume that the nonlinear dynamics governed by  $f$  and  $g$  are unknown. We will also assume that the zero vector is the equilibrium point of (1). To keep the notation simple, we omit the dependence of the state and the control input on time. A discrete-time version of the nonlinear system (1), which can be obtained by using, for instance, the fourth order Runge-Kutta discretization scheme, is given by

$$\mathbf{x}_{k+1} = \ell(\mathbf{x}_k, \mathbf{u}_k), \quad \mathbf{x}_0 = \mathbf{x}_{\text{init}}, \quad (2)$$

where  $\mathbf{x}_k = \mathbf{x}(k\Delta t) + O(\Delta t^4)$ ,  $\mathbf{u}_k = \mathbf{u}(k\Delta t)$  and  $\Delta t > 0$  is the sampling time period used in the Runge-Kutta discretization scheme.

**Assumption 1.** The vector field  $F$  is Lipschitz continuous in  $\mathcal{X} \times \mathcal{U}$  and the Lipschitz constant  $K_F$  of the unknown nonlinear system (1) is known a priori, that is,

$$\|F(\mathbf{x}, \mathbf{u}) - F(\mathbf{y}, \mathbf{v})\| \leq K_F \|(\mathbf{x}, \mathbf{u}) - (\mathbf{y}, \mathbf{v})\| \quad (3)$$

for all  $(\mathbf{x}, \mathbf{u}), (\mathbf{y}, \mathbf{v}) \in \mathcal{X} \times \mathcal{U}$ .

**Remark 1.** There is no loss of generality in assuming that  $F$  is globally Lipschitz continuous given that local Lipschitz continuity over a compact set implies global Lipschitz continuity [26]. In contrast to [22] which assumes that the function  $g$  is known, in this paper we do not make such assumptions.

### A. Control Barrier Functions

Consider a continuously differentiable function  $h : \mathcal{X} \rightarrow \mathbb{R}$  for the control-affine system (1) that satisfies

$$h(\mathbf{x}) \geq 0, \quad \forall \mathbf{x} \in \mathcal{X}_0 \quad \text{and} \quad h(\mathbf{x}) < 0, \quad \forall \mathbf{x} \in \mathcal{X}_d \quad (4)$$

where  $\mathcal{X}_0$  denotes the safe set and  $\mathcal{X}_d$  the set of states that must be avoided.

**Assumption 2.** The set  $\mathcal{X}_0$ , which is the complement of  $\mathcal{X}_d$  in  $\mathcal{X}$ , i.e.,  $\mathcal{X}_0 = \mathcal{X} \setminus \mathcal{X}_d$ , is control (positively) invariant, that is, there exists a control law  $\pi(\mathbf{x}) : \mathcal{X}_0 \rightarrow \mathcal{U}$  such that for all initial conditions  $\mathbf{x}_{\text{init}} \in \mathcal{X}_0$ , the solution  $\varphi(\cdot; \mathbf{x}_{\text{init}})$  to (1) with  $\varphi(0; \mathbf{x}_{\text{init}}) = \mathbf{x}_{\text{init}}$  satisfies  $\varphi(t; \mathbf{x}_{\text{init}}) \in \mathcal{X}_0$  for all  $t \geq 0$ .

The function  $h$  is said to be a Control Barrier Function (CBF) when it satisfies the following definition:

**Definition 1.** Let  $\mathcal{X}_0 \subset \mathbb{R}$  be the 0-superlevel set of  $h : \mathcal{X} \rightarrow \mathbb{R}$ , then  $h$  is a Control Barrier Function (CBF) for (1), if there exists a class  $\mathcal{K}_\infty$  function<sup>1</sup>  $\alpha : \mathbb{R} \rightarrow \mathbb{R}$  such that

$$\sup_{\mathbf{u} \in \mathcal{U}} L_f h(\mathbf{x}) + L_g h(\mathbf{x})\mathbf{u} \geq -\alpha(h(\mathbf{x})), \quad \forall \mathbf{x} \in \mathcal{X} \quad (5)$$

where  $L_f h(\mathbf{x})$  and  $L_g h(\mathbf{x})$  are the Lie derivatives of  $h(\mathbf{x})$  with respect to  $f$  and  $g$ , respectively.

Let us denote the following (state-dependent) set of inputs:

$$K_{\text{CBF}}(\mathbf{x}) = \{\mathbf{u} \in \mathcal{U} : L_f h(\mathbf{x}) + L_g h(\mathbf{x})\mathbf{u} \geq -\alpha(h(\mathbf{x}))\}. \quad (6)$$

Then, the selection of an input  $\mathbf{u}(t)$  from  $K_{\text{CBF}}(\mathbf{x}(t))$  at each time  $t \geq 0$  will ensure that the set  $\mathcal{X}_0$  will be forward invariant relative to (1). In other words, given an initial condition  $\mathbf{x}_{\text{init}} \in \mathcal{X}_0$ , the solution to (1)  $\mathbf{x}(t)$  will also remain in  $\mathcal{X}_0$  if  $\mathbf{u}(t) \in K_{\text{CBF}}(\mathbf{x}(t))$  for all  $t \geq 0$ . Further, given a nominal controller  $k(\mathbf{x})$ , a feedback controller  $\mathbf{u}_S(\mathbf{x})$  can be designed that can guarantee that the system will remain in the safe set  $\mathcal{X}_0$  by solving the following quadratic program (QP):

$$\text{CBF-QP} \quad \mathbf{u}_S(\mathbf{x}) := \underset{\mathbf{u} \in \mathcal{U}}{\text{argmin}} \quad \|\mathbf{u} - k(\mathbf{x})\|^2 \quad (7a)$$

$$\text{s.t.} \quad L_f h(\mathbf{x}) + L_g h(\mathbf{x})\mathbf{u} \geq -\alpha(h(\mathbf{x})). \quad (7b)$$

Because  $f$  and  $g$  are unknown, one cannot compute the solution of QP (7) and use it to design a feedback controller guaranteeing safety for the unknown nonlinear system (1). Further, if the dynamics of the system (1) were known, the safe controller  $\mathbf{u}_S(\mathbf{x})$  would not be implementable as the constraint (7b) must be satisfied for uncountable  $t \in [0, T]$  where  $T$  is the total time horizon. Therefore, we consider finite time intervals  $t_i = i\Delta t$ ,  $t_K = T$  and compute control inputs at those time intervals. This might lead to violation of constraint (7b). For this reason, we make the following assumption:

**Assumption 3.** The discretization step  $\Delta t > 0$  is sufficiently small so that satisfaction of constraint (7b) at  $t_k$  implies constraint satisfaction (7b) for all  $t \in [t_k, t_k + \Delta t]$ .

**Remark 2.** In view of Theorem 3 from [28], Assumption 3 always holds true provided that  $F$  and  $\alpha$  are Lipschitz continuous.

### B. Koopman Operator Theory

Given a nonlinear system,  $\dot{\mathbf{x}} = f(\mathbf{x})$ , the Koopman operator  $\mathcal{K}$  is a linear infinite-dimensional operator which linearly propagates a set of observables  $\Phi(\mathbf{x})$  which are functions of states. In other words,  $\mathcal{K}[\Phi(\mathbf{x})] = \Phi \circ \mathcal{F}_t$ , where  $\Phi(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^N$ ,  $N > n$ ,  $\circ$  is the composition operator and  $\mathcal{F}_t$  is the flow map of the uncontrolled dynamics  $\dot{\mathbf{x}} = f(\mathbf{x})$  which is given by

$$\mathcal{F}_t(\mathbf{x}(t_0)) = \mathbf{x}(t_0) + \int_{t_0}^{t_0+t} f(\mathbf{x}(\tau))d\tau. \quad (8)$$

<sup>1</sup>A function  $\alpha(\cdot) : \mathbb{R} \rightarrow \mathbb{R}$  belongs to the class of  $\mathcal{K}_\infty$  functions if it is strictly increasing and in addition,  $\alpha(0) = 0$  and  $\lim_{r \rightarrow \infty} \alpha(r) = \infty$  [27].

For the nonlinear control affine system (1), the time derivative of  $\Phi(\mathbf{x})$  along the trajectories of (1) is given by

$$\begin{aligned}\dot{\Phi}(\mathbf{x}) &= \nabla_{\mathbf{x}}\Phi(\mathbf{x})[f(\mathbf{x}) + g(\mathbf{x})\mathbf{u}] \\ &= \nabla_{\mathbf{x}}\Phi(\mathbf{x})f(\mathbf{x}) + \nabla_{\mathbf{x}}\Phi(\mathbf{x})g(\mathbf{x})\mathbf{u} \\ &= \mathcal{K}\Phi(\mathbf{x}) + \nabla_{\mathbf{x}}\Phi(\mathbf{x})\sum_{i=1}^m g_i(\mathbf{x})u_i.\end{aligned}\quad (9)$$

Further, it is assumed that  $\nabla_{\mathbf{x}}\Phi(\mathbf{x})g_i(\mathbf{x})$  belongs to the span of  $\Phi(\mathbf{x})$ . In other words, there exists a constant matrix  $C_i$  such that  $\nabla_{\mathbf{x}}\Phi(\mathbf{x})g_i(\mathbf{x}) = C_i\Phi(\mathbf{x})$ . The previous assumption is reasonable as the authors of [29] have shown that for sufficiently large number of Koopman observables, the system governed by (1) can be equivalently modelled as a Koopman Bilinear Form (KBF) as follows:

$$\dot{\mathbf{z}} = \mathcal{K}\mathbf{z} + \sum_{i=1}^m C_i\mathbf{z}u_i, \quad (=: \psi(\mathbf{x}, \mathbf{u})), \quad (10)$$

where  $\mathbf{z} := \Phi(\mathbf{x})$ . To preserve the bilinearity, we apply Euler discretization to (10) to obtain the following discrete-time bilinear control system:

$$\mathbf{z}_{k+1} = \mathcal{K}_d\mathbf{z}_k + \sum_{i=1}^m D_i\mathbf{z}_k u_i, \quad (=: \psi_d(\mathbf{x}, \mathbf{u})), \quad (11)$$

where  $\mathcal{K}_d := (\mathcal{K}\Delta t + I)$ ,  $D_i = C_i\Delta t$  and  $\Delta t > 0$  is the sampling time period. The following assumption is needed for  $\mathbf{x}(t)$  satisfying (1) and  $\mathbf{z}(t)$  satisfying (10) to be well defined and finite for all  $t \geq 0$ .

**Assumption 4.** We assume that the vector fields  $F$  and  $\psi$  are forward complete<sup>2</sup>.

**Remark 3.** The motivation for using the Euler discretization scheme is twofold. First, the bilinear form of the model is maintained so that the model is affine in control. Second, since the bilinear model is affine in control, it can directly be used for the computation of safe inputs based on the QP formulation (CBF-QP).

### III. PROBLEM STATEMENT

The goal of the problem considered in this paper is to characterize a CBF for the unknown nonlinear system (1) and design a corresponding feedback controller that will guarantee safety. Since (1) is unknown, computing control inputs that render (1) safe using the QP formulation is practically impossible. We are given instead a dataset  $\mathcal{D}$  which consists of state-input pairs i.e.  $\mathcal{D} = \{(\mathbf{x}_k, \mathbf{u}_k) \in \mathcal{X} \times \mathcal{U} : k \in \{1, 2, \dots, N_d\}\}$  which satisfy the following assumption:

**Assumption 5.** The data points in  $\mathcal{D}$  are ergodic in  $\mathcal{X} \times \mathcal{U}$  with respect to the uniform distribution.

Next, we formulate the combined learning and control design problem:

**Problem 1.** Under Assumptions 1-5, simultaneously learn the Koopman based bilinear model for (1) and a CBF that satisfies (4) for the learned model. Further, learn a CBF for the unknown nonlinear system (1) and compute a control input  $\mathbf{u}(t)$  that will render the nonlinear system (1) safe.

<sup>2</sup>For the definition of forward completeness, the reader may refer to [30].

### IV. KOOPMAN BASED CBF

The equivalent form of the CBF-QP (7) in the lifted space  $\Phi(\mathbf{x})$  can be represented as follows:

$$\mathbf{u}_S(\mathbf{x}) := \underset{\mathbf{u} \in \mathcal{U}}{\operatorname{argmin}} \quad \|\mathbf{u} - k(\mathbf{x})\|^2 \quad (12a)$$

$$\text{s.t.} \quad \frac{\partial h(\mathbf{x})}{\partial \mathbf{z}} \nabla_{\mathbf{x}}\Phi(\mathbf{x})F(\mathbf{x}, \mathbf{u}) \geq -\alpha(h(\mathbf{x})). \quad (12b)$$

Since  $f$  and  $g$  are unknown, one cannot compute the control inputs  $\mathbf{u}_S(\mathbf{x})$  that would guarantee safety of the unknown nonlinear system (1) by solving the constrained convex QP (Quadratic program) (12). To this end, we use the learned Koopman based bilinear model (KBF) (10) and the CBF for the learned model (as described in Section V) to solve the following modified QP:

$$\mathbf{u}_S(\mathbf{z}) := \underset{\mathbf{u} \in \mathcal{U}}{\operatorname{argmin}} \quad \|\mathbf{u} - k_L(\mathbf{z})\|^2 \quad (13a)$$

$$\text{s.t.} \quad \frac{\partial h(\mathbf{x})}{\partial \mathbf{z}} \left( \mathcal{K}\mathbf{z} + \sum_{i=1}^m C_i\mathbf{z}u_i \right) \geq -\alpha(h(\mathbf{x})) \quad (13b)$$

where  $k_L(\mathbf{z})$  is the nominal feedback controller for the bilinear system (10). The control input computed by solving (13) is then fed back to the original nonlinear system (1) to guarantee safety. In this paper, we choose  $\alpha(y) = \lambda y$ , where  $\lambda > 0$ .

### V. LEARNING KOOPMAN BASED BILINEAR MODEL AND A CONTROL BARRIER FUNCTION

Consider a dataset  $\mathcal{D}$  which consists of state-input pairs i.e.,  $\mathcal{D} = \{(\mathbf{x}_k, \mathbf{u}_k) \in \mathcal{X} \times \mathcal{U} : k \in \{1, 2, \dots, N_d\}\}$ . We construct feedforward neural networks for the encoder (observables)  $\Phi : \mathbb{R}^n \rightarrow \mathbb{R}^N$ , decoder  $\zeta : \mathbb{R}^N \rightarrow \mathbb{R}^n$  (transforms the lifted state back to the original state) and the CBF  $h : \mathcal{X} \rightarrow \mathbb{R}$ . To simultaneously learn the Koopman based bilinear system (10) and a CBF, the following loss function is minimized

$$\mathcal{L}(\Phi, h) = \beta_1 \mathcal{L}_{\text{dyn}} + \beta_2 \mathcal{L}_{\text{recons}} + \beta_3 \mathcal{L}_{\text{barr}}, \quad (14)$$

where  $\beta_2$  and  $\beta_3$  are positive real constants. The individual loss functions in (14) are defined as follows:

The *dynamics loss*  $\mathcal{L}_{\text{dyn}}$  captures the error between the state  $\mathbf{z}_{k+1}$  and the state propagated from  $\mathbf{z}_k$  to  $\mathbf{z}_{k+1}$  via the Koopman based lifted bilinear system (11). In particular,

$$\mathcal{L}_{\text{dyn}} = \sum_{k=1}^{N_d-1} \|\Phi(\mathbf{x}_{k+1}) - \mathcal{K}_d\Phi(\mathbf{x}_k) - \sum_{j=1}^m [\mathbf{u}_i]_j D_j \Phi(\mathbf{x}_k)\|_2^2 \quad (15)$$

where  $\mathbf{u}_i$  denotes the  $i^{\text{th}}$  data point and  $[\mathbf{u}_i]_j$  denotes the  $j^{\text{th}}$  component of input  $\mathbf{u}_i$ . The matrices  $\mathcal{K}_d$  and  $D := [D_1, D_2, \dots, D_m]$  are updated at every epoch using the EDMD algorithm [5] as follows:

$$\begin{bmatrix} \tilde{K}_d & D_1 & \dots & D_m \end{bmatrix} = \Gamma \eta^T (\eta \eta^T)^{-1} \quad (16)$$

where the matrices  $\eta$  and  $\Gamma$  are given by

$$\begin{aligned}\eta &= \begin{bmatrix} \begin{pmatrix} 1 \\ \mathbf{u}_1 \end{pmatrix} \otimes \Phi(\mathbf{x}_1), & \dots, & \begin{pmatrix} 1 \\ \mathbf{u}_{N_d-1} \end{pmatrix} \otimes \Phi(\mathbf{x}_{N_d-1}) \end{bmatrix}, \\ \Gamma &= [\Phi(\mathbf{x}_2), \dots, \Phi(\mathbf{x}_{N_d})] \end{aligned} \quad (17)$$

where  $\otimes$  denotes the Kronecker product. Note that for a given  $\Phi$ , the matrices  $\mathcal{K}_d$  and  $D$  are updated optimally at every epoch.

The *reconstruction loss*  $\mathcal{L}_{\text{recons}}$  is used to penalize the error between the lifted state  $\mathbf{z}_k$  and the state which is obtained by applying the decoder and then the encoder to the lifted state. In particular,

$$\mathcal{L}_{\text{recons}} = \sum_{k=1}^{N_d} \|\mathbf{x}_k - \zeta(\Phi(\mathbf{x}_k))\|_2^2 \quad (18)$$

The role of *control barrier loss*  $\mathcal{L}_{\text{barr}}$  (inspired by [23]) is to penalize any violation of the conditions in (4) by the (candidate) CBF  $h(\mathbf{x})$  and is defined as follows:

$$\begin{aligned} \mathcal{L}_{\text{barr}} = & \frac{1}{N_d} \sum_{i=1}^{N_d} \max(0, -h(\mathbf{x}_i^a)) + \max(0, h(\mathbf{x}_i^b)) \\ & + \max(0, \nabla_{\mathbf{x}} h(\mathbf{x}_i) \psi(\Phi(\mathbf{x}_i), \mathbf{u}_i)) \end{aligned} \quad (19)$$

where  $\mathbf{x} \in \mathcal{X}$ ,  $\mathbf{x}_i^a \in \mathcal{X}_0$  and  $\mathbf{x}_i^b \in \mathcal{X}_d$ . The first term in (19) corresponds to the requirement that  $h(\mathbf{x}) \geq 0$  for  $\mathbf{x} \in \mathcal{X}_0$ , the second term to  $h(\mathbf{x}) < 0$  for  $\mathbf{x} \in \mathcal{X}_d$  and the last term to the condition (7b) be satisfied. However, given a finite number of points in  $\mathcal{X}_0$ ,  $\mathcal{X}_d$  and  $\mathcal{X}$ , minimizing the control barrier loss function would still not guarantee that  $h(\mathbf{x})$  is a CBF. To this end, we first introduce a finite input set  $U = \{\mathbf{u}_1^c, \mathbf{u}_2^c, \dots, \mathbf{u}_R^c\}$  where  $\mathbf{u}_i^c \in \mathcal{U}$  for all  $i \in \{1, 2, \dots, R\}$  and  $R \in \mathbb{N}_{>0}$ . Next, we introduce a first-order logic expression:

$$\begin{aligned} \mathcal{E}(\mathbf{z}, \mathbf{x}) = & \left( h(\mathbf{x}^a) < 0 \bigvee h(\mathbf{x}^b) \geq 0 \right) \bigvee \\ & \left( \bigwedge_{\mathbf{u}_i^c \in U} \nabla_{\mathbf{z}} h(\mathbf{x}) \psi(\mathbf{x}, \mathbf{u}_i^c) < \beta - \alpha(h(\mathbf{x})) \right) \end{aligned} \quad (20)$$

where  $\beta > 0$ , computation of which is detailed in Theorem 2.  $\mathcal{E}(\mathbf{z}, \mathbf{x})$  returns “true” if there exists a pair  $(\mathbf{z}, \mathbf{x})$  that violates at least one of the CBF conditions and returns “false” otherwise. We use the class of Sequential Modulo Theories (SMT) solvers [31] to generate counterexamples that satisfy the falsification constraint. These counterexamples are then added to either set  $\mathcal{X}_0$  or set  $\mathcal{X}_d$  depending on where the counterexamples lie. This process is repeated until the SMT solver is not able to generate any counterexamples that satisfy the falsification expression  $\mathcal{E}(\mathbf{z}, \mathbf{x})$  (i.e., when  $\mathcal{E}(\mathbf{z}, \mathbf{x})$  returns false). In this paper, we use the dReal algorithm as a SMT solver due to its  $\delta$ -completeness property that is defined as follows:

**Definition 2.** [31] Let  $\phi(\mathbf{x})$  be a quantifier first-order logic constraint. An algorithm is said to be  $\delta$ -complete if for any  $\phi(\mathbf{x})$ , the algorithm always returns one of the following answers correctly:  $\phi$  does not have a solution (unsatisfiable), or there is a solution  $\mathbf{x} = \mathbf{b}$  that satisfies  $\phi^\delta(\mathbf{b})$  where  $\phi^\delta(\mathbf{b})$  is a small variation of the original constraint.

**Remark 4.** Our approach is offline and we assume that we have access to a high-fidelity simulator which return the state-input data pairs in environments where  $\mathcal{X} = \mathcal{X}_0$ .

Due to the  $\delta$ -completeness property of dReal algorithm, there is a guarantee that the SMT never fails to generate any counterexample if one exists [31].

#### A. CBF for the unknown nonlinear system

In this section, we prove that the CBF computed for the Koopman based bilinear model is also valid for the original nonlinear system with unknown dynamics.

**Theorem 1.** Under Assumptions 1-5, if the SMT solver is not able to generate any counterexamples for the falsification

constraint, the proposed learning framework in Section V computes a CBF for the Koopman bilinear form (10) corresponding to the unknown nonlinear system (1).

*Proof.* Since the dReal (SMT) algorithm is  $\delta$ -complete (Definition 2), if the SMT solver is not able to generate any counterexamples, then there does not exist any  $\mathbf{z}$  such that (20) holds. Consequently, the following expression is true

$$\begin{aligned} \neg \mathcal{E}(\mathbf{z}, \mathbf{x}) = & (h(\mathbf{x}^a) \geq 0) \bigwedge (h(\mathbf{x}^b) < 0) \bigwedge \\ & \left( \bigvee_{\mathbf{u}_i^c \in U} \nabla_{\mathbf{z}} h(\mathbf{x}) \psi(\mathbf{x}, \mathbf{u}_i^c) \geq \beta - \alpha(h(\mathbf{x})) \right) \end{aligned}$$

where  $\mathbf{x} \in \mathcal{X}$ ,  $\mathbf{x}^a \in \mathcal{X}_0$  and  $\mathbf{x}^b \in \mathcal{X}_d$ . Therefore, there exists a  $s \in \{1, 2, \dots, R\}$  such that

$$\nabla_{\mathbf{z}} h(\mathbf{x}) \psi(\mathbf{x}, \mathbf{u}_s^c) \geq \beta - \alpha(h(\mathbf{x})) \geq -\alpha(h(\mathbf{x})), \quad (21)$$

where we have used the fact that  $\beta > 0$ . Consequently,

$$\sup_{\mathbf{u} \in U} \nabla_{\mathbf{z}} h(\mathbf{x}) \psi(\mathbf{x}, \mathbf{u}) \geq \nabla_{\mathbf{z}} h(\mathbf{x}) \psi(\mathbf{x}, \mathbf{u}_s^c) \geq -\alpha(h(\mathbf{x}))$$

which implies that  $h(\mathbf{x})$  is a CBF for the bilinear system (10).  $\square$

**Theorem 2.** Under Assumptions 1-5, if the SMT solver is not able to generate any counterexamples for the falsification constraint and if  $\Phi(\mathbf{x})$  is injective, then the CBF  $h(\mathbf{x})$  for the Koopman based bilinear system (10) is also a CBF for the original nonlinear system with unknown dynamics (1). In other words, the learned CBF  $h(\mathbf{x})$  satisfies

$$\frac{\partial h(\mathbf{x})}{\partial \mathbf{z}} \nabla_{\mathbf{x}} \Phi(\mathbf{x}) F(\mathbf{x}, \mathbf{u}) \geq -\alpha(h(\mathbf{x})),$$

for all  $\mathbf{x} \in \mathcal{X}$  and  $\mathbf{u} \in U$ .

*Proof.* Let  $(\mathbf{y}, \mathbf{v})$  be a training sample and  $(\mathbf{x}, \mathbf{u})$  be an arbitrary point belonging to  $\mathcal{X} \times \mathcal{U}$  (we use the notation  $\mathbf{a}$  and  $\mathbf{b}$ , i.e.,  $(\mathbf{a}, \mathbf{b}) := [\mathbf{a}^T, \mathbf{b}^T]^T$ ). Let  $\delta > 0$  be such that  $\|(\mathbf{x}, \mathbf{u}) - (\mathbf{y}, \mathbf{v})\| \leq \delta$ . Further, let

$$\mu := \max_{(\mathbf{y}, \mathbf{v}) \in \mathcal{D}} \|\nabla_{\mathbf{y}} \Phi(\mathbf{y}) F(\mathbf{y}, \mathbf{v}) - \psi(\mathbf{y}, \mathbf{v})\| \quad (22)$$

and  $M > 0$  such that  $\|\frac{\partial h(\mathbf{x})}{\partial \mathbf{z}}\| < M$ . In addition, let  $\tau := \max\{\|(\mathbf{y}, \mathbf{u})\| : (\mathbf{y}, \mathbf{u}) \in \mathcal{D}\}$ . Then, using the triangle inequality, we have

$$\begin{aligned} & \|\nabla_{\mathbf{x}} \Phi(\mathbf{x}) F(\mathbf{x}, \mathbf{u}) - \psi(\mathbf{x}, \mathbf{u})\| \\ & \leq \|\nabla_{\mathbf{x}} \Phi(\mathbf{x}) F(\mathbf{x}, \mathbf{u}) - \nabla_{\mathbf{y}} \Phi(\mathbf{y}) F(\mathbf{y}, \mathbf{v})\| + \\ & \quad \|\nabla_{\mathbf{y}} \Phi(\mathbf{y}) F(\mathbf{y}, \mathbf{v}) - \psi(\mathbf{y}, \mathbf{v})\| + \|\psi(\mathbf{y}, \mathbf{v}) - \psi(\mathbf{x}, \mathbf{u})\| \\ & \leq \|\nabla_{\mathbf{x}} \Phi(\mathbf{x}) F(\mathbf{x}, \mathbf{u}) - \nabla_{\mathbf{x}} \Phi(\mathbf{x}) F(\mathbf{y}, \mathbf{v})\| + \\ & \quad \|\nabla_{\mathbf{x}} \Phi(\mathbf{x}) F(\mathbf{y}, \mathbf{v}) - \nabla_{\mathbf{y}} \Phi(\mathbf{y}) F(\mathbf{y}, \mathbf{v})\| + \\ & \quad \|\nabla_{\mathbf{y}} \Phi(\mathbf{y}) F(\mathbf{y}, \mathbf{v}) - \psi(\mathbf{y}, \mathbf{v})\| \\ & \quad + \|\psi(\mathbf{y}, \mathbf{v}) - \psi(\mathbf{x}, \mathbf{u})\| \end{aligned} \quad (23)$$

Since  $\Phi$ ,  $\psi$  and  $F$  are Lipschitz continuous functions with Lipschitz constants  $K_\Phi$ ,  $K_\psi$  and  $K_F$ , respectively, we have

$$\begin{aligned} & \|F(\mathbf{x}, \mathbf{u}) - F(\mathbf{y}, \mathbf{v})\| \leq K_F \|(\mathbf{x}, \mathbf{u}) - (\mathbf{y}, \mathbf{v})\| \leq K_F \delta, \\ & \|\nabla_{\mathbf{x}} \Phi(\mathbf{x})\| \leq K_\Phi, \quad \|\nabla_{\mathbf{y}} \Phi(\mathbf{y})\| \leq K_\Phi, \\ & \|F(\mathbf{y}, \mathbf{v})\| \leq K_F \|(\mathbf{y}, \mathbf{v})\| \leq K_F \tau \\ & \|\psi(\mathbf{y}, \mathbf{v}) - \psi(\mathbf{x}, \mathbf{u})\| \leq K_\psi \|(\mathbf{y}, \mathbf{v}) - (\mathbf{x}, \mathbf{u})\| \leq K_\psi \delta \\ & \|\nabla_{\mathbf{y}} \Phi(\mathbf{y}) F(\mathbf{y}, \mathbf{v}) - \psi(\mathbf{y}, \mathbf{v})\| \leq \mu \end{aligned}$$

Therefore, (23) becomes

$$\begin{aligned} & \|\nabla_{\mathbf{x}}\Phi(\mathbf{x})F(\mathbf{x}, \mathbf{u}) - \psi(\mathbf{x}, \mathbf{u})\| \\ & \leq K_{\Phi}K_F\delta + 2K_{\Phi}K_F\tau + \mu + K_{\psi}\delta < \frac{\beta}{M} \end{aligned} \quad (24)$$

where  $\beta > 0$  is a sufficiently large constant. Then, choosing  $\beta$  to satisfy (24) implies that

$$\begin{aligned} & \nabla_{\mathbf{z}}h(\mathbf{x})\psi(\mathbf{x}, \mathbf{u}) - \nabla_{\mathbf{x}}h(\mathbf{x})F(\mathbf{x}, \mathbf{u}) \\ & \leq \left\| \frac{\partial h}{\partial \mathbf{z}} \right\| \|\nabla_{\mathbf{x}}\Phi(\mathbf{x})F(\mathbf{x}, \mathbf{u}) - \psi(\mathbf{x}, \mathbf{u})\| < M \frac{\beta}{M} = \beta \end{aligned} \quad (25)$$

In view of (25) and Theorem 1, we have

$$\begin{aligned} \nabla_{\mathbf{x}}h(\mathbf{x})F(\mathbf{x}, \mathbf{u}) & > \nabla_{\mathbf{z}}h(\mathbf{x})\psi(\mathbf{x}, \mathbf{u}) - \beta \\ & > -\alpha(h(\mathbf{x})). \end{aligned} \quad (26)$$

Hence,  $h(\mathbf{x})$  satisfies the Koopman based equivalent form of CBF (12) and the result follows.  $\square$

**Remark 5.** The value of  $\beta$  depends on  $K_{\Phi}$  and  $\mu$  which can be tuned and reduced. The value of  $K_{\Phi}$  can be reduced using spectral normalization of neural networks whereas  $\mu$  can be reduced by learning a better Koopman based bilinear model for the unknown system. Consequently, the value of  $\beta$  can be reduced and hence the computational burden on the SMT solver can be reduced.

**Remark 6.** Theorem 1 only provides sufficient conditions as to when the learned CBF for the bilinear system acts as a CBF for the unknown system.

### B. Spectral normalization of Koopman based observables

Consider a function  $p(\mathbf{x}, \theta)$  that is characterized by a neural network as follows:

$$p(\mathbf{x}, \theta) = W^{L+1}(\gamma(W^L(\gamma(W^{L-1}(\dots \gamma^1 \mathbf{x}) \dots)))) \quad (27)$$

where  $\theta := \{W^1, W^2, \dots, W^L\}$  constitutes the collection of weights of the neural network and  $\gamma$  is the activation function. Spectral normalization stabilizes the neural network training by constraining the Lipschitz constant of a function which is characterized by a neural network [32]. We can upper bound the Lipschitz constant of the neural network by leveraging the spectral norm of each layer  $q^l(\mathbf{x}) = \gamma(W^l \mathbf{x})$  for layer  $l$  as follows. For the linear map  $q(\mathbf{x}) = W\mathbf{x}$ , the spectral norm denoted by  $\|q\|_{\text{lip}}$  is given by  $\|q\|_{\text{lip}} = \sup_{\mathbf{x}} \sigma(\nabla_{\mathbf{x}}q(\mathbf{x})) = \sigma(W)$ . For activation functions such as Tanh and ReLU, the spectral norm is equal to 1. Therefore,

$$\|p(\mathbf{x}, \theta)\|_{\text{lip}} \leq \|q^{L+1}\|_{\text{lip}} \cdot \|\gamma\|_{\text{lip}} \dots \|q^1\|_{\text{lip}} = \prod_{i=1}^{L+1} \sigma(W^i). \quad (28)$$

**Proposition 1.** For a given  $K_{\Phi} > 0$ , if the collection of weights  $\theta$  at every hidden layer of the neural network that characterizes  $\Phi(\mathbf{x})$  are normalized as follows:

$$\bar{W} = W/\sigma(W) \cdot K_{\Phi}^{\frac{1}{L+1}}, \quad (29)$$

then the Lipschitz constant of  $\Phi(\mathbf{x})$  is upper bounded by  $K_{\Phi}$ .

*Proof.* Using  $\bar{W} = W/\sigma(W) \cdot K_{\Phi}^{\frac{1}{L+1}}$ , we have

$$\sigma(\bar{W}^i) = \sigma(W^i/\sigma(W^i) \cdot K_{\Phi}^{\frac{1}{L+1}}) = K_{\Phi}^{\frac{1}{L+1}}. \quad (30)$$

Hence, using (28), we have

$$\|\Phi(\mathbf{x})\|_{\text{lip}} \leq \prod_{i=1}^{L+1} \sigma(\bar{W}^i) = \prod_{i=1}^{L+1} K_{\Phi}^{\frac{1}{L+1}} = K_{\Phi} \quad (31)$$

Therefore, the result follows.  $\square$

**Remark 7.** Note that for a given  $K_{\Phi} > 0$ , there is always a trade off between how small  $K_{\Phi}$  can be made and how accurately one can learn the set of observables  $\Phi$  that represents the lifted state for the Koopman based model (11). Further, we use spectral normalization to ensure that the learned  $h(\mathbf{x})$  is Lipschitz continuous.

### C. Algorithm

Algorithm 1 summarizes our proposed approach to compute control inputs that would guarantee safety for the unknown nonlinear system (1).

#### Algorithm 1 Safe controller for unknown nonlinear system

---

**Input:**  $N_d, \Delta t, n, m, T, U, \mathcal{D} = \{\mathbf{x}_k, \mathbf{u}_k\}_{k=1}^{N_d}, \mathbf{x}_{\text{init}}, \mathbf{x}_{\text{goal}}$   
**Output:**  $\Phi, \zeta, \mathcal{K}_d, D_i \forall i \in \{1, \dots, m\}, U_S$

---

```

1: function LEARNING( $\mathcal{D}$ )
2:   Repeat:
3:      $\Phi, \zeta \leftarrow \text{NN}_{\theta}(\mathbf{x}, \mathbf{z})$  ▷ Encoder and decoder
4:      $h(\mathbf{x}) \leftarrow \text{NN}_{\theta}(\mathbf{x})$  ▷ Candidate CBF
5:      $[\mathcal{K}_d, \dots, D_m] \leftarrow \Gamma \eta^T (\eta \eta^T)^{-1}$  ▷ Update bilinear system
6:     Compute total loss  $\mathcal{L}(\theta)$  from Eqn. (14)
7:      $\theta \leftarrow \theta + \alpha_{\text{NN}} \nabla_{\theta} \mathcal{L}(\theta)$  ▷ Update weights
8:   until convergence
9:   return  $\Phi, \zeta, [\mathcal{K}_d, \dots, D_m], h(\mathbf{x})$ 
10: end function
11: function FALSIFICATION( $X$ )
12:   Use dReal algorithm to verify conditions (20)
13:   return satisfiability ▷ True or False
14: end function
15: function SAFE CONTROL( $\mathbf{x}$ )
16:    $\Phi, \zeta, [\mathcal{K}_d, \dots, D_m], h(\mathbf{x}) \leftarrow \text{LEARNING}(\mathcal{D})$ 
17:    $\mathbf{x} \leftarrow \mathbf{x}_{\text{init}}$ 
18:   Repeat:
19:      $\mathbf{u}(\mathbf{x}) \leftarrow \text{Eqn. (13)}, U \leftarrow \text{Append}(\mathbf{u}(\mathbf{x}))$ 
20:      $\mathbf{x}_{\text{next}} \leftarrow \text{Apply control input } \mathbf{u}(\mathbf{x}) \text{ to (Eqn. (12))}$ 
21:      $\mathbf{x} \leftarrow \mathbf{x}_{\text{next}}$ 
22:   until convergence of  $\mathbf{x} \rightarrow \mathbf{x}_{\text{goal}}$ 
23:   return  $U_S$ 
24: end function
25: function MAIN()
26:   while Satisfiable do
27:     Add counterexamples to  $\mathcal{D}$ 
28:      $\Phi, \zeta, [\mathcal{K}_d, \dots, D_m], h(\mathbf{x}) \leftarrow \text{LEARNING}(\mathcal{D})$ 
29:      $S \leftarrow \text{FALSIFICATION}(\mathcal{D})$ 
30:   end while
31:    $U_S \leftarrow \text{SAFE CONTROL}(\mathbf{x}_{\text{init}})$ 
32: end function

```

---

The function LEARNING (Lines 1 to 10 of Algorithm 1) takes the data snapshots  $\mathcal{D}$  of state-input pairs and returns the learned bilinear model and the CBF. For a given learned bilinear model, the function FALSIFICATION (Lines 11 to 13 of Algorithm 1) returns whether the falsification constraint (20) is true or false. If it returns false, the SMT solver generates a counterexample that satisfies (20). Given the current state  $\mathbf{x}$ , the function SAFE CONTROL (Lines 15 to 24 of Algorithm 1) uses the learned observables, CBF and the bilinear model to generate an input by solving the QP (13). This input is then fed back to the unknown nonlinear system (1) to get the subsequent state  $\mathbf{x}_{\text{next}}$ . This process is repeated until the desired state is reached.

### D. Limitations

Next, we briefly discuss the main limitations of our approach. First, the SMT solver is not scalable, which restricts our approach to low-dimensional systems. Second, there is no guarantee that the algorithm terminates, i.e., it is not able to generate any more counterexamples.

## VI. RESULTS

In this section, we provide numerical simulations to validate the ability of our proposed approach to compute control inputs that would keep the state of the unknown nonlinear system in a given safe set. The nominal controller  $k_L(z)$  for the learned Koopman based bilinear system (10) is computed using the MPC package `do-mpc` [33] in Python. We consider the differential drive robot with dynamics:

$$\dot{x} = r \sin(\theta), \quad \dot{y} = r \cos(\theta), \quad \dot{\theta} = (r/L)\omega \quad (32)$$

where  $r$  is the radius of the wheels,  $L$  is the distance between wheels,  $\mathbf{x} = [x, y, \theta]^T$  is the state,  $\mathbf{u} = \omega$  is the control input. We choose  $r = 0.1\text{m}$ ,  $L = 0.1\text{m}$ ,  $N = 5$ ,  $R = 10$ ,  $\beta_1 = 2$ ,  $\beta_2 = 0.05$ ,  $\beta_3 = 1$ ,  $\mathcal{X} = [-5, 5]^2 \times [-0.2, 0.2]$ ,  $\mathcal{U} = [-1, 1]$ . Further, we choose Tanh to be the activation function with learning rate set to  $10^{-3}$ . In our simulations, we have considered observable functions of the form  $\Phi(\mathbf{x}) = [\mathbf{x}^T, \phi_1(x), \dots, \phi_{N-n}(x)]^T$  which are injective. The set  $\mathcal{U}$  is generated by uniformly sampling  $M$  inputs from  $\mathcal{U}$ . The initial conditions  $\mathbf{x}_0$  are sampled from a square with center at  $(-2.5\text{m}, -2.5\text{m})$  and side equal to  $2\text{m}$ . A circular obstacle is placed with center at origin and radius equal to  $1\text{m}$ .  $\mathcal{X}_0$  denotes the set of points that lie outside this circular obstacle. Fig. 1 shows 50 different safe trajectories generated using our approach starting from a fixed  $\mathbf{x}_0$  (sampled uniformly from square  $[-3.5, -1.5] \times [-3.5, -1.5]$ ) to a goal position  $\mathbf{x}_{\text{goal}}$  which is chosen to be  $(2.5\text{m}, 2.5\text{m})$ .

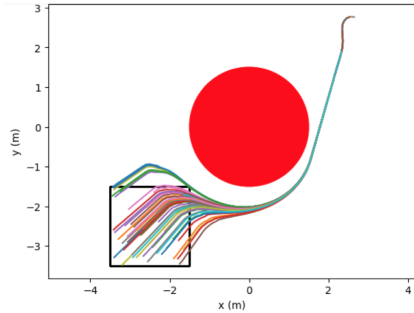


Fig. 1: 50 safe trajectories generated using our proposed approach from initial conditions sampled from a square region to the same goal position.

## VII. CONCLUSION

In this paper, we proposed a learning framework to simultaneously learn a Koopman based bilinear model for an unknown nonlinear system and a valid Control Barrier Function for the learned model. We proved that the CBF for the bilinear model also acts as a CBF for the unknown nonlinear system. Through numerical simulations, we verified our proposed approach on a differential robot for a collision avoidance problem. In our future work, we will consider extending our approach to CBFs to systems with (unknown) dynamics of higher relative degree.

## REFERENCES

- [1] T. Chen and J. Shan, "Koopman-operator-based attitude dynamics and control on  $so(3)$ ," *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 11, pp. 2112–2126, 2020.
- [2] V. Zinage and E. Bakolas, "Koopman operator based modeling and control of rigid body motion represented by dual quaternions," in *American Control Conference (ACC)*, IEEE, 2021, pp. 2992–2997.
- [3] G. Mamakoukas, M. Castano, X. Tan, and T. Murphey, "Local Koopman operators for data-driven control of robotic systems," in *Robotics: Science and Systems*, 2019.
- [4] C. W. Rowley, "Low-order models for control of fluids: Balanced models and the Koopman operator," in *Advances in Computation, Modeling and Control of Transitional and Turbulent Flows*. World Scientific, 2016, pp. 60–67.
- [5] J. H. Tu, "Dynamic mode decomposition: Theory and applications," Ph.D. dissertation, Princeton University, 2013.
- [6] V. Zinage and E. Bakolas, "Far-field minimum-fuel spacecraft rendezvous using Koopman operator and  $\ell_2/\ell_1$  optimization," in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 2992–2997.
- [7] V. Zinage and E. Bakolas, "Koopman operator based modeling for quadrotor control on  $SE(3)$ ," *IEEE Control Systems Letters*, vol. 6, pp. 752–757, 2021.
- [8] B. Lusch, J. N. Kutz, and S. L. Brunton, "Deep learning for universal linear embeddings of nonlinear dynamics," *Nature communications*, vol. 9, no. 1, pp. 1–10, 2018.
- [9] E. Kaiser, J. N. Kutz, and S. L. Brunton, "Data-driven discovery of Koopman eigenfunctions for control," *Machine Learning: Science and Technology*, vol. 2, no. 3, p. 035023, 2021.
- [10] I. Abraham and T. D. Murphey, "Active learning of dynamics for data-driven control using Koopman operators," *IEEE Transactions on Robotics*, vol. 35, no. 5, pp. 1071–1083, 2019.
- [11] G. Mamakoukas, M. L. Castano, X. Tan, and T. D. Murphey, "Derivative-based Koopman operators for real-time control of robotic systems," *IEEE Trans. on Robot.*, vol. 37, no. 6, pp. 2173–2192, 2021.
- [12] L. Lafort, "Proving the correctness of multiprocess programs," *IEEE Transactions on Software Engineering*, no. 2, pp. 125–143, 1977.
- [13] A. Robey, H. Hu, L. Lindemann, H. Zhang, D. V. Dimarogonas, S. Tu, and N. Matni, "Learning control barrier functions from expert demonstrations," in *IEEE Conference on Decision and Control*, 2020, pp. 3717–3724.
- [14] M. Srinivasan, A. Dabholkar, S. Coogan, and P. A. Vela, "Synthesis of control barrier functions using a supervised machine learning approach," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 7139–7145.
- [15] C. Folkestad, Y. Chen, A. D. Ames, and J. W. Burdick, "Data-driven safety-critical control: synthesizing control barrier functions with Koopman operators," *IEEE Control Systems Letters*, vol. 5, no. 6, pp. 2012–2017, 2020.
- [16] R. Cheng, G. Orosz, R. M. Murray, and J. W. Burdick, "End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 3387–3395.
- [17] L. Wang, E. A. Theodorou, and M. Egerstedt, "Safe learning of quadrotor dynamics using barrier certificates," in *International Conference on Robotics and Automation*. IEEE, 2018, pp. 2460–2465.
- [18] A. Taylor, A. Singletary, Y. Yue, and A. Ames, "Learning for safety-critical control with control barrier functions," in *Learning for Dynamics and Control*. PMLR, 2020, pp. 708–717.
- [19] A. J. Taylor, A. Singletary, Y. Yue, and A. D. Ames, "A control barrier perspective on episodic learning via projection-to-state safety," *IEEE Control Systems Letters*, vol. 5, no. 3, pp. 1019–1024, 2020.
- [20] V. Dhiman, M. J. Khojasteh, M. Franceschetti, and N. Atanasov, "Control barriers in bayesian learning of system dynamics," *IEEE Transactions on Automatic Control*, 2021.
- [21] K. Long, V. Dhiman, M. Leok, J. Cortés, and N. Atanasov, "Safe control synthesis with uncertain dynamics and constraints," *arXiv preprint arXiv:2202.09557*, 2022.
- [22] P. Jagtap, G. J. Pappas, and M. Zamani, "Control barrier functions for unknown nonlinear systems using Gaussian processes," in *IEEE Conference on Decision and Control*. IEEE, 2020, pp. 3699–3704.
- [23] Y.-C. Chang, N. Roohi, and S. Gao, "Neural Lyapunov control," *Advances in neural information processing systems*, vol. 32, 2019.
- [24] A. Abate, D. Ahmed, M. Giacobbe, and A. Peruffo, "Formal synthesis of Lyapunov neural networks," *IEEE Control Systems Letters*, vol. 5, no. 3, pp. 773–778, 2020.
- [25] V. Zinage and E. Bakolas, "Neural koopman lyapunov control," *Neurocomputing*, 2023.
- [26] J. Heinonen, *Lectures on Lipschitz analysis*. University of Jyväskylä, 2005, no. 100.
- [27] H. Khalil, *Nonlinear Systems*. Prentice-Hall, Upper Saddle River, NJ, 1996, vol. 3.
- [28] V. Dhiman, M. J. Khojasteh, M. Franceschetti, and N. Atanasov, "Control barriers in bayesian learning of system dynamics," *IEEE Transactions on Automatic Control*, 2021.
- [29] D. Bruder, X. Fu, and R. Vasudevan, "Advantages of bilinear Koopman realizations for the modeling and control of systems with unknown dynamics," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4369–4376, 2021.
- [30] S. Sastry, *Nonlinear systems: analysis, stability, and control*. Springer Science & Business Media, 2013, vol. 10.
- [31] S. Gao, J. Avigad, and E. M. Clarke, " $\delta$ -complete decision procedures for satisfiability over the reals," in *International Joint Conference on Automated Reasoning*. Springer, 2012, pp. 286–300.
- [32] P. L. Bartlett, D. J. Foster, and M. J. Telgarsky, "Spectrally-normalized margin bounds for neural networks," *Advances in neural information processing systems*, vol. 30, 2017.
- [33] S. Lucia, A. Tăulea-Codrean, C. Schoppmeyer, and S. Engell, "Rapid development of modular and sustainable nonlinear model predictive control solutions," *Control Engineering Practice*, vol. 60, pp. 51–62, 2017.