Controlling the Shape of Soft Robots Using the Koopman Operator

Ajai Singh, Jiefeng Sun, and Jianguo Zhao

***Abstract*— In nature, animals with soft body parts demonstrate remarkable control over their shape, such as an elephant trunk wrapping around a tree branch to pick it up. However, most research on robotic manipulators focuses on controlling the end effector, partly because the manipulator's arm is rigidly articulated. With recent advances in soft robotics research, controlling a soft manipulator into many different shapes will significantly improve the robot's functionality, such as medical robots morphing their shape to navigate the digestive system and deliver drugs to specific locations. However, controlling the shape of soft robots is challenging due to their highly nonlinear dynamics that are computationally intensive. In this paper, we leverage a physics-informed, data-driven approach using the Koopman operator to realize the shape control of soft robots. We simulate the dynamics of a soft manipulator using a physics-based simulator (PyElastica) to generate the input-output data, which is then used to identify an approximated linear model based on the Koopman operator. We then formulate the shape-control problem as a convex optimization problem that is computationally efficient. Our linear model is over 12 times faster than the physics-based model in simulating the manipulator's motion. Further, we can control a soft manipulator into different shapes using model predictive control. We envision that the proposed method can be effectively used to control the shapes of soft robots to interact with uncertain environments or enable shape-morphing robots to fulfill diverse tasks. This paper is complemented with a video.**

I. Introduction

Rigid-bodied robots have long been the heart of various industries such as manufacturing, but soft robots made from soft materials have recently emerged in robotics research [1]. Unlike rigid ones, soft robots can exploit their inherent mechanical compliance to interact with humans or external environments, leading to many applications such as manipulation, locomotion, medical devices, etc [2], [3].

The soft robotics community has begun to explore how a robot's shape can enhance its functional capabilities, drawing inspiration from biological organisms [4]. Various living organisms can change their body shape to adapt to different environments and respond to external stimuli. For example, an octopus can squeeze its body through gaps much smaller than its body size [5], and moth larvae can curl up to roll away from predators [6]. Inspired by biological organisms, researchers have developed robots that utilize different shapes for distinct functions. Shah *et al.* investigated how a soft robot can use different shapes for crawling or rolling in different environments [7]. Hwang *et al.* leveraged a novel kirigami composite to develop a morphing drone that can autonomously transform from ground to aerial vehicle [8]. Other recent research, including [9], shows how shape morphing can enhance a robot's functionality by reaching a desired position while avoiding obstacles.

Controlling the shape of a soft robot is challenging due to its highly nonlinear dynamics. Researchers have developed various physics-based models using methods such as Cosserat Rod theory [9], and model reduction method [10], among many others [11]. Although such models can achieve high-fidelity simulation for various soft robots [12], they generally require extensive computational time, making them unsuitable for shape control.

In this paper, we aim to leverage existing physics-based models to develop computationally efficient data-driven models for controlling the shape of soft robots. We use the open-source simulation software PyElastica [9] to generate sufficient input-output data for a soft robot. Using the data, we establish a data-driven model based on Koopman operator theory [13] to obtain a finite-dimensional approximation of the soft robot [14]. The Koopman operator can represent a nonlinear dynamical system with a finite-dimensional linear model to approximate the original dynamics of a soft robot. With such a linear model, we can directly use existing control methods such as model predictive control (MPC) to control a soft robot's shape.

Researchers have recently used the Koopman operator theory to control soft robots, as demonstrated in various studies [14]–[16]. In particular, the work presented in [14] shows promising results in controlling the robot's tip to trace a desired trajectory, while [15] used the Koopman operator to model a soft robotic swimmer. However, controlling a soft robot's shape differs from existing problems (e.g., tip position control) as the shape is specified by continuous curves or surfaces. Therefore, our work contributes an additional application of Koopman operator-based system identification and control for the shape control of soft robots. Specifically, *the contribution of this paper* is to develop a data-driven method to control a soft robot's shape by leveraging existing physics-based models and Koopman operator theory.

The rest of the paper is organized as follows. Section II formulates the shape control problem for a soft robot. In Section III, we provide the mathematical underpinnings of the Koopman operator, its approximation from data, and the algorithm used for system identification using the Koopman operator approach. Section IV presents MPC used with the identified linear system. Section V details the simulation setup, including the generation of the input-output data for a soft robot using PyElastica [9]. Finally, section VI presents

This work is partially supported by the National Science Foundation under Grant CMMI-2126039.

Ajai Singh, Jiefeng Sun, and Jianguo Zhao are with the Department of Mechanical Engineering, Colorado State University, Fort Collins, CO 80523, USA. Ajai.Singh@colostate.edu, J.Sun@colostate.edu, Jianguo.Zhao@colostate.edu

the results of system identification and shape control.

II. Shape Control Problem

In this section, we formulate the shape control problem using a general soft manipulator. We will show how to use this framework to solve the shape control problem in subsequent sections.

Given a soft manipulator of length L, we divide it into $N-1$ segments with equal length. The shape for the i-th ($i = 2, \ldots, N$) segment is specified by the section at its top. At a discrete time step t_k, we use $g_i(t_k) \in SE(3)$ to represent the top section's position and orientation in the inertia frame as shown in Fig. 1.

$$g_i(t_k) = \begin{bmatrix} R_i(t_k) & p_i(t_k) \\ 0 & 1 \end{bmatrix} \tag{1}$$

where $R_i(t_k) \in SO(3)$ represents the orientation, and $p_i(k) \in \mathbb{R}^3$ presents the position for the section's centroid. Denote the i-th segment as L_i. For each segment, we assume we can apply an actuation input u, such as forces/torques at each segment's top or torques generated using spline functions. In practice, these actuation inputs might be generated by artificial muscles embedded within soft materials [17]. In our setup, we apply continuous torque to the system generated by spline functions in PyElastica. With such a setup, the shape of the soft manipulator can be approximated by $g_i(t_k) \in SE(3)$ at time step t_k.

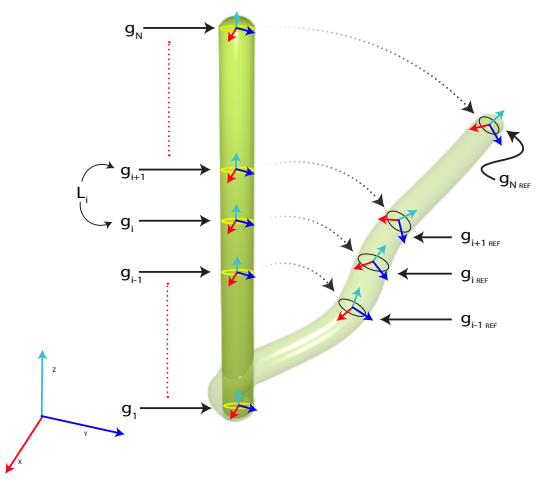

Fig. 1. Illustration for the shape control problem for a soft manipulator divided into $N-1$ equal length segments with the top of each segment shown as a yellow cross-section. The manipulator shown in solid green color is its initial shape ($t = 0$), and the shape shown in faded green shows the target shape.

Given a desired shape for the soft manipulator represented by $g_{i_{ref}}$ ($i = 1, \ldots, N-1$), the shape control problem can be formulated as finding the control input $u(t_k)$ that minimizes the distance between $g_i(t_k)$ and $g_{i_{ref}}$. Note that the distance in $SE(3)$ can be defined separately for the position and orientation, with the Euclidean distance for position and the geodesic distance for orientation. This paper will focus on a simplified problem by only considering the position distance. In this case, the problem can be formulated as follows:

$$\underset{u(t_k)}{\text{minimize}} \quad \sum_{i=1}^{N-1} ||p_{i_{ref}} - p_i(t_k)||_2^2 \tag{2a}$$

subject to

$$x(t_{k+1}) = f(x(t_k), u(t_k)), \tag{2b}$$

$$h(x(t_k), u(t_k)) \leq 0 \tag{2c}$$

where $p_{i_{ref}} \in \mathbb{R}^3$ is the desired position for the i^{th} segment. $x(t_k) \in \mathbb{R}^n$ is the combination of all p_i at time instant t_k, but in general, it can include all the states used to describe the dynamics of the soft manipulator. $u(t_k) \in \mathbb{R}^m$ is the control input of the system at t_k. $h(x(t_k), u(t_k)) \leq 0$ are the various constraints applied to the state and control variables, which are commonly known as polyhedral constraints.

Generally, the dynamics for a soft robot (i.e., $x(t_{k+1}) = f(x(t_k), u(t_k))$) is highly nonlinear, involving complicated physics-based models [11]. Such models can only be solved numerically with considerable computation time, preventing them from real-time shape control of soft robots. Inspired by recent work on using the data-driven method to identify approximated models from either numerical or experimental data for controlling soft robots for manipulation [14]–[16], we aim to obtain a data-driven model using Koopman operator theory. Then we use the model to control the shapes of soft robots.

III. System Identification using Koopman Operator Theory

Given the complicated dynamics of a soft robot, we will use Koopman operator theory to directly identify a computationally efficient linear model using the input-output data generated by physics-based models (e.g., PyElastica [9]). In this section, we briefly review the preliminaries for Koopman operators.

Given a nonlinear dynamical system, the Koopman operator first maps the states of the original system using scalar functions (also called observables) of the states into lifted space with new state variables. The new system in the lifted space with the new state variables is generally an infinite dimensional linear system. Unlike the linearization about a point that becomes inaccurate when operating away from the linearizing point, the Koopman operator describes the evolution of the scalar observable throughout the state space in a linear fashion. This makes the Koopman operator approach preferable when realizing linear representation of nonlinear systems [18].

We briefly review the Koopman operator framework for control systems, as described in [14]. Assume a discrete nonlinear dynamical system given by:

$$\begin{aligned} x(t_{k+1}) &= f(x(t_k)) \\ y(t_k) &= g(x(t_k)) \end{aligned} \tag{3}$$

where $x(t_k)$, $x(t_{k+1}) \in \mathbb{R}^n$ are the state vector at time instant t_k, t_{k+1} respectively, $y(t_k) \in \mathbb{R}^r$ is the output of the system at t_k. To simplify notations, we use $x(t_k)$ and x_{t_k} interchangeably in the following discussions.

To map the state x to a lifted space, we use a basis or observation function $\phi(x(t_k)) : \mathbb{R}^n \to \mathbb{R} \in \mathcal{F}$, where $\mathcal{F}$ is the space of all basis functions. The Koopman operator $\mathcal{K} : \mathcal{F} \to \mathcal{F}$ is defined as:

$$(\mathcal{K}\phi)(x_{t_k}) = \phi(f(x_{t_k})) = \phi(x_{t_k+1}) \tag{4}$$

which means the Koopman operator simply updates the observation of the state in the lifted space from the current time step to the next step.

$\mathcal{K}$ is generally an infinite dimensional linear operator, but we can use a finite subspace to approximate it. Let the finite dimensional approximation of $\mathcal{K}$ be $\bar{\mathcal{K}}$. $\bar{\mathcal{K}}$ operates on $\bar{\mathcal{F}} \subset \mathcal{F}$ which is the subspace spanned by a finite set of basis functions. $\bar{\mathcal{K}}$ can be obtained using Extended Dynamic Mode Decomposition (EDMD) as discussed in [19], and this approximation is achieved by solving the following optimization problem

$$\underset{A}{\text{minimize}} \quad \sum_{k=0}^{K-1} ||\psi(x_{t_{k+1}}) - A\psi(x_{t_k})||_2^2 \tag{5}$$

where $\psi(x) = [\psi_1(x)\ , \psi_2(x)\ , \psi_3(x)\ , \ldots,\ \psi_{n_b}(x)]^\top$ with $\{\psi_i : \mathbb{R}^n \to \mathbb{R}\}_{i=1}^{n_b}$ represents the n_b basis functions, $A \in \mathbb{R}^{n_b \times n_b}$ is the finite dimensional approximation of the Koopman operator, n_b is the total number of basis functions, K is the cardinality of the data-set given by $\mathcal{D} = \{x_{t_k}\}_{k=0}^{K}$ and $^\top$ is the transpose operator.

Solving the minimization problem of Eq. (5), we can represent the nonlinear dynamical system given by Eq. (3) as the following discrete linear dynamical system

$$\begin{aligned} z(t_{k+1}) &= Az(t_k) \\ \tilde{y}(t_k) &= Cz(t_k) \end{aligned} \tag{6}$$

where $z(t_k) = \psi(x(t_k)) \in \mathbb{R}^{n_b}$ and $\tilde{y}(t_k)$ is the output. The matrix $C \in \mathbb{R}^{r \times n_b}$ is obtained just like A by solving

$$\underset{C}{\text{minimize}} \quad \sum_{k=1}^{K} ||y(t_k) - C\psi(x(t_k))||_2^2 \tag{7}$$

Similarly, for a nonlinear dynamical system with control inputs, the methodology discussed can be used. Consider a discrete nonlinear dynamical system given by

$$\begin{aligned} x(t_{k+1}) &= f(x(t_k), u(t_k)) \\ y(t_k) &= g(x(t_k)) \end{aligned} \tag{8}$$

where $u(t_k) \in \mathbb{R}^m$ is the control input. Then in this case to approximate the Koopman Operator, the minimization problem in Eq. (5) changes to

$$\underset{A,\,B}{\text{minimize}} \quad \sum_{k=0}^{K-1} ||\psi(x(t_{k+1})) - (A\psi(x(t_k)) + Bu(t_k))||_2^2 \tag{9}$$

Thus, by solving the minimization problem of Eq. (9), the finite approximation of the Koopman operator is approximated by A and B, and it acts as one step predictor of the nonlinear dynamical system described by Eq. (8). The minimization problem for the output equation, i.e., for C matrix remains the same as given by Eq. (7) We will use EDMD [14], [19] to construct the linear model of a soft robot since EDMD is a data-driven method that approximates the leading Koopman eigenfunctions, eigenvalues, and modes whereas other data-driven methods such as generalized Laplace analysis, Ulam Galerkin method, and Dynamic Mode Decomposition cannot approximate the three quantities [19]. One could also use neural networks to approximate the three quantities, but neural networks require a lot of training data and tuning.

IV. Model Predictive Control for Koopman operator based linear system

In this section, we briefly review model predictive control (MPC) and then show how MPC can be used alongside the Koopman operator for shape control.

Many model-based controllers have been developed for soft robots [20], [21], but most rely on simplifying assumptions and are primarily designed for static control. While such controllers have been proven to be very efficient for the static control of soft robotic manipulators, they are unsuitable for dynamic control. Dynamic control can be achieved by supplementing a piece-wise constant curvature model with data-driven trajectory optimization. However, the downside of this approach is that the training tends to be task-specific [14]. More realistic physics-based models have also been proposed, but they are computationally expensive.

From Section III, the Koopman operator is used to approximate a linear system of a nonlinear dynamical system from data. In [14], the authors constructed an MPC controller from a linear Koopman representation of a nonlinear dynamical system. We use a similar approach to construct an MPC for shape control of a soft manipulator. Since the identified model is linear, the MPC optimization offers computational advantages over nonlinear ones as the MPC optimization problem is convex, which can be solved very efficiently with any method for convex optimization. To implement the Koopman-based MPC, we first define the objective function as follows:

$$\begin{aligned} J = {} & z(t_{N_h})^\top Q(t_{N_h}) z(t_{N_h}) + \\ & \sum_{i=0}^{N_h-1} \{z(t_i)^\top Q(t_i) z(t_i) + u(t_i)^\top R(t_i) u(t_i)\} \end{aligned}$$

where $N_h \in \mathbb{N}$ is the prediction horizon, $Q(t_i) \in \mathbb{R}^{n_b \times n_b}$, $R(t_i) \in \mathbb{R}^{m \times m}$ are positive semidefinite matrices. Here Q and R are constant matrices or vectors also known as weight or cost matrices. Then we can iteratively solve a convex quadratic program over a receding horizon as:

$$\underset{u(t_i)}{\text{minimize}} \quad J \tag{10a}$$

subject to

$$z(t_{i+1}) = Az(t_i) + Bu(t_k), \tag{10b}$$

$$z(0) = \psi(x(t_k)) \tag{10c}$$

Every time the optimization routine is called, the predictions need to be set to the current lifted state $\psi(x(t_k))$. While

the size of the cost and constraint matrices depend on the dimension of the lifted state n_b, [22] shows that these can be rendered independent of n_b by transforming the problem into its so-called dense-form [14]. We use the above framework to solve the shape control problem presented in section II.

V. Simulation Setup

In this paper, we use PyElastica, an open-source, physics-based simulator for soft robots based on the Cosserat rod theory, to generate the data for the Koopman operator. We choose PyElastica due to its relative accuracy, accessibility, and ease of setup. When using PyElastica, we set up a simulation that requires the user to define a system of rods, establish initial and boundary conditions on the rod, run the simulation, and collect data for post-processing. The detailed process can be found in [23].

The physical parameters for our manipulator are listed in Table I. The robot base is fixed as the boundary condition, and actuation is achieved by applying torques distributed along the arm's length. The torques are decomposed into orthogonal functions in local normal and bi-normal directions, determined by continuous splines with N independent control points. We do not apply torque in the axial direction to twist the robot.

TABLE I

List of physical parameters for the simulation setup

Physical Parameter	Value	Unit
Number of tracking points	6	N/A
Starting Position of the rod (vector)	[0.0, 0.0, 0.0]	N/A
Direction in which rod extends(vector)	[0.0, 0.0, 1.0]	N/A
Normal vector of rod	[1.0, 0.0, 0.0]	N/A
Length of rod	1.00	meter (m)
Radius of the tip	0.05	meter (m)
Radius of the base	0.05	meter (m)
Density of the rod	1.0×10^3	kg/m^3
Energy dissipation constant	10.00	N/A
Young's Modulus	1.00×10^7	Pa
Poisson's Ration	0.50	N/A

*The vectors are defined with respect to the standard right-hand coordinate system.

Note that the spline control points for normal and bi-normal directions are distinct, requiring two splines. To generate these two splines for each simulation case, we randomly generate two torques in normal and bi-normal directions for each control point. Initially, the robot assumes a straight and upright shape. In the simulation, the two torque splines are kept constant as a step input for the system. We use the position Verlet integration algorithm as the time stepping algorithm with a time step $\Delta t = 0.0001$ s such that we can capture the transient behavior of the soft manipulator accurately. We then simulate the system for 20 s to ensure that the soft manipulator has reached a steady state.

To construct a linear model of the soft robotic system using the Koopman operator, we collect data for system identification by simulating the robot with random inputs. We simulate the manipulator for 30 cases. For each simulation case, we collect 2000 snapshots with a sampling time of $T_s = 0.01$ s. These data sets are used for approximating the Koopman Operator. The system identification is conducted by lifting the collected snapshots using the basis function with delays defined in section III. We use first-order polynomials with delay $d = 1$ as the basis function and then performed least square regression as shown in Eqs. (9) and (7) to obtain the $A \in \mathbb{R}^{47\times47}$, $B \in \mathbb{R}^{47\times10}$, and $C \in \mathbb{R}^{18\times47}$ matrices, respectively [14]. Note that we choose first-order polynomials as the basis functions because they can generate more accurate predictions compared with other choices such as higher-order polynomials, radial basis functions, etc. For the problem, we have selected 47 basis functions with degree 1. For the 47 functions, the first 18 are defined as the output of the system, the next 28 are delay coordinates, and a constant 1 is added so that we do not lift the inputs. One might consider using higher-degree polynomials or more basis functions to minimize the prediction error $\dot{\phi}(x) = A\phi(x)$. However, increasing the polynomial order generally leads to more functions in ϕ, and thus more derivatives $\dot{\phi}$ must be expressed by ϕ. Consequently, the increase in polynomial order causes the derivatives $\dot{\phi}$ to grow in complexity, making it more challenging for $\dot{\phi}$ to be expressed by ϕ [24].

VI. Results

In this section, we first quantify the identified linear model in terms of accuracy and speed by comparing it with the physics-based model. We demonstrate that it can realize tip and shape control much faster with the identified linear model and an MPC.

A. The Accuracy and Speed of the Identified Linear Model

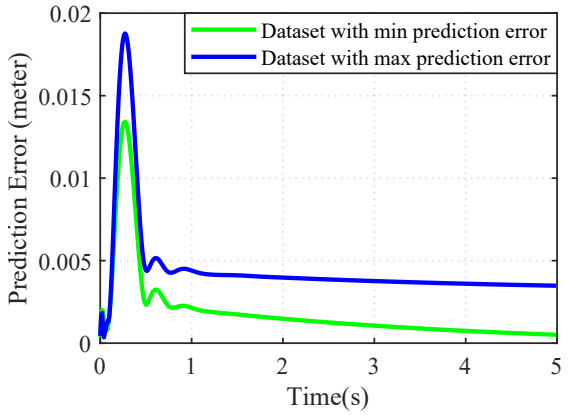

Fig. 2. Prediction error plots for datasets having the least (green) and maximum (blue) prediction error. The error at each time step was calculated using Eq. (11). For clarity of the plot, only 2 of 6 prediction errors are shown.

The accuracy of the Koopman model is estimated by calculating the error at time step t_i defined as:

$$Err(t_i) = \frac{1}{L}\sqrt{\frac{1}{N_x}\sum_{j=1}^{N_x}(x^j(t_i) - x^{j_{ref}}(t_i))^2} \qquad (11)$$

where $L = 1$ is the length of the soft manipulator, $N_x = 18$ is the number of states, $x^j(t_i)$, $x^{j_{ref}}(t_i)$ are the j^{th} state vector at time t_i approximated by Koopman based model and actual physics-based model, respectively. This accuracy

TABLE II

MEAN TIME COMPARISON OF PHYSICS AND KOOPMAN-BASED MODEL

# Time steps	Physics-based	Koopman-based
1	1.7 ms ± 353 µs	188 µs ± 29.5 µs
10	2.21 ms ± 112 µs	159 µs ± 50.7 µs
100	9.14 ms ± 386 µs	696 µs ± 23.5 µs
1000	60.8 ms ± 1.13 ms	4.75 ms ± 111 µs
10000	568 ms ± 7.2 ms	44.6 ms ± 640 µs
100000	5.6 s ± 49.3 ms	435 ms ± 5.79 ms

* The time shown is the mean time per loop.

depends on the number of basis functions and the types of basis functions. We validate the identified linear model against 6 different data sets with random inputs generated with the same method mentioned in section V.

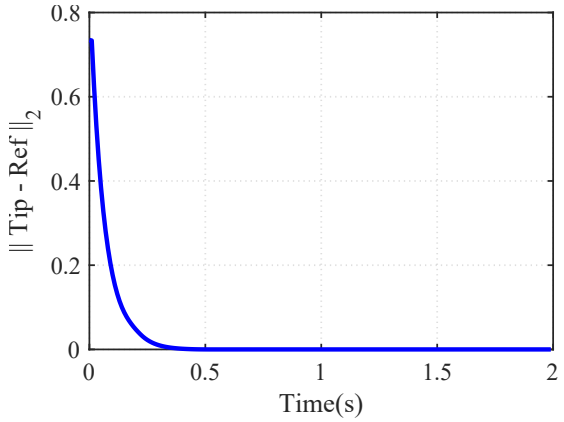

Fig. 3. Results for the control of the tip of the manipulator. The plot shows the Euclidean distance between the tip and the reference set point.

The linear model generated by the Koopman operator has a maximum mean RMSE of 35E-4 meters (obtained by taking the average of all the Err over the entire simulation time) and a minimum mean RMSE of 6.78E-4 across all states. Fig. 2 shows the prediction error for two datasets between the Koopman-based linear model and the physics-based model. The results for these two datasets are chosen because the error curves for all the other four cases are between these two error curves.

The Koopman-based model is more time-efficient than the Physics-based model. We measure the average time for multiple simulations while accounting for interference from other operating system tasks. For the Koopman-based model, we run it seven times with 100 loops, simulating the model for a specific time step Δt. The physics-based model is run seven times with only 10 loops due to the much longer time required for each loop. The time comparison tests are run on a computer with 16 GB Random Access Memory (RAM) and a 2.6 GHz CPU. Table II shows the comparison between the mean time taken by both models to run for a finite number of time steps. We can see the Koopman-based linear model is over 12 times faster than the physics-based model, except in the case of a single step.

Controlling a soft robot's tip represents a special case of the shape control problem proposed in section II. Unlike controlling a soft robot's shape, which requires controlling multiple points, we control only one point, i.e., the tip of the manipulator. This problem can also be called set point tracking in three-dimensional space. To address this problem, we use the MPC with a prediction horizon $N_h = 25$ steps. The desired reference set points are chosen as [0.1, 0.2, 0.3]. To demonstrate the control of the tip, we move the tip from the rest position, i.e., from [0, 0, 1] to [0.1, 0.2, 0.3] where the elements of the vector are the x, y, z coordinates of the tip. The result is shown in Figure 3, which shows the Euclidean distance between the robot's tip and the reference point over time. The plot clearly shows that the MPC controller can move the tip to the desired location within 0.5 s.

B. Shape Control with Koopman-based MPC

We further demonstrate the shape control of a soft manipulator by controlling it to form different shapes, specifically three letters: 'C', 'S', and an inverted 'U'. The linear MPC controller derives the optimal control input over the prediction horizon of $N_h = 25$. The controller has a single constraint i.e. the dynamics of the soft manipulator for various shapes. Its objective is to move tracked points to the desired location. A cost function is chosen to penalize the distance between reference and robot points. We generate reference shapes using the physics-based model with the shooting method, which are then used as references for the MPC controller. Note that since the bottom of the manipulator is rigidly fixed to the ground, the segment close to the ground needs to resemble a vertical shape due to the spline used to interpolate the torques in PyElastica. If we do not consider this segment, however, the desired shapes are indeed similar to the letters 'C' and 'S'.

The results for the three shapes are illustrated in Fig. 4, where we plot the robot's final shape and the desired shape. From the figure, we can see the robot can accomplish the desired shape. To quantify the error between the final and desired shape, we plot the RMSE in meters for different tracking points for the morphed shapes in Fig. 5. Root Mean Squared Error is calculated as the Euclidean distance between the final position generated by the controller for a tracking point and the corresponding reference point. As seen from the error in Fig. 5, the final position error can be quite small ($< 5\%$ with respect to the manipulator's length). Note that we only used points for the shape control for the simulations, but we can also add orientations into the formulation of the shape control problem.

VII. CONCLUSION

In this paper, we employ a data-driven method based on the Koopman operator to establish an approximate linear model of a soft manipulator using the input-output data from an accurate yet computationally inefficient physics-based model. This linear model is both accurate and over 12 times faster than the physics-based model. Combining with a linear MPC, we successfully control a soft robot's shape, a task that is highly challenging when using a physics-based model. Our future work will extend this method to control a soft robot made from a combination of several rods

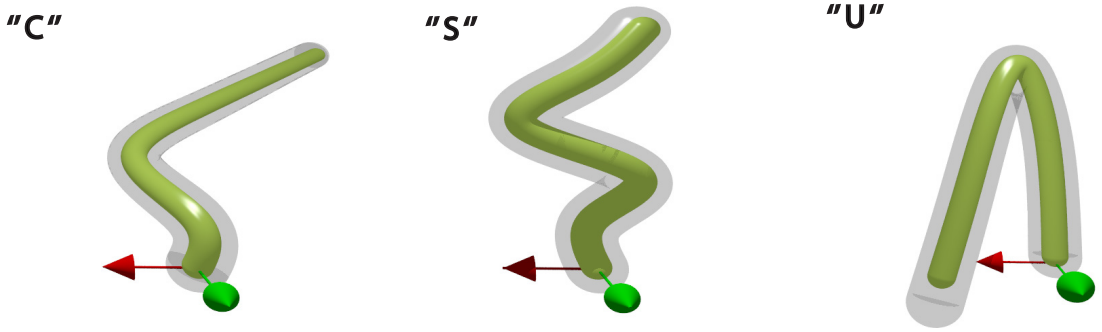

Fig. 4. Results for controlling the soft robot to three different shapes. The object in solid green is the shape of the actual soft robot, and the gray envelope is the reference shape. Note that the reference shape has been scaled to 10% of the actual reference for better visualizations.

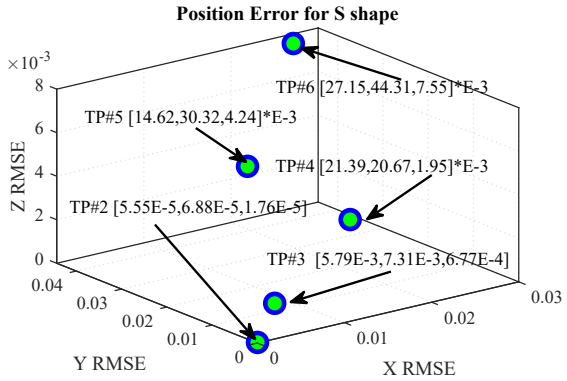

Fig. 5. Position error of the tracking points (TP) corresponding to the S shape acquired by the soft manipulator as shown in Figure 4. The vector following each TP shows RMSE for X, Y, and Z for a particular TP. (Note that the error for TP 1 was always 0 as it was static, hence it was ignored in the error plot.)

to form a polyhedron (e.g., tetrahedron, Octahedron, etc.) to generate more diverse shapes, which can be potentially experimentally verified through soft robot prototypes driven by artificial muscles.

REFERENCES

[1] D. Rus and M. T. Tolley, "Design, fabrication and control of soft robots," *Nature*, vol. 521, no. 7553, pp. 467–475, 2015.

[2] M. Cianchetti, C. Laschi, A. Menciassi, and P. Dario, "Biomedical applications of soft robotics," *Nature Reviews Materials*, vol. 3, no. 6, pp. 143–153, 2018.

[3] P. Polygerinos, N. Correll, S. A. Morin, B. Mosadegh, C. D. Onal, K. Petersen, M. Cianchetti, M. T. Tolley, and R. F. Shepherd, "Soft robotics: Review of fluid-driven intrinsically soft devices; manufacturing, sensing, control, and applications in human-robot interaction," *Advanced Engineering Materials*, vol. 19, no. 12, p. 1700016, 2017.

[4] D. Shah, B. Yang, S. Kriegman, M. Levin, J. Bongard, and R. Kramer-Bottiglio, "Shape changing robots: bioinspiration, simulation, and physical realization," *Advanced Materials*, vol. 33, no. 19, p. 2002882, 2021.

[5] K. J. Quillin, "Kinematic scaling of locomotion by hydrostatic animals: ontogeny of peristaltic crawling by the earthworm lumbricus terrestris," *Journal of Experimental Biology*, vol. 202, no. 6, pp. 661–674, 1999.

[6] R. Armour and J. Vincent, "J bionic eng. 2006, 3, 195-208; c) l. van griethuijsen, b. trimmer," *Biol. Rev*, vol. 89, pp. 656–670, 2014.

[7] D. S. Shah, J. P. Powers, L. G. Tilton, S. Kriegman, J. Bongard, and R. Kramer-Bottiglio, "A soft robot that adapts to environments through shape change," *Nature Machine Intelligence*, vol. 3, no. 1, pp. 51–59, 2021.

[8] D. Hwang, E. J. Barron III, A. T. Haque, and M. D. Bartlett, "Shape morphing mechanical metamaterials through reversible plasticity," *Science Robotics*, vol. 7, no. 63, p. eabg2171, 2022.

[9] N. Naughton, J. Sun, A. Tekinalp, T. Parthasarathy, G. Chowdhary, and M. Gazzola, "Elastica: A compliant mechanics environment for soft robotic control," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3389–3396, 2021.

[10] O. Goury and C. Duriez, "Fast, generic, and reliable control and simulation of soft robots using model order reduction," *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1565–1576, 2018.

[11] G. Mengaldo, F. Renda, S. L. Brunton, M. Bächer, M. Calisti, C. Duriez, G. S. Chirikjian, and C. Laschi, "A concise guide to modelling the physics of embodied intelligence in soft robotics," *Nature Reviews Physics*, pp. 1–16, 2022.

[12] J. Sun and J. Zhao, "Modeling and simulation of soft robots driven by embedded artificial muscles: an example using twisted-and-coiled actuators," in *American Control Conference (ACC)*, 2022, pp. 2911–2916.

[13] 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. [Online]. Available: https://www.pnas.org/doi/abs/10.1073/pnas.17.5.315

[14] D. Bruder, X. Fu, R. B. Gillespie, C. D. Remy, and R. Vasudevan, "Data-driven control of soft robots using koopman operator theory," *IEEE Transactions on Robotics*, vol. 37, no. 3, pp. 948–961, 2020.

[15] M. L. Castaño, A. Hess, G. Mamakoukas, T. Gao, T. Murphey, and X. Tan, "Control-oriented modeling of soft robotic swimmer with koopman operators," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, 2020, pp. 1679–1685.

[16] E. Kamenar, N. Ćrnjarić-Žic, D. Haggerty, S. Zelenika, E. W. Hawkes, and I. Mezić, "Prediction of the behavior of a pneumatic soft robot based on koopman operator theory," in *2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO)*. IEEE, 2020, pp. 1169–1173.

[17] J. Sun, B. Tighe, Y. Liu, and J. Zhao, "Twisted-and-coiled actuators with free strokes enable soft robots with programmable motions," *Soft robotics*, vol. 8, no. 2, pp. 213–225, 2021.

[18] 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.

[19] 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.

[20] C. Della Santina, A. Bicchi, and D. Rus, "On an improved state parametrization for soft robots with piecewise constant curvature and its use in model based control," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1001–1008, 2020.

[21] C. Della Santina, C. Duriez, and D. Rus, "Model based control of soft robots: A survey of the state of the art and open challenges," *arXiv preprint arXiv:2110.01358*, 2021.

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

[23] X. Zhang, F. Chan, T. Parthasarathy, and M. Gazzola, "Modeling and simulation of complex dynamic musculoskeletal architectures," *Nature Communications*, vol. 10, no. 1, pp. 1–12, 2019.

[24] V. Cibulka, T. Haniš, and M. Hromčík, "Data-driven identification of vehicle dynamics using koopman operator," in *2019 22nd International Conference on Process Control (PC19)*. IEEE, 2019, pp. 167–172.