IEEE Robotics and Automation Letters (RAL) paper presented at
2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Kinematics-Informed Neural Networks: Enhancing Generalization Performance of Soft Robot Model Identification

Taerim Yoon[1], Yoonbyoung Chai[1], Yeonwoo Jang[2], Hajun Lee[2], Junghyo Kim[2], Jaewoon Kwon[3], Jiyun Kim[2] and Sungjoon Choi[1]

***Abstract*— A hybrid system combining rigid and soft robots (e.g., soft fingers attached to a rigid arm) ensures safe and dexterous interaction with humans. Nevertheless, modeling complex movements involving both soft and rigid robots presents a challenge. Additionally, the difficulty of obtaining large datasets for soft robots, due to the risk of damage by repetitive and extreme actuations, hiders the utilization of data-driven approaches. In this study, we present a Kinematics-Informed Neural Network (KINN), which incorporates rigid body kinematics as an inductive bias to enhance sample efficiency and provide holistic control for the hybrid system. The model identification performance of the proposed method is extensively evaluated in simulated and real-world environments using pneumatic and tendon-driven soft robots. The evaluation result shows employing a kinematic prior leads to an 80.84% decrease in positional error measured in the L1-norm for extrapolation tasks in real-world tendon-driven soft robots. We also demonstrate the dexterous and holistic control of the rigid arm with soft fingers by opening bottles and painting letters. The codes and dataset are made available[1].**

I. Introduction

Guaranteeing the safety of nearby humans in the face of unexpected accidents is getting more attention as more robots are permeating our daily lives. Hybrid systems comprising both soft and rigid robots can be advantageous for ensuring safe and dexterous interactions. For instance, incorporating a rigid arm with soft fingers can expand the workspace while ensuring interaction safety. However, controlling them poses challenges, stemming from both the complexities of modeling and identifying them [1]. In addition, hybrid systems often rely on separated control of the rigid and soft components, leading to inefficient and uncoordinated motion. In this paper, we present a modeling and identification method applicable to a broad class of soft robots, along with an efficient control methodology for the hybrid system.

Modeling and identifying the kinematics of soft robots are crucial for achieving precise control. However, this process often necessitates delicate and model-specific derivations, relying on equations of motion (e.g., [2]–[4]) or geometry (e.g., [5]–[8]). On the other hand, the black-box approach relies on regression models such as Gaussian process regression [9], [10] or neural networks [11], [12] to bypass modeling complexities. However, implementing the black-box method demands a substantial volume of data, which can potentially harm soft robots due to repetitive and extreme actuations [13]. Moreover, the black-box approach often fails to model the region outside the training data due to overfitting [1]. Thus, it becomes vital to establish a sample-efficient modeling that enables generalization to unseen motor commands.

A recent study on physics-informed modeling (PIM) [14]–[18] showcases their accurate modeling or computational efficient identification by integrating physics principles with black-box models. In this work, we also aim to leverage physics knowledge in modeling and identifying the kinematic behavior of soft robots. To be specific, we model an underlying kinematic structure to achieve the generalizing behavior of the model-based approach while leveraging the black-box nature of neural networks to avoid model-specific derivations. This enables accurate predictions of positions for diverse soft robots with unseen actuation commands, a capability we refer to as extrapolation. Furthermore, modeling the underlying kinematics unifies their modeling with rigid robots, allowing a holistic control of hybrid systems.

To this end, we propose a *Kinematics-Informed Neural Network* (KINN) that models the underlying kinematics of soft robots for two objectives: 1) identifying and 2) controlling the soft robot. We aim to identify the underlying kinematics by iteratively searching for discrete and continuous parameters using a genetic algorithm [19]. The result of the identification is evaluated in simulated and real-world environments using two widely adopted configurations: a pneumatic and a tendon-driven soft robot. The identification results with the tendon-driven soft robot in real-world show an 80.84% reduction of the positional error when employing the kinematics prior to extrapolation tasks. We also experi-

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) and the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. 2021R1A4A3033149). The contributions from the Institute of Information & Communications Technology Planning & Evaluation (IITP) were funded by four separate grants: No. 2019-0-00079, Artificial Intelligence Graduate School Program (Korea University), No. 2022-0-00871, Development of AI Autonomy and Knowledge Enhancement for AI Agent, No. 2022-0-00480, Development of Training and Inference Methods for Goal-Oriented Artificial Intelligence Agents, and No. 2022-0-00612, Geometric and Physical Commonsense Reasoning based Behavior Intelligence for Embodied AI.

[1]Taerim Yoon, Yoonbyoung Chai, and Sungjoon Choi are with the Department of Artificial Intelligence, Korea University, 145 Anam-ro, Seongbuk-gu, Seoul, Korea (email: taerimyoon@korea.ac.kr; yoonbyungchai@korea.ac.kr; sungjoon-choi@korea.ac.kr).

[2]Yeonwoo Jang, Hajun Lee, Junghyo Kim, and Jiyun Kim are with the department of Material Science and Engineering, Ulsan National Institute of Science and Technology, 50 UNIST-gil, Ulju-gun, Ulsan, Korea (email: yjang1@unist.ac.kr; hajuny.lee@gmail.com; kjhsk23@unist.ac.kr; jiyunkim822@gmail.com).

[3]Jaewoon Kwon is with NAVER LABS, Gyeonggi-do, 13561, South Korea. (email: cumuli10@gmail.com)

[1]*https://github.com/terry97-guel/KINN*

ment with the proposed controller in the real-world to follow the end-effector trajectories that only can be reached through extrapolation. Lastly, the complex manipulation tasks of the hybrid system involving painting letters and opening the bottles are demonstrated to showcase the control methods.

II. Related Work

A. Modeling and Identifying Soft Robot

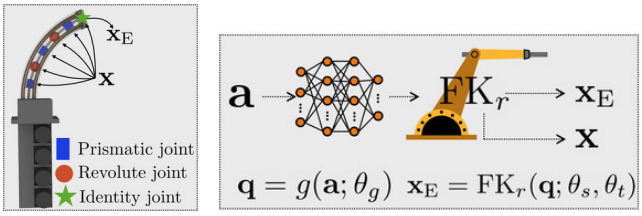
Analyzing soft robots by modeling their material behavior is a commonly embraced approach. Han et al. [20] identified material parameters by deriving their gradients through the modeling of soft robots using the finite-element method (FEM). A recent work of Dubied et al. [2] employed gradient from Projection Dynamics (PD) [21] to identify and control soft robots. However, they still suffer from unstable convergence and require a slow optimization process for control [22]. Furthermore, the limited number of parameters involved in reduced-order modeling [1] can result in an inability to adequately represent the data.

A piece-wise constant curvature (PCC) model [5] provides a simple and computationally efficient framework to model soft robots. Quevedo et al. [23] identified the kinematics of a tendon-driven soft robotic neck with explicit parameterization of the PCC. However, the PCC has a risk of overly reducing the modeling capacity because of its strong assumption of geometry. For instance, one of the recognized issues with the PCC is its inability to model soft robots with complex configurations, such as the Euler spiral [24].

The Cosserat rod model (CR) describes the motion of soft robots through the lengthwise and temporal movement of multi-sectional rods [4]. Nevertheless, both the PCC and CR often require model-specific derivation, which limits their applicability to diverse types of soft robots.

Soft robot can be analyzed through the modeling of the underlying kinematics known as pseudo-rigid bodies (PRB) [12]. Venkiteswaran et al. [25] modeled PRB of a magnetically-actuated continuum manipulator to estimate deformation and contact force. Santina et al. [7] identified and controlled a pneumatic robot with the augmented formulation of a rigid-bodied robot under PCC assumptions. Our work aligns with these studies in modeling soft robots with PRB. In this paper, our primary focus lies in the integration of PRB with neural networks to avoid the need for model-specific derivations while retaining the generalizing behavior.

Black-box methods offer an alternative to modeling soft robots, avoiding the burden of model-specific derivation. By directly approximating forward/inverse kinematic function with the Gaussian process regression, soft robots are controlled to the desired position [9], [10]. Similarly, Bern et al. [11] used a neural network to fit paired data of actuation command and the position of soft robots, solving inverse kinematics using L-BFGS optimizer [26]. Zheng et al. [27] presented the closed loop controller with a neural network that is robust to external disturbance. However, black-box methods can be prone to overfitting, resulting in errors in position control when faced with unseen commands, posing a potential risk to the safety of nearby humans [1].



(a) (b)

Fig. 1. (a) The Pseudo Rigid Body (PRB) of the soft robotic finger is shown, whose position of joints and end-effector are denoted as $\mathbf{x}$ and $\mathbf{x}_E$. (b) Forward Kinematics of the KINN is illustrated.

To improve model learning, informing neural networks with physic principles were studied. Bartholdt et al. [14] utilized a neural network that maps actuation commands to the parameters of a Cosserat rod model for tendon-driven soft robots. Won et al. [28] incorporated a differentiable simulator with a neural network to facilitate the connection of gradient chains and identify the model parameters. Sun et al. [15] incorporated a spring-damper model with recurrent neural networks to model soft actuators. Our study contributes to the existing research field of incorporating the physics prior into neural networks. Specifically, we incorporate neural networks with rigid body kinematics to identify underlying kinematics that allow controlling soft robots and rigid robots simultaneously.

B. Hybrid System

A hybrid system comprising both rigid and soft parts can offer advantages by complementing each other. Uppalapati et al. [29] demonstrated a hybrid system providing both safe interaction and rigid support necessary for harvesting berries. Stokes et al. [30] presented a hybrid system design for efficient locomotion, which utilizes rigid wheels for flat terrain and soft legs for walking in rough terrain and grasping objects. However, existing approaches often rely on separate models for soft robots and rigid robots, which can result in uncoordinated motion. In contrast, we aim to generate coordinated motion by holistic control with a unified model.

III. Methods

A. Modeling

In our study, we model the kinematics of soft robots, leveraging prior knowledge of rigid-body kinematics to enhance efficiency in subsequent tasks involving identification and control. To be specific, we model the underlying kinematics structure, named a pseudo rigid body (PRB) shown in Figure 1a by using the Kinematics-Informed Neural Network (KINN) that relates the actuation commands to the corresponding joint values of the PRB.

Consider a soft robot actuated with n_a actuation commands represented by $\mathbf{a} \in \mathbb{R}^{n_a}$ with n_j end effectors whose true position is denoted as $\mathbf{x}_{\text{gt}} \in \mathbb{R}^{n_j \times 3}$. Forward kinematics, denoted as FK, maps actuation commands $\mathbf{a}$ to the estimated position denoted as $\mathbf{x}_E$. For kinematics parameters denoted as θ, forward kinematics is described as follows:

$$\mathbf{x}_E = \mathrm{FK}(\mathbf{a}; \theta)$$

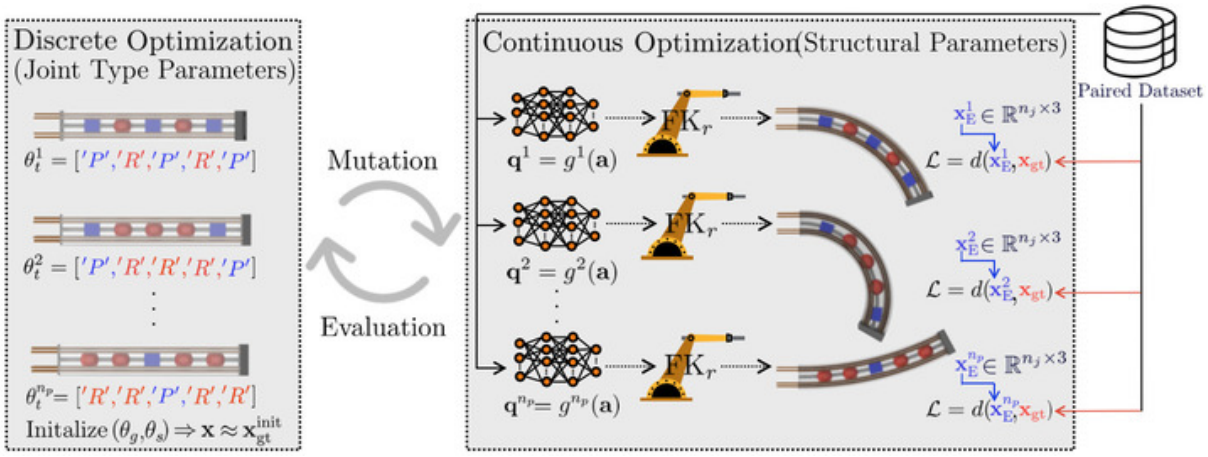


Fig. 2. Process of kinematic architecture search. The joint sequence θ_t is mutated from the archive, followed by continuous optimization over θ_s, θ_g. According to the evaluation, θ_t is selected and archived, completing optimization over discrete variables.

The PRB is modeled with n_t joints, where their joint values are denoted as $\mathbf{q} \in \mathbb{R}^{n_t}$. The joint values $\mathbf{q}$ represent the rotation of revolute joints or the translational movement of prismatic joints. We also denote the position of ith joints as $\mathbf{x} \in \mathbb{R}^{n_t \times 3}$, the positional offset as $p_i^{\text{off}} \in \mathbb{R}^3$, rotational offset as $R_i^{\text{off}} \in SO(3)$, and axis direction as $S_i \in \mathbb{S}^2$ for $i \in [1, n_t]$, where $\mathbb{S}^2$ is 3d-unit sphere and $SO(3)$ is a special orthogonal group which we parameterize using Euler angles. The forward kinematics of the KINN is composed of two layers, as shown in Figure 1b. The actuation $\mathbf{a}$ is initially mapped to joint values $\mathbf{q}$ using a neural network represented as g, with parameters denoted as θ_g. Subsequently, it undergoes rigid-body kinematics denoted as FK_r.

The parameters of FK_r consist of the structural parameter and the joint type parameter, each denoted as θ_s and θ_t. The structural parameter θ_s encodes the axis and offset of the joints in PRB, which is written as $\theta_s = \{p_1^{\text{off}}, R_1^{\text{off}}, S_1, ..., p_{n_t}^{\text{off}}, R_{n_t}^{\text{off}}, S_{n_t}\}$. The joint type parameter θ_t represents a sequence of joint types whose ith element, denoted as θ_t^i, falls into one of four categories: $'P'$ for prismatic joint, $'R'$ for revolute joint, $'F'$ for fixed joint containing only offset and axis information, and $'I'$ for identity joints, placeholders with no additional information.

To account for the complex behavior of soft robots, the number of joints in the PRB, n_t, is often chosen to be sufficiently larger than the number of end-effectors, n_j. For example, the soft robot with $n_j = 1$ end-effector, depicted in Figure 1a, is modeled as a serial chain composed of $n_t = 7$ joints whose joint type sequence is $\theta_t = ['P', 'R', 'P', 'R', 'P', 'I']$.

B. Identification

The model identification process involves finding the optimal $\theta = (\theta_g, \theta_s, \theta_t)$ kinematic parameters that minimize the cost function defined by the distance measure $d(\cdot, \cdot)$. Identifying the kinematic parameters θ is closely related to neural architecture search (NAS) [31], as it involves optimizing discrete parameters and continuous parameters of the model. Similarly, we iteratively optimize continuous parameters and discrete parameters to find the architecture, namely, kinematic architecture search (KAS), as it searches the underlying kinematics as shown in Figure 2.

1) Continuous Optimization of Structural Parameters: We jointly optimize structural parameters and parameters of g, denoted as (θ_s, θ_g). Since all the parameters are differentiable through auto-gradient framework [32], we perform optimization with ADAM [33] as Equation 1.

$$(\hat{\theta}_s, \hat{\theta}_g) = \arg\min_{(\theta_s,\theta_g)} \mathbb{E}_{(\mathbf{a},\mathbf{x}_{\text{gt}})\in D}[d(\mathbf{x}_{\text{gt}}, \text{FK}(\mathbf{a}; \theta_g, \theta_s, \theta_t))] \tag{1}$$

However, naively optimizing with Equation 1 may result in placing the joints of the PRB outside the soft robot as we do not constrain the range of joint values $\mathbf{q}$. This can result in abnormal behaviors, including abrupt changes in position, leading to sub-optimal results in subsequent control tasks. We address this with two effective regularization techniques by leveraging the underlying kinematic structure.

We first collect the rest position of the robot, denoted as $\mathbf{x}_{\text{gt}}^{\text{init}}$, from the soft robot configuration. We then initialize θ_s and θ_g so that $\mathbf{x}_{\text{E}} \approx \mathbf{x}_{\text{gt}}^{\text{init}}$. This is achieved by setting joint values as zeros $q \approx 0_{n_t}$ and the positional offset $p^{\text{off}} \in \theta_s$ to place joints in equal distance from the base position to the initial position $\mathbf{x}_{\text{gt}}^{\text{init}}$. To make $q \approx 0_{n_t}$, we remove the bias term of network g and initialize weights as $\theta_g \sim \mathcal{N}(0, \sigma_{\text{init}}^2)$, $\sigma_{\text{init}} \ll 1$. As a result, the position is initialized to have a straight configuration with $\mathbf{x} \approx \mathbf{x}_{\text{gt}}^{\text{init}}$ as shown in the left-hand side of Figure 2. We introduce another regularizing term to make the joints aligned. Specifically, the joint position $\mathbf{x}$ is constrained so that they are aligned in line using cosine distance as written as follows:

$$\mathcal{L}_{\text{reg}} = \sum_{i=0}^{n_t-1} d_{\text{c}}(\mathbf{x}_{\text{gt}}[k(i)] - \mathbf{x}_{\text{gt}}[\lambda(i)], \mathbf{x}[i+1] - \mathbf{x}[i]) \tag{2}$$

where d_{c} stands for cosine distance, and two indexing functions, denoted as k, and λ, each maps joint index $i \in [1, n_t]$ to index of the closest joints $j \in [0, n_j]$ in the direction of

the kinematic tree's root and leaf, respectively. To elaborate with an example, Figure 1a contains $n_t = 6$ PRB joints and $n_j = 1$ end-effector. Two indexing functions, k and λ, each map any index $i \in [1, n_t]$ to 0 and 1, respectively, corresponding to the base and the end of the soft robot.

2) Discrete Optimization of Joint Type Parameters: We search for the joint type parameters θ_t with the genetic algorithm (GA) [19]. Let n_p denote the number of agents in the archive, and let $\theta_t^i = [{}^1\theta_t^i, {}^2\theta_t^i, \ldots, {}^{n_p}\theta_t^i]$ represent the joint types for agents in the ith generation. The archive maintains a collection of agents with different types of joints that are repeatedly mutated and evaluated. To initiate the process, the first generation θ_t^1 is selected randomly and fed into the continuous optimization algorithm. The agents are evaluated based on the L1 norm of the error on the validation dataset, and a fixed ratio of best-performing agents are selected, bisected at a random position, and merged together until the archive contains n_p agents.

C. Controlling Position and Bending

Our objective is to control the position of a soft robot, denoted as $\mathbf{x}_{\text{tar}}$, with bending direction $\mathbf{u} \in \mathbb{S}^3$ and distance d as shown in Figure 3a, where $\mathbb{S}^3$ is unit 3-sphere. This is achieved by solving optimization problem shown in Equation 3 for action improvement, denoted as $\delta\mathbf{a}$, using the damped least-square method [34], where the control gains are fixed as $K_p = 1$, $K_c = 0.001$ throughout the experiments.

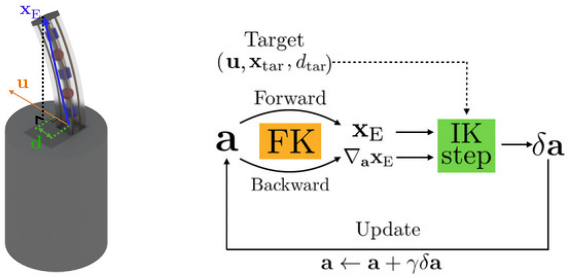
$$\begin{aligned} \delta\mathbf{a} &= \arg\min_{\delta\mathbf{a}} \|e - J\delta\mathbf{a}\|^2 \text{ where,} \\ e &= [K_p(\mathbf{x}_{\text{tar}} - \mathbf{x}_{\text{E}})), K_c(\mathbf{d}_{\text{tar}} - \mathbf{u}^T\mathbf{x}_{\text{E}})]^T, \\ J &= [\nabla_{\mathbf{a}}\mathbf{x}_{\text{E}}, \nabla_{\mathbf{a}}\mathbf{u}^T\mathbf{x}_{\text{E}}]^T \end{aligned} \tag{3}$$

Although it is possible to directly compute $\nabla_{\mathbf{a}}\mathbf{x}_{\text{E}}$ with the auto-gradient framework, the computation can be slow because of the recursive computation during forward and backward computation of FK_r. By leveraging the explicit form of $\nabla_{\mathbf{q}}\mathbf{x}_{\text{E}}$, we only calculate $\nabla_{\mathbf{a}}\mathbf{q}$ with backward computation and $\nabla_{\mathbf{a}}\mathbf{x}_{\text{E}} = \nabla_{\mathbf{a}}\mathbf{q}\nabla_{\mathbf{q}}\mathbf{x}_{\text{E}}$ is obtained by chain rule.

D. Controlling a Hybrid System

We demonstrate simultaneous control of a hybrid system comprising both soft and rigid-bodied robots by naturally extending the control method discussed earlier for soft robots. Consider rigid-bodied parts of a hybrid system with $n_{a^{\text{r}}}$ actuation commands, denoted as $\mathbf{a}^{\text{r}} \in \mathbb{R}^{n_{a^{\text{r}}}}$. The position and rotation (i.e., Euler angles) of the rigid-bodied parts are defined as $\mathbf{x}^{\text{r}}$ and $\mathbf{w}^{\text{r}}$, respectively. We denote each of the target position and rotation as $\mathbf{x}_{\text{tar}}$ and $\mathbf{w}^{\text{r}}_{\text{tar}}$, while the target direction and distance are denoted as $\mathbf{u}$ and d_{tar}. Finally, the inverse kinematics of the hybrid system is solved as Equation 4, by applying damped least-squares.

$$\begin{bmatrix} \nabla_{\mathbf{a}^{\text{r}}}\mathbf{x}_{\text{E}} & \nabla_{\mathbf{a}}\mathbf{x}_{\text{E}} \\ \nabla_{\mathbf{a}^{\text{r}}}\mathbf{w}^{\text{r}} & 0_{3\times n_a} \\ 0_{1\times n_{a^{\text{r}}}} & \mathbf{u}^T J \end{bmatrix} \begin{bmatrix} \delta\mathbf{a}_{\text{r}} \\ \delta\mathbf{a} \end{bmatrix} = \begin{bmatrix} K_p(\mathbf{x}_{\text{tar}} - \mathbf{x}_{\text{E}}) \\ K_w(\mathbf{w}^{\text{r}}_{\text{tar}} - \mathbf{w}^{\text{r}}) \\ K_c(d_{\text{tar}} - \mathbf{u}^T\mathbf{x}_{\text{E}}) \end{bmatrix} \tag{4}$$



(a) (b)

Fig. 3. (a) The bending direction $\mathbf{u}$, bending distance d, and the end-effector of the robot $\mathbf{x}_{\text{E}}$ is shown. (b) Inverse kinematics is solved for the given target, $\mathbf{u}$, $\mathbf{x}_{\text{tar}}$, and d_{tar}, by updating $\mathbf{a}$ with $\delta\mathbf{a}$.

IV. Experiment on Soft Robot Model Identification

In this section, we evaluate the generalization of the proposed model identification method against unseen motor actuation for safe and practical behavior in downstream control problems. Specifically, we evaluated the performance of the proposed method on the tendon-driven soft robotic finger in both simulated and real-world environments, while evaluation with the pneumatic soft robot was conducted solely in a simulated environment. We collected data from these three configurations to train our proposed model, which is divided into interpolation and extrapolation datasets as shown in Figure 4. Specifically, examples with the top 30% action strength, denoted as $\|\mathbf{a}\|_1$, are assigned to the extrapolation datasets, as the position deviates more from the rest pose for large actuation. The remaining 70% constitute the interpolation datasets, which are further randomly subdivided into training, validation, and test datasets in an 8:1:1 ratio.

A. Baselines

We set baseline methods to compare with the proposed method. Each of the baselines is listed below including their acronyms which we will use for the rest of the paper.

1) PureNN: We set a neural network without any physics prior [11] as the baseline to compare the effect of kinematic prior. Two hidden layers of 16 neurons are implemented with a learning rate of 0.0015. The activation function is implemented with Mish in contrast to Sigmoid used in the work of [11], as it provided improved performance for model identification. In addition, the positional data is whitened by subtracting the mean and dividing by the standard deviation.

2) GP: We also conduct a model identification using the Gaussian process regression [10]. Similar to PureNN, whitening is applied to the actuation and position data. We utilized the Squared Exponential (SE) Kernel to fit a regressor to predict the robot's position, setting the length scale as the median distance between data points.

3) PCCNN: To compare with other physics prior, we experiment with a neural network informed with PCC model [16]. PCCNN incorporates the PCC modeling by integrating neural network g_{PCC} and forward kinematics of PCC. Its forward kinematics is composed of two parts. Firstly, g_{PCC}

TABLE I

MODEL IDENTIFICATION RESULTS OF OUR PROPOSED METHOD AND BASELINES ON TWO SIMULATION ENVIRONMENTS AND REAL-WORLD

Method		Pneumatic (Simulated)		Tendon (Simulated)		Tendon (Real-World)	
		Interpolation Error (mm)	Extrapolation Error (mm)	Interpolation Error (mm)	Extrapolation Error (mm)	Interpolation Error (mm)	Extrapolation Error (mm)
Black Box	PureNN [11]	4.605	49.40	2.698	22.93	**2.141**	25.21
	GP [10]	**3.189**	43.67	3.030	6.3531	6.373	12.37
Model-Based	FEM [21]	-	-	4.759	32.903	13.23	16.23
	PCC [5]	-	-	11.555	21.585	10.46	21.03
	Bayes-CR	-	-	41.802	51.063	12.00	32.09
Physics-Informed	PCCNN [16]	306.46	359.28	13.669	22.935	12.38	24.56
	KINN (Ours)	4.983	**10.58**	**1.339**	**3.410**	2.711	**4.831**

predicts PCC parameters ϕ, θ, r which corresponds to plane orientation, rotation angle of arc, and radius of arc [5]. Then, the forward kinematics of the PCC model is performed to obtain the position.

4) FEM: We also compare a recent work of model identification using a differentiable simulator. Specifically, we use Differentiable Projective Dynamics [21] to identify Young's modulus E and Poisson ratio ν using L-BFGS [26].

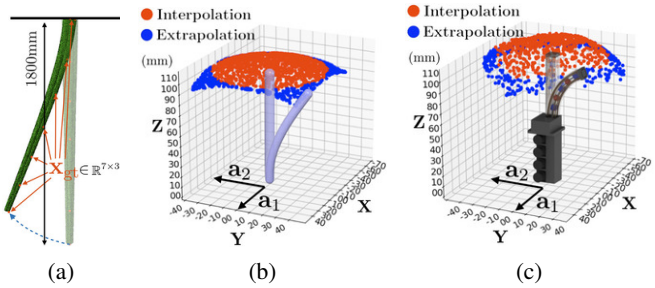
5) PCC: We compare with a model-specific method that uses geometric relationships of tendon-driven soft robots under PCC assumption [5]. Denoting r_p as the pulley radius, l_0 as the initial length, and $\mathbf{a}$ as the rotation angle, the pulled tendon length l_i for the ith actuation is given by $l_i = l_0 - r_p a_i$. The explicit PCC parameters $[\phi, \theta, r]$ are obtained by the geometric relationship between pulled tendon length l and the resulting position [10]. The effective radius r_p and initial length l_0 are searched using L-BFGS.

6) Bayes-CR: The soft robot is simulated using the Cosserat rod model with SoRoSim [35]. The length and Young's Modulus are searched using Bayesian optimization. We fit the surrogate function s using the Gaussian process regression with an SE kernel function to estimate the error, and the acquisition function is implemented with Expected Improvement (EI). We set the scale length of the kernel as the median of the distance between data points, and the exploration rate of the acquisition function as $\xi = 0.01$.

B. Model ID of Pneumatic Soft Robot in Simulation

We simulate a pneumatic soft robot with ABAQUS [36] to gather a highly feasible dataset as shown in Figure 4a. The configuration of the soft robot roughly follows that of [6]. We utilize the Yeoh Hyperelastic Model, characterized by a strain energy density expressed as $U = 0.11(I_1 - 3) + 0.02(I_1 - 3)^2 + \frac{1}{0.1}(J_{\text{el}} - 1)^2$, where I_1 represents the first invariant of the stress tensor, and J_{el} denotes the elastic volume strain. The density is fixed at 1049 kg/m^3. The robot is 1800mm in length and is horizontally divided into three equally spaced sections, where one of the four chambers is selected for actuation in each section. The robot is actuated by three independent chambers with pressure $\mathbf{a} \in \mathbb{R}^3$ ranging from $[0, 0, 0]$ to $[30, 3, 3]$ in kPa making a linear workspace. Seven tracking points were attached to the soft robot to collect 5,644 pairs of actuation commands and positions.

The identification result is evaluated by the error between the predicted position and ground-truth position, denoted as



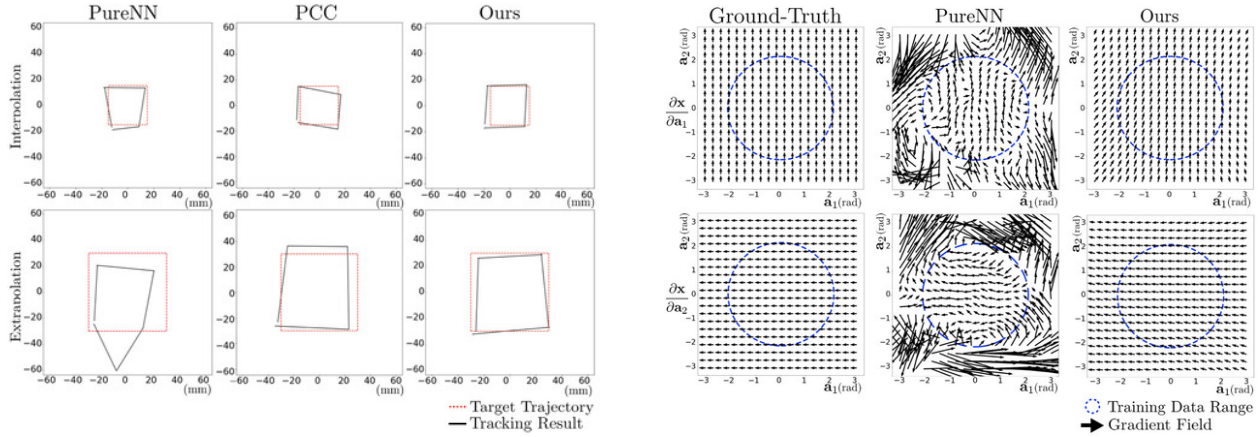
(a) (b) (c)

Fig. 4. Configuration and a workspace of (a) the pneumatic soft robot, tendon-driven soft robotic finger (b) in simulation, and (a) in real-world. Red and blue dots indicate interpolation and extrapolation regions, respectively.

$\|\mathbf{x}_{\text{E}} - \mathbf{x}_{\text{gt}}\|_1$, in both interpolation and extrapolation regions as summarized in Table I. As the employed model of the PCC and Bayes-CR can not describe the pneumatic robot, and FEM suffered non-convergence of the solution, they are excluded from the experiments. For interpolation, our proposed model exhibits competitive performance when compared to the GP, and the difference in the error of 1.794mm is deemed negligible in the context of the total length, which is 1800mm. On the other hand, our model shows significant improvement in extrapolation with a 33.09mm less error compared to the GP, as the kinematics prior plays a substantial role in generalizing.

C. Model ID of Tendon-Driven Soft Finger in Simulation

A tendon-driven soft finger is replicated in a simulated environment using Elastica [37] which enables a more accurate simulation of the Cosserat rod model compared to SoRoSim [35] employed in Bayes-CR, at the cost of increased computational requirements. Figure 4b shows the configuration of the soft finger whose length is 120mm and contains four tendons. The material parameters are set as Young's modulus with 3×10^4 kPa, Poission's Ratio of 0.50, and density $1000\text{kg}/\text{m}^3$. Two independent actuation commands denoted as $\mathbf{a}_1$ and $\mathbf{a}_2$, control four tendons where a negative value means pulling the tendon located on the opposite side. The actuation commands are sampled from a uniform distribution over $\mathbf{a} \in [-0.4, 0.4]$ in $\text{N} \cdot \text{m}$ to collect 2,000 pairs of data points. The identification results are summarized in Table I. Compared to PureNN, leveraging kinematics prior resulted in a decrease in errors of 1.0423mm and 20.181mm for interpolation and extrapolation. Furthermore,



(a) Trajectory Tracking (b) Gradient Fields

Fig. 5. (a) Desired trajectories and tracking results of the end-effector are shown for interpolation and extrapolation tasks. (b) The gradient fields $\frac{\partial \mathbf{x}_E}{\partial \mathbf{a}}$ of the KINN and PureNN are shown. The PureNN accurately models within the training data domain but fails to generalize to the extrapolation region. In contrast, the KINN accurately predicts the position and gradient field outside the training data domain.

TABLE II
TRAJECTORY FOLLOWING RESULT

Measure	Position Error in Trajectory Following (mm)			Cosine Distance in Gradient Prediction	
Model	PureNN	PCC	Ours	PureNN	Ours
Interpolation	2.773	2.374	**2.295**	0.1856	**0.01309**
Extrapolation	12.99	5.093	**2.690**	1.2369	**0.03088**

our proposed method outperformed the one incorporating the PCC prior with 12.005mm and 23.051mm less error for interpolation and extrapolation, respectively.

D. Model ID of Tendon-Driven Soft Finger in Real-World

Our model identification is tested on the tendon-driven soft finger in real-world environments. The soft finger is fabricated by printing its mold from a 3D printer and polymerizing it with Ecoflex 00–30 (Smooth-On). The actuation system is similar to the one from Section IV-C, where each of the four tendons (Nitinol wire, 0.5mm, Alfa Aesar) is attached to a motor (Dynamixel XC-330, Robotis). Five motion capture devices (FLEX13, Opti-Track) are used to collect 1259 paired data points of rotation angle $\mathbf{a}$ of the motor and the end-effector position $\mathbf{x}_E$ where $\mathbf{a} \in [-\pi, \pi]$ in rad. The collected data of interpolation and extrapolation are shown in Figure 4c.

Our proposed model exhibits comparable performance for interpolation task to PureNN, where the difference is negligible considering 2.1mm of the intrinsic error[2] due to hysteresis and sensors. On the other hand, our method significantly outperforms PureNN in the extrapolation task, showcasing improvement in the error of 20.379mm, which accounts for 80.84% of the error associated with PureNN. These properties enable effective control in the extrapolation region, as will be demonstrated in Section V.

[2]To calculate the intrinsic error, the soft robot is repeatedly reset and actuated with the fixed point of $\mathbf{a} = (\pi/2 \text{ rad}, 0)$ and standard deviation was calculated.

V. EXPERIMENT ON SOFT ROBOT CONTROL

A. End-Effector Trajectory Tracking Task

We assess the generalization performance using a single tendon-driven soft finger from Section IV-D. The KINN is trained exclusively using interpolation data and deployed to track trajectories within and outside the scope of the training data. The resulting trajectory is observed by motion capture devices, and we compare the tracking error between the proposed model and PureNN with no inductive bias [11]. We also compare with the learned PCC model from Section IV-D, where actuation is searched using L-BFGS [26].

Figure 5a shows that both the KINN and PureNN can follow the trajectory within the training region with the error of 2.295mm and 2.773mm where performance gap is negligible considering 2.1mm intrinsic error from Section IV-D. However, for the extrapolation region, PureNN fails with the error of 12.99mm. In contrast, the KINN successfully follows the trajectory in the extrapolation region with the error of 2.690mm. We also observe a meaningful decrease in the error of $2.403\ \text{mm}$ when compared to the PCC model.

It is essential to accurately predict gradient fields as the control methods exploit them. Figure 5b shows the ground-truth and predicted gradient fields generated by the KINN and PureNN, respectively. We exclude PCC as it is derived in a model-specific manner to precisely model the ground-truth gradient fields. The ground-truth gradient fields are derived from structural and actuation principles of this specific robot where the two actuation commands, $\mathbf{a_1}$ and $\mathbf{a_2}$, each generate a movement along the $+y$ and $-x$ axes, as illustrated in Figure 4c. The predicted gradient fields are obtained by using auto-gradient [32]. We measure the cosine distance between the learned gradient fields and the ground-truth as only the direction of the ground-truth gradient fields are given. Our model significantly outperforms PureNN in both interpolation and extrapolation domains with a difference of 0.1725 and 1.2060, respectively.

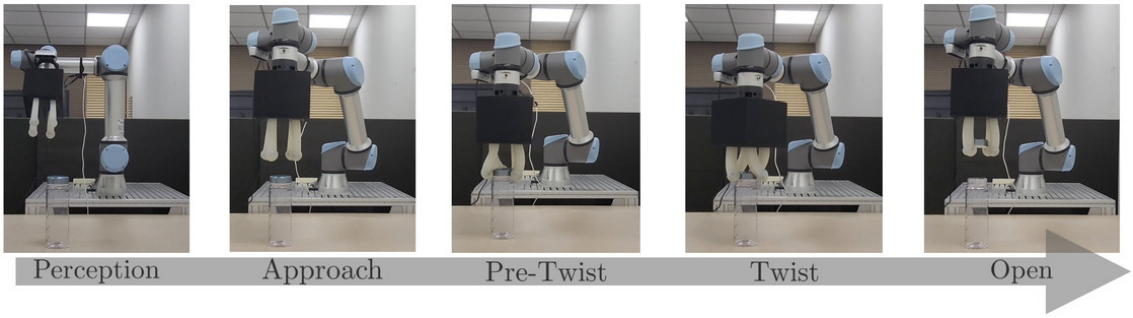


Fig. 6. Primitives to open the bottle cap. (a) The target object is located using the segmentation module. (b) The robot approaches the top of the bottle. (c) Finger and arm pose pre-twisting motion. (d) Twist with finger and arm. (e) Lift the cap to open.

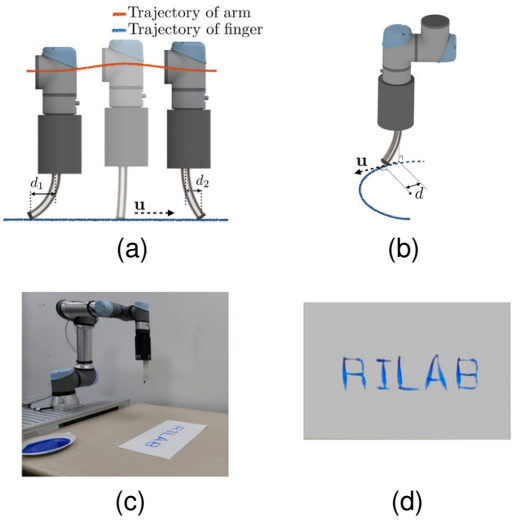


Fig. 7. (a) Illustration of (a) linear stroke and (b) circular stroke. (c) The hybrid system of UR-5e and soft robotic finger painting in a real-world environment and (d) the painted letters.

B. Letter Painting Task

We demonstrate the coordinated movement of the hybrid system as shown in Figure 7c. In detail, the system is composed of a UR-5e manipulator and a soft robotic finger where the brush is glued to the end using Sil-Poxy (Smooth-On). The hybrid system is controlled to paint letters with a brush using two primitives, namely, *Linear Stroke* and *Circular Stroke*, as illustrated in Figure 7a and Figure 7b, respectively. The linear stroke is used to paint straight lines by bending a finger with a distance of d to interpolate between d_1 and d_2 at the start and end of the motion, respectively, with the constant direction of $\mathbf{u}$. The circular stroke draws a circle by aligning the bending direction $\mathbf{u}$ tangential to the circle. By combining two primitives, it generates accurate movements to smoothly stroke with the brush and paint letters as shown in Figure 7d.

C. Bottle Cap Opening Task

Opening the bottle requires dexterous motion and power to make frictional contact to firmly grasp and twist the bottle cap. We demonstrate the capability to effectively coordinate the motion of a rigid arm and four soft robotic fingers, arranged in a square formation, to successfully open a bottle cap in real-world by utilizing the *twist-primitive*, illustrated in Figure 6, to integrate their movements holistically.

The twist-primitive involves capturing and extracting the point clouds of the target object using a depth camera (RealSense-d435) and the segmentation module from [38]. The center of the object is calculated using the median of the point clouds collected in three-dimensional space. The hybrid system then approaches 10cm above the target object. Next, pre-twist motion and twisting motion are performed to rotate 67.5 of the bottle cap. During the twisting motion, the bending direction $\mathbf{u}$ is set as the direction towards the center of the bottle cap, and the bending distance d is set to be 1/3 of the radius of the perceived bottle cap. By repeating the two aforementioned motions twice, the hybrid system successfully opens the water bottle. Our model identification provides generalization performance so that planning outside of the training data is possible where the maximum rotation of the deployed motion, measured in L1-norm $\|(\mathbf{a}_1, \mathbf{a_2})\|_1$, is 1.168π rad exceeding π rad of training data by 16.8%.

D. Limitations

While the proposed method is effective in reducing the reliance on extreme actuation commands, additional investigation is necessary to address potential changes in material properties induced by actuation. In this work, the elongation of the embedded tendons was partly solved by employing super-elastic tendons and careful calibration for the upright pose.

VI. Conclusion

We propose a Kinematics-Informed Neural Network (KINN) to model soft robots with pseudo-rigid bodies. This framework enables the adoption of rigid body kinematics to soft robots for sample-efficient model identification and control, along with the holistic control of both soft and rigid robots. Our experimental findings demonstrate the generalizing capability of the proposed method by effectively identifying and controlling the soft robots within and beyond the training data. Moreover, the proposed method is demonstrated to generate the coordinated motion of rigid robots and soft robots to paint letters and open bottles.

REFERENCES

[1] Thomas George Thuruthel, Yasmin Ansari, Egidio Falotico, and Cecilia Laschi. Control strategies for soft robotic manipulators: A survey. *Soft Robotics*, 5(2):149–163, 2018. PMID: 29297756.

[2] Mathieu Dubied, Mike Yan Michelis, Andrew Spielberg, and Robert Kevin Katzschmann. Sim-to-Real for Soft Robots Using Differentiable FEM: Recipes for Meshing, Damping, and Actuation. *IEEE Robotics and Automation Letters*, 7(2):5015–5022, April 2022.

[3] Ke Wu and Gang Zheng. FEM-Based Nonlinear Controller for a Soft Trunk Robot. *IEEE Robotics and Automation Letters*, 7(2):5735–5740, April 2022.

[4] Federico Renda, Frederic Boyer, Jorge Dias, and Lakmal Seneviratne. Discrete Cosserat Approach for Multisection Soft Manipulator Dynamics. *IEEE Transactions on Robotics*, 34(6):1518–1533, December 2018.

[5] Robert J. Webster and Bryan A. Jones. Design and Kinematic Modeling of Constant Curvature Continuum Robots: A Review. *The International Journal of Robotics Research*, 29(13):1661–1683, November 2010.

[6] Andrew D. Marchese and Daniela Rus. Design, kinematics, and control of a soft spatial fluidic elastomer manipulator. *The International Journal of Robotics Research*, 35(7):840–869, June 2016.

[7] Cosimo Della Santina, Robert K. Katzschmann, Antonio Biechi, and Daniela Rus. Dynamic control of soft robots interacting with the environment. In *2018 IEEE International Conference on Soft Robotics (RoboSoft)*, pages 46–53, Livorno, April 2018. IEEE.

[8] Valentin Falkenhahn, Alexander Hildebrandt, Rudiger Neumann, and Oliver Sawodny. Model-based feedforward position control of constant curvature continuum robots using feedback linearization. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 762–767, Seattle, WA, USA, May 2015. IEEE.

[9] Ge Fang, Xiaomei Wang, Kui Wang, Kit-Hang Lee, Justin D. L. Ho, Hing-Choi Fu, Denny Kin Chung Fu, and Ka-Wai Kwok. Vision-Based Online Learning Kinematic Control for Soft Robots Using Local Gaussian Process Regression. *IEEE Robotics and Automation Letters*, 4(2):1194–1201, April 2019.

[10] Peiyi Wang, Zhiqiang Tang, Wenci Xin, Zhexin Xie, Sheng Guo, and Cecilia Laschi. Design and Experimental Characterization of a Push-Pull Flexible Rod-Driven Soft-Bodied Robot. *IEEE Robotics and Automation Letters*, 7(4):8933–8940, October 2022.

[11] James M. Bern, Yannick Schnider, Pol Banzet, Nitish Kumar, and Stelian Coros. Soft Robot Control With a Learned Differentiable Model. In *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*, pages 417–423, New Haven, CT, USA, May 2020. IEEE.

[12] Pierre Schegg and Christian Duriez. Review on generic methods for mechanical modeling, simulation and control of soft robots. *PLOS ONE*, 17(1):e0251059, January 2022.

[13] Curtis C. Johnson, Tyler Quackenbush, Taylor Sorensen, David Wingate, and Marc D. Killpack. Using First Principles for Deep Learning and Model-Based Control of Soft Robots. *Frontiers in Robotics and AI*, 8:654398, May 2021.

[14] Max Bartholdt, Mats Wiese, Moritz Schappler, Svenja Spindeldreier, and Annika Raatz. A Parameter Identification Method for Static Cosserat Rod Models: Application to Soft Material Actuators with Exteroceptive Sensors. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 624–631, Prague, Czech Republic, September 2021. IEEE.

[15] Wentao Sun, Nozomi Akashi, Yasuo Kuniyoshi, and Kohei Nakajima. Physics-Informed Recurrent Neural Networks for Soft Pneumatic Actuators. *IEEE Robotics and Automation Letters*, 7(3):6862–6869, July 2022.

[16] Silvia Terrile, Andrea López, and Antonio Barrientos. Use of Finite Elements in the Training of a Neural Network for the Modeling of a Soft Robot. *Biomimetics*, 8(1):56, January 2023.

[17] Sourav Sanyal and Kaushik Roy. RAMP-Net: A Robust Adaptive MPC for Quadrotors via Physics-informed Neural Network. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1019–1025, London, United Kingdom, May 2023. IEEE.

[18] Jonas Nicodemus, Jonas Kneifl, Jörg Fehr, and Benjamin Unger. Physics-informed Neural Networks-based Model Predictive Control for Multi-link Manipulators. *IFAC-PapersOnLine*, 55(20):331–336, 2022.

[19] Sourabh Katoch, Sumit Singh Chauhan, and Vijay Kumar. A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*, 80(5):8091–8126, February 2021.

[20] David Hahn, Pol Banzet, James M. Bern, and Stelian Coros. Real2Sim: visco-elastic parameter estimation from dynamic motion. *ACM Transactions on Graphics*, 38(6):1–13, December 2019.

[21] Tao Du, Kui Wu, Pingchuan Ma, Sebastien Wah, Andrew Spielberg, Daniela Rus, and Wojciech Matusik. DiffPD: Differentiable Projective Dynamics. *ACM Transactions on Graphics*, 41(2):1–21, April 2022.

[22] James Walker, Thomas Zidek, Cory Harbel, Sanghyun Yoon, F. Sterling Strickland, Srinivas Kumar, and Minchul Shin. Soft Robotics: A Review of Recent Developments of Pneumatic Soft Actuators. *Actuators*, 9(1):3, January 2020.

[23] Fernando Quevedo, Jorge Munoz Yanez-Barnuevo, Juan A. Castano, Concepcion A. Monje, and Carlos Balaguer. Model Identification of a Soft Robotic Neck. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8640–8645, Las Vegas, NV, USA, October 2020. IEEE.

[24] Cosimo Della Santina and Daniela Rus. Control Oriented Modeling of Soft Robots: The Polynomial Curvature Case. *IEEE Robotics and Automation Letters*, 5(2):290–298, April 2020.

[25] Venkatasubramanian Kalpathy Venkiteswaran, Jakub Sikorski, and Sarthak Misra. Shape and contact force estimation of continuum manipulators using pseudo rigid body models. *Mechanism and Machine Theory*, 139:34–45, September 2019.

[26] Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528, August 1989.

[27] Gang Zheng, Yuan Zhou, and Mingda Ju. Robust control of a silicone soft robot using neural networks. *ISA Transactions*, 100:38–45, May 2020.

[28] Manu Lahariya, Craig Innes, Chris Develder, and Subramanian Ramamoorthy. Learning physics-informed simulation models for soft robotic manipulation: A case study with dielectric elastomer actuators. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11031–11038, Kyoto, Japan, October 2022. IEEE.

[29] Naveen Kumar Uppalapati, Benjamin Walt, Aaron Havens, Armeen Mahdian, Girish Chowdhary, and Girish Krishnan. A Berry Picking Robot With A Hybrid Soft-Rigid Arm: Design and Task Space Control. In *Robotics: Science and Systems XVI*. Robotics: Science and Systems Foundation, July 2020.

[30] Adam A. Stokes, Robert F. Shepherd, Stephen A. Morin, Filip Ilievski, and George M. Whitesides. A Hybrid Combining Hard and Soft Robots. *Soft Robotics*, 1(1):70–74, March 2014.

[31] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search.

[32] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[33] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. 2014. Publisher: arXiv Version Number: 9.

[34] S. Chiaverini, B. Siciliano, and O. Egeland. Review of the damped least-squares inverse kinematics with experiments on an industrial robot manipulator. *IEEE Transactions on Control Systems Technology*, 2(2):123–134, June 1994.

[35] Anup Teejo Mathew, Ikhlas Ben Hmida, Costanza Armanini, Frederic Boyer, and Federico Renda. SoRoSim: A MATLAB Toolbox for Hybrid Rigid–Soft Robots Based on the Geometric Variable-Strain Approach. *IEEE Robotics & Automation Magazine*, 30(3):106–122, September 2023.

[36] Michael Smith. *ABAQUS/Standard User's Manual, Version 6.9*. Dassault Systèmes Simulia Corp, United States, 2009.

[37] Noel Naughton, Jiarui Sun, Arman Tekinalp, Tejaswin Parthasarathy, Girish Chowdhary, and Mattia Gazzola. Elastica: A Compliant Mechanics Environment for Soft Robotic Control. *IEEE Robotics and Automation Letters*, 6(2):3389–3396, April 2021.

[38] Yu Xiang, Christopher Xie, Arsalan Mousavian, and Dieter Fox. Learning RGB-D Feature Embeddings for Unseen Object Instance Segmentation.