CORRECTION POSTED 30 OCTOBER 2023

SOFT ROBOTS

Control of soft robots with inertial dynamics

David A. Haggerty[1]†*, Michael J. Banks[1]†, Ervin Kamenar[1,2]†, Alan B. Cao[3], Patrick C. Curtis[1], Igor Mezić[1], Elliot W. Hawkes[1]

Soft robots promise improved safety and capability over rigid robots when deployed near humans or in complex, delicate, and dynamic environments. However, infinite degrees of freedom and the potential for highly nonlinear dynamics severely complicate their modeling and control. Analytical and machine learning methodologies have been applied to model soft robots but with constraints: quasi-static motions, quasi-linear deflections, or both. Here, we advance the modeling and control of soft robots into the inertial, nonlinear regime. We controlled motions of a soft, continuum arm with velocities 10 times larger and accelerations 40 times larger than those of previous work and did so for high-deflection shapes with more than 110° of curvature. We leveraged a data-driven learning approach for modeling, based on Koopman operator theory, and we introduce the concept of the static Koopman operator as a pregain term in optimal control. Our approach is rapid, requiring less than 5 min of training; is computationally low cost, requiring as little as 0.5 s to build the model; and is design agnostic, learning and accurately controlling two morphologically different soft robots. This work advances rapid modeling and control for soft robots from the realm of quasi-static to inertial, laying the groundwork for the next generation of compliant and highly dynamic robots.

INTRODUCTION

The automation and robotics revolution has transformed manufacturing and heavy industry, leading to higher throughput, repeatability, and quality across numerous sectors (*1*, *2*). Unfortunately, robots are most often relegated to cages and isolated sections of manufacturing sites because of the inherent danger they present to human operators through their fast-moving, heavy, and rigid structures. Efforts toward allowing these robots to perform safely with human collaborators have focused on software control, but absolute guarantees of safety are not possible (*3*–*6*).

In contrast, soft robots are safe by construction because of their low stiffness and mass, but modeling and control of these systems are challenging (*7*–*12*). This is because of their inherent nonlinearity, their high dimensionality, and the imprecise measurement of their position in space. Past work has sought to overcome these obstacles through a variety of modeling methods, each of which constrains the design of control implementations. Most of these modeling approaches fall into two categories: analytical reduced order modeling (ROM) and machine learning (ML).

In soft robot ROM for control, the aim is to develop an analytical model based on simplifying assumptions such as (piecewise) constant curvature deformations (*13*–*16*). For an approximately constant-curvature system, this approach allows for the accurate prediction of dynamics given appropriate estimation of parameters. However, developing these analytical models is nontrivial and labor intensive, and each model applies only to the single system that was modeled. These models tend to be valid only in a neighborhood around the equilibrium point where the system has been linearized (*14*). Controllers based on ROM models have been applied to soft robots in the past, but these have yet to achieve the real-time control of fast, inertial motions (*17*, *18*).

In the ML modeling of soft robot dynamical behaviors, many neural net–based approaches exist. Most of this work focuses on the development of predictors using neural nets such as long short-term memory (*19*, *20*) or recurrent neural networks (*18*, *21*). These methodologies generate highly accurate predictors of the dynamics. However, training these systems has a high computational cost. Moreover, their structure is nonlinear, requiring specialized control algorithms (*10*). One example is a feedforward neural net controller, which has been successfully coupled to a model-free closed-loop controller and applied to a high-deflection, yet quasi-static soft arm (*22*, *23*). In addition, there are approaches that leverage a neural net–based dynamical model in a closed-loop control (*17*, *21*). However, neural net approaches to soft robot modeling have not yet resulted in the closed-loop control of high-speed, inertial, and nonlinear dynamics.

Koopman operator theory (KOT) (*24*) is an alternative modeling paradigm, introduced to the field of ML and data-driven modeling in the early 2000s (*25*, *26*). KOT-based ML has two qualities that make it an attractive strategy for soft robot control: It is data driven, eliminating the need for complicated analytical models, and it identifies a globally linear model, allowing for fast and efficient control design. The Koopman operator is a representation of a dynamical system in terms of the evolution of observables on a function space. Although the evolution of a dynamical system on state space may be nonlinear, its evolution in function space—described by the potentially infinite dimensional Koopman operator—is always linear. This is in contrast to a state-space linearization, which builds a linear approximation of the nonlinear dynamics only valid in a small region of the workspace. The Koopman methodology has been applied to control systems, with most of the work combining a Koopman operator approximation method, dynamic mode decomposition, with control (DMDc) (*27*–*32*). In particular, model predictive control (MPC) is commonly used (*33*). When DMDc and MPC are applied to soft robots (*34*–*37*), the control is accurate but

[1]Department of Mechanical Engineering, University of California, Santa Barbara, CA 93106, USA. [2]Faculty of Engineering, University of Rijeka, Rijeka, Croatia. [3]Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106, USA.
*Corresponding author. Email davidhaggerty@ucsb.edu
†These authors contributed equally to this work.

only shown to be so in quasi-static control in a low-deflection regime (approximately 18° of curvature). It is important to note that simple linearized models are likely to work at these low deflections because the full nonlinearity of the dynamics may only be explored at high deflections. Some studies (*34–36*) show imperfect yet functional controllers using purely state-space linear MPC, suggesting the quasi-linearity of these systems.

In addition to MPC, the combination of the Koopman operator and the linear quadratic regulator (K-LQR) optimal control scheme has shown promise in rigid robot applications (*38*, *39*) and the control of fluid dynamics problems (*40*). Mamakoukas *et al.* (*41*) show promise in a 1–degree-of-freedom soft robotic fish application using a similar Koopman structure.

Even with these many advances in the field, existing soft arm control implementations (*34–36*, *41*, *42*) have yet to be demonstrated in the inertial, nonlinear regime. To compare with other works, we introduce the following definitions of the "inertial regime" and "nonlinear dynamics." We define the inertial regime for soft arms to be when the inertial force experienced by the tip F_{tip} is of the order of its weight $F_{tip} = ma_{tip} \asymp mg$, meaning $a_{tip} \asymp g$. Here, m is the mass of the tip of the arm, a_{tip} is the acceleration of the tip during closed-loop control, and $g = 9.81$ m/s^2. We define nonlinear dynamics to be motions that fail to be adequately captured by a state-space linearization. Thus, an open challenge remains: modeling and control of inertial dynamics in highly nonlinear soft robots.

In this work, we advance modeling and control of soft, continuum arms into the inertial regime. Previous work has considered quasi-static motions, with accelerations below 0.03 *g*. Our work demonstrates movements in closed-loop control with accelerations greater than 1 *g* (see Table 1). We control these inertial movements in a highly nonlinear, high-deflection regime across two variations of our soft arm, each with different dimensions, numbers of actuators, and workspaces. The first demonstrates curvatures up to 110° (robot #1) and the second up to 180° (robot #2) (Fig. 1).

This capability is enabled by the introduction of the static Koopman pregain, which maps held inputs to converged robot configurations. After being learned from data, we used it as a pregain term in the LQR implementation. The static Koopman pregain greatly increases the accuracy of static pointing tasks and improves the stability of dynamic tasks.

We show that our approach requires minimal training and low computational cost, both for determining the model and for controlling the robot. Collecting our training data takes less than 5 min, and the computation of the model takes less than a second, as opposed to the long training times required by many neural net–based approaches. Our approach estimates both the static and dynamic control Koopman operators, enabling the use of low-latency, efficient optimal control methods; this enables real-time tracking of fast-moving reference positions, even if field-deployed on a low-power microcontroller.

RESULTS

In this section, we first outline our approach that enables modeling and control in the inertial, nonlinear regime yet requires relatively little training data and low computational power. Next, we systematically tested the speed and accuracy of the resulting closed-loop controller in a series of circular reference tracking tests. The soft arm was further tested in a tip-tracking test with a rapidly changing, user-defined reference position designed to test the soft arm's responsiveness to changes in commands in real time. Last, we tested our methodology on the dynamic catching and throwing of a ball. This leverages the inertial dynamics of our soft arm to demonstrate its effectiveness in real-world tasks.

Dynamic and static Koopman operator optimal control

The successful real-time control of a soft arm in the inertial and nonlinear regime requires both a model that captures these dynamics and a control methodology that adapts to the motion of the robot

Table 1. Comparison with existing soft, continuum arms shows advances in speed, acceleration, and deflection during closed-loop control. This work demonstrates a 10× increase in reference tracking tip speed (Speed) and a 4× improvement in tip deflection angle (Deflection) and advances closed-loop control of soft robot arms into the inertial regime $a_{tip} > g = 9.81$ m/s^2. The acceleration (Accel) of the soft arm's tip a_{tip} was computed using the centripetal acceleration of soft arms for which circular reference tracking data are available. The distance from the base to the tip of each arm is also given (Length). Note that closed-loop deflection data do not include the large-deflection open-loop tests present in some works. LQR, linear quadratic regulator; RNN, recurrent neural network; ROM, (analytical) reduced order model; PCC, piecewise constant curvature; MPC, model predictive control; LSTM, long short-term memory; TRPO, trust region policy optimization; GPR, Gaussian process regression; TO, trajectory optimization; FFC, feedforward compensator; SM, sliding mode; AF, analytical feedback; R1, robot #1; R2, robot #2.

Robot	Length (m)	Speed (m/s)	Accel (m/s^2)	Deflection (°)	Model	Control method
This work	0.37	1.52	11.6	R1: 110	Koopman	LQR
				R2: 180		
(*23*)	0.3	0.12	0.065	45	None	NN FFC
(*50*)	0.3	0.1	0.1	20	ROM	SM
(*37*)	0.15	0.094	0.29	18	Koopman	MPC
(*15*)	0.38	0.09	0.032	27	PCC ROM	AF
(*17*)	0.44	0.05		19	LSTM	TRPO
(*42*)	0.25	0.035	0.012	7	Koopman	MPC
(*36*)	0.7	0.03	0.032	8	Koopman	MPC
(*18*)	0.22	0.002	0.0016	11	RNN	GPR

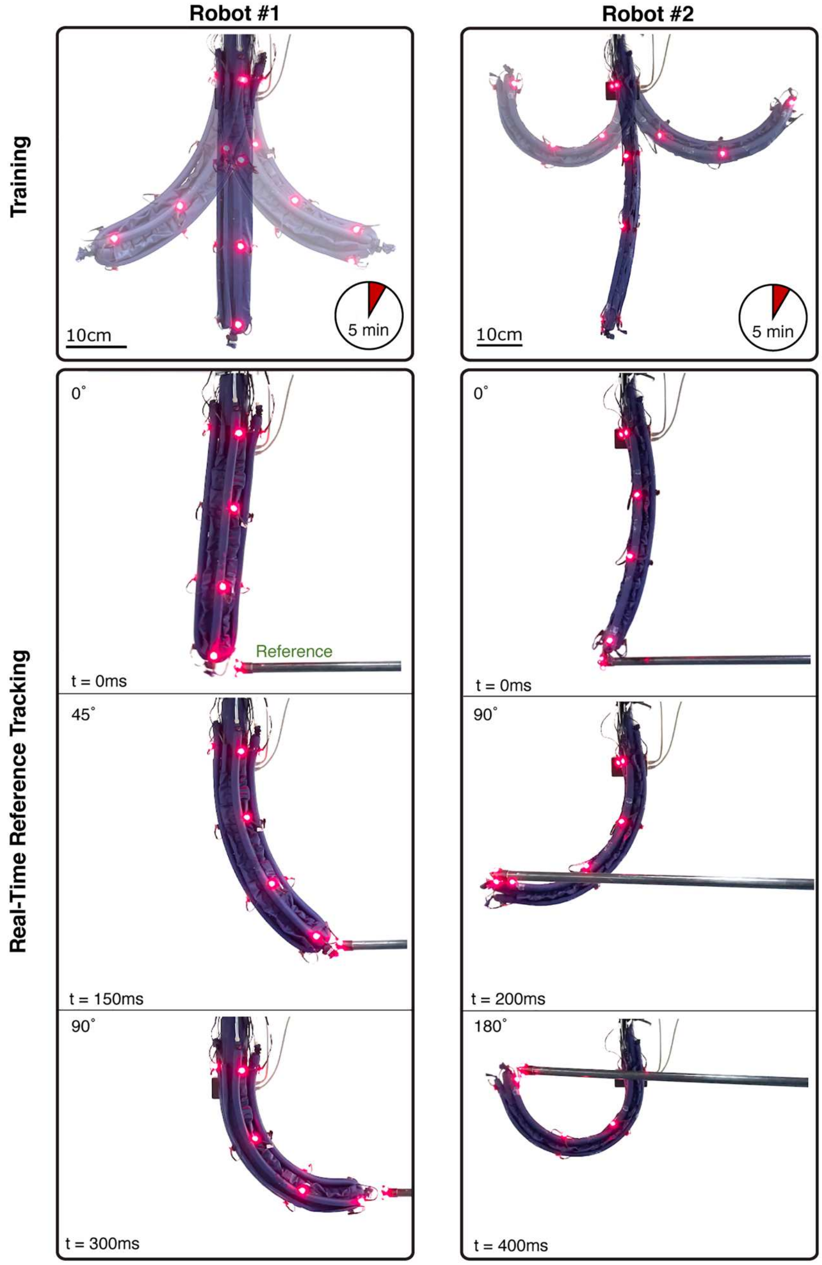

Fig. 1. Inertial, nonlinear soft arm control. Using a combined static and dynamic Koopman framework, we achieved the closed-loop control of soft robotic arms exceeding 10× the tip speed, 40× the tip acceleration, and 6× the angular displacement of existing soft arms. This achievement brings soft robotics into the inertial, nonlinear regime. Only 5 min of training was required to achieve an optimal controller capable of high-deflection, high-accuracy closed-loop tracking of a reference (the tip of a pole moved rapidly by a human). The same methodology was applied to both a low–slenderness ratio, four-muscle arm (robot #1) and a high–slenderness ratio, three-muscle arm (robot #2). Both arms achieved their highest deflection in under half a second.

in real time. We achieved this by building a controller that leverages both the dynamic and the static Koopman operators of the soft arm system. The Koopman operators describe the evolution in time of functions defined on the robot configurations and inputs. These functions are called observables, and the approximation of the Koopman operators involves training on data that are augmented by a chosen basis of observables. The data are collected through a series of training experiments, performed by commanding step inputs with randomly distributed magnitudes. These training data are partitioned into dynamic and static components, which are used to train the two separate Koopman operators (see Materials and Methods). Both the training and model computation processes are fast, requiring only 5 min (approximately 18,000 samples at 60-Hz collection rate) for training data collection, and the matrix pseudo-inverses used in the model construction take less than a second on an ordinary laptop computer.

The observables used to train the dynamic Koopman model are time-delayed measurements of the position of motion tracking points placed on the soft arm. This turned out to be sufficient to build a linear model of its nonlinear dynamics. Previous work considered adding a single time delay to hundreds of monomials (*42*). However, inspired by the fact that, for ergodic systems, the limit of infinitely many time delay observables results in DMD's convergence to the true Koopman operator (*31*, *32*, *43*), we included only time delay observables. Our results show that time delay–only observables are sufficient to capture the dynamics of this nonlinear system (see fig. S2) and that without time delays, the eigenvalues in the high-frequency and dissipative regions of the unit circle and their corresponding Koopman modes are missing (Fig. 2). This time delay–only approach avoids the added computational cost of many monomial observables and also eliminates the large tails associated with monomials that magnify noisy measurements far from the origin. We express this dynamic Koopman operator as a pair of matrices A and B giving the uncontrolled and controlled dynamics, respectively. These can be used to build the K-LQR controller described in Materials and Methods.

The resulting feedback controller is able to command the soft arm to follow a fast-changing reference position but suffers from steady-state error. Introducing integral control (for example, linear quadratic integral control) is one of the commonly used approaches to minimizing steady-state error (*44*). This method, however, is sensitive to measurement noise and requires a trade-off between speed of response and tracking accuracy. As a consequence, the implementation for our goals of highly inertial tasks resulted in poor tracking performance outside of the quasi-linear and quasi-static regimes. Instead, we addressed the steady-state error by introducing a static Koopman pregain, a control concept we developed for the current work. The static Koopman operator was first formally described in our recent modeling work (*44*), but no connection to control design was made. Unlike the dynamic Koopman operator, this operator is a map between functions defined on two different spaces. In our application, the static Koopman operator is used to map functions defined on the space of inputs to functions defined on the space of robot configurations. We learn this operator from the static partition of the training data so that static positions in the workspace of the soft arm correspond to the values of the inputs required to reach those positions after all transient motions dissipate. This operator is then used as a pregain term that augments the LQR controller. Sensor noise is known to cause tracking issues in soft robots attempting to perform real-time tracking of aggressive control inputs (*11*). Our control structure mitigates this problem by balancing the noise-sensitive dynamic K-LQR term with the sensor-agnostic static Koopman pregain.

The construction of the controller and the computation of the optimal input are also fast processes that have low computational overhead. The solution of the Riccati equation involved in computing the LQR control gain takes less than a second, and computing the optimal input at a given time step only requires two small matrix multiplications. This is easily achievable in real time on a low-cost microcontroller.

Closed-loop circle tracking in the inertial, nonlinear regime

With our control architecture in place, we first sought to characterize the performance across a range of deflections and soft arm speeds in a planar circular reference tracking (smooth changes in reference position). We commanded the soft arm's tip to trace out circular paths in the X-Y plane with three radii (100, 180, and

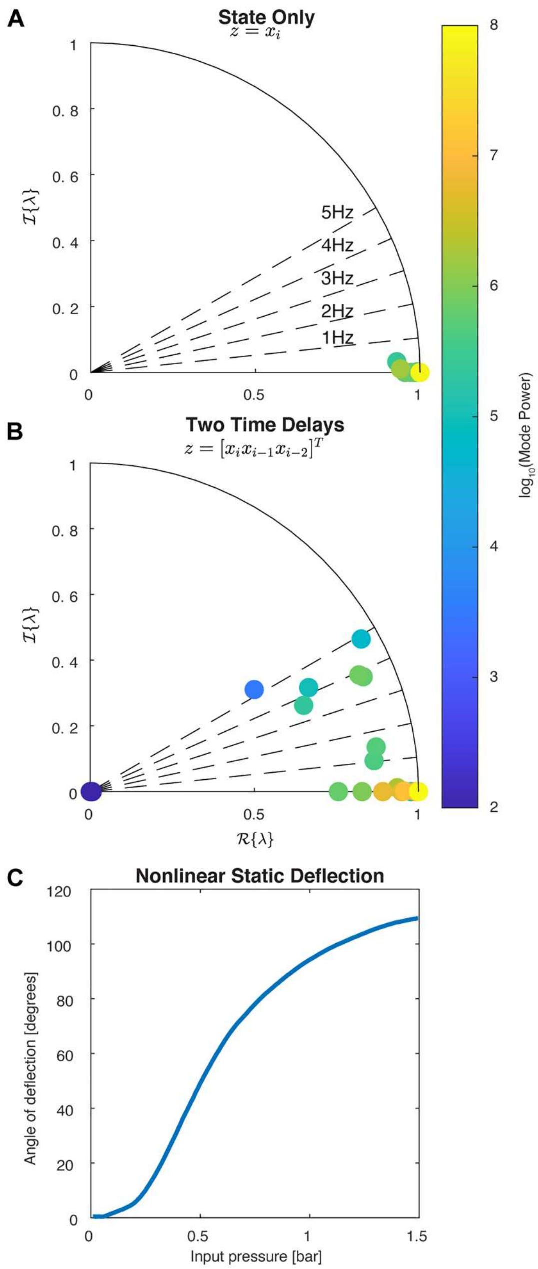

Fig. 2. Nonlinear and inertial dynamics of the soft arm. The eigenvalue plots for Koopman models with state-only (**A**) and state plus time delay observables (**B**) are shown. The dashed radial lines signify sections of the unit circle corresponding to modes with 1- to 5-Hz dynamics. The eigenvalues are shaded corresponding to the logarithm of their maximum achieved mode power evaluated over the training data (see Materials and Methods). Using state-only observables results in a simple linearized model that does not capture any transient dynamics. The addition of two time delay observables allows the modeling of dynamics up to 5 Hz. This is the model we chose for our experiments. (**C**) Presentation of the input-output nonlinearty of the system, which exhibits a sigmoidal deflection response. Modeling this nonlinearity is essential for acceptable reference tracking performance in the high-deflection regime.

220 mm) and six frequencies (0.1, 0.3, 0.5, 0.7, 0.9, and 1.1 Hz), as shown in Fig. 3. The same controller was used for all references, as described in Materials and Methods.

These results show that the soft arm tracks the reference with consistent performance throughout the full range of deflections and speeds tested (Fig. 3, left, and movie S2). The fastest and highest deflection circle tracking result demonstrates a tip speed of 1.5 m/s, a speed-to-length ratio of 3.23 s^{-1}, and a tip acceleration of 11.6 m/s^2 in closed-loop control. This is approximately an order of magnitude faster and 40 times higher acceleration than any soft continuum arm of which we are aware (see Table 1). The system was trained exclusively on step inputs, and as such, the model had no a priori knowledge of the control objective, nor had it been trained on circular behaviors.

In addition, we show that the relative contribution of the dynamic K-LQR input versus the static Koopman pregain increases with increasing speed and deflection (Fig. 3, right). For relatively low speeds and deflections, the dynamic K-LQR input is quite small, and the static Koopman pregain dominates. As accelerations increase and inertia becomes nonnegligible, the dynamic component increases in magnitude to compensate for the static term's inability to account for inertial effects. This suggests that for any soft robot performing a noninertial task, the incredibly simple static Koopman pregain could be sufficient for control.

Closed-loop, real-time reference tracking in the inertial, nonlinear regime

We next sought to characterize the controller performance for a less structured and more challenging control objective: tracking a real-time, user-defined reference. To do so, we commanded the controller to decrease the Euclidean distance between the tip of the soft arm and a motion tracker point located on the tip of a pole. A human operator moved the pole across random trajectories within the reachable workspace of the soft arm. Throughout the test, the robot remains in contact almost continuously while achieving speeds exceeding 0.7 m/s (as shown in Fig. 4 and movies S1 and S4).

To demonstrate the generalizability of our approach for different soft arms, we also tested our approach on a morphologically different second arm. The second arm is longer and more slender and has three instead of four side muscles. This results in larger curvatures and a helical actuation pattern, as discussed in the "Robot design" section. Despite these differences, no changes were needed in the learning and control algorithm, aside from updating the number of inputs. This second system was exposed to 5 min of step input training data, the model and controller were calculated and deployed, and the system was commanded to again track the tip of the user-operated pole. Results of this test are shown in the Supplementary Materials (movie S5), and stills from the testing are shown in Fig. 1.

Dynamic throwing and catching

With the viability of our method shown in the above characterization tests, we finally demonstrated how its capabilities translate to sample robotic tasks. We challenged our soft continuum arm in two ways: first, to catch a ball swinging through the air as we demonstrate in Fig. 5, and second, to receive an object from an operator and to throw it into a reference bin as shown in Fig. 6. Both tests are shown in movie S3. The catching component of this demonstration is similar to the ball catching performed by a two-link arm with a soft joint (*45*) but here completed with a fully soft continuum robot arm.

DISCUSSION

We present a data-driven framework for the modeling and control of inertial and nonlinear soft robots. We used KOT to enable the application of linear control methods to this highly nonlinear, inertial system. We introduce a K-LQR with static Koopman pregain capable of accurately controlling two different soft robots that exhibit high deflections and inertial motions. Advancing the state of the art, the proposed method allows the construction and deployment of both a model and an optimal controller from less than 5 min of training data—to the best of the authors' knowledge, the shortest in soft robotics (Fig. 7). Compared with existing MPC-based controllers, K-LQR is computationally less expensive and can be deployed on a simple microprocessor, enabling cheap and scalable use in a variety of environments outside the research laboratory. Despite its simplicity, our controller allows our soft arm to undergo controlled accelerations substantially greater than previous examples, even exceeding 1*g* (Table 1).

Although the presented demonstration of our modeling and control approach focused on soft robots, its implications could be much broader. The approach's ability to explore the dynamical features of a complex, nonlinear, inertial system could offer advantages in modeling and control of myriad robotic systems. Furthermore, its speed, versatility, low computational cost, and ease of use potentially expand the accessibility of robotics to new user groups. As such, we believe that our approach has the potential to make field-deployable, dynamical, soft robotic systems notably closer to realization.

MATERIALS AND METHODS

Here, we first introduce KOT, the mathematical underpinning of our modeling effort. In the "Approximation of Koopman operators for control systems: DMDc" section, we describe a practical method to build the model for our control system from data. In the "K-LQR" section, we describe how this model is embedded into a real-time feedback controller. Our modeling and control insight is the addition of a static Koopman operator pregain described in the "Static Koopman pregain" section. The design and fabrication of our soft arms and a description of the pneumatic circuitry that drives them are then presented. A block diagram detailing the full training process, modeling, and control architecture is given in fig. S4.

Koopman operator theory

The state space representation of a dynamical system involves defining an n-dimensional state space manifold M with states $x \in M$ and discrete-time evolution given by

$$x^+ = S(x) \tag{1}$$

Here, S is the possibly nonlinear state transition function $S: M \rightarrow M$ and x^+ is the time-shifted state. In our application, $M = \mathbb{R}^n$.

This nonlinearity is often critical to modeling a system in state space, but it complicates the design of control algorithms. We instead turned to an operator-theoretic perspective of dynamics of observables (*24*). Observables are complex-valued functions defined on the state space $f: M \rightarrow \mathbb{C}$. We will restrict ourselves to

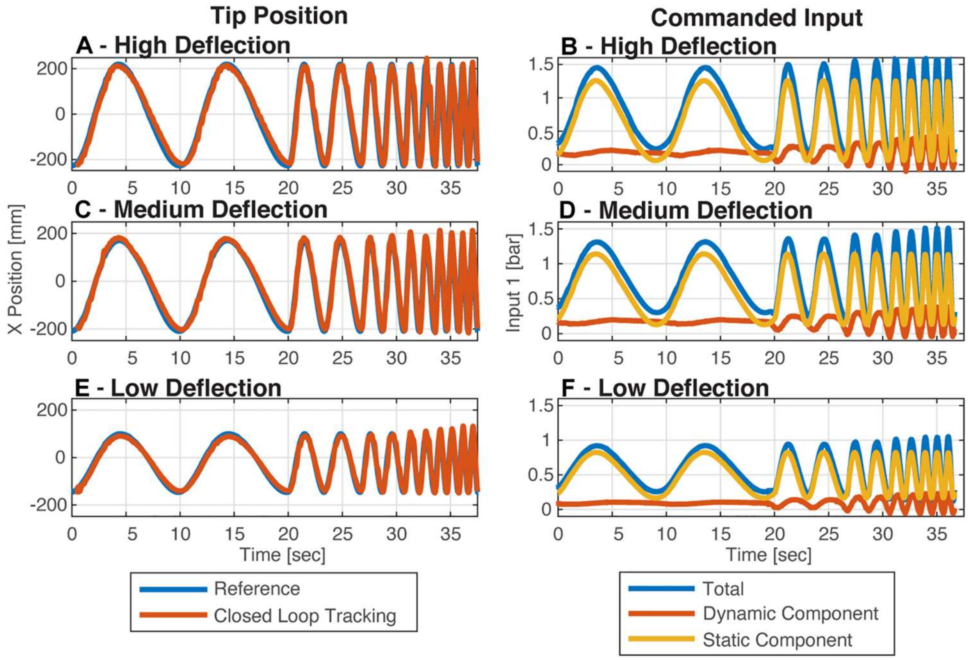

Fig. 3. Closed-loop, real-time reference tracking experiments. The soft arm tracked circular reference trajectories in the *X-Y* plane with frequencies ranging from 0.1 to 1.1 Hz (0.2-Hz step) at (**A** and **B**) high, (**C** and **D**) medium, and (**E** and **F**) low deflections. (A), (C), and (E) show the *X* positions (red) over time compared with their respective references (blue). The *Y* and *Z* positions are shown in fig. S3. (B), (D), and (F) show the relative contributions of the static Koopman pregain (yellow) and dynamic K-LQR (red) to the total input (blue). At quasi-static speeds, only the static Koopman pregain is required for effective performance (that is, the quantity $x - x_{ref}$ is approximately zero); as inertial effects increase, the LQR component increases its contribution to maintain performance. Only the commanded inputs to one of the four side muscles is shown, but the results are similar for all muscles.

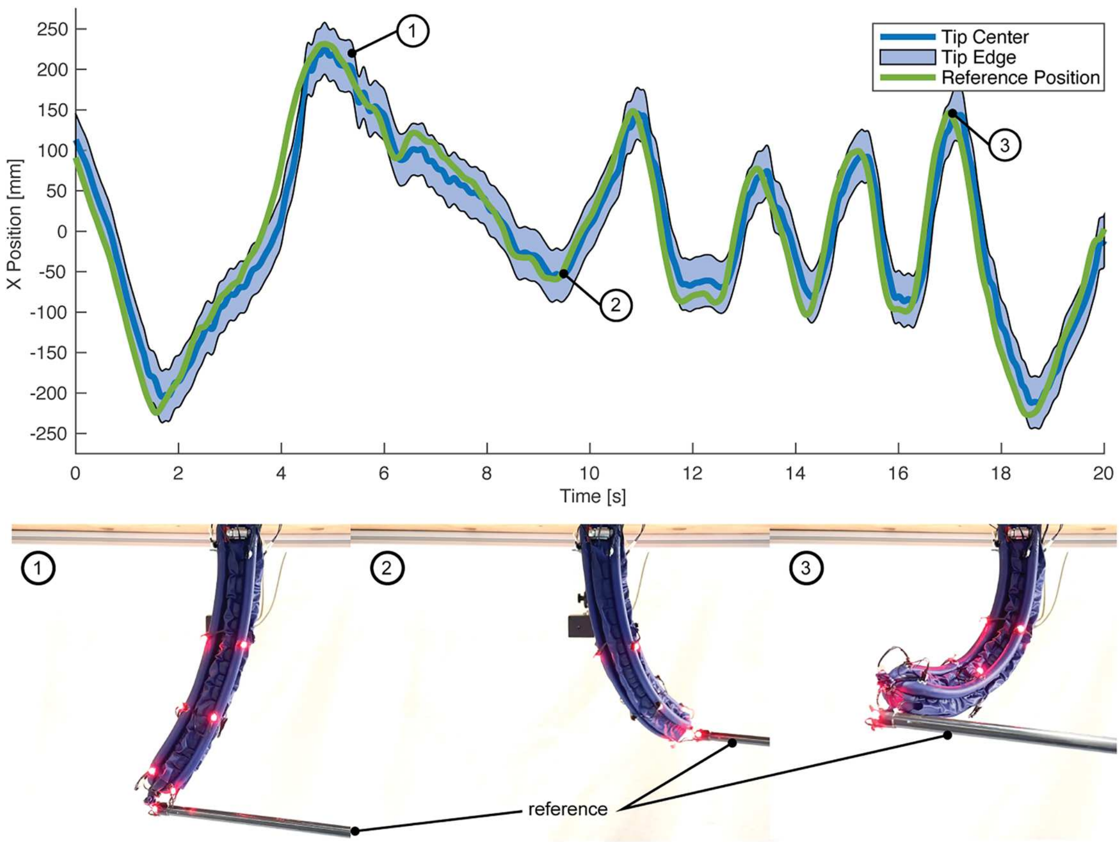

Fig. 4. Arbitrary reference tracking throughout the high-deflection workspace. (**A**) The *X* position of the soft arm tip is shown as it tracks a moving reference commanded randomly by an operator. Contact between the green lines and the blue band indicates points where the soft arm is touching the reference marker. (1) to (3) show images of the soft arm performing this behavior. Of note, the robot rarely loses contact with the moving reference.

real-valued observables $f : M \to \mathbb{R}$. The set of all possible observables

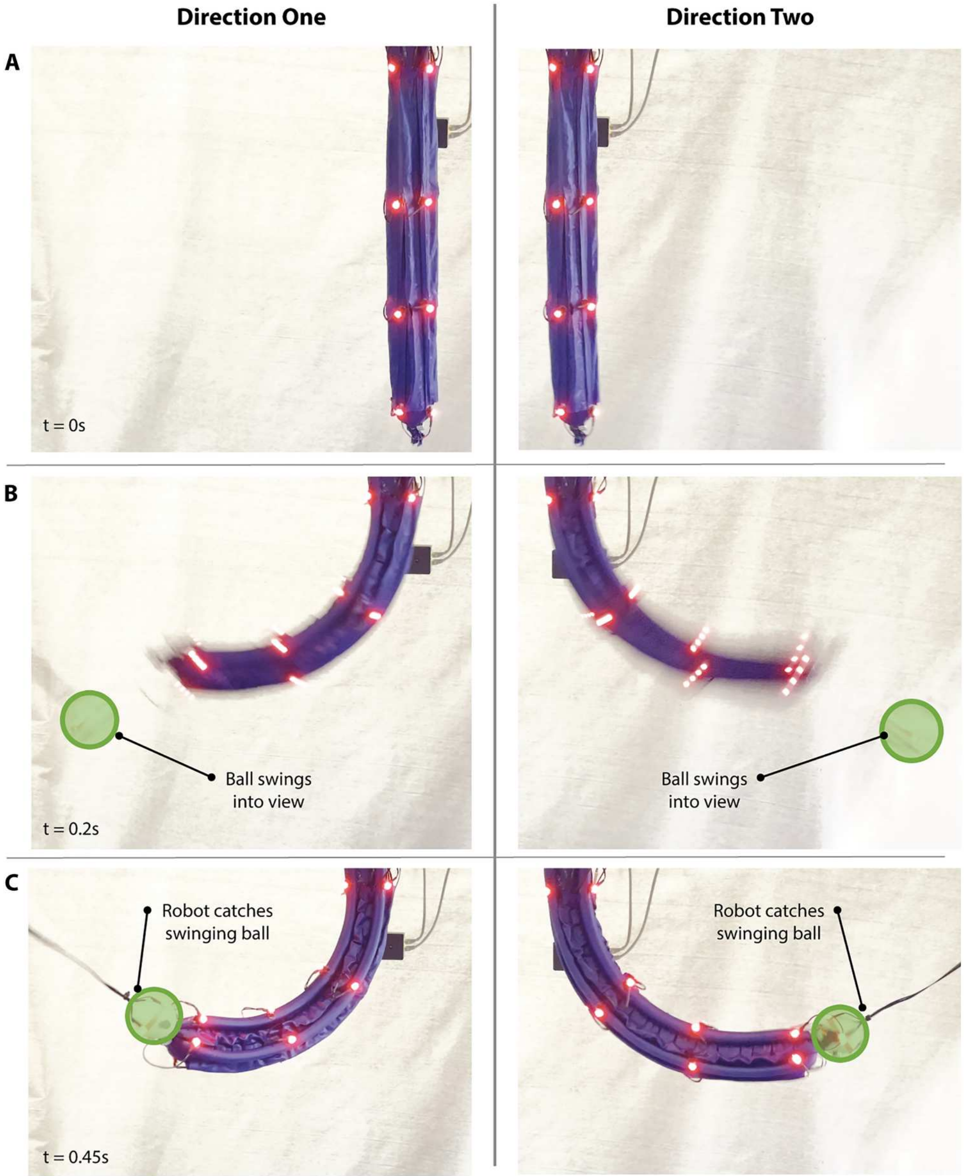

Fig. 5. Dynamic tracking of arbitrary trajectory (catching a swinging ball). (**A**) The soft arm stays in the neutral position while the ball is outside the workspace. (**B**) Once visible, the soft arm rapidly responds to reach the ball (outlined swinging into the workspace). (**C**) The soft arm tip intercepts the ball and catches it (with small magnets on both the soft arm tip and swinging ball).

forms a vector space that is usually infinite dimensional. The Koopman operator $\mathscr{K}$ is defined by $\mathscr{K}f := f \circ S$.

This operator describes the evolution of observables under the action of the dynamics (*1*). Although the underlying state space system is nonlinear, the Koopman operator $\mathscr{K}$ is always linear (*24*–*26*, *45*). This is true without restriction on the dynamics or observables.

We wanted to exploit this linearity to enable the design of an efficient optimal control scheme. This requires extending the Koopman framework to systems of the form $x^+ = S(x, u)$, where $u \in \mathbb{R}^p$ is a p-dimensional vector of user-specified inputs. In full generality, the Koopman operator for systems with input acts on observables of the form $f : M \times \mathscr{U} \to \mathbb{C}$, where $\mathscr{U}$ is the space of all control sequences indexed by time $\overline{u}(\cdot) : \mathbb{N} \to \mathbb{R}^p$. We redefine the state transition function to include inputs $S : M \times \mathbb{R}^p \to M$ and introduce the left shift operator $T : \mathscr{U} \mapsto \mathscr{U}$, which simply chooses the next input in a sequence $(T\overline{u})(k) = \overline{u}(k+1)$. When the observables are defined on both the states and inputs, their Koopman evolution is given by

$$(\mathscr{K}f)(x, \overline{u}(\cdot)) := f(S(x, \overline{u}(0)), T\overline{u}(\cdot)) \tag{2}$$

Elements of $\mathscr{U}$ are infinite dimensional, which puts the observables $f : M \times \mathscr{U} \to \mathbb{R}$ on an infinite dimensional domain, so they cannot be manipulated on a computer. We introduce the simplifying assumption that knowing only the input at the current time step is enough to predict the future dynamics. We can now define observables of the form $f : M \times \mathbb{R}^p \to \mathbb{R}$. This results in a Koopman

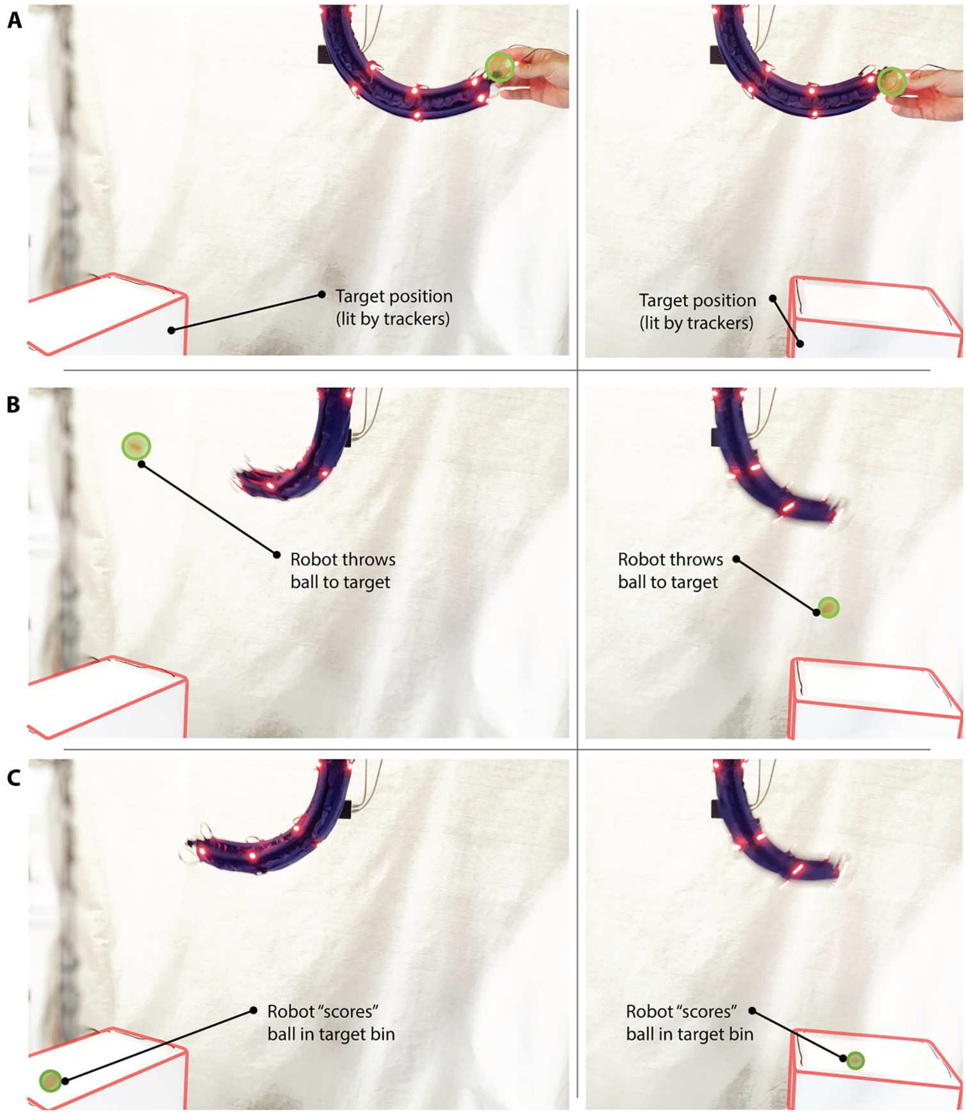

Fig. 6. Implications of the methodology: Completing example tasks. (**A**) The soft arm identifies the objective and approaches it (operator's hand). (**B**) After the operator's hand is removed and the ball is supported by the soft arm, the objective changes to the bin (LED-designated bin in bottom left and right, respectively). The soft arm now flings the ball at the objective. (**C**) The ball successfully enters the bin in two different, arbitrary locations, achievable only by working in the inertial regime.

operator $\mathscr{K}$ defined by

$$(\mathscr{K}f)(x,u)f(S(x,u),u) \quad (3)$$

We sought a finite dimensional linear input/output system that approximates the action of $\mathscr{K}$ on a finite set of chosen observables. This process is described in the next section.

Approximation of Koopman operators for control systems: DMDc

We followed the process outlined in (*33*). The Koopman operator in its fully infinite dimensional form is not practically realizable, so we sought a finite dimensional approximation. The first step is to choose some finite dictionary of observables $\{g_j\ (x,u)\}_{j=1}^{m+p}$. We chose m observables, which are functions of purely the states; p, which are functions of the inputs; and none, which are coupled functions of both the states and inputs

$$\{g_j(x,u)\}_{j=1}^{m+p} = \{f_j(x)\}_{j=1}^{m} \cup \{h_j(u)\}_{j=m+1}^{m+p} \quad (4)$$

It is simple to allow arbitrary input observables, but we only deal with the case where $h_j(u) = u_j$. This decoupling restricts our choices of observables, but it allows us to define a vector of observables $z(x) = [f_1(x) \cdots f_m(x)]^T$ called the lifted state, which allows us to represent our dynamics as a linear input-output system

$$z^+ = Az + Bu \quad (5)$$

Here, A and B are the state transition and input matrices, respectively. This simplification has the benefit of enabling the later use of the fast and efficient linear optimal control methods described in

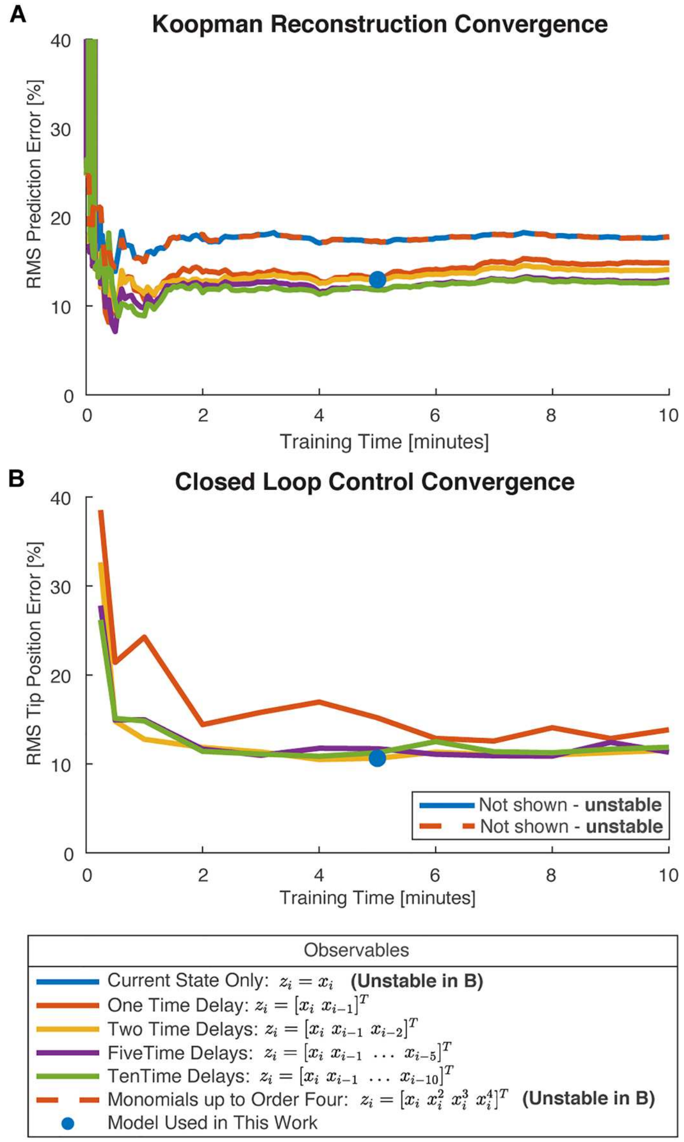

Fig. 7. Convergence of the Koopman model and control system. (**A**) The dynamic Koopman model requires the addition of five time delay observables and only 1 min of training data to reach minimum prediction error. To determine this error, the single-step prediction error of the dynamic Koopman model was collected for all points as the soft arm moved on a circular path in the *X-Y* plane (inset), and the root mean square average was taken. For comparison, a Koopman model using monomials of the state up to order 4 gave no improvement over the state-only model. This reconstruction was performed on a model trained on zero sinusoidal trajectories. (**B**) In the closed-loop control, the combination static/dynamic Koopman controller required only 5 min of training data and two time delays to reach minimum prediction error; accordingly, we used this controller design for every experiment. Each controller was commanded to move the soft arm's tip to a sequence of points in the workspace of the soft arm, and the average root mean square error for all these points was calculated. Using zero time delays resulted in the soft arm being unable to stabilize at any reference position, so that line is not shown.

the "K-LQR" section while still capturing the dynamics of the system as demonstrated in fig. S2.

The states are retrieved from the observables using the output equation

$$x = Cz \tag{6}$$

where C is the output matrix.

Here, we outline the approximation of the matrices A, B, and C using a process called extended dynamic mode decomposition with control (*33*). When restricted to time delay observables, we call this Hankel-DMDc. We wanted to approximate these matrices using K measurements of the states $\{x_1, \dots, x_K\}$, time-shifted states $\{x_1^+, \dots, x_K^+\}$, and inputs $\{u_1, \dots, u_K\}$ collected from experimental data. First, we built data matrices whose columns are the data vectors

$$X := [x_1 \dots x_K] \tag{7}$$

$$X^+ := [x_1^+ \dots x_K^+] \tag{8}$$

$$U := [u_1 \dots u_K] \tag{9}$$

Next, we built the lifted data matrices using our chosen vector of observables $z(x)$

$$X_{\text{lift}} := [z(x_1) \dots z(x_K)] \tag{10}$$

$$X_{\text{lift}}^+ := [z(x_1^+) \dots z(x_K^+)] \tag{11}$$

The desired matrices A and B satisfy the equation

$$X_{\text{lift}}^+ = AX_{\text{lift}} + BU \tag{12}$$

To approximate A and B, we recasted this equation as a minimization problem

$$\min_{A,B} \|X_{\text{lift}}^+ - AX_{\text{lift}} - BU\|_F \tag{13}$$

which has the solution

$$[A \;\; B] = X_{\text{lift}}^+ \left(\begin{bmatrix} X_{\text{lift}} \\ U \end{bmatrix} \right)^\dagger \tag{14}$$

where † is the Moore-Penrose pseudoinverse. Because we prescribe our first n observables to be the states $x \in M$, we can compute the output matrix using a partial identity matrix

$$C = \begin{bmatrix} I_{n\times n} & 0_{n\times m-n} \\ 0_{m-n\times n} & 0_{m-n\times m-n} \end{bmatrix} \tag{15}$$

The action of the matrices A and B on the lifted state via Eq. 5 approximates the action of the Koopman operator $\mathscr{K}$ in Eq. 3. Under certain assumptions, this representation of the Koopman operator converges to the true Koopman operator (*30*). True convergence requires infinite data samples, which are uniformly distributed in state space and a collection of observables, which span an invariant subspace of the Koopman operator's underlying function space. We discuss our method of generating training data in the "Training and observables" section.

K-LQR

To date, similar investigations have used MPC to control their soft robotic systems (*34–36*). Using predictions of the dynamics and a tunable prediction horizon, this architecture calculates input sequences that move the system toward a desired reference position. This enables the use of explicit input and state constraints, but the real-time constrained optimizations involved in this method demand a high computational overhead.

In our inertial soft arm controller, explicit constraints are less important than keeping computational cost and latency low. For unconstrained linear optimal control problems with quadratic cost, the LQR provides an analytical solution that does not require predictions of the dynamics in real time (*46*). For our controller, we began with the application of LQR to the dynamic Koopman representation of a dynamical system [previously demonstrated for a robotic fish (*41*)] and augmented it via the introduction of the static Koopman term, described in the "Static Koopman pregain" section.

Here, we describe the dynamic K-LQR control law. Although originally introduced for linear dynamical systems in state space, LQR can also be applied to a vector of observables z of a nonlinear control system as long as a linear, finite dimensional representation of the Koopman operator (A, B) exists. Given the system

$$z^{+} = Az + Bu \tag{16}$$

$$x = Cz \tag{17}$$

we define the global cost function

$$J = \sum_{i=1}^{K} [(z_i - z_{\text{ref}})^T Q (z_i - z_{\text{ref}}) + u_i^T R u_i] \tag{18}$$

where $x_{\text{ref}} = Cz_{\text{ref}}$ is the desired position and Q and R are diagonal lifted state and input penalty matrices, respectively.

The computation of the minimizing control input is a classical method in optimal control (*46*) and is given by $u_i = -K(z_i - z_{\text{ref}})$, where the matrix K is the LQR gain. This control law results in steady-state errors in much of the soft arm's workspace. This is remedied in the next section by the addition of a pregain term based on the static Koopman operator.

Static Koopman pregain

Unfortunately, dynamic K-LQR alone resulted in substantial disagreement between reference positions and the resulting states. This is because the nonzero inputs required to hold these positions result in a nonzero input penalty term. Any attempt to decrease the input penalty resulted in system instability. The addition of a pregain term is a classical method in control theory that addresses this problem. In this section, we introduce a data-driven method to compute the pregain using a static Koopman operator, which we term the static Koopman pregain.

A core assumption of this component of our model is that when held for enough time, all transient dynamics dissipate, and the robot achieves a static pose. Therefore, the set of admissible step inputs u_{static} corresponds to a set of input-mediated fixed points x_{static}. We sought a mapping from the data matrix of step inputs, U_{static}, to the data matrix of stationary states, X_{static}. Ideally, this mapping would be linear to enable us to use fast, optimal control. The Koopman framework usually requires the domain and range to be the same, but this requirement can be relaxed if we consider the static Koopman operator (*44*). The static Koopman operator contrasts with the dynamic Koopman operator, which describes the evolution of observables f: $M \rightarrow \mathbb{R}$ under the action of the mapping T: $M \rightarrow M$. If we define observables on the inputs as g: $\mathbb{R}^p \rightarrow \mathbb{R}$, the static Koopman operator $\mathscr{K}_{\text{stat}}$ is defined as $\mathscr{K}_{\text{stat}} f(x_{\text{stat}}) = g(u_{\text{stat}})$.

We desired to approximate the action of the static Koopman operator with a finite dimensional matrix G. To do so, we first constructed the data matrix U_{static} with unique step inputs as the columns of the matrix. By feeding these inputs to the system and allowing transient dynamics to dissipate, we are left with a unique stationary state, x_{static}; these states represent the columns of X_{static}. The matrix G is then computed using

$$G = U_{\text{static}} X_{\text{static}}^{\dagger} \tag{19}$$

The matrix G serves as a linear mapping from stationary states to inputs.

Last, we are ready to bias our control law with the addition of a feedforward pregain term Gz_{ref}, resulting in

$$\begin{aligned} u_i &= -K(z_i - z_{\text{ref}}) + Gz_{\text{ref}} \\ z_{i+1} &= Az_i + Bu_i \\ x_i &= Cz_{i+1} \end{aligned} \tag{20}$$

This signal is the optimal stabilizing solution taking the present initial state to the desired state, x_{ref}.

As shown in Fig. 3, the pregain term $u_{\text{stat}} = Gz_{\text{ref}}$ outweighs the dynamic term $u_{\text{dyn}} = -K(z_i - z_{\text{ref}})$ in most tests. This allows the input penalty weights in the dynamic term to be optimized without fear of sacrificing steady-state error. In addition, the static Koopman term provides enough of a steady input to counter the fluctuations caused by measurement noise introduced by the state measurements in the dynamic term. This is the reason our system does not experience the destabilizing effects of noise in fast-moving reference tests described in (*11*).

Training and observables

With the mathematical underpinning of our modeling and control methodology described (see fig. S4), we now turn to the particular choices made to suit our particular robotic applications. Given the soft arms described in the "Robot design" section, we collected training data through a series of experiments, performed by commanding step inputs with randomly distributed magnitudes. The only prior knowledge of the soft arm's dynamics required is an upper bound for the length of time required for the dissipative dynamics to die down while inputs are held. Each step input was held for this amount of time so that the soft arm converges to a steady state, efficiently probing both the dynamic and static response. The data were separated into training and validation sets, and the training data were further partitioned into dynamic and static components, which were used to train dynamic and static Koopman operators (see the "Dynamic and static Koopman operator optimal control" section).

Choosing observables is difficult in practice. We chose to implement DMDc with time delay observables (also known as Hankel DMDc) because of their provable convergence as the number of time delays goes to infinity under certain assumptions on the dynamics (*31*, *32*, *43*). In reality, adding more time delays gives a diminishing return in prediction accuracy (see Fig. 7A). A single time delay with hundreds of monomials was used in (*34*–*36*, *42*), but we found that time delay–only observables offer better results, with improvements in reconstruction with up to 10 observables (see Fig. 7A). To create our observables, we used the current measurement of the *X*-*Y*-*Z* positions of the motion trackers x_k and appended two time-delayed versions of the same states $z_k = [x_k \ \ x_{k-1} \ \ x_{k-2}]^T$. Each time delay looks 1/60 s into the past. This proved to be sufficient for closed-loop control. For reconstruction, more time delays give further increases to the model's accuracy, as shown in Fig. 7A.

The synergy of step inputs and time delays allows the discovery of system eigenvalues in the important 1- to 5-Hz range (the span of natural frequencies of the arm), as shown in Fig. 2. Without time delays, these eigenvalues and their corresponding Koopman modes are missed (Fig. 2). For comparison with the Koopman model used in (*42*), we tested the addition of monomial observables up to order 4 with no new dynamic modes of any meaningful mode power learned. Monomial observables also failed to give any improvement to the reconstruction or closed-loop pointing accuracy of the model and controller (Fig. 7).

With the goal of minimizing training time and model complexity, we found that up to five time delays and 1 min of step input training is best for modeling our system before considering control, but only two time delays and 5 min of step input training is ideal when control is considered. We first compared the prediction ability of different dynamic Koopman models as we varied the number of time delays and total training time (Fig. 7A). The addition of a single time delay substantially reduced error; however, additional time delays continued to offer marginal improvements up to five delays. We also found that after only approximately 1 min of training, the model reached its minimum error. Second, we built K-LQR controllers as described in the "K-LQR" section, augmented with a static Koopman operator as a pregain term, with varied time delays and training time. We then quantified the error with closed-loop control (Fig. 7B). In this case, two time delays outperformed one delay but were comparable to three or more, resulting in our decision to use two delays for control. We also found that after approximately 5 min of training (50 unique step inputs), the error converged; we used this amount of training time for the remaining experiments. Note that a direct linearization of the system was unstable during controlled motions, suggesting the nonlinearity of the system.

Robot design

For this investigation, we constructed two distinct soft arms to evaluate the viability of the proposed methodology across nonlinear dynamical systems. For each, we aimed to meet the following objectives: (i) high-deflection, nonlinear dynamics for which linearization fails; (ii) inertial dynamics, for which quasi-static approximations fail; and (iii) enough morphological diversity such that their analytical models would not be readily transferrable.

To this end, the first arm was designed to have four actuators (two antagonistic pairs) longitudinally aligned with the main body to produce planar actuation. This design is behaviorally similar to others present in the literature (*15*, *16*, *47*). When fabricated with appropriate pretension, this construction allows for approximately 110° of curvature when fully actuated. With a length of 45 cm and a maximum diameter (main body diameter plus the diameter of the fully inflated muscles) of 6.25 cm, the slenderness ratio of this device was 7.2 (the ratio of length to maximum diameter).

The second arm was designed with three actuators, all of which were affixed to the body such that a torsional deflection would be induced when inflated. This produces a helical actuation that is markedly different from that of the first embodiment. With a length of 53 cm and a maximum diameter of 3.8 cm, this device exhibited a slenderness ratio of 13.9. The muscles were affixed with pretensions such that, when fully actuated, this device is capable of achieving approximately 180° of curvature.

For objective (i), with an angle of curvature of at least 110° for both arms, the nonlinearity metric was well achieved (see Fig. 2). For objective (ii), both systems were fabricated out of airtight fabric, using fabric pneumatic artificial muscles (fPAMs) as described in (*48*), which exhibit a fast response time and low hysteresis (on the order of 1%), achieving accelerations in excess of *g*. For (iii), the factor of approximately two difference in slenderness ratios, the change in actuator numbers, and the inclusion of helical actuation all combined to produce two systems with meaningfully different behavior [see, for example, the model presented in (*49*) compared with (*13*)].

Robot fabrication

Both arms were constructed out of 30 Denier silicone-polyurethane impregnated ripstop nylon (Sil-nylon, Rockywoods Fabrics), actuated by fPAMs (*48*) built out of the same material. The main body was fabricated such that one side of the fabric weave cell was parallel to the longitudinal axis, while the other was perpendicular. This orientation makes the soft arm axially and transversely stiff but torsionally compliant. The muscles were fabricated such that each side of the cell was offset by approximately 45° with respect to the longitudinal axis, which instead makes the actuator torsionally stiff but compliant axially and transversely. Moreover, when these muscles are inflated, they shorten in the longitudinal direction as a McKibben does, up to 35% based on the pretensioning induced during adhesion to the main body.

Each of these components was cut from a sheet of fabric, rolled into a tube, and sealed with a lap joint using room-temperature-vulcanizing (RTV) silicone adhesive (Smooth-on Silpoxy). Once each component was fashioned, a jig was produced to hold the main body and pretensioned muscles in place while the RTV cured. Last, between each muscle, a fabric sleeve, exhibiting the same fabric bias as the muscles, was attached to the main body to allow for motion capture tracker wires to be routed without occluding the view of the light-emitting diodes (LEDs).

Pneumatic circuit design

Each soft arm body was held at a constant pressure of approximately 1 bar for the entirety of testing, supplied by a discrete source. For each muscle of both soft arms, Festo VEAB-L-26-D2-Q4-V1-1R1 proportional pressure valves were used to command individual pressures continuously. These three-port valves were chosen for three reasons: their fast response times (<10 ms), accurate response (0.75% full-scale absolute accuracy and 0.4% full-scale repeatability error), and the ability to accept forced exhaust through their third port. However, this accuracy requires a lower flow rate, which precluded the use in the much larger main body (primarily because of persistent leaks). Additional information on the general control circuitry configuration can be found in the Supplementary Materials.

Supplementary Materials

This PDF file includes:
Discussion
Figs. S1 to S5

Other Supplementary Material for this manuscript includes the following:
Movies S1 to S5

REFERENCES AND NOTES

1. I. Ermolov, in *Industrial Robotics Review* (Springer International Publishing, 2020), pp. 195–204.
2. W. Ji, L. Wang, Industrial robotic machining: A review. *Int. J. Adv. Manuf. Technol.* **103**, 1239–1255 (2019).
3. A. S. Bisen, H. Payal, Collaborative robots for industrial tasks: A review. *Materials Today Proc.* **52**, 500–504 (2022).
4. S. Bragança, E. Costa, I. Castellucci, P. M. Arezes, in *A Brief Overview of the Use of Collaborative Robots in Industry 4.0: Human Role and Safety* (Springer International Publishing, 2019), pp. 641–650.
5. D. Kragic, J. Gustafson, H. Karaoguz, P. Jensfelt, R. Krug, Interactive, collaborative robots: Challenges and opportunities, in *International Joint Conference on Artificial Intelligence* (IJCAI, 2018), pp. 18–25.
6. F. Vicentini, Collaborative robotics: A survey. *J. Mech. Des.* **143**, 1–29 (2021).
7. D. Rus, M. T. Tolley, Design, fabrication and control of soft robots. *Nature* **521**, 467–475 (2015).
8. C. Majidi, Soft robotics: A perspective—Current trends and prospects for the future. *Soft Robot.* **1**, 5–11 (2014).
9. T. George Thuruthel, Y. Ansari, E. Falotico, C. Laschi, Control strategies for soft robotic manipulators: A survey. *Soft Robot.* **5**, 149–163 (2018).
10. J. Wang, A. Chortos, Control strategies for soft robot systems. *Adv. Intell. Syst.* **4**, 2100165 (2022).
11. L. Shi, Z. Liu, K. Karydis, Koopman operators for modeling and control of soft robotics. arXiv:2301.09708 [cs.RO] (7 February 2023).
12. O. Yasa, Y. Toshimitsu, M. Y. Michelis, L. S. Jones, M. Filippi, T. Buchner, R. K. Katzschmann, An overview of soft robotics. *Annu. Rev. Control Robot. Auton. Syst.* **6**, 1–29 (2023).
13. R. K. Katzschmann, C. Della Santina, Y. Toshimitsu, A. Bicchi, D. Rus, Dynamic motion control of multi-segment soft robots using piecewise constant curvature matched with an augmented rigid body model, in *2019 2nd IEEE International Conference on Soft Robotics (RoboSoft)* (IEEE, 2019), pp. 454–461.
14. R. K. Katzschmann, M. Thieffry, O. Goury, A. Kruszewski, T.-M. Guerra, C. Duriez, D. Rus, Dynamically closed-loop controlled soft robotic arm using a reduced order finite element model with state observer, in *2019 2nd IEEE International Conference on Soft Robotics (RoboSoft)* (IEEE, 2019), pp. 717–724.
15. C. Della Santina, R. K. Katzschmann, A. Bicchi, D. Rus, Model-based dynamic feedback control of a planar soft robot: Trajectory tracking and interaction with the environment. *Int. J. Robot. Res.* **39**, 490–513 (2020).
16. C. Della Santina, R. L. Truby, D. Rus, Data-driven disturbance observers for estimating external forces on soft robots. *IEEE Robot. Autom. Lett.* **5**, 5717–5724 (2020).
17. A. Centurelli, L. Arleo, A. Rizzo, S. Tolu, C. Laschi, E. Falotico, Closed-loop dynamic control of a soft manipulator using deep reinforcement learning. *IEEE Robot. Autom. Lett.* **7**, 4741–4748 (2022).
18. X. Wang, N. Rojas, A data-efficient model-based learning framework for the closed-loop control of continuum robots. arXiv:2204.10454 [cs.RO] (22 April 2022).
19. T. G. Thuruthel, B. Shih, C. Laschi, M. T. Tolley, Soft robot perception using embedded soft sensors and recurrent neural networks. *Sci. Robot.* **4**, eaav1488 (2019).
20. R. L. Truby, C. Della Santina, D. Rus, Distributed proprioception of 3D configuration in soft, sensorized robots via deep learning. *IEEE Robot. Autom. Lett.* **5**, 3299–3306 (2020).
21. T. G. Thuruthel, E. Falotico, F. Renda, C. Laschi, Model-based reinforcement learning for closed-loop dynamic control of soft robotic manipulators. *IEEE Transact. Robot.* **35**, 124–134 (2019).
22. S. Neppalli, B. Jones, W. McMahan, V. Chitrakaran, I. Walker, M. Pritts, M. Csencsits, C. Rahn, M. Grissom, Octarm—A soft robotic manipulator, in *IEEE/RSJ International Conference on Intelligent Robots and Systems* (IEEE, 2007), p. 2569.
23. D. Braganza, D. M. Dawson, I. D. Walker, N. Nath, A neural network controller for continuum robots. *IEEE Transact. Robot.* **23**, 1270–1277 (2007).
24. B. O. Koopman, Hamiltonian systems and transformation in hilbert space. *Proc. Natl. Acad. Sci. U.S.A.* **17**, 315–318 (1931).
25. I. Mezić, Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dyn.* **41**, 309–325 (2005).
26. I. Mezić, A. Banaszuk, Comparison of systems with complex behavior. *Phys. D Nonlinear Phenom.* **197**, 101–133 (2004).
27. J. L. Proctor, S. L. Brunton, J. N. Kutz, Dynamic mode decomposition with control. *SIAM J. Appl. Dyn. Syst.* **15**, 142–161 (2016).
28. J. L. Proctor, S. L. Brunton, J. N. Kutz, Generalizing Koopman theory to allow for inputs and control. *SIAM J. Appl. Dyn. Syst.* **17**, 909–930 (2018).
29. L. Shi, K. Karydis, ACD-EDMD: Analytical construction for dictionaries of lifting functions in Koopman operator-based nonlinear robotic systems. *IEEE Robot. Autom. Lett.* **7**, 906–913 (2022).
30. M. Korda, I. Mezić, On convergence of extended dynamic mode decomposition to the Koopman operator. *J. Nonlinear Sci.* **28**, 687–710 (2018).
31. H. Arbabi, I. Mezic, Computation of transient Koopman spectrum using Hankel-dynamic mode decompoisition, in *APS Division of Fluid Dynamics (Fall)* (American Physical Society, 2017), p. G1.009.
32. H. Arbabi, I. Mezic, Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the Koopman operator. *SIAM J. Appl. Dyn. Syst.* **16**, 2096–2126 (2017).
33. M. Korda, I. Mezić, Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. *Automatica* **93**, 149–160 (2018).
34. D. Bruder, B. Gillespie, C. D. Remy, R. Vasudevan, Modeling and control of soft robots using the Koopman operator and model predictive control. arXiv:1902.02827 [cs.RO] (1 July 2019).
35. D. Bruder, C. D. Remy, R. Vasudevan, Nonlinear system identification of soft robot dynamics using Koopman operator theory, in *2019 International Conference on Robotics and Automation (ICRA)* (IEEE, 2019), pp. 6244–6250.
36. D. Bruder, X. Fu, R. B. Gillespie, C. D. Remy, R. Vasudevan, Koopman-based control of a soft continuum manipulator under variable loading conditions. arXiv:2002.01407 [cs.RO] (4 February 2020).
37. J. Chen, Y. Dang, J. Han, Offset-free model predictive control of a soft manipulator using the Koopman operator. *Mechatronics* **86**, 102871 (2022).
38. H. Yin, M. Welle, D. Kragic, Policy learning with embedded Koopman optimal control. *Proc. Mach. Learn. Res.* **144**, 1–14 (2018).
39. I. Abraham, T. D. Murphey, Active learning of dynamics for data-driven control using Koopman operators. *IEEE Transact. Robot.* **35**, 1071–1083 (2019).
40. A. Gibson, "Application of Koopman linear quadratic regulator to the control of a spherical microbubble," thesis, University of Colorado, Colorado Springs (2022).
41. G. Mamakoukas, M. L. Castano, X. Tan, T. D. Murphey, Derivative-based Koopman operators for real-time control of robotic systems. *IEEE Transact. Robot.* **37**, 2173–2192 (2021).
42. D. Bruder, X. Fu, G. Brent, D. Remy, R. Vasudevan, Data-driven control of soft robots using Koopman operator theory. *IEEE Transact. Robot.* **37**, 948–961 (2021).
43. I. Mezić, On numerical approximations of the Koopman operator. *Mathematics* **10**, 1180 (2022).
44. I. Mezić, Koopman operator, geometry, and learning of dynamical systems. *Not. Am. Math. Soc.* **68**, 1087–1105 (2021).
45. Y. Huang, M. Hofer, R. D'Andrea, Offset-free model predictive control: A ball catching application with a spherical soft robotic arm, in *2021 International Conference on Intelligent Robots and Systems (IROS)* (IEEE/RSJ, 2021), pp. 563–570.
46. B. D. Anderson, J. B. Moore, *Optimal Control: Linear Quadratic Methods* (Courier Corporation, 2007).
47. M. M. Coad, L. H. Blumenschein, S. Cutler, J. A. Reyna Zepeda, N. D. Naclerio, H. El-Hussieny, U. Mehmood, J.-H. Ryu, E. W. Hawkes, A. M. Okamura, Vine robots: Design, teleoperation, and deployment for navigation and exploration. *IEEE Robot. Autom. Mag.* **27**, 120–132 (2020).
48. N. D. Naclerio, E. W. Hawkes, Simple, low-hysteresis, foldable, fabric pneumatic artificial muscle. *IEEE Robot. Autom. Lett.* **5**, 3406–3413 (2020).
49. L. H. Blumenschein, N. S. Usevitch, B. H. Do, E. W. Hawkes, A. M. Okamura, Helical actuation on a soft inflated robot body, in *2018 IEEE International Conference on Soft Robotics (RoboSoft)* (IEEE, 2018), pp. 245–252.

50. A. Kazemipour, O. Fischer, Y. Toshimitsu, K. W. Wong, R. K. Katzschmann, A robust adaptive approach to dynamic control of soft continuum manipulators. arXiv:2109.11388 [cs.RO] (26 February 2021).

Acknowledgments

Funding: This work was supported in part by the NSF grant no. 1935327 and the ARO-MURI W911NF1710306: From Data-Driven Operator Theoretic Schemes to Prediction, Inference, and Control of Systems. Part of the work performed by E.K. was funded via the Fulbright Foundation. **Author contributions:** D.A.H. designed the robots and test apparatus, wrote the manuscript, prepared the movies, and performed the experiments. M.J.B. designed the modeling and control algorithm, prepared the figures, and wrote the manuscript. E.K. designed the electronics, performed the experiments, and wrote the manuscript. A.B.C. ran the experiments, performed the analysis, and prepared the figures. P.C.C. performed experiments. I.M. advised the design of the modeling and control algorithm, the paper, and the experiments. E.W.H. advised the design of the robot, wrote the manuscript, and advised the paper and experiments. **Competing interests:** D.A.H. is the founder and CEO of Vine Medical Inc., which has no financial interest in the subject work, nor did it support the investigation in any way. The other authors declare that they have no competing interests. **Data and materials availability:** All data needed to support the conclusions of this paper can be found in the Supplementary Materials or at https://doi.org/10.5281/zenodo.8184777.

Submitted 28 June 2022
Accepted 2 August 2023
Published 30 August 2023
10.1126/scirobotics.add6864

Science Robotics

Control of soft robots with inertial dynamics

David A. Haggerty, Michael J. Banks, Ervin Kamenar, Alan B. Cao, Patrick C. Curtis, Igor Mezi#, and Elliot W. Hawkes

Sci. Robot. **8** (81), eadd6864. DOI: 10.1126/scirobotics.add6864

Science Robotics (ISSN 2470-9476) is published by the American Association for the Advancement of Science. 1200 New York Avenue NW, Washington, DC 20005. The title *Science Robotics* is a registered trademark of AAAS.