Local Koopman Operators for Data-Driven Control of Robotic Systems

Giorgos Mamakoukas*, Maria Castaño†, Xiaobo Tan†, and Todd Murphey*
*Department of Mechanical Engineering, Northwestern University, Evanston, Illinois 60208, USA
†Department of Electrical and Computer Engineering, Michigan State University, East Lansing, Michigan 48824, USA
giorgosmamakoukas@u.northwestern.edu

***Abstract*—This paper presents a data-driven methodology for linear embedding of nonlinear systems. Utilizing structural knowledge of general nonlinear dynamics, the authors exploit the Koopman operator to develop a systematic, data-driven approach for constructing a linear representation in terms of higher order derivatives of the underlying nonlinear dynamics. With the linear representation, the nonlinear system is then controlled with an LQR feedback policy, the gains of which need to be calculated only once. As a result, the approach enables fast control synthesis. We demonstrate the efficacy of the approach with simulations and experimental results on the modeling and control of a tail-actuated robotic fish and show that the proposed policy is comparable to backstepping control. To the best of our knowledge, this is the first experimental validation of Koopman-based LQR control.**

I. Introduction

Optimal control theory has reached a level of maturity such that there are a number of available schemes suitable for problems with known dynamics. Examples of such include linear quadratic regulator (LQR) [1], linear model predictive control (LMPC) [2], nonlinear model predictive control (NMPC) [3], feedback linearization [4], differential dynamic programming (DDP) [5], sequential action control (SAC) [6] and variants of the above [7]–[9]. The plethora of available techniques allows one to compute satisfactory solutions for nonlinear and high-dimensional systems. At the same time, it is often imperative that control solutions be calculated in real time. Unfortunately, the high nonlinearity and dimensionality present in many robotic systems are often an obstacle to the real-time implementation of nonlinear feedback control schemes [10]. Further, many robotic applications involve dynamics that are unknown, or ever-changing. These challenges call for feedback policies that can use data to adapt their models [11] and that make the necessary approximations to reach a good balance between model accuracy and computational efficiency. This is why, together with the evolution of machine learning tools, there is increasing interest in data-driven modeling and control approaches that can run in real-time.

In light of these challenges, the Koopman operator has recently drawn attention in the robotics community, as it can help address both the difficulty with nonlinearity, as well as the need to incorporate data in the model [12], [13]. Specifically, the Koopman operator propagates a nonlinear system in a linear manner without loss of accuracy by evolving functions of the states, termed observables [14]. The linear representation allows one to control the nonlinear system using tools from linear optimal control [15], [16], which is typically much easier and faster to implement than nonlinear methods. As a result, it enables online feedback for high-dimensional nonlinear systems. Interestingly enough, beyond the reduction in feedback complexity, controlling the linear representation instead of the original nonlinear system can even lead to better performance [17].

The Koopman operator can be readily combined with machine learning tools to help learn unknown dynamics from data [18]–[24]. With regard to robotic tasks involving fluid environments, uncertain terrains or other complicated dynamics such as those of bipedal walking or running, the ability to use data to learn or adapt the model is significant. As a result, the Koopman operator is a promising framework for data-driven system identification. More importantly, however, and as we detail later in Section II, the Koopman operator framework differs from standard system identification schemes in that it places the learning task in the context of seeking linear transformations of the states, which is useful for control purposes [25]–[27].

A downside of the Koopman operator, however, is that, unless a finite-dimensional invariant subspace exists [17], it is infinite-dimensional and presents practical challenges in modeling and control. For this reason, recent studies try to obtain a finite-dimensional approximation to the Koopman operator that describes the dynamics with high fidelity [13], [16]. In this trade-off between the dimensionality and the modeling error of the linear representation, the challenge becomes finding the minimum number of basis functions for the desired accuracy [24]. Choosing observable functions that best approximate the Koopman operator, however, remains an open research question. To the best of our knowledge, there has not been a systematic way of choosing the Koopman basis functions for general nonlinear systems. Rather, most efforts have relied on trial-and-error [28] and machine learning tools [24], or are system-specific [12].

In this work, we introduce a way of constructing the basis functions for the Koopman operator using higher-order derivatives of general, but known, nonlinear dynamics, where the values of the linear coefficients may be unknown. Using a data-driven, least-squares technique with a closed-form solution, we obtain the coefficients for the linear transformation, based upon which an LQR policy is found. In particular, the LQR

gains need to be computed only once and the actual feedback control value is computed online with negligible cost. As a result, our approach differs from other data-driven efforts that require more intensive online calculations of the control [28]. We validate our approach with simulation and experimental results using a tail-actuated robotic fish and compare our method to backstepping control, a sophisticated and well-studied feedback scheme [29]–[31].

The organization of the paper is as follows. Section II reviews the Koopman operator and explains how it is used in the present study for data-driven control. Section III describes the control synthesis approach that uses LQR feedback. Section IV illustrates the approach and demonstrates its performance using the system of a tail-actuated robotic fish. Section V discusses the findings of this paper, as well as ideas for further expanding this work.

II. Koopman Operator

This section reviews the Koopman operator, methods to obtain a finite approximation of the operator, and explains its relevance to system identification and optimal control.

The Koopman operator $\mathcal{K}$ is an infinite-dimensional *linear* operator that evolves functions of the state $s \in \mathbb{R}^n$ (i.e., $\Psi(s)$, commonly referred to as observables) of a dynamical system. That is,

$$\frac{d}{dt}\Psi(s) = \mathcal{K}\Psi(s) \quad \text{and} \quad \Psi(s_{k+1}) = \mathcal{K}_d\Psi(s_k), \tag{1}$$

for continuous-time and discrete-time systems, respectively. In other words, it allows one to evolve the nonlinear dynamics in a linear setting without loss of accuracy. Contrary to dynamics that are linearized around a fixed point and become inaccurate away from the linearization point, the Koopman operator evolves a nonlinear system with full fidelity throughout the state space.

Expressing nonlinear systems in a linear manner is a desirable property for many reasons. For example, Koopman eigenfunctions reveal state partitions along which the nonlinear dynamics evolve linearly. The ability to obtain geometric properties of nonlinear systems using the Koopman eigenvalues has drawn the attention of the scientific community. Work in [32], for example, investigates the global stability of a system using the eigenfunctions of the Koopman operator, whereas work in [33] extends the local linearization around a stationary point to the whole basin of attraction.

In addition to studying the behavior of complex systems, the Koopman framework enables the use of feedback that is as simple as linear optimal control, while capturing the original nonlinear dynamics. The ability to control complex systems with linear feedback is rather promising for robotic applications that remain challenging with nonlinear schemes, such as underwater locomotion. At the same time, the infinite-dimensional nature of the Koopman operator renders any practical use prohibitive.

A. Koopman Invariant Subspaces

There exist nonlinear systems that admit a finite-dimensional linear Koopman representation. Work in [17] shows that, for certain systems, there exist Koopman invariant subspaces that lead to finite-dimensional linear representations of nonlinear systems. The authors also demonstrate that the LQR control based on the linear representation could outperform LQR control calculated based on the original, nonlinear dynamics [34]. Unfortunately, Koopman invariant subspaces have only been found for a limited class of polynomial systems. Even more, there is no finite-dimensional Koopman invariant subspace for systems with multiple fixed points; the representation of the Koopman operator has no closure [17].

Recent studies have focused on approximating the infinite dimensional operator $\mathcal{K}$ with a finite representation $\tilde{\mathcal{K}} \in \mathbb{R}^{w\times w}$ that captures the original nonlinear dynamics with acceptable accuracy [13], [16]. These efforts have largely benefited from advances in machine learning, which make it possible to use data-driven regression schemes to obtain a finite-dimensional approximation to the Koopman operator. In this paper, we adopt the least-squares method shown in [13], which we detail next.

B. Data-driven Finite-dimensional Approximation to Koopman Operators

In the absence of a finite-dimensional Koopman invariant subspace, a linear propagation of the states will induce errors. The challenge is then to obtain an approximation to the Koopman operator that will linearly evolve the nonlinear system with tolerable error.

To obtain an approximation to the Koopman operator, $\tilde{\mathcal{K}}$, one chooses a set of observable functions $\Psi(s) = [\psi_1(s), \psi_2(s), \dots, \psi_w(s)] \in \mathbb{R}^w$ (which can include the states themselves) and uses data to solve a least-squares minimization problem. To allow for the effect of actuation, (1) is modified such that the observables include control terms as well [16], [28]. For the discrete-time case, this minimization takes the form

$$\tilde{\mathcal{K}}_d^* = \underset{\tilde{\mathcal{K}}_d}{\operatorname{argmin}} \sum_{k=0}^{P-1} \frac{1}{2}\|\Psi(s_{k+1}, u_{k+1}) - \tilde{\mathcal{K}}_d\Psi(s_k, u_k)\|^2, \tag{2}$$

where P is the number of measurements. Each measurement is a set of an initial state s_k, final state s_{k+1}, and the actuation applied at the same instants, u_k and u_{k+1}, respectively.

The above expression has a closed-form solution, given by

$$\tilde{\mathcal{K}}_d^* = \mathcal{A}\mathcal{G}^\dagger, \tag{3}$$

where

$$\begin{aligned} \mathcal{A} &= \frac{1}{P}\sum_{k=0}^{P-1} \Psi(s_{k+1}, u_{k+1})\Psi(s_k, u_k)^T \\ \mathcal{G} &= \frac{1}{P}\sum_{k=0}^{P-1} \Psi(s_k, u_k)\Psi(s_k, u_k)^T \end{aligned} \tag{4}$$

and $\dagger$ is the Moore-Penrose pseudoinverse. For a derivation of (3), the reader can refer to the Appendix. Note that the

time spacing δt between measurements s_k and s_{k+1} must be consistent for all P training measurements. Last, one can switch between the continuous-time and discrete-time operators via $\mathcal{K} = \log(\mathcal{K}_d)/t_s$, where t_s is the time between measurements s_k and s_{k+1}.

We should note that the data-driven approximation of the Koopman operator is not inherently different from other system identification techniques. The Koopman operator can be approximated using any of the standard regression techniques, such as ridge or lasso regression [35], [36]. However, the Koopman operator places the system identification task in a meaningful context. Contrary to system identification tools that may try to estimate unknown parameters or, more generally, the nonlinear dynamics of a system [23], [37], searching for a data-driven Koopman operator translates to searching for a linear representation of the nonlinear system.

C. Synthesis of Basis Functions

Here, we motivate the use of higher-order derivatives of known nonlinear dynamics to populate the observables. The proposed method is a data-driven way of constructing the observables in order to approximate a Koopman invariant subspace with a finite number of functions. We should note that the method is not meant to contribute to system identification of completely unknown dynamics, but rather to capturing with minimal error the evolution of an existing nonlinear model using a linear representation for the purposes of computational efficiency and control performance. As such, it does require that a model of the dynamics already exists, but not requiring that the linear coefficients of the terms are known. When one is not available, system identification tools can be used, such as in [23], to characterize the underlying system.

This method is inspired by work in [17] and [34]. The former study identifies Koopman invariant subspaces for a very limited class of nonlinear systems (whose dynamics have a specific polynomial structure). Their proposed methodology of populating the Koopman observable functions is using the Carleman linearization approach, appropriate for the types of systems they consider. Despite commenting on the challenge of non-closure, their approach is specific to polynomial systems. Further, their suggested approach is to identify eigenfunctions from data and use those for control, which they illustrate in [34].

For the systems shown to admit a finite-dimensional Koopman invariant subspace, it is straightforward to show that the terms in the observable function $\Psi(s)$ capture all higher-order derivatives of the original states. This is precisely the reason why the linear representation matches the nonlinear dynamics with no error. In cases where an invariant subspace has not been found, there is no closure of the higher-order derivatives. However, this way of reasoning allows one to infer information about the priority of certain functions over others in populating $\Psi(s_k)$. That is, the evolution of a nonlinear equation $\dot{x}(t) = g(t)$ can be approximated with a Taylor series as

$$\begin{aligned} g(t) &\approx g(t_0) + g'(t)\Big|_{t=t_0}\delta t + \frac{1}{2}g''(t)\Big|_{t=t_0}\delta t^2 + \dots \\ &\quad + g^{(n)}(t)\Big|_{t=t_0}\frac{\delta t^n}{n!} \\ &= \begin{bmatrix} 1 & \delta t & \frac{\delta t^2}{2} & \dots & \frac{\delta t^n}{n!} \end{bmatrix} \begin{bmatrix} g(t_0) \\ g'(t_0) \\ g''(t_0) \\ \vdots \\ g^{(n)}(t_0) \end{bmatrix}, \end{aligned} \tag{5}$$

where $\delta t = t - t_0$.

For a fixed t, (5) can be written in the form of (1), where the derivatives of the function $g(t)$ are equivalent to the observables $\Psi(s_k)$. That is,

$$\underbrace{\begin{bmatrix} g(t) \\ g'(t) \\ g''(t) \\ \vdots \\ g^{(n)}(t) \end{bmatrix}}_{\Psi(s_{k+1})} \approx \underbrace{\begin{bmatrix} 1 & \delta t & \frac{\delta t^2}{2} & \dots & \frac{\delta t^n}{n!} \\ 0 & 1 & \delta t & \dots & \frac{\delta t^{n-1}}{(n-1)!} \\ 0 & 0 & 1 & \dots & \frac{\delta t^{n-2}}{(n-2)!} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix}}_{\tilde{\mathcal{K}}_d} \underbrace{\begin{bmatrix} g(t_0) \\ g'(t_0) \\ g''(t_0) \\ \vdots \\ g^{(n)}(t_0) \end{bmatrix}}_{\Psi(s_k))}. \tag{6}$$

For t close to t_0, it suffices in (5) to just use the first few derivatives, where each additional derivative has a decreasing effect on the update of the function $g(t)$ considered. Note that, in (6), all derivatives of $g(t)$ are assumed to be different functions. Further, the highest derivative is not propagated at all with this representation.

We argue that populating the observables $\Psi(s_k)$ with the next higher-order derivatives instead of randomly choosing basis functions generates, locally in time, the most accurate linear representation of the nonlinear dynamics. However, due to the fact that there is no closure of the higher-order derivatives and the series will have to be truncated at some point, the analytical expression of (6) would lead to a very inaccurate Koopman operator, as is commented in [17]. For this reason, we use data-driven techniques to approximate $\tilde{\mathcal{K}}_d$, as shown in Section II. B, even when a model is known, to propagate, more accurately than an analytical model of the form in (6), the last derivatives in terms of existing terms in the observables.

In other words, linearly representing nonlinear dynamics without error requires the existence of Koopman invariant subspaces. The latter are formed by populating the observable functions with higher-order derivatives and provided that there is a finite-dimensional closure. Even when no closure exists, populating the basis functions with higher-order derivatives creates increasingly better approximations to the invariant subspaces. Data-driven methods can then be used to improve the linear approximation of the nonlinear dynamics.

III. LQR ON KOOPMAN OPERATOR

Consider a linear system with states $s \in \mathbb{R}^n$, control $u \in \mathbb{R}^m$, dynamics given by

$$\frac{d}{dt}s = As + Bu, \tag{7}$$

and a performance objective

$$J = \int_0^\infty (s - s_{des})^T Q(s - s_{des}) + u^T R u dt, \tag{8}$$

where $Q \succeq 0 \in \mathbb{R}^{n\times n}$ and $R \succ 0 \in \mathbb{R}^{m\times m}$ are weights on the deviation from the desired states and the applied control, respectively. For linear systems of the form in (7), the linear quadratic regulator (LQR) controller calculates the minimizer to (8) in one iteration [1]. The control solution has the form of a state feedback law of the form

$$u = -K_{LQR}(s - s_{des}), \tag{9}$$

where $K_{LQR} \in \mathbb{R}^{m\times n}$, the LQR gains, need to be calculated only once for each minimization task defined by (8). Next, we show how we modify the LQ optimization problem to control the Koopman representation of a nonlinear system.

Consider an approximate Koopman operator $\tilde{\mathcal{K}}$, such that

$$\frac{d}{dt}\Psi(s,u) \approx \tilde{\mathcal{K}}\Psi(s,u). \tag{10}$$

Let $\Psi(s,u) = [\Psi_s(s), \Psi_{s,u}(s,u)]^T$, where $\Psi_s(s) \in \mathbb{R}^{w_s}$ are the functions that depend only on the states, and $\Psi_{s,u}(s,u) \in \mathbb{R}^{w_u}$ are those that depend on the input as well, where $w = w_s + w_u$. This notation allows us to re-write (10) as

$$\frac{d}{dt}\begin{bmatrix}\Psi_s(s)\\ \Psi_{s,u}(s,u)\end{bmatrix} \approx \begin{bmatrix}\tilde{\mathcal{K}}_s & \tilde{\mathcal{K}}_{s,u}\\ \tilde{\mathcal{K}}_{u,s} & \tilde{\mathcal{K}}_{u,u}\end{bmatrix}\begin{bmatrix}\Psi_s(s)\\ \Psi_{s,u}(s,u)\end{bmatrix}, \tag{11}$$

where $\tilde{\mathcal{K}}_s \in \mathbb{R}^{w_s\times w_s}$ and $\tilde{\mathcal{K}}_{s,u} \in \mathbb{R}^{w_s\times w_u}$ are sub-matrices of $\tilde{\mathcal{K}}$ that describe the dynamics of the functions $\Psi_s(s)$ that depend only on the states. Note that the dynamical equation (10) has been modified from (1) to allow for control inputs [16], [28].

In order to obtain a state- and control-affine form, we can linearize the Koopman representation with respect to $\Psi_s(s)$ and u. Note that in the Koopman representation, the states are expanded from s to Ψ_s (which can include the states s of the original system, too). We can then write

$$\begin{aligned}\frac{d}{dt}\Psi_s(s) \approx\ & \frac{\partial}{\partial \Psi_s(s)}(\tilde{\mathcal{K}}_s\Psi_s(s) + \tilde{\mathcal{K}}_{s,u}\Psi_{s,u}(s,u))\cdot\Psi_s(s)\\ &+ \frac{\partial}{\partial u}(\tilde{\mathcal{K}}_s\Psi_s(s) + \tilde{\mathcal{K}}_{s,u}\Psi_{s,u}(s,u))\cdot u\\ =\ & (\tilde{\mathcal{K}}_s + \tilde{\mathcal{K}}_{s,u}\frac{\partial}{\partial\Psi_s(s)}\Psi_{s,u}(s,u))\cdot\Psi_s(s)\\ &+ (\tilde{\mathcal{K}}_{s,u}\frac{\partial}{\partial u}\Psi_{s,u}(s,u))\cdot u. \end{aligned}\tag{12}$$

This form is equivalent to the linearized dynamics of the original system with $(\tilde{\mathcal{K}}_s + \tilde{\mathcal{K}}_{s,u}\frac{\partial}{\partial\Psi_s(s)}\Psi_{s,u}(s,u)) \equiv A(s,u)$ and $(\tilde{\mathcal{K}}_{s,u}\frac{\partial}{\partial u}\Psi_{s,u}(s,u)) \equiv B(s)$ for the purposes of designing a linear controller. This form allows one to employ linear control policies, after evaluating the terms $A(s,u)$ and $B(s)$.

For the purposes of speed, we wish to avoid constantly re-evaluating the expression in (12) when calculating the feedback control input. To do that, we choose the basis functions such that it is not required to re-evaluate A and B. That is, we choose $\Psi_{s,u}(s,u) = u$ and (12) becomes

$$\frac{d}{dt}\Psi_s(s) \approx \tilde{\mathcal{K}}_s\Psi_s(s) + \tilde{\mathcal{K}}_{s,u}\cdot u,$$

where $\tilde{\mathcal{K}}_s$ and $\tilde{\mathcal{K}}_{s,u}$ are fixed.[1] If one wishes to update the Koopman operator online, these matrices would vary in response to how incoming state measurements change the solution to (3).

Then, we define a similar optimization problem to (8)

$$\begin{aligned}J_{\tilde{\mathcal{K}}} = \int_0^\infty & (\Psi_s(s) - \Psi_s(s_{des}))^T Q_{\tilde{\mathcal{K}}}(\Psi_s(s) - \Psi_s(s_{des}))\\ &+ u^T R u dt,\end{aligned}\tag{13}$$

where $Q_{\tilde{\mathcal{K}}} \succeq 0 \in \mathbb{R}^{w_s\times w_s}$ penalizes the deviation from the desired state of the observable functions $\Psi(s_{des})$. We set

$$Q_{\tilde{\mathcal{K}}} = \begin{bmatrix}Q & 0\\ 0 & 0\end{bmatrix},$$

so that a meaningful comparison can be made with regards to the original nonlinear system and the associated objective shown in (8). The LQR feedback law for (13) becomes

$$u = -K_{LQR}(\Psi(s) - \Psi(s_{des})). \tag{14}$$

Note that the control solution only updates based on the functions $\Psi_s(s)$, thereby significantly reducing the computational time compared to other feedback schemes that forward-predict the evolution of the system, compute an optimal response, and then perform a line search over the entire time horizon to decide the control solution.

To validate our proposed method for the synthesis of the basis functions, we implement the Koopman-based LQR policy described in this section using a tail-actuated robotic fish, shown in Fig. 1.

IV. RESULTS

The states of the robotic fish are $s = [x, y, \psi, v_x, v_y, \omega]^T$, where x, y are the world-frame coordinates, ψ is the orientation, v_x and v_y are the body-frame linear velocities (surge and sway, respectively), and ω is the body-frame angular velocity. We use α to indicate the angle of the tail. The tail is actuated with $\alpha(t) = \alpha_o + \alpha_a \sin(\omega_a t)$, where $\alpha_a, \alpha_o, \omega_a$ are the amplitude, bias, and frequency of the tail beat. The ranges of the bias and the amplitude are $\alpha_0 \in [-50°, 50°]$ and $\alpha_a \in [0, 30°]$, respectively. To simplify the problem, we keep the frequency fixed at $\omega_a = 2\pi$ rad/s. These actuation constraints are applied throughout the simulations and experiments.

[1]Choosing $\Psi_{s,u}(s,u) = u$ allows us to simplify (12) at the expense, however, of less accurate approximation of the dynamics. For example, if $\cos(x)u$ appears in the dynamics, it will be approximated in the Koopman model as $c_1 u$, where $c_1 \in \mathbb{R}$.

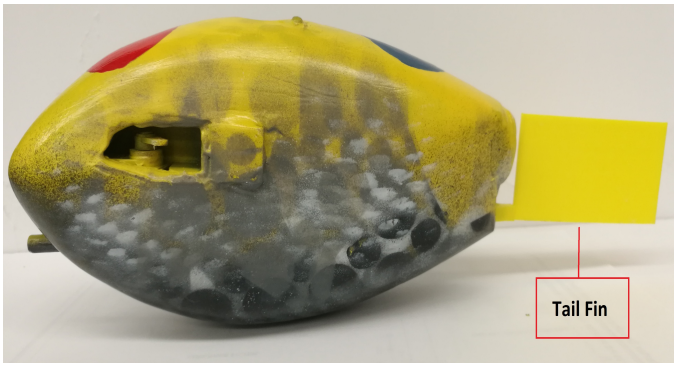


Fig. 1: Tail-actuated robotic fish used in the experiments, developed by the Smart Microsystems Lab at Michigan State University. It maneuvers in water by oscillating its tail fin.

We then describe the dynamics of the system with an average model suitable for tail-actuated systems [38] given by

$$\dot{s} = \begin{bmatrix} \dot{x} \\ \dot{z} \\ \dot{\psi} \\ \dot{v}_x \\ \dot{v}_y \\ \dot{\omega} \end{bmatrix} = f(s) \triangleq \begin{bmatrix} v_x \cos(\psi) - v_y \sin(\psi) \\ v_x \sin(\psi) + v_y \cos(\psi) \\ \omega \\ f_1(s) + K_f f_4(\alpha_0, \alpha_a, \omega_a) \\ f_2(s) + K_f f_5(\alpha_0, \alpha_a, \omega_a) \\ f_3(s) + K_m f_6(\alpha_0, \alpha_a, \omega_a) \end{bmatrix}, \quad (15)$$

where

$$\begin{aligned}
f_1(s) = &\ \frac{m_2}{m_1} v_y \omega - \frac{c_1}{m_1} v_x \sqrt{v_x^2 + v_y^2} + \frac{c_2}{m_1} v_y \sqrt{v_x^2 + v_y^2} \arctan(\frac{v_y}{v_x}) \\
f_2(s) = &\ -\frac{m_1}{m_2} v_x \omega - \frac{c_1}{m_2} v_y \sqrt{v_x^2 + v_y^2} - \frac{c_2}{m_2} v_x \sqrt{v_x^2 + v_y^2} \arctan(\frac{v_y}{v_x}) \\
f_3(s) = &\ (m_1 - m_2) v_x v_y - c_4 \operatorname{sgn}(\omega) \omega^2 \\
f_4(\alpha_0, \alpha_a, \omega_a) = &\ \frac{m}{12 m_1} L^2 \omega_a^2 u_1 \\
f_5(\alpha_0, \alpha_a, \omega_a) = &\ \frac{m}{4 m_2} L^2 \omega_a^2 u_2 \\
f_6(\alpha_0, \alpha_a, \omega_a) = &\ -\frac{m}{4 J_3} L^2 c \omega_a^2 u_2 \\
u_1 = &\ \alpha_a^2 (3 - \frac{3}{2}\alpha_o^2 - \frac{3}{8}\alpha_a^2) \\
u_2 = &\ \alpha_a^2 \alpha_o
\end{aligned}$$

and $m_1 = m_b - m_{a_x}, m_2 = m_b - m_{a_y}, J_3 = J_{b_z} - J_{a_z}, c_1 = \frac{1}{2}\rho S C_D, c_2 = \frac{1}{2}\rho S C_L, c_4 = \frac{1}{J_3} K_D, c_5 = \frac{1}{2 J_3} L^2 m c$. Parameter m_b is the mass of the robotic fish, m_{a_x} and m_{a_y} are the hydrodynamic derivatives that represent the added masses of the robotic fish along the x and y directions, respectively, J_{a_z} and J_{bz} are the added inertia effect and the inertia of the body about the z-axis, respectively, m is the mass of the water displaced by the tail per unit length, ρ is the water density, L is the tail length, c is the distance from the body center to the pivot point of the actuated tail, C_D, C_L, K_D are drag force, lift, and drag moment coefficients, respectively, and K_f and K_m are scaling coefficients that are measured experimentally [39].

A. Simulation Results

In this section, we present simulation results on the data-driven LQR control on the tail-actuated robotic fish. First, we populate the observable functions with the first-order derivatives of the terms that appear in (15). For example, $\frac{d}{dt} v_y \omega = \dot{v}_y \omega + v_y \dot{\omega}$, where $\dot{v}_y$ and $\dot{\omega}$ are given by (15). In this way, we end up with the system states, the control inputs, and 52 additional scalar functions, such that $\Psi_x(s) \in \mathbb{R}^{60}$.[2]

Simulation Parameters			
Parameter	Value	Parameter	Value
m_b	0.725 kg	ρ	1000 kg/m^3
m_{ax}	–0.217 kg	S	0.03 m^2
m_{ay}	–0.7888 kg	C_D	0.97
J_{bz}	2.66×10^{-3} kg · m^2	C_L	3.9047
J_{az}	-7.93×10^{-4} kg · m^2	K_D	4.5×10^{-3}
L	0.071 m	K_f	0.7
d	0.04 m	K_m	0.45
c	0.105 m		

TABLE I: Simulation parameters for the tail-actuated fish model dynamics (15).

Next, we train an approximate Koopman operator using (3). To generate data s_k and s_{k+1}, we sample $P = 3000$ initial conditions for the states with uniform distributions given by $\mathcal{U}_{x_0}(-0.5\,\text{m},\ 0.5\,\text{m})$, $\mathcal{U}_{y_0}(-0.1\,\text{m},\ 0.1\,\text{m})$, $\mathcal{U}_{\psi_0}(-\pi/4\,\text{rad},\ \pi/4\,\text{rad})$, $\mathcal{U}_{v_x}(0,\ 0.04\,\text{m/s})$, $\mathcal{U}_{v_y}(-0.0025\,\text{m/s},\ 0.0025\,\text{m/s})$, $\mathcal{U}_{\omega}(-0.5\,\text{rad/s},\ 0.5\,\text{rad/s})$. For each sample, we apply random inputs generated from a uniform distribution given by $\mathcal{U}_{\alpha_0}(-50°,\ 50°)$ for the tail angle bias and $\mathcal{U}_{\alpha_a}(0, 30°)$ for the tail angle amplitude of oscillations. Then, for each sample of initial conditions s_k and controls u_k, we use dynamics (15) and parameters shown in Table I to propagate the states with the given control for $\delta t = 0.005$ s and obtain the final states s_{k+1}. We use the set of s_k, s_{k+1}, u_k to compute the approximate discrete Koopman operator (3). Note that the value of u_{k+1} can be arbitrary, since we are not trying to predict the evolution of the control-dependent basis functions.

Once we have trained the Koopman operator, we convert it to the continuous time via $\tilde{\mathcal{K}} = \log(\tilde{\mathcal{K}}_d)/\delta t$, extract the state- and control- linearization matrices A and B, choose the weight matrices Q and R and compute the infinite-horizon LQR gains.

Fig. 2 shows the performance of the system (15) using LQR-feedback calculated from the Koopman representation. The desired trajectory is described by $s_{des} = [0, 0, 0, 0.02, 0, 0.05 \cdot 135 \cdot \pi/9 \cdot \cos(0.05t + \pi/2)]$ that generates a figure-8 shape. The weights are $Q = \text{diag}(0, 0, 0, 10^8, 0.01, 10^8)$ and $R = \text{diag}(0.01, 0.01)$. As is seen in the figure, the system tracks the desired states successfully.

B. Experimental Results

We use the robotic fish (see Fig. 1) to collect state measurements and train the Koopman operator. In a single

[2]Using separate functions for the time derivatives of each individual term that appears in the dynamics is similar to using the time derivatives of the entire equation of a state (e.g. $\ddot{v}_y(t)$). Despite increasing the number of basis functions, we prefer the first approach because it does not require knowing the coefficients of the individual terms in advance (e.g. $\frac{m_2}{m_1}$). As a result, it can be readily used for other robotic tail-actuated fish that have a different morphology.

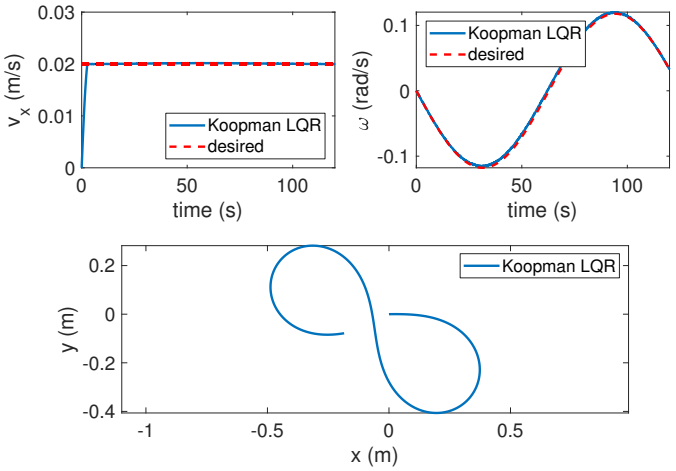


Fig. 2: LQR-controlled robotic fish in simulation. The LQR gains are generated once using the learned Koopman operator. The desired trajectory is given in terms of the forward and angular velocity. Despite using fixed LQR gains, the controlled system successfully tracks the desired trajectories that were designed to produce a figure 8 pattern.

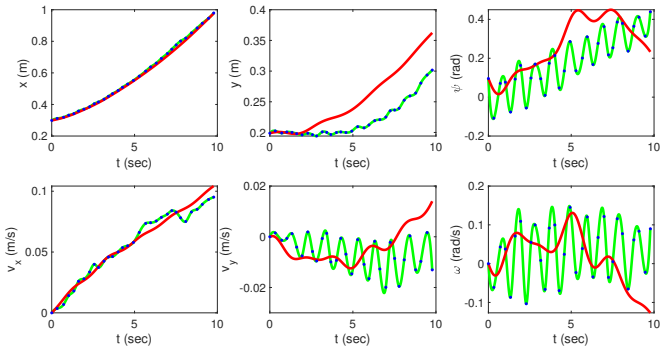
experimental run, we apply constant tail bias and amplitude for the oscillations of the tail fin. We run 22 experimental runs, with two trials for eleven different combinations of actuation parameters. The tail beat frequency is $\omega_a = 2\pi\,\mathrm{rad/s}$ for all runs. The actuation patterns used for the experimental data are shown in Table II.

During the runs, an overhead camera captures the coordinates and orientation of the robotic fish at about $4\,\mathrm{Hz}$. We then use Kalman filtering to estimate the body-fixed velocities of the robotic fish. To train the Koopman operator, we need the same time difference between pairs of measurements s_k and s_{k+1}, as we explain in Section II. For this reason, and to decrease the time between measurements to arbitrary levels (without the constraints of our sampling and filtering methods), we interpolate data at $\delta t = 0.005\,\mathrm{s}$. Last, we use the interpolated data to obtain a Koopman operator.

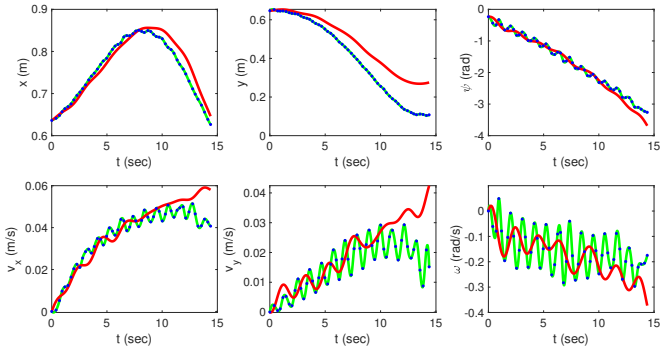
To measure how well the Koopman model captures the nonlinear dynamics of the robotic fish, we use $\tilde{\mathcal{K}}$, learned from the experimental data, to propagate the identified model continuously based on the initial states of each of the 22 experimental runs. Then, we compare the resulting simulated trajectories against the corresponding experimental ones. For the purposes of illustration, we show two such comparisons in Fig. 3. The linear Koopman model, despite not perfect, reasonably follows the experimental data for at least five seconds. We believe that the modeling is worsened by the average model (15) used to describe the longer-term behavior of the dynamics, rather than the original dynamic model. To improve the fitness, one might wish to avoid the average model and instead use Kirchoff's equations for a rigid body in fluid environment. Alternatively, one could also use a system identification algorithm, such as SINDy [23], to first obtain a model for the nonlinear dynamics of the system. We plan to

Actuation values	
Amplitude (°)	Bias (°)
15	0
20	0
20	40
20	45
20	50
25	0
25	45
25	50
30	0
30	45
30	50

TABLE II: Amplitude and bias for the tail-beat oscillations of the robotic fish used to collect experimental data to train the Koopman operator. The actuation is kept consistent through each of the 22 runs (2 trials repeated for each combination of the controls.



(a) $\alpha_0 = 0, \alpha_a = 30°$



(b) $\alpha_0 = 45°, \alpha_a = 25°$

Fig. 3: Fitness between Koopman model and experimental measurements. The green line shows data interpolated from experimental measurements (blue dots) every $dt = 0.005\,\mathrm{s}$. The red line shows the evolution of the states using the Koopman model. The actuation is constant for each of the two runs and is indicated in the caption.

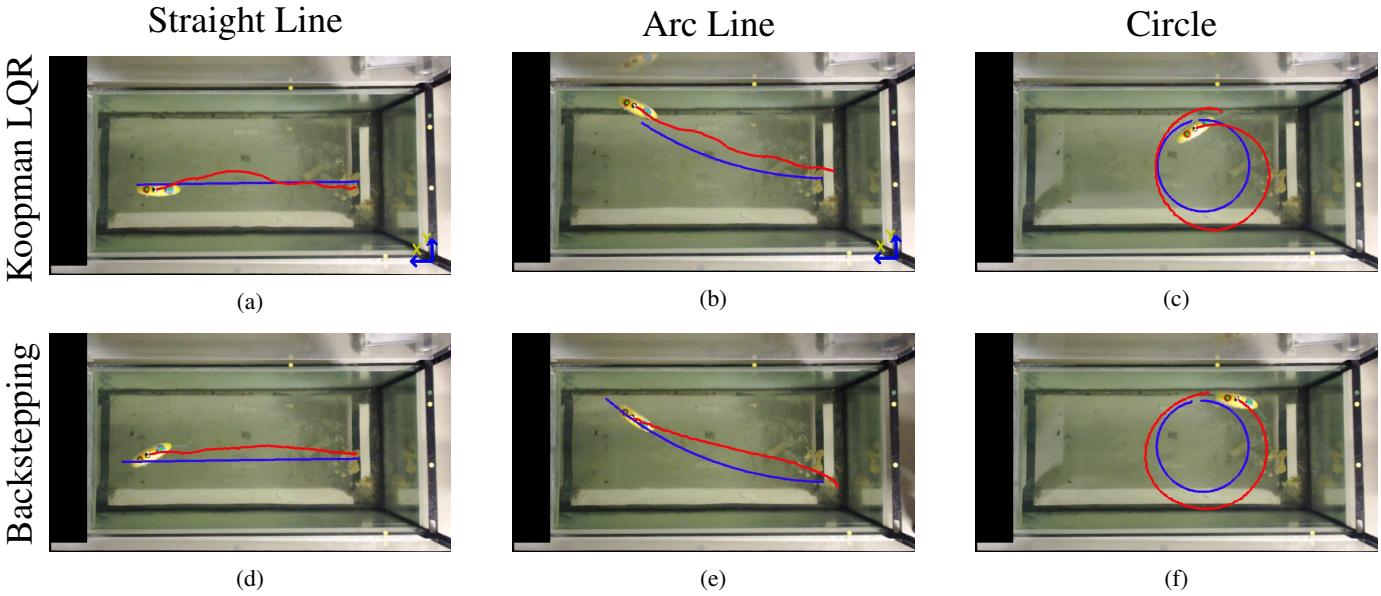


Fig. 4: Examples of trajectory tracking using Koopman LQR (top rows) and backstepping (botttom rows) on the tail-actuated robotic fish. The desired trajectory is shown in blue. Feedback is implemented at 1 Hz. A video of the experiments is available at https://youtu.be/khqU0W1Nn2g.

explore these avenues in future work.

Next, we use the Koopman operator, choose Q and R to define the minimization problem (13), and calculate the infinite-horizon LQR gains. We then run experiments using the feedback policy in (14) to track a line, an arc, and a circle. Feedback is implemented at a slow rate (1 Hz) due to speed limitations in the image processing (about 4 Hz) used for the estimation of the states.

We compare our method to backstepping feedback that uses the model in (15) to calculate control responses online. Backstepping is a widely-used method that offers a systematic way of synthesizing control for a system in strict-feedback form [30], [31]. Via Lyapunov analysis, it guarantees the stability of the system and it allows the accommodation of input constraints. Backstepping control generally requires low computational effort, often required for online applications.

As is seen in Fig. 4, the proposed data-driven policy is comparable to the more sophisticated, model-based backstepping controller. This result is promising, given that Koopman-based LQR does not require the special form of dynamics that is needed for backstepping control.

V. Discussion and Future Work

In this paper, we use the Koopman operator framework to develop data-driven linear representations of nonlinear systems, suitable for real-time feedback. We advocate for a specific way of structuring the observable functions that aims at minimizing the representation error. We control the data-driven linear model using LQR feedback that requires only one computation of the gains. We then demonstrate the efficacy of our approach with simulation and experimental results using a case study of a tail-actuated robotic fish.

While we validate our approach with an example of tracking in a fluid environment, the proposed method can be used for any system that can benefit from data-driven methods or reduction of the nonlinearity. However, underwater robotics is perhaps the most suitable application for this method, due to the inherent environmental uncertainty, the highly nonlinear dynamics, and the need for controllers that use limited computation (to preserve battery use or due to limited computational power). While this method could certainly be applied to other systems, it perhaps would not be as useful for low-dimensional systems, with known dynamics and few nonlinearities.

In the future, we will extend both the experimental and theoretical aspects of this work. Specifically, we plan on validating our method on a gliding robotic fish moving in the 3D underwater environment, currently under development. We also hope to compare our method against using traditional system identification tools to obtain the nonlinear dynamics. Further, we wish to provide formal guarantees for the optimality of our proposed structure of the basis functions, as well as describe a process that can provably capture a nonlinear system in a linear setting up to the desired order of accuracy.

VI. Acknowledgments

This material is based upon work supported by the National Science Foundation (IIS-1717951, IIS-1715714, DGE1424871). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

VII. Appendix

A. Derivation of (3)

Consider the optimization problem

$$\tilde{\mathcal{K}}_d^* = \underset{\tilde{\mathcal{K}}_d}{\operatorname{argmin}} \sum_{k=0}^{P-1} \frac{1}{2} \|\Psi(s_{k+1}, u_{k+1}) - \tilde{\mathcal{K}}_d \Psi(s_k, u_k)\|^2.$$

This expression is quadratic in $\tilde{\mathcal{K}}_d$ so we use matrix calculus to differentiate and find the optimal solution.

$$\begin{aligned}
0 =& \frac{\partial}{\partial \tilde{\mathcal{K}}_d} \sum_{k=0}^{P-1} \frac{1}{2} \|\Psi(s_{k+1}, u_{k+1}) - \tilde{\mathcal{K}}_d \Psi(s_k, u_k)\|^2 \\
=& \frac{\partial}{\partial \tilde{\mathcal{K}}_d} \sum_{k=0}^{P-1} \frac{1}{2} (\Psi(s_{k+1}, u_{k+1}) - \tilde{\mathcal{K}}_d \Psi(s_k, u_k))^T \\
& \cdot (\Psi(s_{k+1}, u_{k+1}) - \tilde{\mathcal{K}}_d \Psi(s_k, u_k)) \\
=& \frac{\partial}{\partial \tilde{\mathcal{K}}_d} \sum_{k=0}^{P-1} \frac{1}{2} (\Psi(s_{k+1}, u_{k+1})^T \Psi(s_{k+1}, u_{k+1}) \\
& - \Psi(s_{k+1}, u_{k+1})^T \tilde{\mathcal{K}}_d \Psi(s_k, u_k)) \\
& - \Psi(s_k, u_k)^T \tilde{\mathcal{K}}_d^T \Psi(s_{k+1}, u_{k+1}) \\
& + \Psi(s_k, u_k)^T \tilde{\mathcal{K}}_d^T \tilde{\mathcal{K}}_d \Psi(s_k, u_k) \\
=& \sum_{k=0}^{P-1} \frac{1}{2} \frac{\partial}{\partial \tilde{\mathcal{K}}_d} \big[- 2\Psi(s_{k+1}, u_{k+1})^T \tilde{\mathcal{K}}_d \Psi(s_k, u_k) \\
& + \Psi(s_k, u_k)^T \tilde{\mathcal{K}}_d^T \tilde{\mathcal{K}}_d \Psi(s_k, u_k) \big] \\
=& \sum_{k=0}^{P-1} \frac{1}{2} \big[- 2\Psi(s_{k+1}, u_{k+1}) \Psi(s_k, u_k)^T \\
& + 2\tilde{\mathcal{K}}_d \Psi(s_k, u_k) \Psi(s_k, u_k)^T \big] \\
=& \sum_{k=0}^{P-1} -\Psi(s_{k+1}, u_{k+1}) \Psi(s_k, u_k)^T + \tilde{\mathcal{K}}_d \Psi(s_k, u_k) \Psi(s_k, u_k)^T \\
=& - \sum_{k=0}^{P-1} \Psi(s_{k+1}, u_{k+1}) \Psi(s_k, u_k)^T \\
& + \tilde{\mathcal{K}}_d \sum_{k=0}^{P-1} \Psi(s_k, u_k) \Psi(s_k, u_k)^T.
\end{aligned}$$

Using the substitutions in (4) to simplify the above terms yields

$$0 = -\mathcal{A} + \tilde{\mathcal{K}}_d \mathcal{G} \Longleftrightarrow \tilde{\mathcal{K}}_d = \mathcal{A}\mathcal{G}^\dagger,$$

where $\dagger$ is the Moore-Penrose pseudoinverse.

References

[1] H. Kwakernaak and R. Sivan, *Linear Optimal Control Systems*. Wiley, New York, 1972.

[2] A. Bemporad, F. Borrelli, and M. Morari, "Model predictive control based on linear programming—the explicit solution," *IEEE Transactions on Automatic Control*, vol. 47, no. 12, pp. 1974–1985, 2002.

[3] F. Allgöwer and A. Zheng, *Nonlinear Model Predictive Control*. Birkhäuser, Basel, 2012.

[4] A. Isidori, *Nonlinear Control Systems*. Springer, London, 1995.

[5] D. H. Jacobson and D. Q. Mayne, *Differential Dynamic Programming*. American Elsevier, New York, 1970.

[6] A. R. Ansari and T. D. Murphey, "Sequential action control: closed-form optimal control for nonlinear and nonsmooth systems," *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1196–1214, 2016.

[7] E. Theodorou, Y. Tassa, and E. Todorov, "Stochastic differential dynamic programming," in *American Control Conference*, 2010, pp. 1125–1132.

[8] E. Todorov and W. Li, "A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems," in *American Control Conference*, 2005, pp. 300–306.

[9] G. Mamakoukas, M. A. MacIver, and T. D. Murphey, "Feedback synthesis for underactuated systems using sequential second-order needle variations," *The International Journal of Robotics Research*, vol. 37, no. 13-14, pp. 1826–1853, 2018.

[10] D. Mayne, *Nonlinear model predictive control: Challenges and opportunities*. Springer, 2000.

[11] Z.-S. Hou and Z. Wang, "From model-based control to data-driven control: Survey, classification and perspective," *Information Sciences*, vol. 235, pp. 3–35, 2013.

[12] A. Mauroy and J. Goncalves, "Linear identification of nonlinear systems: A lifting technique based on the Koopman operator," in *Conference on Decision and Control*, 2016, pp. 6500–6505.

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

[14] B. O. Koopman, "Hamiltonian systems and transformation in Hilbert space," *Proceedings of the National Academy of Sciences*, vol. 17, no. 5, pp. 315–318, 1931.

[15] M. O. Williams, M. S. Hemati, S. T. Dawson, I. G. Kevrekidis, and C. W. Rowley, "Extending data-driven Koopman analysis to actuated systems," in *IFAC Symposium on Nonlinear Control Systems*, 2016, pp. 704–709.

[16] J. L. Proctor, S. L. Brunton, and J. N. Kutz, "Generalizing Koopman theory to allow for inputs and control," *SIAM Journal on Applied Dynamical Systems*, vol. 17, no. 1, pp. 909–930, 2018.

[17] S. L. Brunton, B. W. Brunton, J. L. Proctor, and J. N. Kutz, "Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control," *PloS One*, vol. 11, no. 2, p. e0150171, 2016.

[18] I. Mezić, "On applications of the spectral theory of the Koopman operator in dynamical systems and control theory," in *Conference on Decision and Control*, 2015, pp. 7034–7041.

[19] M. Budišić, R. Mohr, and I. Mezić, "Applied Koopmanism," *Chaos*, vol. 22, no. 4, p. 047510, 2012.

[20] M. Korda and I. Mezić, "Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control," *Automatica*, vol. 93, pp. 149–160, 2018.

[21] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. N. Kutz, "On dynamic mode decomposition: Theory and applications," *Journal of Computational Dynamics*, vol. 1, no. 2, pp. 391–421, 2014.

[22] A. Mauroy and I. Mezić, "Global stability analysis using the eigenfunctions of the Koopman operator," *IEEE Transactions on Automatic Control*, vol. 61, no. 11, pp. 3356–3369, 2016.

[23] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences*, vol. 113, no. 15, pp. 3932–3937, 2016.

[24] B. Lusch, J. N. Kutz, and S. L. Brunton, "Deep learning for universal linear embeddings of nonlinear dynamics," *Nature Communications*, vol. 9, no. 1, p. 4950, 2018.

[25] B. C. Daniels and I. Nemenman, "Automated adaptive inference of phenomenological dynamical models," *Nature Communications*, vol. 6, p. 8133, 2015.

[26] I. G. Kevrekidis, C. W. Gear, J. M. Hyman, P. G. Kevrekidid, O. Runborg, and C. Theodoropoulos, "Equation-free, coarse-grained multiscale computation: Enabling microscopic simulators to perform system-level analysis," *Communications in Mathematical Sciences*, vol. 1, no. 4, pp. 715–762, 2003.

[27] Z. Hou and S. Jin, "A novel data-driven control approach for a class of discrete-time nonlinear systems," *IEEE Transactions on Control Systems Technology*, vol. 19, no. 6, pp. 1549–1558, 2011.

[28] I. Abraham, G. De La Torre, and T. D. Murphey, "Model-based control using Koopman operators," in *Robotics: Science and Systems*, vol. 13, 2017.

[29] P. V. Kokotovic, "The joy of feedback: Nonlinear and adaptive," *IEEE Control Systems*, vol. 12, no. 3, pp. 7–17, 1992.

[30] H. K. Khalil, *Nonlinear Systems*. Prentice Hall, Upper Saddle River, NJ, 2002.

[31] R. Fierro and F. L. Lewis, "Control of a nonholomic mobile robot: Backstepping kinematics into dynamics," *Journal of Robotic Systems*, vol. 14, no. 3, pp. 149–163, 1997.

[32] A. Mauroy and I. Mezić, "Global stability analysis using the eigenfunctions of the Koopman operator," *IEEE Transactions on Automatic Control*, vol. 61, no. 11, pp. 3356–3369, 2016.

[33] Y. Lan and I. Mezić, "Linearization in the large of nonlinear systems and Koopman operator spectrum," *Physica D: Nonlinear Phenomena*, vol. 242, no. 1, pp. 42–53, 2013.

[34] E. Kaiser, J. N. Kutz, and S. L. Brunton, "Data-driven discovery of Koopman eigenfunctions for control," *arXiv preprint arXiv:1707.01146*, 2017.

[35] K. P. Murphy, *Machine learning: A Probabilistic Perspective*. The MIT Press, Cambridge, MA, 2012.

[36] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*. Springer, New York, 2013.

[37] L. Ljung, *System Identification*. Prentice Hall, Upper Saddle River, NJ, 1998.

[38] J. Wang and X. Tan, "Averaging tail-actuated robotic fish dynamics through force and moment scaling," *IEEE Transactions on Robotics*, vol. 31, no. 4, pp. 906–917, 2015.

[39] M. L. Castaño and X. Tan, "Model predictive control-based path following for tail-actuated robotic fish," 2018.