Accounting for Hysteresis in the Forward Kinematics of Nonlinearly-Routed Tendon-Driven Continuum Robots via a Learned Deep Decoder Network

Brian Y. Cho[1], Daniel S. Esser[2], Jordan Thompson[1], Bao Thach[1],
Robert J. Webster III[2], and Alan Kuntz[1]

Abstract—Tendon-driven continuum robots have been gaining popularity in medical applications due to their ability to curve around complex anatomical structures, potentially reducing the invasiveness of surgery. However, accurate modeling is required to plan and control the movements of these flexible robots. Physics-based models have limitations due to unmodeled effects, leading to mismatches between model prediction and actual robot shape. Recently proposed learning-based methods have been shown to overcome some of these limitations but do not account for hysteresis, a significant source of error for these robots. To overcome these challenges, we propose a novel deep decoder neural network that predicts the complete shape of tendon-driven robots using point clouds as the shape representation, conditioned on prior configurations to account for hysteresis. We evaluate our method on a physical tendon-driven robot and show that our network model accurately predicts the robot's shape, significantly outperforming a state-of-the-art physics-based model and a learning-based model that does not account for hysteresis.

Index Terms—Surgical Robotics: Steerable Catheters/Needles, Deep Learning Methods, Medical Robots and Systems

I. INTRODUCTION

CONTINUUM robots have garnered significant attention for medical applications due to a high degree of dexterity and their flexible nature—enabling them to perform precise and intricate movements in complex and narrow anatomical structures [1], [2]. The inherent flexibility of continuum robots has the potential to minimize the invasiveness of surgery, mitigating one of the problems of open surgery–the overall trauma to the patient. Tendon-driven continuum robots are a type of continuum robot actuated by tendons routed along the length of the backbone [3], [4]. When the tendons are pulled, they apply forces and moments to the backbone, causing the
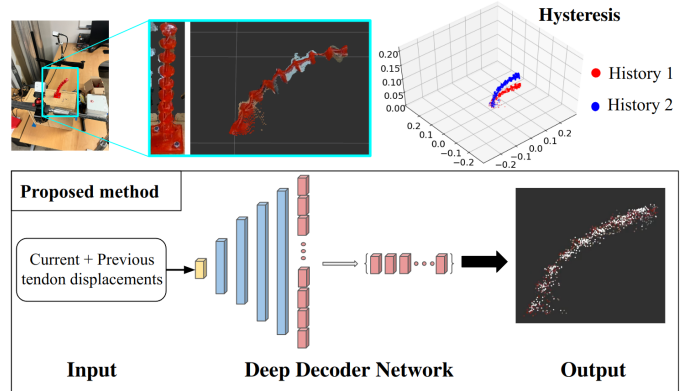
Fig. 1: (Upper left) We leverage point clouds to represent the whole shape of the robot. (Upper right) Hysteresis causes the robot's shape to significantly depend on its prior configuration history. History 1 and History 2 show the sensed shape of the robot at the same tendon-displacement configuration, but having come from different prior configurations. (Bottom) Our proposed deep decoder neural network model aims to learn the tendon robot's entire shape while accounting for hysteresis. Taking as input the current and previous tendon configurations it produces a point cloud (white) that well aligns with the ground truth (red).

robot to bend and take curved shapes, based on its backbone compliance and the tendon routing.

Accurate mechanical models are necessary to plan and control the complex movements of these flexible robots. Physics-based mechanical modeling approaches for tendon-driven robots, e.g., [5], [6], are unable to take into account unmodeled effects such as anisotropic or non-homogeneous material properties and unpredictable friction, leading to a mismatch between model predictions and the actual shape of a physical robot. Machine learning-based models, e.g., [7], [8], have been shown to be able to learn from data to predict robot shape well. However, to date these methods only map a single current configuration to the robot's shape. Continuum robots, including tendon-driven robots, suffer from hysteresis, in which the shape is dependent not only on the current configuration but on prior configurations [9], [10] (see Fig. 1). These concerns make it difficult to predict a robot's shape for either physics-based or learned methods, which in turn can lead to inaccurate planning and control of these robots.

In this work, we propose a deep neural network-based method that learns to predict the entire shape of a tendon-driven robot given both its current configuration, i.e., tendon displacements, as well as the prior configuration. Conditioning the shape model on both the robot's current configuration and

its previous configuration enables the model to account for hysteresis and significantly improves the accuracy of the shape prediction.

As an additional benefit, our method outputs a geometrical representation of the robot's shape *directly* in the form of a point cloud—a representation widely used in other domains for object geometric shape representation [11]–[13]. This is in contrast to existing methods, e.g., [8], which output parameterizations of shape such as the robot's backbone. These parameterizations require additional, time-consuming computation to describe the robot's full shape in order to be useful for control or planning via collision detection or task-space metric evaluation. The concept is particularly well-motivated when considering point clouds, where fast nearest-neighbor data structures and computational geometry methods enable computationally-efficient processing of point clouds for collision-detection and clearance calculations [14], [15]. Our method's point-cloud output is enabled by our deep-decoder-based neural network architecture as well as a novel point-cloud-based loss function. We have designed our approach to be independent of the particular tendon routing or robot prototype and envision that our method has the potential to be utilized in the various surgical and interventional applications for which robots of this type have been proposed. This includes upper airway surgery, thoracic surgery, laparoscopic surgery in the abdomen, gynecological surgery, and urological surgery.

We evaluate the effectiveness of our proposed method in improving the shape prediction accuracy on a physical tendon-driven robot with both linear and non-linearly routed tendons. We show significant improvement in shape prediction compared to a state-of-the-art physics-based mechanical model. We demonstrate the method's ability to successfully account for hysteresis by comparing it against a version that is not conditioned on prior configurations. The results show that our deep decoder network model accurately predicts the robot's shape and successfully accounts for hysteresis.

II. BACKGROUND AND RELATED WORK

Tendon-driven robots are a class of continuum robots that use cables or tendons to control their motion. They have been widely used in various applications, including surgical robotics [1], [2], [4], [16]. Tendon-driven robots display high dexterity and flexibility, enabling a large workspace and the ability to navigate through confined spaces and curve around anatomical obstacles in the human body. Additionally, they can be designed to be lightweight and compact, which makes them suitable for applications where size and weight are a concern [17], [18]. One of the major challenges in tendon-driven robots is accurately modeling their complex mechanical behavior. This is largely due to non-linear and difficult to model factors such as hysteresis, friction, and anisotropic material properties, resulting in inaccuracies in predicting the shape and motion of tendon-driven robots.

To address this challenge, research effort has been made to handle many unmodeled effects in tendon-driven robots [9], [10], [19], [20]. Of particular concern is a phenomenon displayed by tendon-driven robots called hysteresis [9], [10], [21]–[23]. This manifests as a dependence in the forward

kinematics computation not only on current configuration but on past configurations or paths. Hysteresis occurs in tendon-driven robots due to multiple potential factors, e.g., friction between the tendons and the robot, actuator slack, and tension irregularities.

Researchers have proposed various modeling techniques for continuum robots, from physics-based models, both analytical and numerical [5], [19], [24], to machine learning-based approaches [7], [8], [25]–[28]. Rucker *et al.* [5] present a physics-based model that leverages Cosserat rod theory to model the mechanics of tendon-driven continuum robots. Tendon friction is a known issue in such robots that can introduce hysteretic behaviour in these devices, and various approaches have been proposed to model such effects in catheter-like devices with straight tendons [29], [30], however modeling these effects in generally routed (i.e., not necessarily linearly-routed) continuum robots is an open challenge. Data-driven methods have been leveraged to compute the inverse kinematics [31], [32] and forward kinematics [7], [33]–[36] of continuum robots. However these methods do not consider the full robot shape, rather only computing information about the robot's tip—which may be insufficient for applications requiring, e.g., obstacle avoidance. Kuntz *et al.* [8] propose a forward kinematics method for learning the entire backbone shape of concentric tube robots, however the method requires additional computation to recover the full robot geometry. Further, none of these learned methods account for hysteresis. We build upon these methods in this work.

III. PROBLEM FORMULATION

We consider a tendon-driven robot consisting of a flexible backbone of length l with N tendons arbitrarily, e.g., potentially *non-linearly* [5], routed along its length, with reference base frame $\mathcal{B}$. By pulling a tendon i at the robot base, the compliant backbone bends, and the entire robot changes its shape according to the tendon routing and the tendon displacement defined by q_i for tendon i. We then define the robot's configuration as a vector $\mathbf{q} = [q_i : i = 1, \cdots, N] \in \mathbb{R}^N$. Furthermore, to enable the consideration of hysteresis, we define an augmented *hysteresis configuration* vector $\boldsymbol{\eta}$ which pairs the prior robot configuration along with the current configuration, $\boldsymbol{\eta} = [\mathbf{q}_{\mathrm{prior}}, \mathbf{q}_{\mathrm{current}}] \in \mathbb{R}^{2N}$.

We define the robot's shape as $\mathbf{p}$, noting that $\mathbf{p}$ can take the form of any sufficiently descriptive shape representation. We next define a function that maps the robot configuration, conditioned on the prior configuration, to the robot's shape as the forward kinematics function $\mathcal{F}_{FK} : \boldsymbol{\eta} \rightarrow \mathbf{p}$. The problem then becomes to approximate $\mathcal{F}_{FK}$ as closely as possible.

IV. METHOD

To solve this problem, our method leverages a learning-based approach to map hysteresis configurations to the robot's shape in the form of a point cloud. Specifically, we propose a novel deep decoder neural network model that maps an $\boldsymbol{\eta} = [\mathbf{q}_{\mathrm{prior}}, \mathbf{q}_{\mathrm{current}}]$ to a point cloud $\hat{\mathbf{p}}$ consisting of M 3D points such that $\hat{\mathbf{p}} \in \mathbb{R}^{3 \times M}$, representing the complete surface geometry of the robot. The goal is to output $\hat{\mathbf{p}}$ such that it closely approximates the true shape of the robot $\mathbf{p}$ after
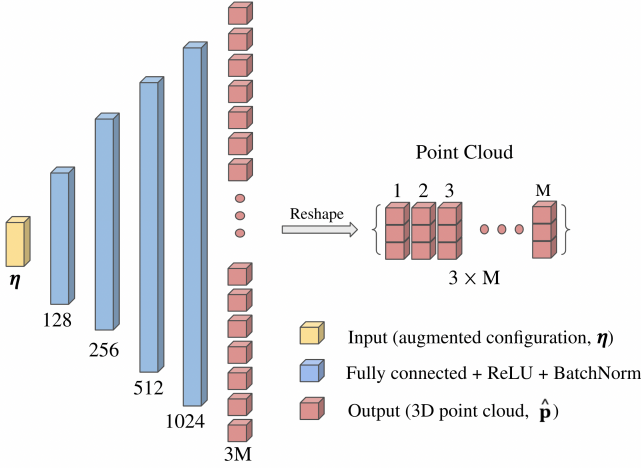
Fig. 2: Network architecture of our novel deep decoder network. The model takes the robot's augmented configuration vector $\boldsymbol{\eta}$ as input and outputs a point cloud of the robot's shape $\hat{\mathbf{p}}$. The model consists of 4 hidden layers, each of which is fully-connected, ReLU activated, and batch normalized followed by an output layer of size $3M$. We reshape the network output (1-dimensional vector) into a $3 \times M$ matrix, where M is the number of points, as the point cloud representation.

having started in $\mathbf{q}_{\text{prior}}$ and then transitioned to $\mathbf{q}_{\text{current}}$, i.e., having executed the two configurations in $\boldsymbol{\eta}$ in sequence. Note that our approach is designed to be an accurate hysteresis-informed computational model that is utilized by a motion planner prior to deployment and assists in open-loop control during deployment.

Our deep decoder network model architecture is presented in Fig 2. The model takes as input $\boldsymbol{\eta}$. This is then passed through four fully-connected layers of increasing size, each with rectified linear unit (ReLU) activation functions [37] and batch normalization [38]. This is then connected to an output layer defining a 1-dimensional vector of size $3M$. We reshape the network output into the point cloud as a $3 \times M$ matrix, a general form of point clouds, where M is the number of points.

We present a novel loss function $\mathcal{L}_{tendon}$ that enables our method to learn the complex and detailed 3D geometry of the tendon-driven robot. $\mathcal{L}_{tendon}$ blends two popular correspondence-free point cloud distance metrics, Chamfer distance (CD) and Earth Mover's Distance (EMD). Chamfer distance C measures the distance between two sets of points (e.g., $\mathbf{p}_a$ and $\mathbf{p}_b$) by summing the distances between each point in one set and its closest neighbor in the other set:

$$C(\mathbf{p}_a, \mathbf{p}_b) = \sum_{x \in \mathbf{p}_a} \min_{y \in \mathbf{p}_b} ||x - y||^2 + \sum_{y \in \mathbf{p}_b} \min_{x \in \mathbf{p}_a} ||x - y||^2.$$

By contrast, EMD, defined here as the function E, measures the distance between two point distributions via:

$$E(\mathbf{p}_a, \mathbf{p}_b) = \min_{\xi : \mathbf{p}_a \to \mathbf{p}_b} \sum_{x \in \mathbf{p}_a} ||x - \xi(x)||_2,$$

where ξ is a bijection pairing points in $\mathbf{p}_a$ with points in $\mathbf{p}_b$. Intuitively, EMD computes the minimum amount of work required to transform one point set into the other, where the work represents the amount of mass moved times the distance it is moved. Our novel loss function is the linear combination, with scaling factor λ (which we set experimentally as discussed in

Sec. V-C), of these two distance metrics between the predicted and ground-truth point clouds:

$$\mathcal{L}_{tendon} = C + \lambda \cdot E.$$

Intuitively for our problem, CD encourages the coarse geometry of our model's predicted point clouds to be similar to that of the ground truth while EMD ensures that the predicted point clouds have a similar distribution to the ground truth. Adding EMD on top of CD helps refine the 3D geometry as well as enforce the points to be evenly distributed on the surface of the tendon robot.

V. DATA COLLECTION AND MODEL TRAINING

To successfully train our method, we must first collect a diverse set of hysteresis configurations and pair each with a dense and accurate point cloud representing the shape of the robot. In this section we first describe how we capture dense and accurate point clouds of the robot's shape and the process by which we segment the robot in the point cloud, separating it from the background. We next describe the process by which we collect a data set of diverse hysteresis configurations using the point cloud capturing method. Finally we describe the training process for our model that leverages the data set.

A. Registration and Segmentation

To collect point cloud data for a given robot shape, we leverage two point cloud sensors pointed at the robot from different angles (see Fig. 3). These sensors collect the point clouds $\mathbf{p}_{\text{cam1}}$ and $\mathbf{p}_{\text{cam2}}$[1]. We seek to unify these two point clouds into a single dense point cloud. To calibrate the coordinate systems of the cameras to the robot base frame $\mathcal{B}$, we leverage the commonly-used correspondence-based registration algorithm of [39]. We compute two homogeneous transformation matrices $\mathbf{T}_1^{\mathcal{B}} : \mathbf{p}_{\text{cam1}} \to \mathcal{B}$ and $\mathbf{T}_2^1 : \mathbf{p}_{\text{cam2}} \to \mathbf{p}_{\text{cam1}}$. We then calculate the transformation matrix $\mathbf{T}_2^{\mathcal{B}} = \mathbf{T}_1^{\mathcal{B}} \mathbf{T}_2^1$. Using the resulting transformation matrices, we align the two point clouds $\mathbf{p}_{\text{cam1}}$ and $\mathbf{p}_{\text{cam2}}$ with the robot base frame $\mathcal{B}$ (see Fig 3). After calibrating the coordinate systems of the two cameras to the robot base frame, we segment the tendon-driven robot, removing the points associated with the background and leaving only the points in $\mathbf{p}_{\text{cam1}}^{\mathcal{B}}$ and $\mathbf{p}_{\text{cam2}}^{\mathcal{B}}$ that correspond to the robot. With the two camera frames registered and the robot segmented, we can concatenate $\mathbf{p}_{\text{cam1}}^{\mathcal{B}}$ and $\mathbf{p}_{\text{cam2}}^{\mathcal{B}}$ to obtain a complete representation of the robot's shape from both camera views, i.e., $\mathbf{p} = [\mathbf{p}_{\text{cam1}}^{\mathcal{B}}, \mathbf{p}_{\text{cam2}}^{\mathcal{B}}]$ (see Fig 3). Note that while we use two cameras, the process above could be used with just one, or with more than two with minimal changes.

B. Hysteresis configuration data set generation

To train our model in a supervised fashion where the predicted outputs are compared against the ground-truth point clouds, we collect a data set $\mathcal{D}^{hys}$. $\mathcal{D}^{hys}$ is a set of data points where each data point is a tuple pairing a hysteresis configuration $\boldsymbol{\eta}$ with its corresponding registered, segmented, and concatenated point cloud $\mathbf{p}$ collected on the robot. More

---

[1]In a slight overload of notation we borrow the use of $\mathbf{p}$ (and associated subscripts) to now also represent a sensed point cloud that lies on the surface of the physical robot.
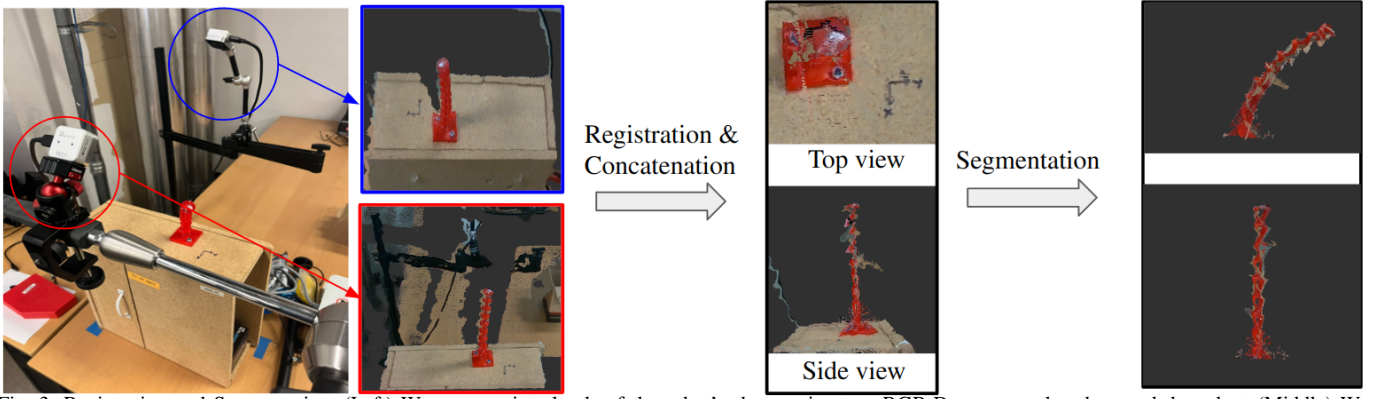
Fig. 3: Registration and Segmentation. (Left) We sense point clouds of the robot's shape using two RGB-D cameras placed around the robot. (Middle) We apply a registration algorithm to align the two camera frames with the robot base frame. We identify five points of correspondence in each of the three frames, specifically the robot's tip when it is at its home, zero-tension position, as well as four points at the corners of the 3D printed structure at the robot's base. The registration algorithm then provides a closed-form solution to generate the optimal rigid transform based on the correspondence set. The concatenated, aligned point clouds are shown. (Right) We then segment the robot's point cloud by removing point cloud points outside of the robot's workspace. Specifically, we eliminate any points in the sensed point clouds which are outside of the robot's workspace, which we approximate for this purpose as a hemisphere of radius l (corresponding to the robot's backbone length) plus a safety margin of ϵ centered at the robot's base. This ensures that only the points corresponding to the robot's shape are retained in the scene.

formally, $\mathcal{D}^{hys} = \{(\boldsymbol{\eta}_1, \mathbf{p}_1), (\boldsymbol{\eta}_2, \mathbf{p}_2), \cdots, (\boldsymbol{\eta}_L, \mathbf{p}_L)\}$ is a data set of L data points.

Remembering that each $\boldsymbol{\eta}$ encodes two robot configurations (a $\mathbf{q}_{\text{prior}}$ and a $\mathbf{q}_{\text{current}}$), we note that for the model to learn how hysteresis influences the robot's shape at a given $\mathbf{q}_{\text{current}}$, we need $\mathcal{D}^{hys}$ to contain multiple hysteresis configurations where $\mathbf{q}_{\text{current}}$ is consistent but $\mathbf{q}_{\text{prior}}$ varies.

The process by which we generate this diverse data set is specific to our robot. The robot on which we evaluate our method (see the red-colored robot in Figs. 1 and 3) is actuated via four tendons, three straight tendons and a helical tendon (see full description in Sec. VI-A).

We first generate a nominal set of configurations for our training set. This is done by first generating a discretization over the combinations of possible tendon displacements on the robot, restricted in a specific way. We restrict this set to combinations that ensure that all three straight tendons are never tensioned simultaneously (which would not be done in practice) and that the helical tendon is not tensioned simultaneously with any straight tendon. To further cover the robot's configuration space, we then add to this set a full grid search-based discretization in tension space, relaxing the restriction to only exclude configurations that simultaneously tension all three straight tendons and excluding configurations already generated by the previous step.

We utilize this nominal set as the set of possible $\mathbf{q}_{\text{current}}$ configurations. We then generate a large set of $\boldsymbol{\eta}$ augmented configurations in the following way (see Alg. 1). First, consider a home configuration as a special case $\mathbf{q}_{\text{home}}$, where all tendons have zero tension. We create the first set of hysteresis configurations as $\boldsymbol{\eta} = [\mathbf{q}_{\text{home}}, \mathbf{q}_{\text{current}}]$ for each $\mathbf{q}_{\text{current}}$ configuration in our nominal set (lines $4-5$). In other words, for each configuration in the nominal set, we first command the robot to the home configuration, $\mathbf{q}_{\text{home}}$, and then to the non-home configuration $\mathbf{q}_{\text{current}}$. We then record the point cloud at $\mathbf{q}_{\text{current}}$.

We next augment this set with more hysteresis configurations where the robot does *not* return to $\mathbf{q}_{\text{home}}$ between configurations but rather first visits a different, randomly selected

---

Algorithm 1: Hysteresis Configuration Generation

Input: NominalSet

1 hysteresis configuration vector $\boldsymbol{\eta} \leftarrow$ []
2 **for** *each configuration in NominalSet* **do**
3      $q_{\text{home}} \leftarrow$ home_configuration()
4   Append q_{home} to $\boldsymbol{\eta}$
5   Append configuration to $\boldsymbol{\eta}$
6   **for** *i from 1 to num_additional_hysteresis* **do**
7   RandomConfiguration $\leftarrow$
           randomly_select_configuration(NominalSet)
8   Append RandomConfiguration to $\boldsymbol{\eta}$
9   Append configuration to $\boldsymbol{\eta}$
10   **end**
11 **end**
12 **return** $\boldsymbol{\eta}$

---

configuration from the nominal configuration set (lines $6-10$). In other words, multiple times for each nominal configuration, we first command the robot to go to a randomly selected configuration and then to the nominal configuration, recording the point cloud at the nominal configuration. This ensures that we have diverse hysteresis configurations, i.e., that for each configuration in the nominal set (i.e., each $\mathbf{q}_{\text{current}}$), we have multiple corresponding hysteresis configurations in which $\mathbf{q}_{\text{prior}}$ is different but $\mathbf{q}_{\text{current}}$ is the same.

C. Model training

To train and validate our model, we split the full data set into 50 test cases and separate 70% of the remaining cases for training and 30% for validation. After each training epoch, the model is evaluated on the validation set. If the validation loss stops improving over 50 consecutive training epochs, we stop the training process to prevent overfitting. We initialize the model weights using Xavier initialization [40] and use a batch size of 32 during training. We use the Adam optimizer with a learning rate of 0.01, decayed by a factor of 0.1 every 100 epochs. We use our loss function $\mathcal{L}_{tendon}$ described prior. By training and evaluating multiple versions of our model, we have determined that $\lambda = 1$ works well.

● Ground truth   ● Learned point clouds

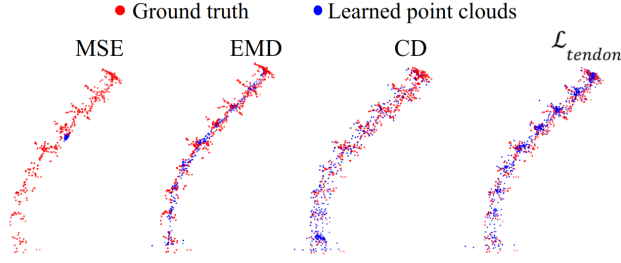MSE   EMD   CD $\mathcal{L}_{tendon}$

Fig. 4: Qualitative loss function comparison. The sensed, ground truth robot shape point cloud is shown in red, and the model-predicted point clouds are shown in blue. The model trained with MSE loss completely fails to achieve the robot's shape. When trained with EMD loss, the model tends to generate points that are uniformly distributed along the robot's structure but the points often cluster around the robot's backbone rather than accurately representing the full shape, including the disks attached along its length. The model trained with the CD loss function produces point clouds that show the robot's geometry but the resulting points are unevenly distributed along the shape. Our proposed loss function, EMD + CD ($\mathcal{L}_{tendon}$), shows the best performance both qualitatively and quantitatively, demonstrating a precise estimation of the robot's shape with evenly distributed points.

VI. EXPERIMENTS AND RESULTS

Here we present experimental details, quantify how significant hysteresis is for our system, analyze the benefit of our novel loss function, and finally compare the performance of our model to a state-of-the-art physics-based model as well as to a version of our model that does not account for hysteresis. Also note that, with the exception of model training, throughout this section when we compare two point clouds—e.g., a model-predicted point cloud and a ground truth point cloud—we employ CD as a metric of difference. CD has been effectively used in various applications to measure the similarity between point clouds where point-to-point correspondence is not available or not desirable, providing robustness and flexibility in geometric evaluation [41]–[43]. We utilize CD rather than the custom loss function we train the model with to decouple the evaluation of loss function performance and model performance.

A. Experimental details

For our experiments we leverage a physical tendon-driven robot, composed of a 3D printed, flexible thermoplastic polyurethane (TPU) material body with a thin nitinol tube embedded in the 3D printed structure with a length of 0.2 m, consisting of 9 circular disks that connect 3 straight-routed tendons at 120 degrees apart and 1 helically-routed tendon, with linear actuators pulling on the tendons to control the robot's shape at the robot's base (see Figs. 1 and 3). Note that our method's current validation is limited to our robot with a specific, commonly used tendon routing pattern. Each disk has a diameter of 0.02 m and is positioned with a distance of 0.02 m between each adjacent pair. We use two RealSense depth cameras (D405) to collect data. Our nominal configuration set generated as described above contains 2773 configurations. For comparison purposes, we define a special $\mathcal{D}^{hys}$ subset, $\mathcal{D}^{hys}_{\text{home}}$, as the first data collected where each $\mathbf{q}_{\text{prior}}$ configuration is the home configuration. The full data set $\mathcal{D}^{hys}$ consists of $\mathcal{D}^{hys}_{\text{home}}$ augmented with two sets where $\mathbf{q}_{\text{prior}}$ was instead randomized (as detailed above). In this way, $\mathcal{D}^{hys}_{\text{home}}$ contains 2773 data points and $\mathcal{D}^{hys}$ contains $2773 \times 3 = 8319$ data points. With this, each nominal

configuration appears in our data set three times, once where the prior configuration was the home configuration and two additional times where the prior configuration was randomly selected. Our method outputs a point cloud of size 512, and we downsample the point clouds of all training and evaluation data to have size 512 as well.

B. Quantification of hysteresis

We first quantify the severity of hysteresis for our robot. To do so, we compare the nominal configuration set of 2773 configurations gathered in two ways. In the first we command the robot to return to the home configuration in between each of the nominal configurations (i.e., with the home configuration as $\mathbf{q}_{\text{prior}}$). In the second, we instead command the robot to visit a random configuration in between each of the nominal configurations (i.e., with a random configuration as $\mathbf{q}_{\text{prior}}$). We then compare the point clouds for each nominal configuration gathered in these two ways. In the case where there is no hysteresis effect, these would result in the same shapes. Instead, the difference between the two is a CD of 0.04 ± 0.1 m and an end-tip position distance of 0.018 ± 0.013 m ($9 \pm 6.5\%$ of robot length), averaged over all 2773 configurations. This implies that our tendon-driven robot displays significant hysteresis. One example of the difference is shown in Fig. 1, top right.

C. Loss function evaluation

To evaluate the impact of our loss function $\mathcal{L}_{tendon}$, we train our method with a variety of loss functions and compare their performance. We compare our model trained with $\mathcal{L}_{tendon}$ against models trained with a standard Mean Squared Error (MSE) loss, EMD loss, and CD by itself as a loss function. In Fig. 4 we show examples of the models' outputs evaluated on one hysteresis configuration as a qualitative example.

Quantitatively, we evaluate the outputs of the model on our 50 test cases. The model trained with the MSE loss fails to model the robot's shape, achieving an average CD of 1.134 ± 0.167 m from the ground truth point clouds. The model trained with the EMD loss generates point clouds evenly distributed along the robot's length but struggles to capture the complete shape of the robot, resulting in an average CD of 0.0163 ± 0.0055 m. While the CD only loss function produces point clouds that start to approximate the robot's geometry, the resulting points are unevenly distributed along the shape. The CD for the model trained on this loss function is 0.0084 ± 0.0065 m. The model trained on our proposed loss function, EMD + CD ($\mathcal{L}_{tendon}$), shows the best performance both qualitatively and quantitatively. It achieves an average CD of 0.0057 ± 0.0024 m, displaying a precise estimation of the robot's shape with evenly distributed point clouds. We find that our proposed loss function ($\mathcal{L}_{tendon}$) provides point clouds that are 199, 2.9, and 1.47 times closer to the ground truth on average than the MSE, EMD, and the CD loss functions, respectively.

D. Comparison to physics-based and non-hysteresis models

We compare against both a physics-based model and a version of our model that does not account for hysteresis. We do so in two experiments, the first where we evaluate on a
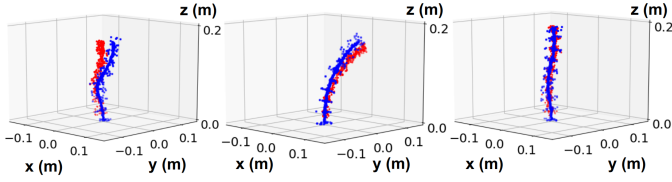
Fig. 5: Physics-based predicted robot shape (blue) vs. ground truth (red). Three examples are shown from the test set, the worst (left), the median (middle), and the best (right). These demonstrate the ranges of error between the physics-based predicted shape and the ground truth sensed shape.

large number of independent hysteresis configurations and the second in which we evaluate on a smaller set of cohesive, sequential trajectories in the robot's workspace.

1) Comparison method details: We first compare against the state-of-the-art physics-based method of [6], which is an extension of the model proposed in [5] to use tendon displacements rather than tension as the input. We obtained an initial guess of the backbone Young's modulus E by hanging weights on one straight routed tendon, and optimizing E to best fit the shape of the manipulator. Then, we calibrated the model on a subset of 30 of the neural network training cases. The eight parameters we calibrated are the backbone Young's modulus E, Poisson's ratio ν, mass density ρ, tendon compliance C_t and tendon offsets $[\delta_1, \ldots, \delta_4]$ which represent potential tension/slack in the home configuration. The physics-based model produces a backbone shape representation. To generate a point cloud representation of the full robot from the model we radially expand the backbone shape to the diameter of the robot, attach the 9 circular disks along the backbone in simulation, and create a point cloud of the resulting geometry (see Fig. 5). We optimize the eight parameters using pattern search to best match the collected point clouds from the training set.

We also compare against a version of our model that does not account for hysteresis (labeled here as the non-hysteresis model). To do so, we change the model's input from the two configuration pair (the hysteresis configuration) to a single configuration—as in traditional forward kinematics—keeping the rest of the network architecture the same. We train the model on the nominal configurations collected by returning to the home configuration in between each.

2) Timing evaluation: To demonstrate the value of our deep decoder network outputting the full shape of the robot directly, we first evaluate the time required by the various methods to compute the full shape of the robot. All computation was performed on a computer with an AMD Ryzen 7 3700X 8-core processor, 64 GB of 3200 Hz DDR4 DRAM, and an NVIDIA GeForce RTX 3060 Ti.

Our model requires an average total training time of 6 minutes 2 seconds (averaged across 10 runs). After training, in deployment our model requires a computation time of 0.32 ± 0.00004 ms from the configuration input to outputting the full predicted robot shape, averaged across 100 runs. For the same evaluation, the non-hysteresis model demonstrates an average computation time of 0.33 ± 0.00011 ms. By contrast, it takes on average 613 ± 206 ms to generate a point cloud from the physics-based model. This is broken up as 394 ms to get the backbone shape from the model and then an additional
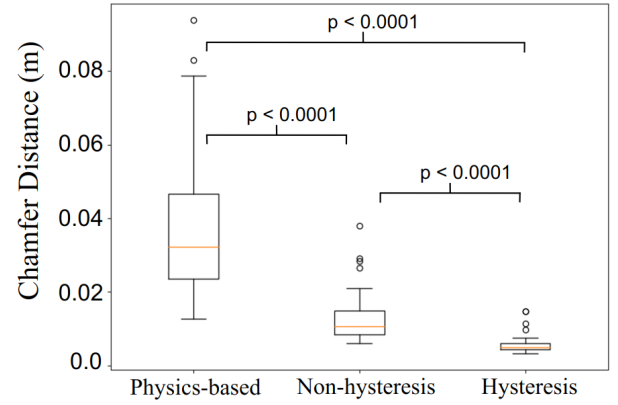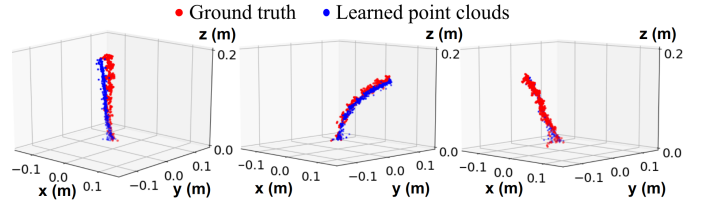

Fig. 6: Boxplot comparison of the physics-based model, the non-hysteresis model, and our full model that accounts for hysteresis (labeled Hysteresis here) across the 50 test cases. The physics-based model achieves an average CD of 0.0374 ± 0.019 m. The non-hysteresis model achieves a CD of 0.013 ± 0.007 m while our full model achieves 0.0057 ± 0.0024 m.

(a) Non-hysteresis model predicted shape compared with the ground truth.
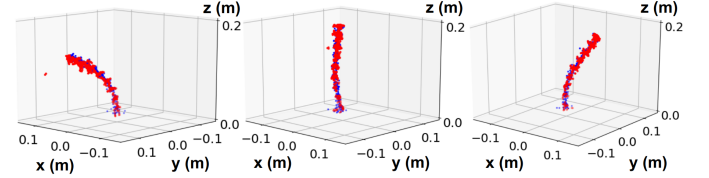
(b) Full hysteresis model predicted shape compared with the ground truth.

Fig. 7: Three examples (the worst, the median, and the best cases from the 50 test cases) comparing the performance of the non-hysteresis (a) and our full model (b). While the non-hysteresis model shows good qualitative performance, our full model estimates the robot's shape more accurately.
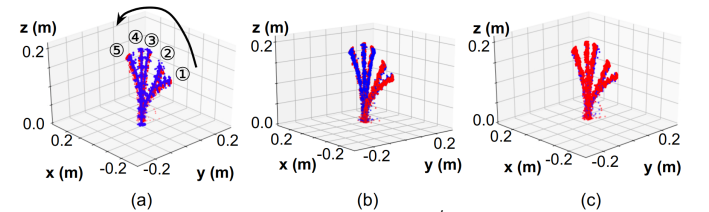

Fig. 8: An example trajectory test case (the 1st of the 6 trajectory cases) comparing the performance of (a) the physics-based model, (b) the non-hysteresis model, and (c) our hysteresis model. The order of the robot trajectory is indicated in (a) using numbers 1 through 5, and this sequence is consistent across all three subfigures. The physics-based model performs the worst, with the non-hysteresis model improving upon the physics-based model, while our model shows improvement over both.

219 ms to augment the backbone with the rest of the robot geometry's point cloud. Our model demonstrates substantial speedup, producing full robot shapes $\approx 1,915$ times more quickly.

3) Single hysteresis configuration evaluation: We next evaluate model accuracy. We show qualitative results for the physics-based model in Fig. 5. We show quantitative results in Fig. 6 comparing all three methods with additional qualitative results for our model and the non-hysteresis model shown in Fig. 7.
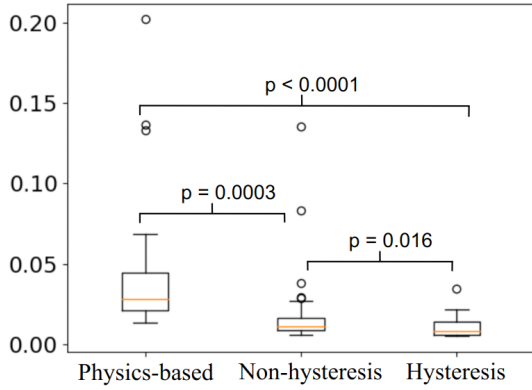
Fig. 9: For the full trajectory evaluation, boxplot comparison of CD averaged across all configurations in all 6 trajectories for the physics-based model, the non-hysteresis model, and our model (labeled Hysteresis here). The physics-based model achieves an average CD of 0.044 ± 0.042 m. The non-hysteresis model achieves 0.015 ± 0.006 m while our model achieves 0.009 ± 0.003 m.

Qualitatively (see Figs. 5 and 7), the physics-based model performs the worst, the non-hysteresis model the next best, and our full model performing the best, more closely capturing the robot's shape. Quantitatively, as can be seen in Fig. 6, when evaluated on our 50 test data points our model with hysteresis significantly outperforms the other two. The physics-based model performs the worst, resulting in an average CD of 0.0374 ± 0.019 m from the ground truth. The non-hysteresis model achieves a CD of 0.013 ± 0.007 m while our hysteresis model achieves 0.0057 ± 0.0024 m. On average, our proposed method produces shapes that are 6.6 times closer to the ground truth than the physics-based model and 2.3 times closer than the non-hysteresis models.

As CD, being a sum over many point pair distances, is a potentially unintuitive metric, we additionally evaluate end-tip position error. The physics-based model exhibits 0.019 ± 0.008 m end-tip position error ($9.5 \pm 4\%$ of robot length), the non-hysteresis model 0.01 ± 0.004 m ($5 \pm 2\%$ of robot length), and our model 0.0076 ± 0.0037 m ($3.8 \pm 1.85\%$ of robot length). However, note that CD serves as a more precise metric than the end-tip position error, given we calibrate/train all the models based on the complete shape of the robot, rather than solely focusing on the end-tip position error.

4) Full trajectory evaluation: For tendon-driven robots to be effectively used in real-world applications, it is imperative that the model accurately predicts the robot's shape as it executes trajectories in its environment.

In this section we evaluate our method's ability to predict the robot's shape across full trajectories, composed of a number of sequentially chosen random configurations. Here we augment the model's training data by supplementing $\mathcal{D}^{hys}$ with a random trajectory dataset consisting of 1200 data points. This dataset consists of configurations randomly selected from the nominal set without returning to $\mathbf{q}_{home}$ between configurations or between pairs of configurations.

We evaluate our model's performance compared with the physics-based model and the non-hysteresis model using 6 trajectories not seen during training, each comprising 5 different sequential randomly-sampled configurations. Qualitatively, as can be seen in Fig. 8, our model accurately estimates the robot's shape for the entire trajectory, outperforming the other

comparison methods. Fig. 9 shows the quantitative result for all three methods. The physics-based model achieves a CD of 0.044 ± 0.042 m from the ground truth averaged over all 6 trajectory cases (i.e., 30 configurations). The non-hysteresis model displays an average CD of 0.015 ± 0.006 m, performing better than the physics-based model. Our model exhibits the best performance with an average CD of 0.009 ± 0.003 m, producing shapes that are 4.9 and 1.7 times closer to the ground truth on average than the physics-based and non-hysteresis models, respectively. The end-tip position error measurement shows 0.02 ± 0.012 m ($10 \pm 6\%$ of robot length) for the physics-based model, 0.011 ± 0.004 m ($5.5 \pm 2\%$ of robot length) for the non-hysteresis model, and 0.009 ± 0.004 m ($4.5 \pm 2\%$ of robot length) for our model.

VII. CONCLUSION

We proposed a deep decoder neural network model that computes the forward kinematics of tendon-driven continuum robots. The model directly outputs a full shape representation of the robot in the form of a point cloud. By augmenting the input with a prior configuration in addition to the current configuration, the model is able to successfully account for hysteresis. We evaluate the accuracy of our method in predicting the robot's shape and compare against a physics-based model and a model that does not account for hysteresis, significantly outperforming both.

In future work, we plan to develop a new approach that accounts for the entire past trajectory of the robot rather than a short history, and to leverage the model in control and planning for these robots, in methods such as, e.g., [16], [44]. We plan to address dynamic conditions of anatomical environments, including hysteresis changes and tissue disturbance due to contact. Future work will focus on validating the effectiveness of the proposed method within anatomical environments to ensure its robustness and accuracy in real-world medical applications. We also intend to apply the method to other continuum robots, such as concentric tube robots, longer catheter systems, and tendon robots with various other nonlinear tendon routings, to further validate the model's applicability across various medical domains.

Further, point clouds as an output geometric shape representation are popular in other domains, and are particularly well-motivated in applications that leverage nearest-neighbor data structures for collision detection in planning and control [14], [15], but are not the only choice for geometric shape representation. Voxel sets and signed distance functions, among others, are options as well. We intend to further explore the ways in which a learned model can output the robot's shape directly using representations such as these.

We believe that this new approach takes steps toward overcoming the limitations of current approaches in modeling the shape of tendon-driven robots. This has the potential to lead to more accurate and efficient prediction of the tendon robot's shape, unlocking the door to safer operation and improved capabilities for these robots.

REFERENCES

[1] P. Dupont, N. Simaan, H. Choset, and C. Rucker, "Continuum robots for medical interventions," *Proceedings of the IEEE*, vol. 110, no. 7, pp. 847–870, 2022.

[2] J. Burgner-Kahrs, D. C. Rucker, and H. Choset, "Continuum robots for medical applications: a survey," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1261–1280, Dec. 2015.

[3] T.-D. Nguyen and J. Burgner-Kahrs, "A tendon-driven continuum robot with extensible sections," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept. 2015, pp. 2130–2135.

[4] T. Kato, I. Okumura, S.-E. Song, A. J. Golby, and N. Hata, "Tendon-driven continuum robot for endoscopic surgery: preclinical development and validation of a tension propagation model," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 5, pp. 2252–2263, 2015.

[5] D. C. Rucker and R. J. Webster, III, "Statics and dynamics of continuum robots with general tendon routing and external loading," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1033–1044, 2011.

[6] K. Oliver-Butler, J. Till, and C. Rucker, "Continuum Robot Stiffness under External Loads and Prescribed Tendon Displacements," *IEEE Transactions on Robotics*, vol. 35, no. 2, 2019.

[7] R. Grassmann, V. Modes, and J. Burgner-Kahrs, "Learning the forward and inverse kinematics of a 6-DOF concentric tube continuum robot in SE(3)," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 5125–5132.

[8] A. Kuntz, A. Sethi, R. J. Webster, III, and R. Alterovitz, "Learning the Complete Shape of Concentric Tube Robots," *IEEE Transactions on Medical Robotics and Bionics*, vol. 2, no. 2, pp. 140–147, 2020.

[9] H.-J. Yu, W.-L. Yang, Z.-X. Yang, W. Dong, Z.-J. Du, and Z.-Y. Yan, "Hysteresis analysis of a notched continuum manipulator driven by tendon," *Mechanical Sciences*, vol. 9, no. 1, pp. 211–219, 2018.

[10] Y.-H. Kim and T. Mansi, "Shape-adaptive hysteresis compensation for tendon-driven continuum manipulators," *arXiv*, 2021.

[11] W. Wu, Z. Qi, and L. Fuxin, "PointConv: Deep Convolutional Networks on 3D Point Clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[12] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017.

[13] B. Thach, B. Y. Cho, A. Kuntz, and T. Hermans, "Learning Visual Shape Control of Novel 3D Deformable Objects from Partial-View Point Clouds," in *International Conference on Robotics and Automation (ICRA)*, 2022, pp. 8274–8281.

[14] Ichnowski, Jeffrey and Alterovitz, Ron, "Fast nearest neighbor search in SE (3) for sampling-based motion planning," in *Workshop on the Algorithmic Foundations of Robotics*. Springer, 2015, pp. 197–214.

[15] J. Ichnowski and R. Alterovitz, "Concurrent nearest-neighbor searching for parallel sampling-based motion planning in SO (3), SE (3), and Euclidean spaces," in *Workshop on the Algorithmic Foundations of Robotics 13*. Springer, 2020, pp. 69–85.

[16] M. Bentley, C. Rucker, and A. Kuntz, "Interactive-Rate Supervisory Control for Arbitrarily-Routed Multitendon Robots via Motion Planning," *IEEE Access*, vol. 10, pp. 80999–81019, 2022.

[17] P. J. Swaney, P. A. York, H. B. Gilbert, J. Burgner-Kahrs, and R. J. Webster, "Design, fabrication, and testing of a needle-sized wrist for surgical instruments," *Journal of Medical Devices*, vol. 11, no. 1, p. 014501, 2017.

[18] A. Leavitt, R. Lam, N. C. Taylor, D. S. Drew, and A. Kuntz, "Toward a millimeter-scale tendon-driven continuum wrist with integrated gripper for microsurgical applications," *arXiv*, 2023.

[19] H. Yuan, L. Zhou, and W. Xu, "A comprehensive static model of cable-driven multi-section continuum robots considering friction effect," *Mechanism and Machine Theory*, vol. 135, pp. 130–149, 2019.

[20] T. Poignonec, P. Zanne, B. Rosa, and F. Nageotte, "Towards in situ backlash estimation of continuum robots using an endoscopic camera," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4788–4795, 2020.

[21] T. Kato, I. Okumura, H. Kose, K. Takagi, and N. Hata, "Tendon-driven continuum robot for neuroendoscopy: validation of extended kinematic mapping for hysteresis operation," *International Journal of Computer Assisted Radiology and Surgery*, vol. 11, no. 4, pp. 589–602, 2016.

[22] D. Baek, J.-H. Seo, J. Kim, and D.-S. Kwon, "Hysteresis compensator with learning-based hybrid joint angle estimation for flexible surgery robots," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6837–6844, 2020.

[23] Y. Guo, B. Pan, Y. Sun, G. Niu, Y. Fu, and M. Q.-H. Meng, "Motion hysteresis compensation based on motor current segmentation for elongated cable-driven surgical instruments," *IEEE Transactions on Automation Science and Engineering*, 2023.

[24] S. Neppalli, M. A. Csencsits, B. A. Jones, and I. D. Walker, "Closed-form inverse kinematics for continuum manipulators," *Advanced Robotics*, vol. 23, no. 15, pp. 2077–2091, 2009.

[25] D. Wu, Y. Zhang, M. Ourak, K. Niu, J. Dankelman, and E. Vander Poorten, "Hysteresis modeling of robotic catheters based on long short-term memory network for improved environment reconstruction," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2106–2113, 2021.

[26] W. Bai, F. Cursi, X. Guo, B. Huang, B. Lo, G.-Z. Yang, and E. M. Yeatman, "Task-based lstm kinematic modeling for a tendon-driven flexible surgical robot," *IEEE Transactions on Medical Robotics and Bionics*, vol. 4, no. 2, pp. 339–342, 2021.

[27] X. T. Ha, D. Wu, M. Ourak, G. Borghesan, J. Dankelman, A. Menciassi, and E. Vander Poorten, "Shape sensing of flexible robots based on deep learning," *IEEE Transactions on Robotics*, vol. 39, no. 2, pp. 1580–1593, 2022.

[28] S. Lilge, T. D. Barfoot, and J. Burgner-Kahrs, "Continuum robot state estimation using Gaussian process regression on SE (3)," *International Journal of Robotics Research*, vol. 41, no. 13-14, pp. 1099–1120, 2022.

[29] J. Jung, R. S. Penning, N. J. Ferrier, and M. R. Zinn, "A modeling approach for continuum robotic manipulators: Effects of nonlinear internal device friction," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 5139–5146.

[30] G. Subramani and M. R. Zinn, "Tackling friction - an analytical modeling approach to understanding friction in single tendon driven continuum manipulators," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 610–617.

[31] W. Xu, J. Chen, H. Y. Lau, and H. Ren, "Data-driven methods towards learning the highly nonlinear inverse kinematics of tendon-driven surgical manipulators," *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 13, no. 3, p. e1774, 2017.

[32] N. Liang, R. M. Grassmann, S. Lilge, and J. Burgner-Kahrs, "Learning-based Inverse Kinematics from Shape as Input for Concentric Tube Continuum Robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 1387–1393.

[33] C. Bergeles, F. Y. Lin, and G. Z. Yang, "Concentric tube robot kinematics using neural networks," in *Hamlyn symposium on medical robotics*, June 2015, pp. 1–2.

[34] M. Kasaei, K. K. Babarahmati, Z. Li, and M. Khadem, "Data-efficient non-parametric modelling and control of an extensible soft manipulator," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 2641–2647.

[35] G. Fagogenis, C. Bergeles, and P. E. Dupont, "Adaptive nonparametric kinematic modeling of concentric tube robots," in *IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 2016.

[36] M. Kasaei, K. K. Babarahmati, Z. Li, and M. Khadem, "A data-efficient neural ode framework for optimal control of soft manipulators," in *The Conference on Robot Learning*. PMLR, 2023, pp. 1–14.

[37] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[38] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *International Conference on Machine Learning*. pmlr, 2015.

[39] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Transactions on pattern analysis and machine intelligence*, no. 5, pp. 698–700, 1987.

[40] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[41] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3d object reconstruction from a single image," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[42] T. Wu, L. Pan, J. Zhang, T. Wang, Z. Liu, and D. Lin, "Balanced chamfer distance as a comprehensive metric for point cloud completion," *Advances in Neural Information Processing Systems*, vol. 34, pp. 29088–29100, 2021.

[43] F. Lin, Y. Yue, S. Hou, X. Yu, Y. Xu, K. D. Yamada, and Z. Zhang, "Hyperbolic chamfer distance for point cloud completion," in *IEEE/CVF International Conference on Computer Vision*, 2023, pp. 14595–14606.

[44] Y. Huang, M. Bentley, T. Hermans, and A. Kuntz, "Toward learning context-dependent tasks from demonstration for tendon-driven surgical robots," in *International Symposium on Medical Robotics*, 2021.