An Exploration of the Koopman Operator for Modeling Controlled Dynamics and Distance Propagation

Michael D. Napoli

***Abstract*— Nonlinear systems are often times unavoidable in engineering and maintain their status as being difficult and computationally inefficient to control optimally. In extreme cases, it may not be realistic, or even possible, to derive these equations analytically. While data-driven alternatives for modeling can be found in various machine learning algorithms, they often learn the dynamics through trial and error, which can be slow and is subject to simulation-faults.**

The Koopman operator is an infinitely-dimensioned matrix which propagates a set of observation functions forward in time and can be used to model nonlinear dynamics in a linear form. This paper explores two methods of deriving the Koopman operator; analytically and through a data-driven least-squares approach, then compares the results with the true model dynamics. Furthermore, to make the observation functions more rigorous, they will incorporate distances from predetermined spherical objects. The inclusion of distances into the observation space is advantageous as it allows the user to implement linear constraints with respect to avoiding spherical obstacles. Ideally, this would produce fast and optimal linear control of a robot which would otherwise be moving through an environment with nonlinear constraints. The results of this paper are two fold: 1). the Koopman operator is easily capable of capturing the dynamics for linear system propagation, but 2). the data-driven approach requires more investigation before integrating into an accurate control synthesis. In short, the work presented here serves as an introduction to the Koopman operator, and its applicability to nonlinear system modeling.

I. Introduction

While nonlinear dynamical systems are ubiquitous to the robotics community, they remain among the most difficult systems to control both reliably and efficiently. Current methods for modeling these systems are based on exact analytical expressions, which can be difficult, even impossible to derive, or complicated approximation methods which incorporate a degree of error and can be tedious to compute. That said, the first step to system design and implementation is developing a comprehensive model which incorporates control inputs, disturbances and other user-defined parameters. The growth of these systems with the intent of robustness inevitably correlates to an increases in nonlinearity, making the system increasingly difficult to control.

The paper explores the calculation and implementation of the Koopman operator, an infinitely-dimensioned matrix which propagates observation functions on the system forward in time and can be used to linearize complex systems. This is a powerful behavior, as it allows otherwise possibly unpredictable behaviors to be controlled or implemented in a linear manner. It is also a powerful alternative when the dynamics of a system are completely unknown.

To test this phenomenon, the operator is calculated analytically, and through the data-driven approach titled extended dynamical mode decomposition (EDMD) which will be discussed more thoroughly in Section II and III. While the analytical derivation of the Koopman operator defeats the purpose of its ability to be derived solely from data, it will be used in order to understand the breadth of its ability to propagate nonlinear functions forward in time.

By comparing the data-driven operator to the analytical, the accuracy of the EDMD approach can be legitimized. Motivation for this paper is likewise propelled by the possibility of being able to accurately and optimally control complex nonlinear systems using linear model-based control methods.

II. Literature Review

The Koopman operator has become an important alternative to nonlinear system modeling in generally uncontrolled environments since its derivation in 1931 by B.O. Koopman [1]. More recently, its derivation for the purpose of implementation was repeated in [2] where the authors presented a formulation in more modern notation and gave examples as to possible applications of this powerful operator. These examples include fluid system evaluations, power system modeling, jets in cross flows and building energy efficiencies, among others. The paper also presents dynamical mode decomposition (DMD) as the primary method in solving for the finite-dimensioned form of the otherwise infinitely-dimensioned operator. The finite-dimensioned operator can also be derived via EDMD, which is considered a data-driven approach to modeling [3]. This is especially powerful in systems for which the dynamics are not fully understood, or cannot be represented analytically. EDMD will be the primary approach presented in Section III-B. In [4] the concept of Koopman generators are used to extend the EDMD method to continuous dynamics, and the authors claim that this formulation has minimal approximation error in comparison to the more popular discrete approach.

The control of nonlinear dynamical systems are equally important to their modeling. That said, implementing optimal controllers are often associated with non-convex problem formulations and are computationally inefficient in comparison to linear alternatives. Many approaches to control work to overcome these issues. For example, various model linearization methods, using non-optimal control policies, or performing nonlinear optimization.

Robotic models require inputs which allow for manipulation and control. There are multiple studies into the reformulation of the Koopman operator to incorporate control variables into the observation space [5, 6, 7]. This is not an exhaustive list by any means, but the papers presented here

are the state-of-the-art in the realm of Koopman operator generation with control and its application in model-based control policies. In [7], the Koopman operator is presented with control variable inputs, and its limitations are addressed.

Interesting research has also been pursued in the realm of bilinear control in [6] where the authors present a bilinear combination of input and state terms such that the operator can be used with linear optimal controllers, but performs with accuracy that more closely resembles nonlinear methods. While control of the system will not be implemented here, bilinear terms are used to decrease approximation errors in Section IV-B and thus the aforementioned paper will be utilized more thoroughly in future work. The formulation of the Koopman operator with inputs is more thoroughly defined in Section III-C.

While there are many implementations of the Koopman operator in practice, the paper [5] will be given special attention as it shows a thorough demonstration of the integration of the Koopman operator for a nonlinear system with a model-based controller.

It is also important to note that the Koopman operator serves as an effective alternative to the modeling of systems with unknown dynamics. In comparison, reinforcement learning (RL) approaches have gained some authority in the topic recently. Most notably, RL approaches have been proven effective for the control of the bipedal robot, Cassie, with very little information on the model dynamics. This is notable as bipedal robotics is among the most unstable systems to control [8]. That said, these approached commonly have problematic integration with real-world systems, unlike the Koopman operator which learns its behavior from preexisting data. Ideally, this would alleviate the time taken to train RL-based models.

III. Preliminaries

This section will discuss the general theory behind the Koopman operator and its derivation using a data-driven learning approach. The approach implemented here is titled extended dynamical mode decomposition, and will be implemented for discussion in Section V.

The methods presented here are from a collection of works, but most heavily draw from [5]. They are thus restated for completeness with the intent that the motivations in using these operators can be thoroughly understood.

It is important to note that this paper will represent the states, inputs and observable functions as *row-vectors*, as opposed to the commonly used *column-vector* notation.

A. Koopman Operator Theory

Ordinarily, a model function for a given dynamical system is calculated in discrete-time using a transition function, F.

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

Where $x_k \in \mathbb{M} \in \mathbb{R}^N$ is a row vector of a state of size N at the time-step, k. F is a discrete model function which maps $F : \mathbb{M} \to \mathbb{M}$. It is common for F to be nonlinear with respect to the states; leading to issues in runtime and the overall complexity of optimal control functions.

The Koopman operator presents a data-driven alternative to analytical derivations of F in terms of a linear operator. To utilize the Koopman operator, an observation function, g, which communicates the state space into the infinitely-dimensioned function space, $\mathbb{C}$, is used.

$$z_k = g(x_k) \tag{2}$$

Where z_k is composed of the observable functions built from the state x_k. In other words, z_k represents the current state translated to the observation space such that $z_k \in \mathbb{C} \in \mathbb{R}^\infty$ and $g \in \mathbb{G} : \mathbb{M} \to \mathbb{C}$. If the function space is defined as a Hilbert space, a similar function to (1) can thus be written in terms of its own propagation function.

$$z_{k+1} = z_k\mathcal{K} = g(x_k)\mathcal{K}. \tag{3}$$

Where $\mathcal{K} \in \mathbb{R}^{\infty\times\infty}$ is the Koopman operator and is used to propagate the observation functions forward in time. Putting (2) and (3) together, the operator can be better represented without intermediate terms.

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

Where $\mathcal{K} : \mathbb{G} \to \mathbb{G}$ and is used to propagate the observation functions forward w.r.t. a constant time-step. Importantly, in this derivation, $\mathcal{K}$ is linear when $\mathbb{G}$ is a vector space, even when $\mathbb{M}$ is a nonlinear domain.

B. Approximating the Koopman Operator

In order to solve for the Koopman operator, the dimensionality of the function space must first be addressed. Being an infinitely-dimensioned space is not realistic when put in terms of real world applications so (2) is redefined as a subset of the original space.

$$z_k = \Psi(x_k) \tag{5}$$

Where $\Psi \in \mathbb{G}^{N_K} \subset \mathbb{G}$ is some observation function with dimension N_K that approaches but does not equal infinity and is dependent on the accuracy desired by the user. More specifically, Ψ can be written as the list of chosen observation functions.

$$\Psi(x_k) = [\psi_1(x_k),\ \psi_2(x_k),\ \cdots,\ \psi_{N_K}(x_k)]. \tag{6}$$

Where each ψ_i term is a function of the state variable. Combining this approach with 4 yields the following.

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

Where $r(x_k)$ is the residual error presented by the approximate operator, Ψ. Likewise, the Koopman operator is now represented as $K \in \mathbb{R}^{N_K\times N_K}$.

Now, assuming data is available which represents the desired behavior, the Koopman operator in finite dimensions can be solved for using a list of data points.

$$X = [x_1, x_2, \cdots, x_P] \tag{8}$$

Where X is the list of state propagation data and P is the number of collected data points. Using this data, the solution for the Koopman operator can be defined as the minimization of the residual error over the entire data set.

$$J = \frac{1}{2}\sum_{k=1}^{P-1} ||r(x_k)||^2 \tag{9}$$

Which can be restated in terms of the Koopman operator approximation from (7).

$$J = \frac{1}{2}\sum_{k=1}^{P-1} ||\Psi(x_{k+1}) - \Psi(x_k)K||^2 \tag{10}$$

In this form, the solution for J is a simple least-squares regression such that $K = G^{\dagger}A$ where $G^{\dagger}$ is the pseudo-inverse of G. More specifically, the matrices G and A are composed of the observation functions for the current state and its propagation forward in time.

$$G = \frac{1}{P}\sum_{p=1}^{P-1} \Psi(x_k)\Psi^{\intercal}(x_k), \tag{11}$$

$$A = \frac{1}{P}\sum_{p=2}^{P} \Psi(x_k)\Psi^{\intercal}(x_{k+1}). \tag{12}$$

It is important to note that not all observation functions may be necessary in the final representation of the model. For this reason, single value decomposition (SVD) will be used to prioritize the higher impact terms when computing $G^{\dagger}$. The single value decomposition equation is restated here for completeness.

$$G = USV^{\intercal} \tag{13}$$

Where $U, V \in \mathbb{R}^{N_K \times N_S}$ and $S \in \mathbb{R}^{N_S \times N_S}$ such that the matrices represent the N_S most prominent terms in the observation space. The pseudo-inverse can be found by exploiting the nature of the SVD results.

$$G^{\dagger} = VS^{-1}U^{\intercal} \tag{14}$$

Finally, the finite dimensioned Koopman operator can be reformulated using the results of the SVD.

$$K = \left(VS^{-1}U^{\intercal}\right)A \tag{15}$$

Where K is a learned approximation of the Koopman operator from data. These steps will be used to formulate the approach referred to as the data-driven Koopman operator in Section V.

C. Koopman Operator With Control

With the understanding that the Koopman operator is composed of a system of observable functions, the incorporation and linearization of control variables is a straightforward process [5]. The observation function, as defined in (5), thus becomes a combination of state and input terms.

$$\Psi(x_k, u_k) = [\psi_1(x_k, u_k), \cdots, \psi_{N_K}(x_k, u_k)] \tag{16}$$

For clarity, both system (5) and (16) will be linearized with respect to their components. For the observation function shown in (5) the model propagation function is

$$\Psi(x_{k+1}) = \Psi(x_k)K. \tag{17}$$

Where the state x_k is easily retrievable from the propagation. To linearize this with respect to x_k, the observables are chosen such that the observation function is differentiable. The resulting equation is equal to an equation of the form $x_{k+1} = x_k A(x_k)$.

$$\Psi(x_{k+1}) = x_k \frac{\partial \Psi}{\partial x}K \tag{18}$$

For the model with inputs, the propagation can likewise be defined as a function of (16).

$$\Psi(x_{k+1}, u_{k+1}) = \Psi(x_k, u_k)K \tag{19}$$

Which can be linearized via a similar method to (18).

$$\Psi(x_{k+1}, u_k) = x_k \frac{\partial \Psi}{\partial x}K + u_k \frac{\partial \Psi}{\partial u}K \tag{20}$$

In this form, the propagation function closely resembles a control-affine function of the form $x_{k+1} = A(x_k, u_k)x_k + B(x_k, u_k)u_k$ and is compatible with most linear model-based control structures. It is important to note that while the input is taken into the list of observables, it does not get propagated by K and is written to the next time-step with no change in magnitude.

IV. Methods

A. Sphere World Environment

in this implementation, a point-robot will be used to demonstrate the performance of the Koopman operator when modeling nonlinear systems. The point-robot will be treated with holonomic dynamics, and thus the only noteworthy state variables are its x-y coordinate pair.

$$x_k = \begin{bmatrix} q_{x,k}, & q_{y,k} \end{bmatrix} \tag{21}$$

Where the row-vector state of the robot, x_k, is made up of the position, $q_{j,k}$, where j represents the axis of reference. Using holonomic dynamics, the input can thus be defined in terms of the derivative of x_k.

$$u_k = \begin{bmatrix} \dot{q}_{x,k}, & \dot{q}_{y,k} \end{bmatrix} \tag{22}$$

Where $\dot{q}_{j,k}$ is the velocity being applied directly to the robot. This can thus be implemented into a linear dynamics function.

$$x_{k+1} = x_k + \alpha u_k \tag{23}$$

Where α is some predetermined step-size. These dynamics may seem trivial, but they are important as it makes the analytical derivation of the Koopman operator in Section IV-B possible. In addition to the robot state and input, the distance of the robot from each of the environment obstacles will also be defined and utilized within the observation space.

$$O(x_k) = \begin{bmatrix} d(x_k, o_W), & d(x_k, o_1), & \cdots, & d(x_k, o_{N_O}) \end{bmatrix} \tag{24}$$

Where $d(x_k, o_n)$ is the distance from the position of x_k to the center point of a circular obstacle, o_n. The obstacle space therefore contains the distance from N_O obstacles and the wall, o_W.

This environment was chosen for the simplicity of its distance equation from each of the obstacles. This can likewise be easily interpreted into a linear constraint equation such that $d(x_k, o_n) \geq R(o_n) + R_{robot}$. Where $R(o_n)$ is the radius of the obstacle o_n and R_x is the radius of the robot.

B. Observation Space

To understand the selection of observation functions, it is first necessary to discuss the distance function implemented for use in (24). Here, the squared L_2-norm is used as it makes the analytical derivation of the Koopman operator approachable.

$$d(x_k, o_n) = (x_k - o_n)(x_k - o_n)^\intercal \tag{25}$$

Using this formulation, and by substituting (23) in for the current step, it is possible to derive the exact analytical solution to the propagation of the distance function.

$$\begin{aligned} (x_{k+1} - o_n)(x_{k+1} - o_n)^\intercal &= (x_k + u_k - o_n)(x_k + u_k - o_n)^\intercal \\ \Rightarrow\ & x_k x_k^\intercal + x_k u_k^\intercal - x_k o_n^\intercal \\ & + u_k x_k^\intercal + u_k u_k^\intercal - u_k o_n^\intercal \\ & - o_n x_k^\intercal - o_n u_k^\intercal + o_n o_n^\intercal \\ \Rightarrow\ & x_k x_k^\intercal + u_k u_k^\intercal + o_n o_n^\intercal \\ & + 2x_k u_k^\intercal - 2x_k o_n^\intercal - 2u_k o_n^\intercal \end{aligned} \tag{26}$$

In other words, the propagation of $d(x_k, o_n)$ is a function of bilinear terms with respect to the state and input. This behavior shows that if $x_k x_k^\intercal$, $u_k u_k^\intercal$ and $x_k u_k^\intercal$ are present in the observation space, an exact solution of the squared distance should be calculable. The distance can then be implemented directly into the constraints of a linear optimization problem as it does not change the convexity of the problem. It should also be noted that the current state is also known.

$$(x_k - o_n)(x_k - o_n)^\intercal = x_k x_k^\intercal - 2x_k u_k^\intercal + o_n o_n^\intercal \tag{27}$$

Since (27) is directly dependent on the current step, the terms shared with (26) can be eliminated and replaced with the preexisting value for distance. Using this information the sphere world observation space, Ψ_s, can be defined using x_k, u_k and $O(x_k)$ along with the appropriate observable functions. From the methods shown in (26), the observable functions are chosen such that the distance can be modeled analytically. The bilinear combinations of $x_k^\intercal x_k$, $u_k^\intercal u_k$ and $x_k^\intercal u_k$ are included because they contain the terms necessary to compute their L_2-norm squared counterpart and can be propagated on an element-by-element basis.

$$\begin{aligned} \Psi_s(x_k, u_k) = [&x_k, u_k, O(x_k), \\ &\texttt{vec}(x_k^\intercal x_k), \texttt{vec}(u_k^\intercal x_k), \\ &\texttt{vec}(x_k^\intercal u_k), \texttt{vec}(u_k^\intercal u_k)] \end{aligned} \tag{28}$$

Where the function $\texttt{vec}$ defines the mapping of matrices elements to the vector space. It can be noted that although $(x_k^\intercal u_k)^\intercal = u_k^\intercal x_k$ and thus only one is necessary to define both terms, both are included so as to make indexing easier in implementation.

Because the incorporation of distance used here is equivalent to $||x_k - o_n||_2^2$, the comparisons made in Section V will make claims based on the $\sqrt{d(x_k, o_n)}$. In this way, the accuracy of the L_2-norm can be evaluated directly.

V. Results

Two approaches to the derivation of the Koopman operator are attempted here. The first is the analytical approach; where the coefficients are derived via similar methods to those shown in (26). The second is the data-driven method as defined in Section III-B. For demonstration purposes, a sinusoidal input which ranged between $[-2.5, 2.5]$ was used to develop the model over a $10[s]$ time frame and is shown in Figure 1. In the model equation the α coefficient was set to 1 for simplicity and the initial position of the robot was set to the origin of the map, $[0, 0]$.

Using these parameters, the models could be tested for relatively dramatic changes in state. Figure 2 shows the development of x over time for all three models. As can

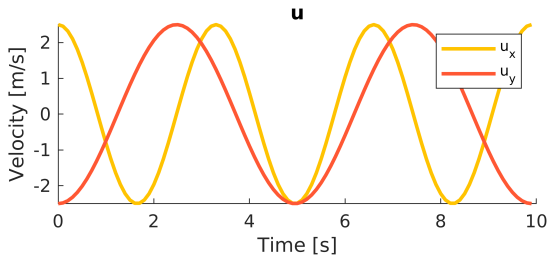

Fig. 1. Sinusoidal input trend over $10[s]$ time period.

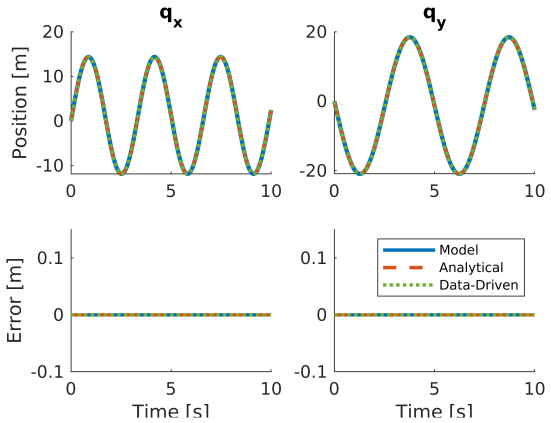

Fig. 2. Propagation of the state, x_k, over time.

be seen, both models represent the state development terms with negligible error. This makes sense as the system is simple and the operators are capable of solving for the exact solution.

More importantly, the system was tested for how it modeled the bilinear terms which were incorporated into the observation space. While $x_k^\intercal x_k$, $u_k^\intercal u_k$ and $x_k^\intercal u_k$ were all factored into the space, only the $x_k^\intercal x_k$ terms are shown here as they were deemed representative of the accuracy of all the bilinear observable functions. This behavior can be seen in Figure 3. Although they are identical both $q_x q_y$ and $q_y q_x$ are shown. This was done so that indexing during implementation would be simpler, but does imply that both must be verified during testing.

It is clear that the behavior of the data-driven model (in green) incorporates an excessive degree of error into the bilinear terms. The reason for this is unclear, but is most likely due to problematic implementation of the least-squares regression algorithm. Interestingly, the data-driven model was observed to be off by a constant gain of $\frac{1}{2}$ for all bilinear combinations of terms.

Another possibility is the derivation of the partial derivatives for use in the linearized propagation function from (20). In an attempt to avoid derivation errors, the finite-difference method (FDM) was also tested. That said, this did not alleviate the error w.r.t. the bilinear terms, leading to a belief that the incorrect approximation of the model is a result of the learning algorithm.

In comparison to the data-driven approximation of the bilinear terms, the analytical Koopman operator matches the model with a high degree of fidelity. In a controls application, this degree of accuracy would be sufficient for incorporation into the bounds or cost if needed.

Here, the evaluation of the bilinear terms serves as an intermediate step to evaluating the distance. Unfortunately, because the bilinear terms for the data-driven approach are highly inaccurate, it is unlikely that the distance functions will show improved results. That said, with the understanding that the bilinear terms were off by a constant gain of $\frac{1}{2}$, the deviation of $d(x_k, o_n)$ from the model can be compared to this gain. Ideally helping with bug testing in the future.

Figure 4 shows the behavior of $\sqrt{d(x_k, o_n)}$ over time as discussed in Section IV and w.r.t. to both the analytical and data-driven approximations of the Koopman operator. As previously mentioned, the data-driven method again incorporates a large degree of error, most likely a direct

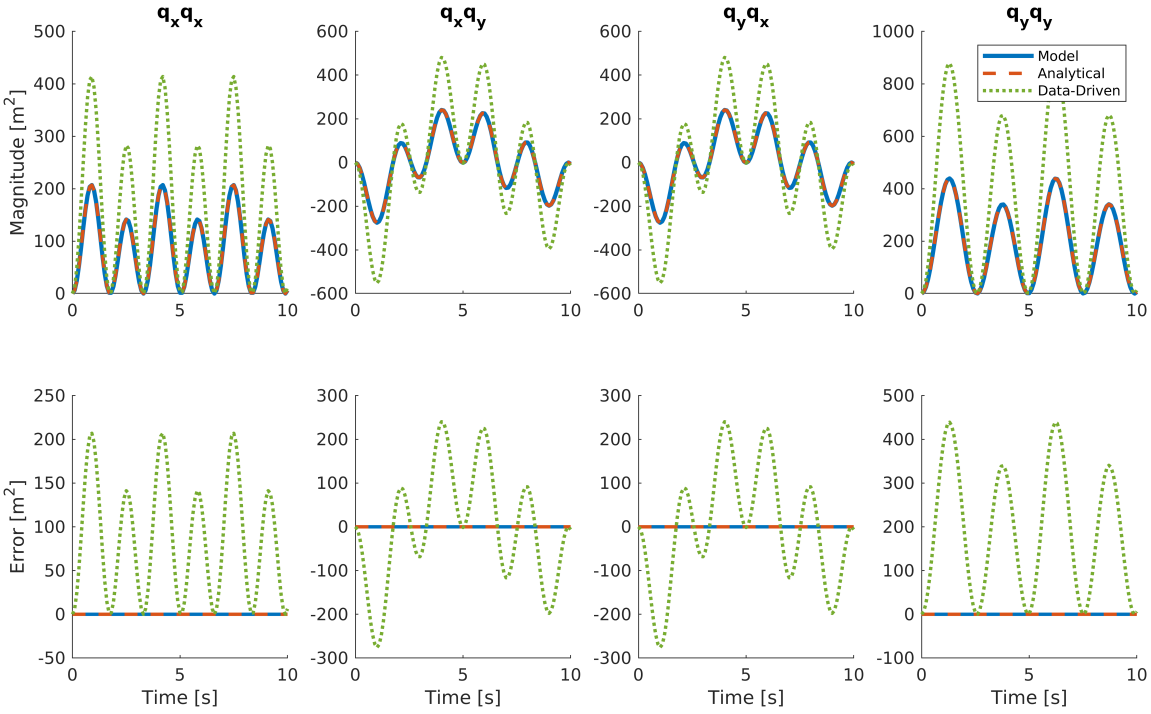

Fig. 3. Propagation of $x_k^\intercal x_k$ terms over time.

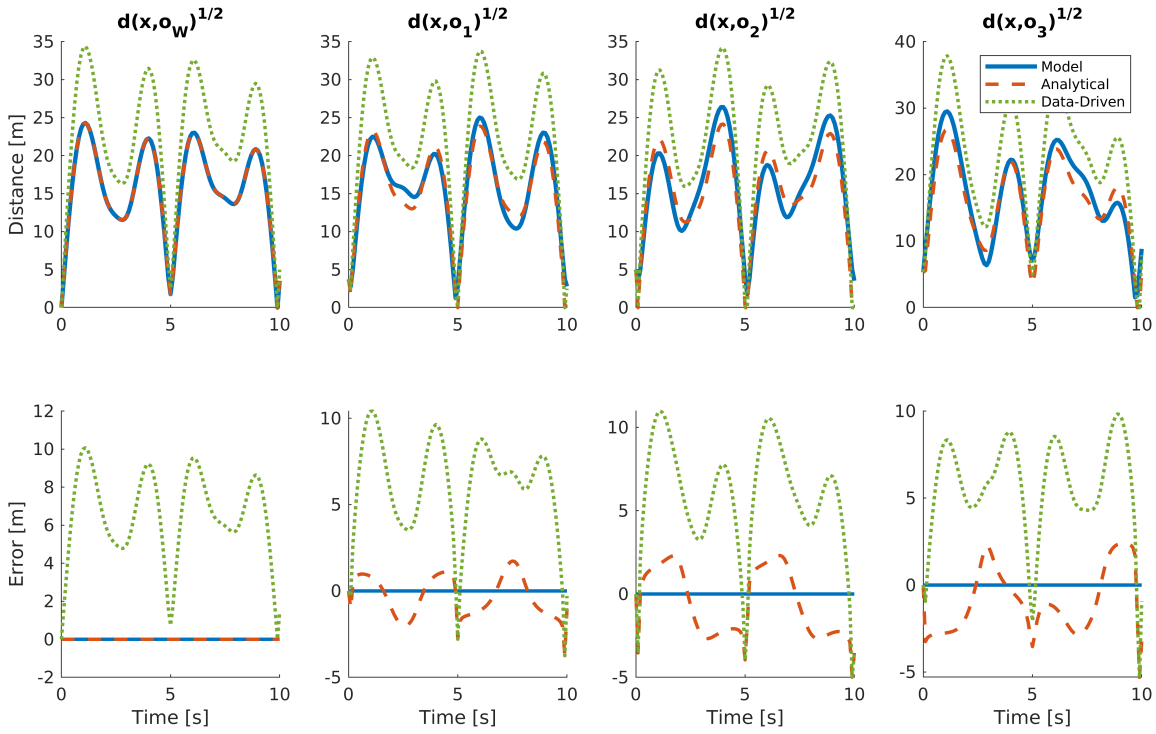

Fig. 4. Propagation of $\sqrt{d(x_n, o_n)}$ terms over time.

result of the error shown in Figure 3. Unfortunately it does not seem as though the distance deviates by a constant gain as observed for the bilinear terms. This points towards the possibility that the fault lies in some combination of the bilinear gains, but is unclear at this time. Possible solutions to this issue will be futher discussed in Section VI.

The analytical Koopman operator has slightly higher accuracy. It is capable of approximating the distance from each obstacle, at all times, with a maximum error of $5.30[m]$ from the fourth obstacle. This error may be considered too large for linear constraint implementation, but is appreciably better than its data-driven alternative which has a peak error of $10.97[m]$ w.r.t. the third obstacle.

VI. Conclusion

The purpose of this paper was to explore the analytically and data-driven Koopman operator and its applications to controlled systems and obstacle distance modeling. While both alternatives were successful in modeling the linear propagation of the state, only the analytical operator was capable of following the bilinear combination of terms accurately.

Both methods struggled in following the squared L_2-norm distance of the robot from each of the obstacles. The analytical operator was capable of somewhat matching the true-model propagation, but showed a consistent deviation from the reference value for all obstacles except the wall, whose center-point was placed at the origin. The data-driven alternative incorporated an aggressive level of noise, making it unrealistic for integration with any type of controller.

The inability for the analytical operator to match the distance for only the obstacles that were not placed at the origin leads to the belief that the center point of each obstacle is not being properly stored in the observable functions. Incorporating a constant term into the observables was considered, but was later excluded because of the reasoning derived in (27). In the future, it may be a better approach to disregard this assumption and calculate the distance using all terms as stated in (26).

While it is not shown here, interesting behavior was demonstrated when the bilinear terms of the analytical operator were given a constant gain. When the gain was set to 1, the operator behaves as seen in the figures shown, but when it was set to 2, it was an exact match of the data-driven approach. This behavior is also true for the distance propagation. The reasoning behind this behavior is unclear but must be addressed before the Koopman operator can be used in any real-time system.

Although the analytical operator was demonstrated here for comparison, it is often difficult, even impossible, to solve for the applicable coefficients in large nonlinear systems. The purpose of the Koopman operator is to be capable of replicating complex systems when a model would be otherwise difficult to compute or is not available at all. With that said, future work for this project should focus in developing more consistent methods of deriving the data-driven Koopman operator with the intent of alleviating the difficulties discussed in Section V. With a more accurate Koopman operator, the user could ideally incorporate the linearized dynamical system into most linear model-based controllers, making for a fast optimal controller.

References

[1] B. O. Koopman. "Hamiltonian Systems and Transformation in Hilbert Space". In: *Proceedings of the National Academy of Sciences* 17.5 (May 1931). Publisher: Proceedings of the National Academy of Sciences, pp. 315–318. DOI: 10.1073/pnas.17.5.315. URL: https://www.pnas.org/doi/abs/10.1073/pnas.17.5.315 (visited on 11/16/2022).

[2] Marko Budišić, Ryan M. Mohr, and Igor Mezić. "Applied Koopmanism". In: *Chaos* 22.4 (Dec. 2012). arXiv:1206.3164 [nlin], p. 047510. ISSN: 1054-1500, 1089-7682. DOI: 10.1063/1.4772195. URL: http://arxiv.org/abs/1206.3164 (visited on 10/05/2022).

[3] Matthew O. Williams, Ioannis G. Kevrekidis, and Clarence W. Rowley. "A Data–Driven Approximation of the Koopman Operator: Extending Dynamic Mode Decomposition". en. In: *J Nonlinear Sci* 25.6 (Dec. 2015), pp. 1307–1346. ISSN: 1432-1467. DOI: 10.1007/s00332-015-9258-5. URL: https://doi.org/10.1007/s00332-015-9258-5 (visited on 10/06/2022).

[4] Sebastian Peitz, Samuel E. Otto, and Clarence W. Rowley. "Data-Driven Model Predictive Control using Interpolated Koopman Generators". In: *SIAM J. Appl. Dyn. Syst.* 19.3 (Jan. 2020). Publisher: Society for Industrial and Applied Mathematics, pp. 2162–2193. DOI: 10.1137/20M1325678. URL: https://epubs.siam.org/doi/abs/10.1137/20M1325678 (visited on 10/11/2022).

[5] Ian Abraham, Gerardo De La Torre, and Todd D. Murphey. "Model-Based Control Using Koopman Operators". In: *Robotics: Science and Systems XIII*. arXiv:1709.01568 [cs, math]. July 2017. DOI: 10.15607/RSS.2017.XIII.052. URL: http://arxiv.org/abs/1709.01568 (visited on 10/05/2022).

[6] Daniel Bruder, Xun Fu, and Ram Vasudevan. "Advantages of Bilinear Koopman Realizations for the Modeling and Control of Systems With Unknown Dynamics". In: *IEEE Robotics and Automation Letters* 6.3 (July 2021). Conference Name: IEEE Robotics and Automation Letters, pp. 4369–4376. ISSN: 2377-3766. DOI: 10.1109/LRA.2021.3068117.

[7] Steven L. Brunton et al. "Koopman Invariant Subspaces and Finite Linear Representations of Nonlinear Dynamical Systems for Control". en. In: *PLOS ONE* 11.2 (Feb. 2016). Publisher: Public Library of Science, e0150171. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0150171. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0150171 (visited on 10/17/2022).

[8] Guillermo A. Castillo et al. "Hybrid Zero Dynamics Inspired Feedback Control Policy Design for 3D Bipedal Locomotion using Reinforcement Learning". en. In: *arXiv:1910.01748 [cs]* (Oct. 2019). arXiv: 1910.01748. URL: http://arxiv.org/abs/1910.01748 (visited on 03/02/2022).