Model-Based Control Using Koopman Operators

Ian Abraham, Gerardo De La Torre, and Todd D. Murphey
Department of Mechanical Engineering
Northwestern University, Evanston, Illinois 60208
Email: i-abr@u.northwestern.edu
gerardo.delatorre@northwestern.edu,
t-murphey@northwestern.edu

Abstract—This paper explores the application of Koopman operator theory to the control of robotic systems. The operator is introduced as a method to generate data-driven models that have utility for model-based control methods. We then motivate the use of the Koopman operator towards augmenting model-based control. Specifically, we illustrate how the operator can be used to obtain a linearizable data-driven model for an unknown dynamical process that is useful for model-based control synthesis. Simulated results show that with increasing complexity in the choice of the basis functions, a closed-loop controller is able to invert and stabilize a cart- and VTOL-pendulum systems. Furthermore, the specification of the basis function are shown to be of importance when generating a Koopman operator for specific robotic systems. Experimental results with the Sphero SPRK robot explore the utility of the Koopman operator in a reduced state representation setting where increased complexity in the basis function improve open- and closed-loop controller performance in various terrains, including sand.

I. INTRODUCTION

Modeling for complex dynamical systems has typically been the first step when designing, control, planning, or state-estimation algorithms. System design and specifications have been dependent on the use of high-fidelity models. However, any derivation of a dynamical model from first principles is typically a demanding task when the complexity of state interactions is high. Moreover, analytical models do not capture external disturbances. As a result, derived models, for use in model-based control settings, often have limited use or poor prediction over longer time spans. Nevertheless, a representation of the behavior of a dynamical system is central to most model-based engineering and scientific application.

Within the field of systems and control theory, model uncertainty has typically been mitigated with the use of robust and adaptive control architectures. Typically, adaptive controllers are self tuning and reactive to incoming state information while robust controllers are designed to be invariant to model uncertainty [1]–[4]. Motion planning for uncertain dynamical systems have also been extensively investigated. Generally, in this approach, uncertainty is explicitly modeled and incorporated into the decision making process [5]–[7]. However, like robust and adaptive control approaches, the need for an explicit uncertainty model often limits its utility in general settings. Machine learning, offers a much more general approach [8]–[10]. In particular, recent advances have utilized large sets

of data to perform model-based control of various dynamical systems [11]. Nonetheless, several questions about the training data, stability, convergence properties, computational complexity, and mechanical property conservation of the models are still open questions that need to be addressed.

Recently, the use of data-driven techniques to mitigate the effects of model uncertainty have sparked interest in the Koopman operator [12]. The Koopman operator is a infinite-dimensional linear operator that is able to exactly capture the behavior of nonlinear dynamical systems. In application, the Koopman operator is approximated with a finite-dimensional linear operator [13]. This approximation can be computed in a solely data-driven manner without any prior information of the dynamical system. Complex fluid flow systems have accurately been modeled using this approach [14]. Furthermore, it has been shown that the spectral properties of the approximate Koopman operator can be examined to investigate system-level behavior like ergodicity and stability [12], [15], [16]. In addition, recent work has shown its utility in human-machine systems [17]. In this paper, we investigate the utility of Koopman operator theory for control in robotic systems.

The work is motivated by the desire to generate or augment dynamical models of robotic systems through data collection. In particular, it is of interest to synthesize model-based controllers using these data-driven models. Thus, the main contribution of this paper is the application of Koopman operator theory to the control of robotic systems. The Koopman operator is shown to have a linearizable data-driven model of the dynamical system that is amenable to model-based control methods. Closed-loop and open-loop controllers are then formulated using the proposed data-driven model. Furthermore, we explore the consequences of the specific choice of basis function as well as complexity order for swing up control of a simulated cart- and vertical take-off and landing (VTOL)-pendulum systems. Last, experiments using the Koopman operator using a Sphero SPRK robot are shown. We conclude the paper with recommendations for future work.

The organization of this paper is as follows. Section II gives an overview of the Koopman operator theory and its application to data-driven approximations of dynamical systems. In addition, Sections III and IV explore the implementation of Koopman operator theory in simulation and experimentation, respectively. Conclusions are in Section V.

II. KOOPMAN OPERATOR

An overview of Koopman operator theory is given in this section. For the purposes of this paper, we focus more on the practical implementation of the theory and omit much of the theoretical presentation. However, the interested reader can find a complete treatment of the Koopman operator in [13].

To begin, consider a discrete-time dynamical system evolving as

$$x_{k+1} = F(x_k), \tag{1}$$

where $x_k \in M$ is the, possibly unobserved, state of the system and $y_k \in \mathbb{C}$. Furthermore, define an observation function

$$y_k = g(x_k), \tag{2}$$

where $g \in \mathbb{G} : M \to \mathbb{C}$ and $\mathbb{G}$ is a function space. For the purposes on this paper, we assume that $\mathbb{G}$ is the L^2 space. The Koopman operator, $\mathcal{K} : \mathbb{G} \to \mathbb{G}$, is defined as

$$[\mathcal{K}g](x) = g(F(x)). \tag{3}$$

Note that the Koopman operator maps elements in $\mathbb{G}$ to elements in $\mathbb{G}$. Therefore, it does not, as done by F, map system states to system states. Furthermore, note that (3) can be written as

$$[\mathcal{K}g](x_k) = g(F(x_k)) = g(x_{k+1}). \tag{4}$$

Therefore, the Koopman operator propagates the output of the system forward. Finally, the observable equation can be easily extended to the case where multiple observations are available, $g : M \to \mathbb{C}^K$.

The Koopman operator defined in (3) is linear when $\mathbb{G}$ is a vector space. This property holds even if the considered discrete-time dynamical system is nonlinear. However, since the Koopman operator maps $\mathbb{G}$ to elements in $\mathbb{G}$ it is infinite dimensional. Therefore, a nonlinear dynamical system given by (1) can be equivalently described by a *linear infinite dimensional operator*. From a practical standpoint, there is not much benefit from this infinite dimensional representation even if the operator could be defined for a specific system of interest. However, the Koopman operator can be approximated with a linear finite dimensional operator using data-driven approaches.

A. Approximating a Koopman Operator

In order to define an approximate Koopman operator the observation function (2) is redefined as

$$y_k = g(x_k) = \Psi(x_k), \tag{5}$$

where $\Psi(x)$ is a user-defined vector-valued function

$$\Psi(x) = [\psi_1(x), \psi_2(x), \dots, \psi_N(x)]. \tag{6}$$

Next, the relation described by (4) is now given as

$$\Psi(x_{k+1}) = \Psi(x_k)K + r(x_k). \tag{7}$$

where $K \in \mathbb{C}^{N \times N}$ and $r(x_k)$ is a residual (approximation error). Note that the matrix K advances Ψ forward one time step. Next, it is assumed that the trajectory of the system has been collected such that

$$X = [x_1, \dots, x_P] \tag{8}$$

where P is the number of recorded data points.

The matrix K can be computed in a number of ways. In this paper, we adopt the least-squares approach, described in [18], where K is determined by minimizing

$$J = \frac{1}{2} \sum_{p=1}^{P-1} |r(x_p)|^2, \tag{9}$$

$$= \frac{1}{2} \sum_{p=1}^{P-1} |\Psi(x_{p+1}) - \Psi(x_p)K|^2. \tag{10}$$

Solving the least-squares problem yields

$$K = G^\dagger A, \tag{11}$$

where $\dagger$ denotes the Moore–Penrose pseudoinverse and

$$G = \frac{1}{P} \sum_{p=1}^{P-1} \Psi(x_p)^{\mathrm{T}} \Psi(x_p), \tag{12}$$

$$A = \frac{1}{P} \sum_{p=1}^{P-1} \Psi(x_p)^{\mathrm{T}} \Psi(x_{p+1}). \tag{13}$$

Note that the computational burden of this approach grows as the dimension of Ψ increases. The approach generally yields a better approximation as the dimension of Ψ increases. Furthermore, the number of data points and their distribution across the state space will have a large effect on the computed K matrix.

The definitions of (8-13) can be generalized. The recorded data points need not come from a single trajectory nor be sequential [18]. Multiple trajectories and trajectories with missing data points can be used. The only requirement is the sum of residuals given in (9) be defined by consecutive states (x_k, x_{k+1}) spaced equally in time. Even this could be avoided by choosing another optimization to solve for K.

B. Approximating Dynamical Systems

For predicting dynamical systems, the approximation to the Koopman operator can be used to generate a data-driven model of a system by defining Ψ as

$$\Psi(x) = [x^{\mathrm{T}}, \psi_1(x), \psi_2(x), \dots, \psi_N(x)]. \tag{14}$$

Note that the state of the system, $x \in \mathbb{R}^n$, is now included in $\Psi(x)$. Thus we can write the approximate dynamical equations of the considered system as

$$x_{k+1} \approx \hat{K}^{\mathrm{T}} \Psi(x_k)^{\mathrm{T}}, \tag{15}$$

where $\hat{K} \in \mathbb{R}^{n \times N}$ is the first n columns of K. Note that equation (15) simply propagates forward the quantities of interest (e.g. system states). Furthermore, in this work, x_{k+1} is described as a linear combination of the system state, x_k, and the functions $\psi_i(x_k)$.

III. CONTROL SYNTHESIS: OPEN- AND CLOSED-LOOP CONTROLLERS

In this section we formulate open- and closed-loop model-based controllers using the Koopman operator. It is first shown that for a differentiable choice of basis function Ψ, the Koopman operator has a linearization that can be computed for model-based control methods. Given the linearizable Koopman operator, a model-based optimal control problem is formulated for open- and closed-loop controllers.

A. Koopman Operator Linearization

By choosing a Ψ that is differentiable, the Koopman operator approximation to the dynamical system can be linearized:

$$x_{k+1} \approx \hat{K}^{\mathrm{T}} \frac{\partial \Psi}{\partial x} x_k \tag{16}$$

$$\approx A(x_k) x_k. \tag{17}$$

Control inputs are readily incorporated to the definition of Ψ as an augmented state,

$$\Psi(x, u) = [x^{\mathrm{T}}, u^{\mathrm{T}}, \psi_1(x, u), \psi_2(x, u), \ldots, \psi_N(x, u)]. \tag{18}$$

This yields the approximate dynamical equations,

$$x_{k+1} \approx \hat{K}^{\mathrm{T}} \Psi(x_k, u_k)^{\mathrm{T}} \tag{19}$$

and the linearization of the approximate dynamical equations,

$$x_{k+1} \approx \hat{K}^{\mathrm{T}} \frac{\partial \Psi}{\partial x} x_k + \hat{K}^{\mathrm{T}} \frac{\partial \Psi}{\partial u} u_k \tag{20}$$

$$\approx A(x_k, u_k) x_k + B(x_k, u_k) u_k. \tag{21}$$

Note that *linearizable equations of motion of a dynamical system can be computed solely from data.*

B. Optimal Control Problem

Control synthesis for trajectory optimization is generated for mobile robot dynamics of the form

$$x_{k+1} = f(x_k, u_k), \tag{22}$$

where $x \in \mathbb{R}^n$ is the state and $u \in \mathbb{R}^m$ is the control input. For a discrete system, we can solve for a trajectory that minimizes the objective defined as

$$J = \sum_{k=0}^{N} \frac{1}{2} (x_k - \tilde{x}_k)^T \mathbf{P} (x_k - \tilde{x}_k) + \frac{1}{2} u_k^T \mathbf{R} u_k, \tag{23}$$

where $\mathbf{P} \in \mathbb{R}^{n \times n}$ and $\mathbf{R} \in \mathbb{R}^{m \times m}$ are positive definite weight matrices on state and control and $\tilde{x}_k$ is the reference trajectory at time k. Note that the accuracy of the system model (22) will largely determine the effectiveness of the synthesized optimal control.

Open-Loop: Open-loop trajectory optimization precomputes the set of trajectory and control actions that minimize the objective function (23) subject to the modeled dynamical constraints in (22). Projection-based optimization [19] is used in discrete time to generate the set of trajectory and control actions given an initial trajectory x_k and control u_k for $k \in [0, N]$. In the experiment, the projection-based optimization algorithm first generates the control actions based on the dynamical model and then at a fixed rate the command signals are sent via Bluetooth communication to the robot. Odometry data is collected only for post-processing and is not used to update the command signals.

Closed-Loop: In the simulated and the experimental work, a discrete-time version of Sequential Action Control (SAC) [20] is used with the Koopman operator to generate closed-loop optimal control calculations. However, any MPC technique can be used with the Koopman operator. Here, SAC operates by first forward simulating an open-loop trajectory for some horizon N for a control-affine dynamical system given by

$$x_{k+1} = f(x_k, u_k) = g(x_k) + h(x_k) u_k. \tag{24}$$

The sensitivity to a control injection for any given discrete time of the objective function is given as

$$\frac{dJ}{d\lambda_k} = \rho_k (f_2(k) - f_1(k)) \tag{25}$$

where

$$f_1(k) = f(x_k, u_{0,k}), \tag{26}$$
$$f_2(k) = f(x_k, u_k^\star) \tag{27}$$

are the dynamics subject to the default control $u_{0,k}$ and derived control $u_k^\star$. The co-state variable $\rho_k \in \mathbb{R}^n$ is computed by backwards simulating the following discrete equation

$$\rho_{k-1} = \frac{\partial l_k}{\partial x} + \frac{\partial f_k}{\partial x}^T \rho_k, \tag{28}$$

where $l_k = \frac{1}{2}(x_k - \tilde{x}_k)^T \mathbf{P}(x_k - \tilde{x}_k) + \frac{1}{2} u_k^T \mathbf{R} u_k$ and $f_k = f(x_k, u_{0,k})$ for some default $u_{0,k}$ subject to $\rho_N = \vec{0}$. The optimal control u_k^* is computed by first defining a secondary objective function as

$$J_u = \sum_{k=0}^{N} \frac{1}{2} (\frac{dJ}{d\lambda_k} - \alpha_d)^2 + \frac{1}{2} \|u_k^\star - u_{0,k}\|_{\mathbf{R}}^2. \tag{29}$$

The objective (29) is now convex in u_k^* and has a minimizer when

$$u_k^* = (\Lambda + R^T)^{-1} h(x_k)^T \rho_k \alpha_d + u_{0,k}, \tag{30}$$

where $\Lambda = h(x_k)^T \rho_k \rho_k^T h(x_k)$. Given the sequence of actions $u_k^\star$, it is then possible to calculate the time of control application $t_k^\star$ as

$$t_k^\star = \mathrm{argmin} \frac{dJ}{d\lambda_k}. \tag{31}$$

The control duration in discrete time is found using an outward line search [21] for a sufficient descent on the cost.

Fig. 1. Sphero SPRK Robot is shown with its clear spherical casing revealing the underlying mechanism. The internal mechanism shifts the center of mass by rolling and rotating within the spherical enclosure, causing the SPRK to roll. RGB LEDs on the top of the SPRK are utilized to track the odometry of the robot through an Xbox Kinect with OpenCV and OpenKinect libraries for image processing and motion capture. ROS [22] is used to transmit and collect data at 20Hz.

IV. Experiments Using Sphero SPRK

In this section, we describe the experimental set-up for use of the Sphero SPRK robot with model-based control algorithms that utilize a state-space model generated via the Koopman operator. In particular, we define data-driven closed- and open-loop model predictive controllers as well as motivate and explore the utility of Koopman operator for control of a robotic system.

In the experiments with the SPRK, trajectory optimization is run both in open-loop form and closed-loop feedback form. Here, the tracked states of the robot are position x, y and velocity $\dot{x}, \dot{y}$ and inputs to the robot are desired velocities u_1, u_2. The objective function parameters are defined as $\mathbf{P} = \mathrm{diag}([60, 60, 0.1, 0.1])$ and $\mathbf{R} = \mathrm{diag}([20, 20])$ and are maintained constant through both open-loop and closed-loop experiments. An additional set of experiments are done to show the use of the Koopman operator for control in a sand environment.

A. SPRK

The SPRK is a differential drive mobile robot enclosed in a spherical case. The dynamics of the SPRK are driven by the nonlinear coupling between the internal mechanism and the outer spherical encasing. In addition, proprietary underlying controllers govern how the command velocities are interpreted to low-level motors. The proprietary embedded software uses the on-board gyro-accelerometers to balance the robot upright while rolling. The caster wheels on top of the internal mechanism ensures constant contact of the lower wheels that are driven via two motors. The embedded software interfaces with heading and velocity (or $x - y$ velocity) command inputs sent via Bluetooth communication. A high fidelity model of the robot would include several internal states characterize the internal mechanism and controller. However, rather than

seeking to approximated a high dimensional model, a reduced state model was sought.

Figure 1 shows a closer look at the SPRK robot. Odometry is collected using a Xbox Kinect with OpenCV [23] image processing. More details about odometry and motion capture are stated in the caption of Fig. 1.

B. SPRK Koopman Operator

The representation of the system consists of the position of the robot (x, y), its velocity $(\dot{x}, \dot{y})$, and the commanded velocity (u_x, u_y). Odometry data from the Kinect paired with recorded velocity commands are used to generate the approximate Koopman operator. The vector-valued functions used in this experiment are polynomial basis functions given as

$$\Psi(x) = [x, y, \dot{x}, \dot{y}, u_x, u_y, 1, \psi_1, \psi_2, \ldots, \psi_M] \quad (32)$$
$$\psi_i(x) = \dot{x}^{\alpha_i} \dot{y}^{\beta_i} \quad (33)$$

where α_i, and β_i are nonnegative integers, index i tabulates all the combinations such that $\alpha_i + \beta_i \leq Q$ and $Q > 1$ defines the largest allowed polynomial degree. We ignore higher order position dependence in the operator in order to prevent any possible overfitting of position-based external disturbances. The approximated Koopman operator was computed using data captured when the robot was operating at velocity under $1 \ m/s$ for the open-loop trails.

V. Results

A. Simulation: Mechanical Energy

In this section, the equations of motion of a double pendulum are approximated with the method described in Section II. The mass of both pendulums are 1 kilogram and the lengths of both are 1 meter. The mass of the pendulums are assumed to be concentrated at their ends. The system is conservative and subject to a gravitational field (9.81 m/s^2).

The state of the system, x, is described by the relative angles of the pendulums with respect to the vertical (θ_1 and θ_2) and their time derivatives ($\dot{\theta}_1$ and $\dot{\theta}_2$). Data was collected by simulating the system multiple times with random initial conditions given by

$$x_0 = [\mathcal{U}(-1, 1)l_{\theta_1}, \mathcal{U}(-1, 1)l_{\theta_2}, \mathcal{U}(-1, 1)l_{\dot{\theta}_1}, \mathcal{U}(-1, 1)l_{\dot{\theta}_2}]$$

where $\mathcal{U}(-1, 1)$ is an uniformly distributed random variable with range -1 to 1. Furthermore, $l_{\theta_1} = l_{\theta_2} = \frac{\pi}{3}$ and $l_{\dot{\theta}_1} = l_{\dot{\theta}_2} = 0.5$. Therefore, the initial condition is uniformly distributed around the origin (and the stable equilibrium) and its range is defined by $L = [l_{\theta_1}, l_{\theta_2}, l_{\dot{\theta}_1}, l_{\dot{\theta}_2}]$. Any data point that fall outside of the range defined by L was not used to approximate the Koopman operator. Data collection occurred at 100 Hz and was stopped when $2,000$ data points were collected.

The vector-valued functions used in this numerical experiment are polynomial basis functions give as

$$\Psi(x) = [\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2, 1, \psi_1, \psi_2, \ldots, \psi_M] \quad (34)$$
$$\psi_i(x) = (\theta_1/l_{\theta_1})^{\alpha_i}(\theta_2/l_{\theta_2})^{\beta_i}(\dot{\theta}_1/l_{\dot{\theta}_1})^{\gamma_i}(\dot{\theta}_2/l_{\dot{\theta}_2})^{\delta_i} \quad (35)$$
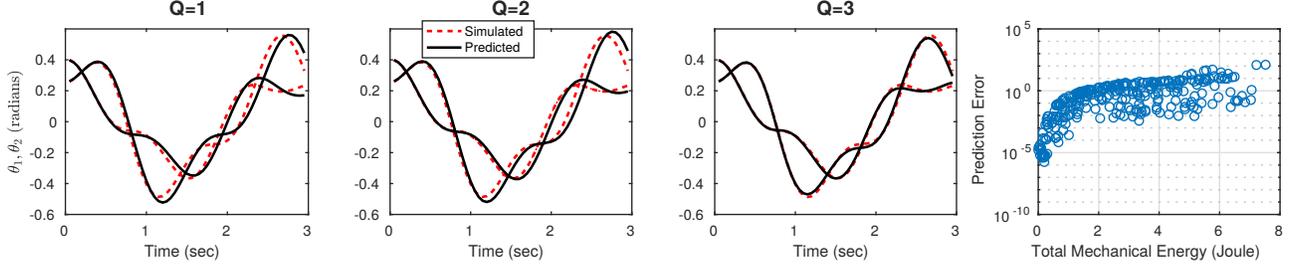
Fig. 2. Simulated trajectories when the approximate Koopman operator was used to propagated the system's configuration. As the complexity Ψ increases, so does the accuracy in prediction. 100 trials with uniformly random initial conditions were conducted to invesgate the relationship between accuracy and total mechanical energy. The prediction error tended to increase with total mechanical energy.

where $\alpha_i, \beta_i, \gamma_i$, and δ_i are nonnegative integers, index i tabulates all the combinations such that $\alpha_i + \beta_i + \gamma_i + \delta_i \leq Q$ and $Q > 1$ defines the largest allowed polynomial degree. Note that $-1 \leq \psi_i \leq 1$ when the state of the system is within the defined range. The polynomial basis functions were scaled by the maximum expected value of the state to prevent numerical instability when higher order polynomials were utilized.

Figure 2 shows a simulated trajectory and the corresponding predicted trajectories when approximated Koopman operators were used to propagate the system's configuration. As expected, the accuracy of the predicted trajectories are improved when Q is increased. Figure 2 also shows how the accuracy of the predicted trajectories are dependent on the initial conditions. The prediction error of a trajectory is computed as

$$\frac{1}{N} \sum_i^N (x_{\text{sim},i} - x_{\text{K},i})^2 \qquad (36)$$

where x_{sim} is the simulated trajectory, x_{K} is the system's trajectory predicted by the approximated Koopman operator, and N is the total run-time of the simulation. The prediction error tended to increase with total mechanical energy. Recall that the dynamics of a double pendulum are described by transcendental functions. Therefore, any approximation by polynomials of these dynamics will deteriorate as the relative angle increases in magnitude. However, when the relative angles are small (total mechanical energy is small) a polynomial approximation is accurate. As expected, selection of Ψ plays a critical role in determining the quality of the computed Koopman operator.

B. Simulation: Inversion and Stabilization of Pendulum Systems

In this section, we describe the results of utilizing the Koopman operator for inverting a cart-pendulum system and a VTOL-pendulum system. In particular, this section overviews the effect that the choice of basis functions has on systems that have components in $SO(n)$ for $n > 1$.

For the cart-pendulum system, the Koopman states are given as

$$\Psi(x) = [\theta, x, \dot{\theta}, \dot{x}, u, 1, \psi_1, \psi_2, \ldots, \psi_M] \qquad (37)$$

where we use

$$\psi_i(x) = \theta^{\alpha_i} x^{\beta_i} \dot{\theta}^{\gamma_i} \dot{x}^{\delta_i} u \qquad (38)$$

as the polynomial basis function set and compare with a Fourier basis function,

$$\psi_i(x) = \prod_{[x]_i} \prod_{\kappa_j} \cos([x]_i \kappa_j) \sin([x]_i \kappa_j) u, \qquad (39)$$

where $[x]_i$ is the ith state of the system and κ_j is the jth basis order such that $\sum_j \kappa_j \leq Q$.

In this simulation, a nominal model given by

$$x_{k+1} = x_k + \begin{bmatrix} \dot{\theta}_k \\ \dot{x}_k \\ u \\ u \end{bmatrix} \delta t \qquad (40)$$

is utilized as an initial guess for the controller in order to boot-strap the data-driven process. Figure 3 presents the use of increasing complexity orders of a polynomial and Fourier basis function for the cart-pendulum system. Both test cases begin with the same initial condition and the same nominal model. At intervals of $20s$, a Koopman operator is computed with either the polynomial or Fourier basis functions using the initial $20s$ of data collected. Due to the existence of the pendulum on $SO(1)$, the Fourier basis function immediately generates a Koopman operator model that allows the controller to balance and stabilize the pendulum. Moreover, the use of the Fourier basis illustrates the concept that increasing complexity on the operator basis set is not always guaranteed to return an improved data-driven model. In particular, when $Q = 2$, the Koopman operator matches the system model identically. As a result, any further additions in complexity using the Fourier basis for this system is not beneficial (this is not always the case if the system has higher order dependencies). In contrast, the polynomial basis function does show improvement as complexity is increased. Although it would require an infinite set of polynomials to approximate a cosine or sine function, the controller using this operator model provides the desired energy pumping cart motion that is commonly witnessed in inverting a pendulum.

Simulated examples are further investigated with the use of a vertical take-off and landing (VTOL) pendulum system
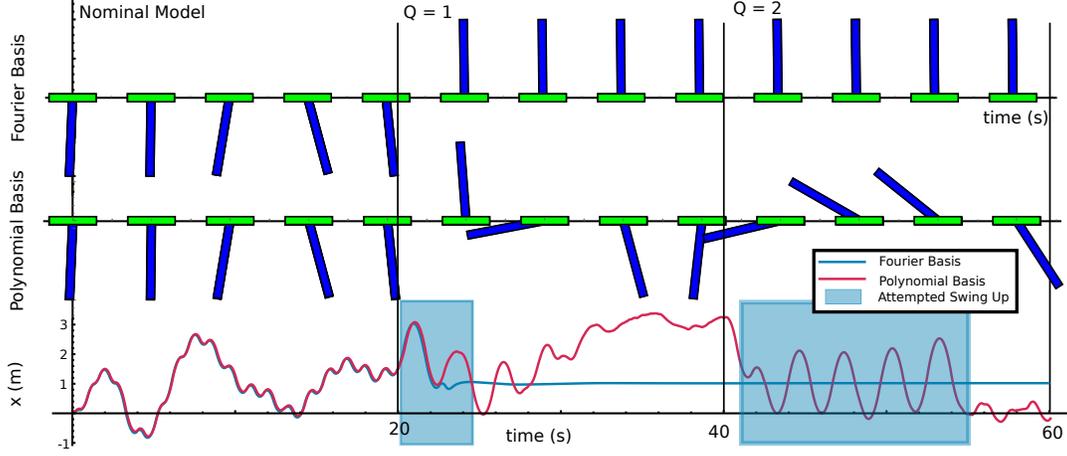
Fig. 3. The progressive improvement in control as the Koopman operator increases the basis order of complexity Q is shown. Each pendulum configuration is taken as a snapshot in time. Koopman operators with complexity Q are trained on the initial first 20 seconds with the nominal model. Note that because of the $SO(1)$ configuration of the pendulum, a Fourier basis of complexity $Q = 1$ is sufficient to invert at stabilize the cart-pendulum. Adding a higher complexity $Q = 2$ does not provide a different Koopman matrix (this does not necessarily hold true for non-simulated systems). It is interesting to note that as the complexity of the polynomial basis increases, so do the number of attempts at swinging up the cart-pendulum. Link to multimedia provided: https://vimeo.com/219458009 .

[24]. For this example, the problem of inverting the pendulum attached to a VTOL is slightly modified. Specifically, it is assumed that a well known model of the VTOL exists, but the interaction between the VTOL and the pendulum remains unknown. Thus, the goal of this simulated example is to generate a Koopman operator that describes the interaction of the VTOL on the pendulum.

In this example, the Koopman operator is redefined as an augmentation to a dynamical system

$$x_{k+1} = f(x_k, u_k) + \tilde{K}^T \Psi(x_k, u_k)^T. \tag{41}$$

By subtracting the current nominal model of the system $f(x_k, u_k)$ from both side in equation (41) and treating x_{k+1} as the measurement of state, we can define the following as a nonlinear process that can be used to generate a Koopman operator:

$$\Psi(x_{k+1}) = x_{k+1} - f(x_k, u_k) = \tilde{K}^T \Psi(x_k, u_k)^T. \tag{42}$$

Given the previous cart-pendulum result, we see that the interaction between the VTOL and the pendulum can be captured solely via a vast set of basis functions across the state of the VTOL-pendulum system. In Fig. 4, the VTOL is shown attempting to invert and balance the pendulum attached with the use of the Koopman operator. Each sequential Koopman operator with increasing complexity is generated from the first 20 seconds worth of data. Originating from the nominal model, it can be seen that the swinging behavior captures a portion of the energy pumping maneuvers required to invert the pendulum. As the Koopman basis order increases, so does the refinement in control authority. When $Q = 2$ for the polynomial basis, it can be seen that swing up attempts are more successful. Once the Koopman operator generated from the Fourier basis functions is used, the controller generates

the appropriate control strategy to swing up and invert the pendulum.

In the following section, our discussion on the use of the Koopman operator is extended to control of a Sphero SPRK robot in a reduced state setting.

C. SPRK Experiments

1) Open-Loop Trajectory Optimization: Figure 5 shows trajectories generated using the open-loop controller with varying Q. The reference trajectory is given as

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \dot{\tilde{x}} \\ \dot{\tilde{y}} \end{bmatrix} = \begin{bmatrix} r\cos(vt) \\ r\sin(2vt) \\ -rv\sin(vt) \\ 2rv\cos(2vt) \end{bmatrix}. \tag{43}$$

where $r = 0.5$ and $v = 1.3$. The reference trajectory was made sufficiently aggressive to excite the system's internal nonlinearities.

As expected, the system improves in performance when tracking the reference trajectory with increasing Q. In particular, as Q goes from 1 to 2, less drift in the resulting open-loop trajectory is visually noted at the end of the path. As Q is further increased, more complexity is added to the description of the SPRK via the Koopman operator which in turn reduces drift and improves the tracking performance. Furthermore, the standard deviation of tracking error across trials is shown to reduce as Q is increased. This implies both consistency in the behavior of the robot subject to the controller. Therefore, it can be concluded that the approximated Koopman operator is better able to represent the dynamics of the system by increasing the complexity of Ψ.
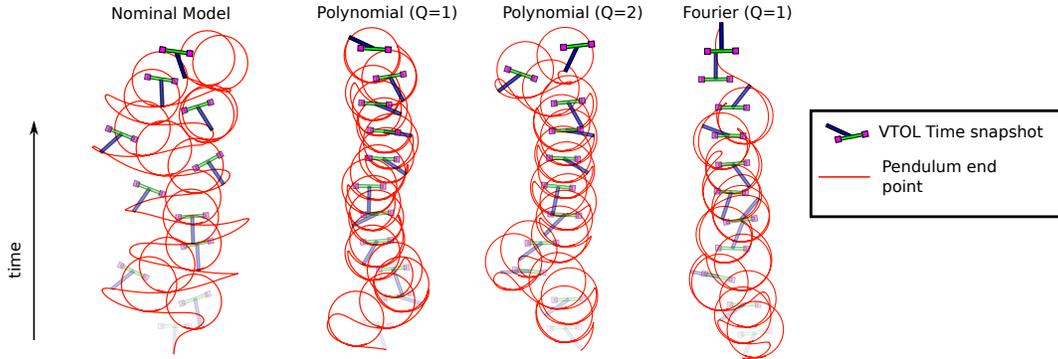
Fig. 4. Each Koopman operator is trained on the residual modeling error of 20 seconds attempted pendulum inversion using the nominal model. As the order of the polynomial basis increases from $1 \rightarrow 2$, the number of swing up attempts also increases. Notably, a first order Fourier basis captures the necessary features that allow the controller to invert and stabilize the pendulum. Link to multimedia provided: https://vimeo.com/219458009 .
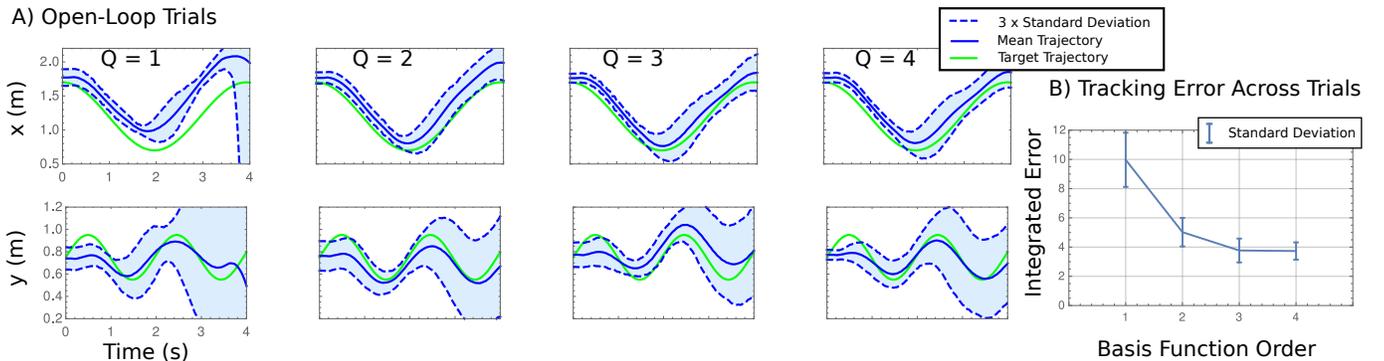

Fig. 5. Here we show reference tracking using open-loop trajectory optimization. The reference trajectory was made sufficiently aggressive to excite the system's internal nonlinearities that cannot be captured completely by the minimal state representation. Respective integrated tracking errors are shown to decrease with an increase in Q. This suggests that the approximate Koopman operator better represents the dynamics of the system with increasing complexity of Ψ.

2) Closed-Loop Trajectory Tracking: Figure 6 shows the experimental results for trajectory tracking on a tarp and sand terrain using closed-loop model-based controllers with the Koopman operator. The optimal control signal was updated at 20 Hz and the reference trajectory was given by equation (43) where r is split into two components, $r_x = 0.7$ and $r_y = 0.4$, with $v = 0.9$. The nominal linear model is given by

$$x_{k+1} = Ax_k + Bu_k, \qquad (44)$$

where A and B are defined as a fully controllable double integrator system.

The effectiveness of the closed-loop controller is benchmarked by comparing the model generated from the Koopman operator to that of a simulated example of the controller knowing the true system model (Fig. 6). Using only the first 20 seconds worth of data from the nominal model controller, we can see in Fig. 6 A) that as the operator increases in complexity, so shows the performance of the controller relative to the benchmark test. Specifically, Fig. 6 B) shows the tracking error for experimental trials with increasing complexity of the Koopman operator. Notably, when $Q = 3$ in sand, the Koopman operator did not have a sparse enough data set that

spans the higher order terms in the operator. This can be fixed by collecting more data that spans the robot's operating region.

Here, the nonlinear dynamics driven by the internal mechanism become more apparent as the order of the operator is increased. In particular, equation (6) provides some insight into the output of the data-driven model of the Koopman operator for the update equation of the SPRK's velocity subject to control inputs. Because the effect of the internal mechanism's configuration (typically described on $SO(3)$) cannot be linearly approximated, the Koopman operator begins to approximate a Taylor expansion (6). Therefore, the Koopman operator captures the inherent nonlinearities that are utilized by the model-based controller with respect to the terrain. However, achieving a representation that performs consistently across all operating terrains seems infeasible with such limited information, without extra structure on the Koopman operator, such as global Lie group structure or mechanical properties (e.g. symmetries).

VI. CONCLUSION

We present Koopman operator theory and focus on the practical implementation of the theory for model-based con-

A) Experimental Results with Sphero SPRK
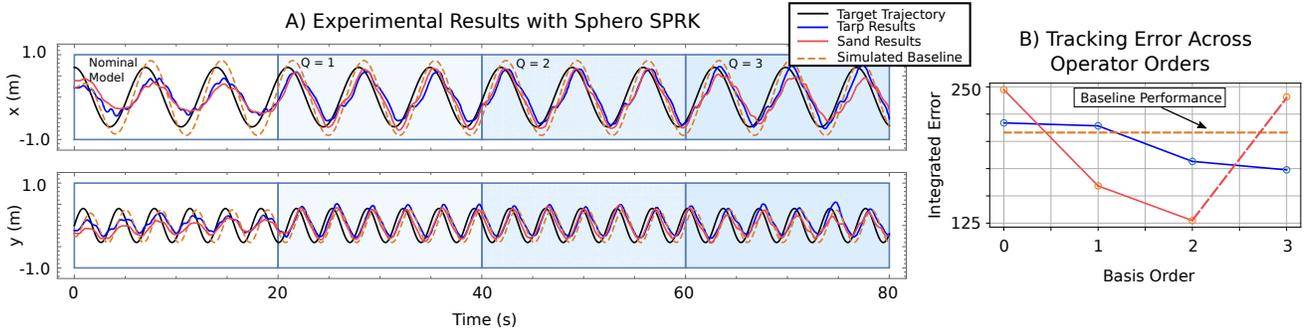
B) Tracking Error Across Operator Orders

Fig. 6. Here, we show closed-loop model-based control using sequentially increasing basis complexity, Q in the Koopman operator. Two examples using the SPRK robot are run on a tarp and on sand. A baseline simulated example is provided to show the best-case performance of the controller subject to the nominal model used. As the complexity of the operator's basis function is increased, so does the performance of the tracking. Note that in B), the $3rd$ order operator used in sand (shown as the dashed red line) did not have a sparse enough set of data to provide a stable model, although it performed better than the nominal model. Link to multimedia provided: https://vimeo.com/219458009 .

$$\begin{bmatrix} \dot{x}_{k+1} \\ \dot{y}_{k+1} \end{bmatrix} = \begin{bmatrix} 0.08x_k - 0.35y_k + 0.76\dot{x}_k + 0.21\dot{y}_k + 1.06u_1 - 0.17u_2 - 0.05\dot{x}_k^2 - 0.19\dot{y}_k^2 - 1.09\dot{x}_k\dot{y}_k^2 - 0.71\dot{y}_k\dot{x}_k^2 + 0.40 \\ -0.06x_k + 0.16y_k + 0.17\dot{x}_k + 0.87\dot{y}_k - 0.38u_1 + 0.57u_2 - 0.20\dot{x}_k^2 - 0.52\dot{y}_k^2 - 0.45\dot{x}_k\dot{y}_k^2 - 3.17\dot{y}_k\dot{x}_k^2 - 0.04 \end{bmatrix} \tag{6}$$

trol. We derive a linearizable data-driven model using the Koopman operator. Closed-loop and open-loop controllers were formulated using the proposed data-driven model. The open-loop experiments reveal the Koopman operator improves performance as the complexity of the basis increases. Closed-loop experiments reveal the Koopman operator is able to capture the nonlinear dynamics of simulated examples with the cart- and VTOL-pendulum and the SPRK robot.

Future research directions include an in-depth analysis of the choice of basis for dynamical system with distinct structure (e.g. conservative systems, mechanical systems, etc.). The relationship between available states and the accuracy of the approximate Koopman operator needs rigorous stability analysis. Moreover, numerical stability analysis and algorithmic optimization is another possible research avenue.

REFERENCES

[1] N. Hovakimyan and C. Cao, *L1 Adaptive Control Theory: Guaranteed Robustness with Fast Adaptation*. SIAM, 2010.
[2] K. J. Åström and B. Wittenmark, *Adaptive control*. Courier Corporation, 2013.
[3] K. Zhou and J. C. Doyle, *Essentials of robust control*. Prentice hall Upper Saddle River, NJ, 1998, vol. 104.
[4] D. S. Bernstein and W. M. Haddad, "LQG control with an H/sup infin-ity/performance bound: a Riccati equation approach," *IEEE Transactions on Automatic Control*, vol. 34, no. 3, pp. 293–305, 1989.
[5] S. C. Ong, S. W. Png, D. Hsu, and W. S. Lee, "Planning under uncertainty for robotic tasks with mixed observability," *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 1053–1068, 2010.
[6] A. Bry and N. Roy, "Rapidly-exploring random belief trees for motion planning under uncertainty," in *International Conference on Robotics and Automation (ICRA)*, 2011, pp. 723–730.
[7] J. Van Den Berg, S. Patil, and R. Alterovitz, "Motion planning under uncertainty using differential dynamic programming in belief space," in *Robotics Research*. Springer, 2017, pp. 473–490.
[8] D. Nguyen-Tuong and J. Peters, "Model learning for robot control: a survey," *Cognitive processing*, vol. 12, no. 4, pp. 319–340, 2011.
[9] M. Jordan and T. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.
[10] C. G. Atkeson, A. W. Moore, and S. Schaal, "Locally weighted learning for control," in *Lazy learning*. Springer, 1997, pp. 75–113.
[11] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou, "Information theoretic mpc for model-based reinforcement learning," *International Conference on Robotics and Automation (ICRA)*, 2017.
[12] 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.
[13] D. Henrion, I. Mezic, and M. Putinar, "Applied Koopmanism," 2016.
[14] I. Mezić, "Analysis of fluid flows via spectral properties of the Koopman operator," *Annual Review of Fluid Mechanics*, vol. 45, pp. 357–378, 2013.
[15] A. Mauroy and I. Mezić, "Global stability analysis using the eigen-functions of the Koopman operator," *IEEE Transactions on Automatic Control*, vol. 61, no. 11, pp. 3356–3369, 2016.
[16] I. Mezić, "On applications of the spectral theory of the Koopman operator in dynamical systems and control theory," in *Decision and Control (CDC)*, 2015, pp. 7034–7041.
[17] A. Broad, T. D. Murphey, and B. Argall, "Learning models for shared control of human-machine systems with unknown dynamics," *Robotics: Science and Systems Proceedings*, 2017.
[18] 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.
[19] J. Hauser, "A projection operator approach to the optimization of trajectory functionals," *IFAC Proceedings Volumes*, vol. 35, no. 1, pp. 377–382, 2002.
[20] 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, Oct 2016.
[21] J. Nocedal and S. J. Wright, "Numerical optimization 2nd," 2006.
[22] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.
[23] G. Bradski, *Dr. Dobb's Journal of Software Tools*, 2000.
[24] T. Luukkonen, "Modelling and control of quadcopter," *Independent research project in applied mathematics, Espoo*, 2011.