Global Control of Soft Manipulator by Unifying Cosserat and Neural Network

Haihong Li, Lingxiao Xun, Gang Zheng

Global Control of Soft Manipulator by Unifying Cosserat and Neural Network

Haihong Li, Lingxiao Xun, and Gang Zheng

***Abstract*—Soft manipulators, a relatively novel class of robots, are increasingly prevalent in the field of robotics. Due to the material nonlinearity and the absence of links and joints which are typically used to derive their kinematics and dynamics, the exact modeling and effective control frameworks of such soft robots are much more difficult than the traditional rigid counterparts. To achieve global control of the end-effector position of the soft manipulator actuated by cables, this work proposes a hybrid control strategy by unifying the piecewise linear strain (PLS) Cosserat model and radial basis function neural network (RBFNN) to approximate the Jacobian matrix of any end-effector position in the whole workspace, and then designs a global control scheme based on the approximation of Jacobian matrix. The theoretical convergence proof and experimental validation are provided for the designed global controller. The results corroborate that the proposed control strategy has excellent accuracy and robustness in tracking desired time-varying trajectories through different experiments.**

***Index Terms*—Soft manipulator, Cosserat, neural network, controller.**

I. Introduction

The biological systems in nature provide significant inspiration of soft structures which are able to bend, extend, shear and twist. Examples include elephant trunk, octopus tentacles, mammalian tongue, robotic fish, etc. Unlike the conventional rigid robots, all of such continually deformable structures have an infinite number of degrees of freedom, which provides soft robots an incredible agility to adapt to new environments and move in narrow spaces [1]. However, it remains a substantial challenge to establish a mathematical model suitable to accurately describe the kinematics and dynamics of such high-dimensional systems due to their inherent properties of compliance and flexibility. Additionally, the development of high-performance control approaches for soft robots is also a challenging research area.

To address the above-mentioned challenges, one of the most used techniques is to deduce the exact models of soft robots. In [2], the kinematic model was derived using geometric information, followed by the application of a computed torque controller to control an eel-like soft robot. The piecewise constant curvature (PCC) has proven to be quite feasible with a vast range of control applications [3], [4]. For example, their use in the inverse kinematic control [5], [6], the closed-loop dynamic feedback controllers [7], [8], [9], and feedforward dynamic control [10]. However, under the condition of external loads or contact with the environment, the validity of the CC assumption is relatively conservative and unsuitable for many applications [11]. Other techniques have also been used to get the models. In [12], the authors employed Euler-Bernoulli beam theory to model the bending mode of the robot, and developed the position and orientation control for an eversion robot with changing structural stiffness. FEM models are suitable to handle material and geometric nonlinearities, and can provide a more accurate representation of deformations of soft robots [13], [14]. In order to provide a general method to control soft robot, an open-loop controller was designed via the FEM model [13]. In [15], the authors employed FEM and visual tracking technique to achieve the feedback control for soft robots. [16] analyzed the dynamics of soft robots, and used the reduced-order model technique to achieve fast convergence of soft robots to a desired position.

The Cosserat beam model is the geometrically nonlinear generalization of the other beam theory-based modeling methods for slender soft robots [17]. [18] overcame the expensive model computational obstacle, and adopted the Cosserat rod model to design a robust controller for continuum robots. The authors of [19] combined the Cosserat kinematic model with integrated sensing and the Jacobian matrix to provide a practically feasible closed-loop control system for application to soft robots. Previous research has also explored the use of discrete Cosserat model-based controllers. For instance, in [20], a first-order approximation assumption of piecewise constant strain (PCS) dynamics was proposed, based on which the closed-loop task-space dynamic controllers were developed. The geometric variable strain Cosserat approach [21] was used to carry out the Jacobian-based inverse kinematic control of the general soft manipulator. However, this model-based control method ignored the external forces and lacked the experimental validation [22]. The piecewise liner strain (PLS) Cosserat model proposed in [23] was used to achieve the local control of end-effector position for the cable-driven soft manipulator [24]. For those mentioned Cosserat-based modeling method where the strains were used to parameterize the manipulator's deformation, the proposed global controllers in the existing literature have only been validated through simulation, but cannot be implemented into the real prototype, as those strains are difficult to be measured in practice.

All of the aforementioned control techniques, for the tracking control of soft continuum robots, require the exact models (either kinematic, or dynamic) to be known. In contrast, model-free controllers have proposed to control the soft robots with the uncertainties and complexities of nonlinear behaviors using machine learning techniques or empirical methods. In [25], the machine learning-based method was introduced to achieve the closed loop kinematic control of

All authors are affiliated with Université de Lille, Inria, CNRS, Centrale Lille, UMR 9189 CRIStAL, Lille, F-59000, France.

continuum robots in the task space. [26] used the kinematic and kinetic models based on artificial neural network (ANN) to control the compact bionic handling arm. For the work of [27], the authors employed the continual learning technique to design controllers able to continuously adapt to changes in robot dynamics. [28] used an ANN to approximate the input-output relation of a cable-driven soft robot, designed the robust controller, and provided the proof of the stability and robustness as well. [29] presented a machine learning based approach towards development of dynamic model and a trajectory optimization method of predictive control for the soft manipulator. In [30], a data-based control framework was introduced to solve the soft robot underwater locomotion problem using deep reinforcement learning. [31] described a Koopman-based approach and its application to model predictive control design for pneumatic soft robots. [32] employed the machine learning approach to estimate the a differentiable model of the quasi-static physics of a soft robot, and then performed gradient based optimization to find the best open-loop control input. [33] presented a method to train the forward kinematic model and its Jacobian together as two neural networks to realize the real-time computation of inverse kinematics on soft robots. Despite their gratifying advances, collecting a representative and sufficient amount of data is still challenging for most learning-based control approaches for soft robots [34]. Furthermore, stability analysis and theoretical convergence proofs are difficult to provide for the learning-based controllers.

The hybrid control method combining model-based and learning-based control approaches has also been a cutting-edge control strategy for soft robots, which can be regarded as an improvement of model-free control while incorporating the strengths of physics models. In [35], a supervised learning method to solve the inverse statics of the cable-actuated soft manipulator with non-constant curvature was introduced. [36] presented a model-based policy learning algorithm for closed-loop predictive control of a soft robotic manipulator. For the work of [37], the authors put forward a hybrid modeling method which combined the first-principles model with machine learning methods to improve overall performance of the nonlinear model predictive controller. Based on FEM and ANN, [38] proposed a transfer learning scheme to minimize the effort of generating real data for neural network training, and achieved the fine control of a real soft pneumatic actuator. In [39], the authors introduced a model-based online learning and adaptive control algorithm for the wearable soft robotic glove, which enabled the soft glove to adapt to diverse hand conditions for reference tracking.

To the best of our knowledge, up to now, the discrete Cosserat static model-based control technique combined with the neural networks is scarcely explored in soft robotics. This motives the work of this paper, where we propose a hybrid approach to globally control the end-effector position of the cable-driven soft manipulator by unifying both the PLS Cosserat static model and the neural networks to track the trajectories within its whole workspace in real time. In summary, compared to other kinematic or static model-based controller design for soft robots, the contributions of this work

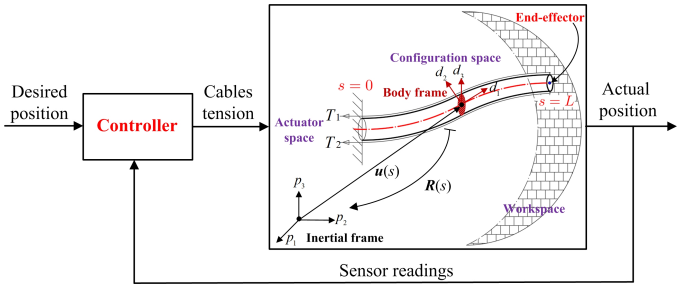

Fig. 1. Schematic of the end-effector position control for the cable-driven soft manipulator.

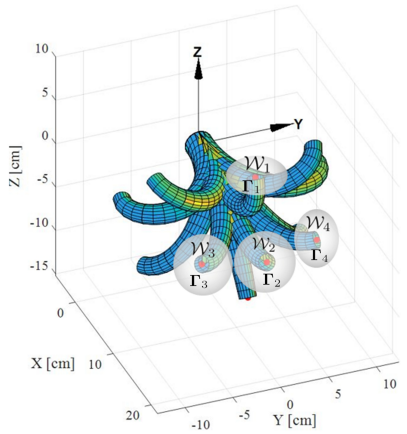

Fig. 2. Sub-workspaces and their associated configurations for the soft manipulator.

include the following:

- An optimization-based approach is presented to obtain the whole workspace of the soft manipulator via PLS Cosserat, and a novel decomposition strategy of the workspace is proposed;
- A method to solve the inverse statics of the soft manipulator modeled by the PLS Cosserat approach using the minimum potential energy principle is implemented to determine the unique configuration of manipulator;
- The radial basis function neural network (RBFNN) is unified with PLS Cosserat to globally approximate the Jacobian matrix of any end-effector position of the soft manipulator in its whole workspace;
- The robust control scheme for the cable-driven soft manipulator is established with complete mathematical proof. Experimental implementations on the soft prototype corroborate the feasibility and robustness of the proposed controller.

II. Problem Statement

In this paper, the objective is to develop an efficient closed-loop controller based on the approximated static model for the purpose of controlling the end-effector position of the cable-driven soft manipulator within its whole workspace $\mathcal{W}_E$.

To realize this objective, PLS Cosserat rod model [23] is adopted in our work to model the soft manipulator, as depicted in Fig. 1. The material-attached reference frame

(d_1, d_2, d_3) of the continuous Cosserat rod can be defined as $\boldsymbol{g}(s) = \begin{pmatrix} \boldsymbol{R}(s) & \boldsymbol{u}(s) \\ \boldsymbol{0}^\top & 1 \end{pmatrix} \in SE(3)$ where $\boldsymbol{u}(s) \in \mathbb{R}^3$ represents the position field of any cross section at $s \in [0, L]$, $\boldsymbol{R}(s) \in SO(3)$ stands for the material orientation in the form of an orthonormal rotation matrix w.r.t. the inertial frame (p_1, p_2, p_3). In the framework of PLS, the configuration space, noted as $\boldsymbol{q} \in \mathbb{R}^{6(N+1)}$, is parameterized by 6 strain components of each interpolation node of soft manipulator divided into N sections. We denote $\boldsymbol{u} \in \mathbb{R}^3$ as the end-effector position of the soft manipulator (workspace), and $\boldsymbol{T} \in \mathbb{R}^{\overline{\mathrm{N}}}$ as tension magnitude vector of $\overline{\mathrm{N}}$ ($\overline{\mathrm{N}} \geq 4$) cables (actuator space) attached at the end-effector of the manipulator, then by applying the principle of virtual work as described in [21], the PLS Cosserat static and kinematic models are derived as

$$\boldsymbol{K}\boldsymbol{q} + \boldsymbol{G}(\boldsymbol{q}) = \boldsymbol{H}\boldsymbol{T} \tag{1a}$$

$$\boldsymbol{u}(\boldsymbol{q}) = \mathbf{P}\boldsymbol{g}(\boldsymbol{q})\mathbf{E} \tag{1b}$$

$$\boldsymbol{\eta} = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{1c}$$

where $\boldsymbol{K} \in \mathbb{R}^{6(N+1)\times 6(N+1)}$ is the generalized stiffness matrix, $\boldsymbol{G}(\boldsymbol{q}) \in \mathbb{R}^{6(N+1)\times 1}$ is the generalized gravitational matrix, and $\boldsymbol{H} \in \mathbb{R}^{6(N+1)\times \overline{\mathrm{N}}}$ is the generalized actuation matrix; $\mathbf{P}$ and $\mathbf{E}$ are the pre-defined selection matrices; $\boldsymbol{\eta} = \left(\boldsymbol{\Omega}^\top, \boldsymbol{V}^\top\right)^\top \in \mathbb{R}^6$ represents velocity twist where $\boldsymbol{\Omega} \in \mathbb{R}^3$ and $\boldsymbol{V} \in \mathbb{R}^3$ stand for the angular and linear velocity, respectively; $\boldsymbol{J}(\boldsymbol{q})$ is the geometric Jacobian matrix describing the relation between the velocity twist and the time derivative of the generalized strain of the soft manipulator. With the above PLS Cosserat models, we can obtain

$$\dot{\boldsymbol{u}} = \boldsymbol{\Gamma}(\boldsymbol{q})\dot{\boldsymbol{T}} \tag{2}$$

where $\boldsymbol{\Gamma}(\boldsymbol{q}) \in \mathbb{R}^{3\times \overline{\mathrm{N}}}$ is the Jacobian matrix, which determines the directional relation between the input and the output of the system. Then the classical pseudo-inverse method, based on the inverse of $\boldsymbol{\Gamma}(\boldsymbol{q})$, can be applied to design robust controller. However, it is evident that the calculation of $\boldsymbol{\Gamma}(\boldsymbol{q})$ requires the knowledge of the generalized strain vector $\boldsymbol{q}$, which is practically difficult to be measured.

In order to solve the mentioned issue of the PLS Cosserat model-based control design for soft robots, this work aims to globally (i.e., for all admissible $\boldsymbol{q}$) approximate $\boldsymbol{\Gamma}(\boldsymbol{q})$ in advance, and thus design a robust pseudo-inverse controller, based only on the measurement of end-effector position, to realize the global control of $\boldsymbol{u}$.

Concretely, we propose to divide the whole workspace $\mathcal{W}_E$ into several sub-workspaces, denoted as $\mathcal{W}_i \subseteq \mathcal{W}_E$, $i = 1, 2, \cdots, m$, where m represents the number of the sub-workspaces, as shown in Fig. 2, choose a representative equilibrium point $\boldsymbol{u}_i \in \mathcal{W}_i$ from the ith sub-workspace, and calculate the associated Jacobian matrix $\boldsymbol{\Gamma}_i$ in advance. Subsequently, the global approximation of $\boldsymbol{\Gamma}(\boldsymbol{q})$ will be realized via a neural network (with parameter α_i to be trained) as

$$\boldsymbol{\Gamma}(\boldsymbol{q}) = \widehat{\boldsymbol{\Gamma}}(\boldsymbol{u}) = \sum_{i=1}^{m} \alpha_i(\boldsymbol{u_i})\boldsymbol{\Gamma}_i$$

where $\boldsymbol{\Gamma}_i$ represents the ith representative Jacobian matrix.

In summary, to design the presented global controller by unifying PLS Cosserat static-model and neural network, the following 4 questions need to be answered:

Q1: How to estimate and divide the whole workspace $\mathcal{W}_E$?
Q2: For each sub-workspace, how to calculate the associated $\boldsymbol{\Gamma}_i$?
Q3: With the pre-calculated $\boldsymbol{\Gamma}_i$, how to construct the neural network to approximate $\boldsymbol{\Gamma}(\boldsymbol{q})$?
Q4: With the obtained Jacobian matrix approximation $\widehat{\boldsymbol{\Gamma}}$, how to design a robust controller to realize the global control objective of $\boldsymbol{u}$?

In the following, III-A will present the detailed solution to Q1, followed by Q2 addressed in III-B. After that, III-C will correspond to the solution to Q3. Finally, III-D will provide the answer to Q4.

III. Robust Controller Design via the PLS Cosserat Static Model and Neural network

In this section, we firstly present a workspace estimation and decomposition approach for the PLS Cosserat static model, which provides the benefit for the soft robotic applications mainly related to their control. Next, the analytical solution of the matrix $\boldsymbol{\Gamma}(\boldsymbol{q})$ is provided, and an optimization approach is presented to solve the inverse PLS Cosserat static model for the purpose of computing $\boldsymbol{\Gamma}_i$. After that, a novel neural network based strategy for the off-line estimation of the Jacobian-based relation matrix between the input and output of the soft manipulator in the whole workspace is proposed. Finally, we design a closed-loop feedback control, and provide the theoretical convergence proof for the designed controller.

A. Workspace Estimation and Decomposition

The subject of workspace determination has been widely studied by the soft robotic community, which includes all its equilibrium points where the end-effector of soft robots can reach. The goal of this subsection is to estimate and decompose the feasible workspace of the soft manipulator based on the PLS Cosserat model by using the optimization-based approach. The workspace $\mathcal{W}_E$ of the robot's end-effector can be defined as

$$\mathcal{W}_E = \left\{\boldsymbol{u} \in \mathbb{R}^3 | \boldsymbol{u} = \mathbf{P}\boldsymbol{g}(\boldsymbol{q})\mathbf{E}, \boldsymbol{K}\boldsymbol{q} + \boldsymbol{G}(\boldsymbol{q}) = \boldsymbol{H}\boldsymbol{T}, \forall \boldsymbol{T} \in \boldsymbol{\mathcal{T}}\right\}$$

where $\boldsymbol{T} \in \mathbb{R}^{\overline{N}}$ represents tension magnitude vector of $\overline{N}$ cables, and $\boldsymbol{\mathcal{T}} = \boldsymbol{\mathcal{T}_1} \times \cdots \times \boldsymbol{\mathcal{T}_{\overline{N}}}$ with $\boldsymbol{\mathcal{T}_i} = \left[T^i_{\min}, T^i_{\max}\right]$ being the minimal and maximal tension bounds of the ith cable for $i = 1, \cdots, \overline{N}$.

To obtain the correct estimation of the whole workspace, an optimization-based method [40] is implemented on the soft manipulator that has been modeled by the PLS Cosserat approach. As a result, the whole workspace estimation algorithm can be formulated as

$$\begin{aligned} &\underset{\boldsymbol{q}^{(i)},\ \boldsymbol{T}^{(i)}}{\arg\min} \ \|\boldsymbol{u}(\boldsymbol{q}^{(i)}) - \boldsymbol{\mathcal{E}}^{(i)}\|_2^2 \\ &s.t. \quad \begin{cases} \boldsymbol{K}\boldsymbol{q}^{(i)} + \boldsymbol{G}(\boldsymbol{q}^{(i)}) - \boldsymbol{H}\boldsymbol{T}^{(i)} = \boldsymbol{0} \\ \boldsymbol{T}^{(i)} \in \boldsymbol{\mathcal{T}} \end{cases} \end{aligned} \tag{3}$$

with

$$\boldsymbol{u}(\boldsymbol{q}^{(i)}) = \mathbf{P}\boldsymbol{g}(\boldsymbol{q}^{(i)})\mathbf{E}$$
$$\boldsymbol{\mathcal{E}}^{(i)} = \boldsymbol{\delta}_x^{(i)}\boldsymbol{\delta}_y^{(i)}\boldsymbol{\delta}_z^{(i)}\boldsymbol{\mathcal{E}}_0$$

where $\boldsymbol{\mathcal{E}}^{(i)}$ represents the selected radiating vector outside the workspace $\mathcal{W}_E$, $\boldsymbol{\mathcal{E}}_0$ is the initial radiating vector, $\boldsymbol{\delta}_{x,y,z}^{(i)}$ stand for the basic rotation matrices about the x-, y-, and z-axis, respectively. The detailed algorithm procedures of the optimization-based method along with the scheme to address the non-convexity problem (i.e., how to accurately determine the interior boundary of the workspace) can be found in [40]. Eventually, we can estimate the end-effector workspace for the studied soft manipulator by solving the minimization problem (3).

The properties of Jacobian matrices of all end-effector positions in a certain sub-workspace are considered to be the same, and thus we can use a nominal Jacobian matrix to replace all Jacobian matrices in this sub-workspace. With the obtained whole workspace $\mathcal{W}_E$, the scheme of workspace decomposition will be introduced. In terms of two different points $\boldsymbol{u}_i \in \mathcal{W}_E$ and $\boldsymbol{u}_{i+l} \in \mathcal{W}_E$, for $l = 1, 2, \cdots, S$ (any point around $\boldsymbol{u}_i$), the Jacobian matrices (i.e., $\boldsymbol{\Gamma}_i$, and $\boldsymbol{\Gamma}_{i+l}$) respectively corresponding to the end-effector positions (i.e., $\boldsymbol{u}_i$, and $\boldsymbol{u}_{i+l}$) can be obtained. Thus, we propose the following decomposition rule along the workspace surface to determine whether $\boldsymbol{u}_i$ and $\boldsymbol{u}_{i+l}$ belong to the same sub-workspace or not:

$$\mathrm{Sgn}(\boldsymbol{\Gamma}_i) = \mathrm{Sgn}(\boldsymbol{\Gamma}_{i+l}) \tag{4}$$

where the symbol $\mathrm{Sgn}(\cdot)$ applies the conventional sign function to the matrix via an element-wise way. The sign for the elements of the Jacobian matrix $\boldsymbol{\Gamma}$ represents the directional relation between cables' tension and end-effector position. If the rule formulated in (4) holds, then $\boldsymbol{u}_i$ and $\boldsymbol{u}_{i+l}$ will be classified into one sub-workspace $\mathcal{W}_i$ where $\boldsymbol{u}_i$ is defined as the representative equilibrium point of the sub-workspace. Otherwise, the two distinct sub-workspaces will be generated for $\boldsymbol{u}_i$ and $\boldsymbol{u}_{i+l}$, respectively. Comparing the elements' sign from the Jacobian matrices around all equilibrium points in $\mathcal{W}_E$ iteratively, we can finally decompose the whole workspace of the soft manipulator via the PLS Cosserat into m different sub-workspaces.

B. Calculation of $\boldsymbol{\Gamma}_i$ for the ith Sub-Workspace

To compute $\boldsymbol{\Gamma}_i$, we need to know the analytical formula of $\boldsymbol{\Gamma}(\boldsymbol{q})$ and the associated strain value of $\boldsymbol{q}$ for a given equilibrium end-effector position $\boldsymbol{u}_i \in \mathcal{W}_i$ which in fact is the well-known problem of inverse static model. Therefore, this subsection will firstly deduce the analytical formula of $\boldsymbol{\Gamma}_i$, and then present how to solve the inverse PLS Cosserat static model.

1) Analytical formula of $\boldsymbol{\Gamma}(\boldsymbol{q})$: With the exact relation described by (2), the matrix $\boldsymbol{\Gamma}(\boldsymbol{q})$ can be formulated as

$$\boldsymbol{\Gamma}(\boldsymbol{q}) = \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{T}} = \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{q}}\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{T}} \tag{5}$$

Obviously, the calculation of the matrix $\boldsymbol{\Gamma}(\boldsymbol{q})$ can be regarded as the multiplication of $\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{q}}$ and $\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{T}}$. The exhaustive derivation of the two parts can be found in Appendix A. From the kinematic model (1b-1c), we have

$$\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{q}} = \begin{bmatrix}\mathbf{0}_3 & \boldsymbol{R}(\boldsymbol{q})\end{bmatrix}\boldsymbol{J}(\boldsymbol{q})$$

Taking the derivative of (1a) with respect to $\boldsymbol{T}$ yields

$$\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{T}} = \left(\boldsymbol{K} + \frac{\partial[\boldsymbol{G}(\boldsymbol{q})}{\partial \boldsymbol{q}}\right)^{\dagger}\boldsymbol{H}$$

Consequently, the analytical solution of $\boldsymbol{\Gamma}(\boldsymbol{q})$ is equivalent to

$$\boldsymbol{\Gamma}(\boldsymbol{q}) = [\mathbf{0}_3\ \boldsymbol{R}(\boldsymbol{q})]\ \boldsymbol{J}(\boldsymbol{q})\left(\boldsymbol{K} + \frac{\partial[\boldsymbol{G}(\boldsymbol{q})]}{\partial \boldsymbol{q}}\right)^{\dagger}\boldsymbol{H} \tag{6}$$

where $(\cdot)^{\dagger}$ implies the pseudo-inverse of $(\cdot)$.

2) Inverse PLS Cosserat Static Model: With the analytical formula of $\boldsymbol{\Gamma}(\boldsymbol{q})$ deduced in Section III-B1, for a given $\boldsymbol{u}_i \in \mathcal{W}_i$, if we can solve the inverse statics of (1a-1b) to get the associated $\boldsymbol{q}_i$, then we can obtain the value of $\boldsymbol{\Gamma}_i$ for $\mathcal{W}_i$. Therefore, the following presents how to solve this inverse problem within an optimization framework.

For a given end-effector position $\boldsymbol{u}_{\mathrm{d}}$, the first intuitive approach to solve the inverse statics is to minimize the following cost function

$$\begin{aligned}&\underset{(\boldsymbol{q},\ \boldsymbol{T})}{\arg\min}\ \|\boldsymbol{u}(\boldsymbol{q}) - \boldsymbol{u}_{\mathrm{d}}\|_2^2\\ &s.t.\quad \begin{cases}\boldsymbol{K}\boldsymbol{q} + \boldsymbol{G}(\boldsymbol{q}) = \boldsymbol{H}\boldsymbol{T}\\ \boldsymbol{T} \in \boldsymbol{\mathcal{T}}\end{cases}\end{aligned} \tag{7}$$

with $\boldsymbol{u}(\boldsymbol{q}) = \mathbf{P}\boldsymbol{g}(\boldsymbol{q})\mathbf{E}$, where $\|\cdot\|$ denotes the standard Euclidean norm. Theoretically, multiple solutions might exist for the above optimization problem, which heavily depends on the initial guess values of $\boldsymbol{q}$ and $\boldsymbol{T}$.

In order to overcome the issue of multiple solutions, this paper proposes to integrate the minimum potential energy principle into the minimal distance based cost function, since such a principle guarantees a unique configuration of any mechanical system in a static equilibrium. Motivated by this idea, for the given end-effector position $\boldsymbol{u}_{\mathrm{d}}$ with its associated strain configuration $\boldsymbol{q}$, the corresponding elastic potential energy $\mathcal{P}_E$ and gravitational potential energy $\mathcal{P}_G$ of total length L of the soft manipulator can be written as

$$\mathcal{P}_E = \frac{1}{2}\int_0^L \boldsymbol{q}^{\top}\boldsymbol{\Phi}^{\top}\boldsymbol{\mathcal{H}}(s)\boldsymbol{\Phi}\boldsymbol{q}\mathrm{d}s$$

and

$$\mathcal{P}_G = \int_0^L \boldsymbol{\mathcal{O}}^{\top}(s)\boldsymbol{\mathcal{M}}(s)\boldsymbol{\mathcal{G}}\mathrm{d}s$$

where $\boldsymbol{\Phi} \in \mathbb{R}^{6\times 6(N+1)}$ is a generalized matrix consisted of coefficient matrices of strain interpolation nodes via the PLS assumption, $\boldsymbol{\mathcal{H}}(s) \in \mathbb{R}^{6\times 6}$ stands for the stiffness matrix of each cross section, $\boldsymbol{\mathcal{O}}(s) = [\mathbf{0}_{3\times 1}^{\top}\ \boldsymbol{u}^{\top}(s)]^{\top}$ represents the position vector of each cross section with respect to the inertial frame, $\boldsymbol{\mathcal{M}}(s) \in \mathbb{R}^{6\times 6}$ is the mass matrix of any cross section, and $\boldsymbol{\mathcal{G}} \in \mathbb{R}^6$ is the gravitational acceleration twist.

Consequently, the inverse problem is solved in this paper by minimizing the following cost function

$$\begin{aligned} &\underset{(\boldsymbol{q},\ \boldsymbol{T})}{\arg\min}\ \|\boldsymbol{u}(\boldsymbol{q})-\boldsymbol{u}_{\mathrm{d}}\|_2^2+\beta(\mathcal{P}_E+\mathcal{P}_G)\\ &s.t.\quad \begin{cases}\boldsymbol{K}\boldsymbol{q}+\boldsymbol{G}(\boldsymbol{q})=\boldsymbol{H}\boldsymbol{T}\\ \boldsymbol{T}\in\boldsymbol{\mathcal{T}}\end{cases}\end{aligned} \tag{8}$$

with β being the positive weight.

To handle the inequality constraint in (8), the nonlinear interior point method [41] is used. Specifically, adding the logarithmic barrier function $\boldsymbol{\mathcal{B}}$ to the objective function in order to remove the inequality constraints, the original NLP problem (8) can be then approximately re-formulated as

$$\begin{aligned} &\underset{\overline{\boldsymbol{x}}=(\boldsymbol{q},\ \boldsymbol{T},\ \boldsymbol{\mu})}{\arg\min}\ \|\boldsymbol{u}(\boldsymbol{q})-\boldsymbol{u}_{\mathrm{d}}\|_2^2+\beta(\mathcal{P}_E+\mathcal{P}_G)-\boldsymbol{\mu}\boldsymbol{\mathcal{B}}\\ &s.t.\quad \boldsymbol{K}\boldsymbol{q}+\boldsymbol{G}(\boldsymbol{q})=\boldsymbol{H}\boldsymbol{T}\end{aligned} \tag{9}$$

with

$$\boldsymbol{\mu}=\begin{bmatrix}\mu_1 & \cdots & \mu_{2\overline{\mathrm{N}}}\end{bmatrix},\ \boldsymbol{\mathcal{B}}=\begin{bmatrix}\log(T^1-T^1_{\min})\\ \vdots\\ \log(T^{\overline{\mathrm{N}}}-T^{\overline{\mathrm{N}}}_{\min})\\ \log(T^1_{\max}-T^1)\\ \vdots\\ \log(T^{\overline{\mathrm{N}}}_{\max}-T^{\overline{\mathrm{N}}})\end{bmatrix},$$

where $\boldsymbol{\mu}$ is the barrier parameter vector. The Newton-type method is then used to find the optimal solution $\overline{\boldsymbol{x}}^*$ of such an optimization problem with the nonlinear equality constraints.

C. Global Estimation of $\boldsymbol{\Gamma}(\boldsymbol{q})$ via Neural Network

Inspired by the thought of Section III-B, a series of Jacobian matrices corresponding to the representative end-effector positions of the m sub-workspaces are computed to construct a set of matrix basis, represented by $\overline{\boldsymbol{\Gamma}}=\begin{bmatrix}\boldsymbol{\Gamma}_1^\top & \boldsymbol{\Gamma}_2^\top & \cdots & \boldsymbol{\Gamma}_m^\top\end{bmatrix}^\top\in\mathbb{R}^{3m\times\overline{\mathrm{N}}}$, and the weight vector $\boldsymbol{\alpha}(\boldsymbol{u})=\begin{bmatrix}\alpha_1(\boldsymbol{u}) & \alpha_2(\boldsymbol{u}) & \cdots & \alpha_m(\boldsymbol{u})\end{bmatrix}\in\mathbb{R}^{1\times m}$ is then defined. Finally, we propose to use the matrix basis $\overline{\boldsymbol{\Gamma}}$ obtained from the PLS Cosserat model to globally approximate $\boldsymbol{\Gamma}(\boldsymbol{q})$ corresponding to the any end-effector position $\boldsymbol{u}$ while its corresponding weight vector $\boldsymbol{\alpha}(\boldsymbol{u})$ will be determined via a trained neural network. In other words, for any end-effector position $\boldsymbol{u}\in\mathcal{W}_E$ in the whole workspace, its associated matrix $\boldsymbol{\Gamma}(\boldsymbol{q})$ approximation problem can be then transformed to a regression one, and it yields

$$\widehat{\boldsymbol{\Gamma}}(\boldsymbol{u})=\sum_{i=1}^{m}\alpha_i(\boldsymbol{u})\boldsymbol{\Gamma}_i$$

To obtain the value of $\boldsymbol{\alpha}(\boldsymbol{u})$, we use the radial basis function neural network (RBFNN) which is a commonly used three-layer feedforward network. Next, I will present how to approximate the parameter $\boldsymbol{\alpha}(\boldsymbol{u})$. According to the input-output relation (2), the regression problem can be then formulated as the following linear optimization one:

$$\underset{\boldsymbol{W}^{\mathrm{T}}}{\arg\min}\ f=\sum_{j=1}^{k}\|\boldsymbol{u}_j-\boldsymbol{u}_{j-1}-(\sum_{i=1}^{m}\alpha_i(\boldsymbol{u}_j)\boldsymbol{\Gamma}_i)(\boldsymbol{T}_j-\boldsymbol{T}_{j-1})\|_2^2 \tag{10}$$

with

$$\begin{aligned}\boldsymbol{\alpha}^\top(\boldsymbol{u}_j)&=\boldsymbol{W}^\top\overline{\boldsymbol{h}}(\boldsymbol{u}_j)\\ \overline{\boldsymbol{h}}(\boldsymbol{u}_j)&=\begin{bmatrix}h_1(\boldsymbol{u}_j) & h_2(\boldsymbol{u}_j) & \cdots & h_n(\boldsymbol{u}_j)\end{bmatrix}^\top\\ h_l(\boldsymbol{u}_j)&=\exp\left(-\frac{\|\boldsymbol{u}_j-\boldsymbol{c}_l\|^2}{2b_l^2}\right),\ l=1,2,\cdots,n.\end{aligned}$$

where $\boldsymbol{u}_j\in\mathbb{R}^3$ is the end-effector position vector, k represents the number of the equilibrium points evenly selected from the workspace, $\boldsymbol{W}^\top\in\mathbb{R}^{m\times n}$ is the estimated weight matrix, n is the number of the neurons, and $\overline{\boldsymbol{h}}(\boldsymbol{u}_j)\in\mathbb{R}^n$ with $h_l(\boldsymbol{u}_j)$ being the Gaussian kernel function of the l^{th} neuron, $\boldsymbol{c}_l$ and b_l are the center and width of the Gaussian kernel function, respectively.

From the objective function of (10), we try to find the pairs of $(\boldsymbol{T}_j,\boldsymbol{T}_{j-1})$ and $(\boldsymbol{u}_j,\boldsymbol{u}_{j-1})$. To generate the training data sets, we start from an equilibrium point j and change a little cable's tension from $\boldsymbol{T}_j$ to $\boldsymbol{T}_{j-1}$, and then the end-effector position will be changed from $\boldsymbol{u}_j$ to $\boldsymbol{u}_{j-1}$. In this way, we can obtain a set of training data. We can select k points and repeat this operation to generate multiple training data sets. Finally, the gradient descent method is used to solve the above optimization problem.

Remark 1: It must be pointed out that one possible solution is to approximate the Jacobian matrix $\boldsymbol{\Gamma}(\boldsymbol{q})$ by using for example back propagation neural network (BPNN) as presented in [42]. Nevertheless, BPNN is susceptible to getting stuck in local minima during the optimization process, and more prone to over-fitting, especially when the data is scarce. In our work, we use a hybrid strategy of both the PLS Cosserat model and RBFNN to calculate the approximation of $\boldsymbol{\Gamma}(\boldsymbol{q})$, which effectively ensures accurate control since it takes full advantage of the model information. On the one hand, the offline calculation method of the Jacobian matrix $\boldsymbol{\Gamma}(\boldsymbol{q})$ by using the RBFNN greatly improves control accuracy, as RBFNN can avoid local minima, over-fitting, and better approximate the real Jacobian matrix. On the other hand, compared to the FEM-based gain-scheduling control of a soft trunk robot [43], this method globally approximates $\boldsymbol{\Gamma}(\boldsymbol{q})$ of any end-effector position $\boldsymbol{u}\in\mathcal{W}_E$, and allows the soft manipulator to track different time-varying trajectories in real time.

D. Global Robust Controller Design

Specifically, according to the deduced input-output relationship in (2), a robust controller (valid for the whole workspace) for the soft manipulator is developed in this subsection. The feedback control law is designed as follows

$$\dot{\boldsymbol{T}}=\widehat{\boldsymbol{\Gamma}}^\dagger(\boldsymbol{u})(-\boldsymbol{\lambda}_1\boldsymbol{e}-\boldsymbol{\lambda}_2\int_0^t\boldsymbol{e}\mathrm{d}s) \tag{11}$$

where $\boldsymbol{\lambda}_1\in\mathbb{R}^{3\times3}$ and $\boldsymbol{\lambda}_2\in\mathbb{R}^{3\times3}$ are constant diagonal, positive-definite matrices of feedback gains; $\boldsymbol{e}=\boldsymbol{u}-\boldsymbol{u}_r$ with $\boldsymbol{u}_r$ being a constant desired end-effector position. With the introduced notations, (2) can be written as

$$\dot{\boldsymbol{e}}=-(\boldsymbol{\lambda}_1\boldsymbol{e}+\boldsymbol{\lambda}_2\int_0^t\boldsymbol{e}\mathrm{d}s)-\Delta\boldsymbol{\Gamma}\widehat{\boldsymbol{\Gamma}}^\dagger(\boldsymbol{\lambda}_1\boldsymbol{e}+\boldsymbol{\lambda}_2\int_0^t\boldsymbol{e}\mathrm{d}s) \tag{12}$$

with $\mathbf{\Gamma} = \widehat{\mathbf{\Gamma}} + \Delta\mathbf{\Gamma}$, where $\Delta\mathbf{\Gamma}$ represents the estimation error of the Jacobian matrix.

Assumption 1: With enough training samples to train the RBFNN described in Section III-C, it is assumed that there exists a constant $\varepsilon > 0$ such that

$$\|\mathbf{\Upsilon}\| \leq \varepsilon < 1, \ \forall \boldsymbol{T} \in \boldsymbol{\mathcal{T}} \tag{13}$$

with $\mathbf{\Upsilon} = \begin{bmatrix} \Delta\mathbf{\Gamma}\widehat{\mathbf{\Gamma}}^{\dagger} & \mathbf{0}_3 \\ \mathbf{0}_3 & \Delta\mathbf{\Gamma}\widehat{\mathbf{\Gamma}}^{\dagger} \end{bmatrix} \in \mathbb{R}^{6\times 6}$, where $\|\mathbf{\Upsilon}\|$ represents the spectral norm of matrix $\mathbf{\Upsilon}$, and ε is a prescribed approximation accuracy error. It is worth noting that the smaller ε is, the more accurate Jacobian matrix we have obtained by using RBFNN.

Theorem 1: In terms of the studied soft manipulator described by (2), for given $\boldsymbol{\lambda}_1$, $\boldsymbol{\lambda}_2$, there exist symmetric positive-definite matrices $\mathbf{P}$ and $\mathbf{Q}$ that fulfill the Lyapunov equation

$$\mathbf{P}\mathbf{A} + \mathbf{A}^{\top}\mathbf{P} = -2\mathbf{Q}, \tag{14}$$

with $\mathbf{A} = \begin{bmatrix} \mathbf{0}_3 & \mathbf{I}_3 \\ -\boldsymbol{\lambda}_2 & -\boldsymbol{\lambda}_1 \end{bmatrix}$, if the approximation accuracy error ε satisfies

$$\varepsilon \leq \frac{\lambda_{\min}(\mathbf{Q})}{\lambda_{\max}(\mathbf{P})\|\mathbf{B}\|}, \tag{15}$$

with $\mathbf{B} = \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 \\ -\boldsymbol{\lambda}_2 & -\boldsymbol{\lambda}_1 \end{bmatrix}$, the proposed controller (11) can then drive any $\boldsymbol{u} \in \mathcal{W}_E$ to the desired constant position $\boldsymbol{u}_r \in \mathcal{W}_E$, i.e.,

$$\lim_{t\to\infty} \|\boldsymbol{u}(t) - \boldsymbol{u}_r(t)\|_2 = 0$$

Proof 1: To prove the stability of the system, we introduce the state variable: $\boldsymbol{\zeta} = \begin{bmatrix} \int_0^t \boldsymbol{e}\mathrm{d}s \\ \boldsymbol{e} \end{bmatrix}$. Then, the system (12) can be re-written as the following state-space form:

$$\dot{\boldsymbol{\zeta}} = \mathbf{A}\boldsymbol{\zeta} + \mathbf{\Upsilon}\mathbf{B}\boldsymbol{\zeta}$$

Note that we want to prove $\boldsymbol{\zeta}(t) \to \mathbf{0}$ when $t \to \infty$, and this is equivalent to prove $V(\boldsymbol{\zeta}) \to 0$ when $t \to \infty$, where $V(\boldsymbol{\zeta})$ is a Lyapunov function defined as

$$V(\boldsymbol{\zeta}) = \boldsymbol{\zeta}^{\top}\mathbf{P}\boldsymbol{\zeta}$$

Then, the derivative of $V(\boldsymbol{\zeta})$ w.r.t. time t yields

$$\begin{aligned} \frac{\partial V}{\partial t} = \dot{V} &= \boldsymbol{\zeta}^{\top}\mathbf{P}\dot{\boldsymbol{\zeta}} + \dot{\boldsymbol{\zeta}}^{\top}\mathbf{P}\boldsymbol{\zeta} \\ &= \boldsymbol{\zeta}^{\top}\left(\mathbf{P}\mathbf{A} + \mathbf{A}^{\top}\mathbf{P}\right)\boldsymbol{\zeta} + \boldsymbol{\zeta}^{\top}\left(\mathbf{P}\mathbf{\Upsilon}\mathbf{B} + \mathbf{B}^{\top}\mathbf{\Upsilon}^{\top}\mathbf{P}\right)\boldsymbol{\zeta} \end{aligned}$$

Logically, if both (14) and (15) hold, we can conclude that

$$\begin{aligned} \dot{V} &= -2\boldsymbol{\zeta}^{\top}\mathbf{Q}\boldsymbol{\zeta} + 2\boldsymbol{\zeta}^{\top}\mathbf{P}\mathbf{\Upsilon}\mathbf{B}\boldsymbol{\zeta} \\ &\leq -2\boldsymbol{\zeta}^{\top}\mathbf{Q}\boldsymbol{\zeta} + 2\|\mathbf{P}\mathbf{\Upsilon}\mathbf{B}\|\boldsymbol{\zeta}^{\top}\boldsymbol{\zeta} \\ &\leq -2\boldsymbol{\zeta}^{\top}\mathbf{Q}\boldsymbol{\zeta} + 2\lambda_{\max}(\mathbf{P})\|\mathbf{\Upsilon}\mathbf{B}\|\boldsymbol{\zeta}^{\top}\boldsymbol{\zeta} \\ &\leq -2\boldsymbol{\zeta}^{\top}\mathbf{Q}\boldsymbol{\zeta} + 2\frac{\lambda_{\max}(\mathbf{P})}{\lambda_{\min}(\mathbf{Q})}\|\mathbf{\Upsilon}\mathbf{B}\|\boldsymbol{\zeta}^{\top}\mathbf{Q}\boldsymbol{\zeta} \\ &\leq -2\left(1 - \frac{\lambda_{\max}(\mathbf{P})}{\lambda_{\min}(\mathbf{Q})}\|\mathbf{\Upsilon}\|\|\mathbf{B}\|\right)\frac{\lambda_{\min}(\mathbf{Q})}{\lambda_{\max}(\mathbf{P})}\boldsymbol{\zeta}^{\top}\mathbf{P}\boldsymbol{\zeta} \\ &\leq -2\left(1 - \varepsilon\frac{\lambda_{\max}(\mathbf{P})}{\lambda_{\min}(\mathbf{Q})}\|\mathbf{B}\|\right)\frac{\lambda_{\min}(\mathbf{Q})}{\lambda_{\max}(\mathbf{P})}V < 0 \end{aligned}$$

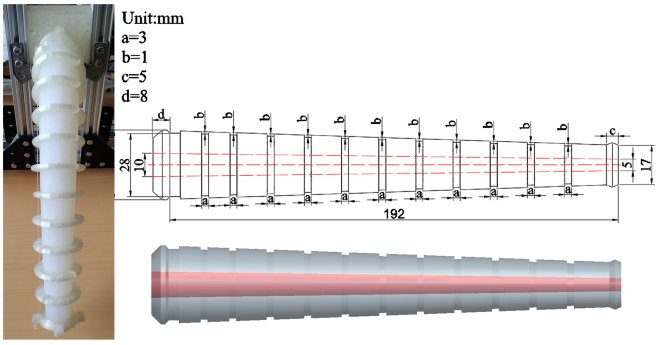

Fig. 3. Soft manipulator prototype actuated by cables [23].

which guarantees that V exponentially converges to 0. Thus, we can derive that $\boldsymbol{\zeta}(t)$ exponentially converges to $\mathbf{0}$.■

The controller designed in (11) is dependent on the pseudo-inverse of the matrix $\widehat{\mathbf{\Gamma}}$, while the RBFNN in Section III-C is not trained to provide this pseudo-inverse. To get rid of the inversion during the run time of the control, $\widehat{\mathbf{\Gamma}}^{\dagger}$ can also be directly obtained by using RBFNN as long as $\mathbf{\Gamma}^{\dagger}$ is selected as the matrix base.

IV. Experimental Testing on the Soft Manipulator

First of all, we describe the soft manipulator control system, parameters setting of workspace estimation, and parameter tuning of RBFNN-based optimization as well. Next, a series of experiments are conducted to demonstrate the effectiveness and robustness of the proposed controller. In the end, the experimental comparison between the control scheme which is only valid in a certain sub-workspace and the proposed robust controller suitable for the whole workspace is carried out.

A. Soft Manipulator Prototype Control System

The conical manipulator prototype is controlled by 4 independent cables which are attached to the tip and pass through the small holes of rigid rings mounted on the isotropic silicone body. To obtain the end-effector position of the soft manipulator, a long hole inside the whole body is made, and a magnetic sensor is placed at the tip of the soft manipulator to measure the end-effector position in real time. In detail, the exact geometric parameters of the investigated soft manipulator are illustrated in Fig. 3, and the identified material parameters from [23] are: Young's modulus $E = 2.563 \times 10^5$ Pa, shear modulus $G = 8.543 \times 10^4$ Pa, and density of material $\rho = 1.41 \times 10^3$ kg/m^3. In addition to the robot prototype, there are micro controller, position sensor, Matlab environment on the laptop, and 4 stepping motors for control of the cable length, as shown in Fig. 4.

As for the optimization-based approach, we propose to discretize the angles with a discretization step size of 0.05 Radian (Unit), 200 successive rays with respective direction vector $\boldsymbol{\mathcal{E}}^{(i)}$, $i = 1, 2, \cdots, 200$ emanating at angular intervals of the angles $\gamma_x = \gamma_y = \gamma_z = 2\pi/200$ from the initial radiating point which is exterior to the workspace. In this case, the PLS Cosserat model is divided into three sections. Finally, the

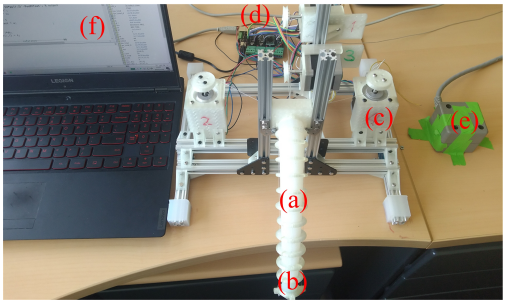

Fig. 4. Experimental setup. (a) A silicone soft manipulator; (b) End-effector; (c) Stepping motor; (d) Micro controller; (e) Polhemus magnetic position sensor; (f) Matlab environment on the laptop.

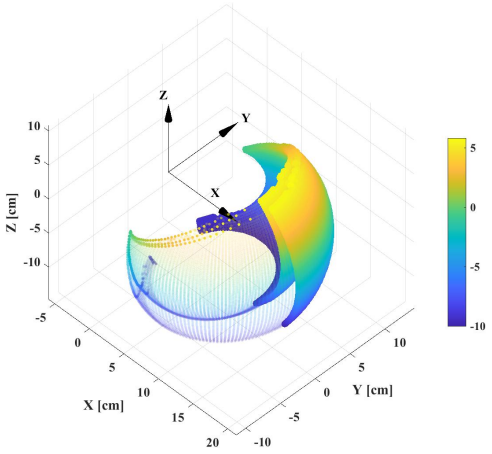

Fig. 5. The whole workspace of the investigated soft manipulator.

whole workspace of the cable-driven soft manipulator shown in Fig. 5 can be obtained.

According to the proposed decomposition rule (4), the whole workspace was finally divided into 8 sub-workspaces where the 8 representative end-effector position points (as illustrated in Fig. 6) that are respectively distributed in their corresponding sub-workspace can be chosen. Calculating their associated configuration variables by solving the inverse model, we can then acquire 8 Jacobian matrices to form a matrix basis for estimating (off-line) $\mathbf{\Gamma}(\boldsymbol{q})$ corresponding to any end-effector position $\boldsymbol{u} \in \mathcal{W}_E$.

In our test, the number of neurons required for the approximation of the Jacobian matrix $\mathbf{\Gamma}(\boldsymbol{q})$ is determined by recording the achievable control performance with a given number of neurons, and then, increasing the number of neurons until the desired tracking effects are obtained. To determine the number of neurons, we have defined the relative error between end-effector position difference and its estimation as

$$e = \frac{1}{k}\sum_{j=1}^{k} \frac{\|\boldsymbol{u}_j - \boldsymbol{u}_{j-1} - \left(\sum_{i=1}^{m} \alpha_i(\boldsymbol{u}_j)\mathbf{\Gamma}_i\right)(\boldsymbol{T}_j - \boldsymbol{T}_{j-1})\|_2}{\|\boldsymbol{u}_j - \boldsymbol{u}_{j-1}\|_2}$$

and compared this metric e by using RBFNN with different number of neurons. The smaller the metric e, the more accurate approximation of $\mathbf{\Gamma}(\boldsymbol{q})$. To determine the number of the neurons, many trials were required in the simulation environment. Initially, 6 neurons were used, and then the

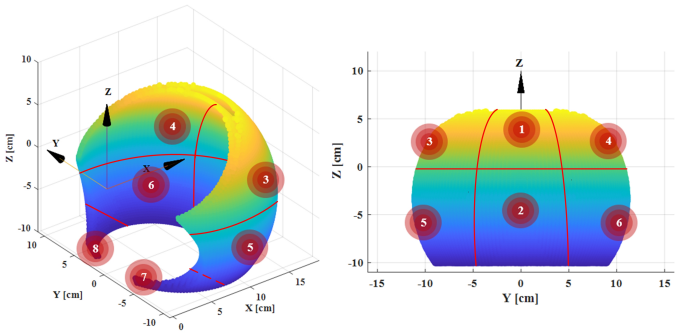

Fig. 6. From the left, the distribution of the representative end-effector positions selected from all sub-workspaces, followed by the front view of the selected points.

number was gradually increased to 27. We noticed that when utilizing more than 24 neurons, e is almost the same as that of using 24 neurons. Based on those simulations, using 24 neurons, the Jacobian matrix has been approximated with sufficient accuracy such that the estimation error of Jacobian matrix satisfies Assumption 1, and the control performance has also reached the desired level. During the training process, the estimated weight matrix in (10) was initialized to zero, and the center of each neuron was dependent on the evenly chosen end-effector position across the whole workspace.

Since we trained (off-line) the RBFNN in the simulation environment, the sufficient data sets can be generated. We tried to evenly choose 40 equilibrium points from the whole workspace of the manipulator. For each point j, we can then collect 4 groups of data pairs near itself by applying a small perturbation to each cable. Thus, 160 sets of data pairs can be obtained. We selected 50% of the data sets as training sets which were enough to satisfy the control performance, and the remaining 50% for validation purpose. The convergence strategy of the RBFNN involves initializing the RBF centers appropriately and iteratively adjusting the weights using optimization techniques to minimize the objective function, and finally we use the gradient descent method to solve the linear optimization problem.

B. Experimental Testing

To evaluate the performance of the proposed controller with the obtained Jacobian matrix, different experiments including slowly time-varying trajectory tracking tests and robustness analysis are performed on the soft manipulator prototype.

1) Slowly Time-varying Trajectory Tracking Tests: In this test, we used the RBFNN-based robust controller to track circle and star-shaped slowly time-varying trajectories. The results have been demonstrated in Fig. 7 and Fig. 8, clearly showcasing the soft manipulator can rapidly track the different slowly time-varying trajectories. Basically, the proposed PLS Cosserat static model-based controller is feasible and effective for the soft manipulator to move in its whole workspace.

2) Robustness Analysis: In an effort to demonstrate the robustness of the proposed controller, a small Teenage Mutant Ninja Turtle which can be considered as an externally permanent disturbance hanged on the soft body. Simultaneously,

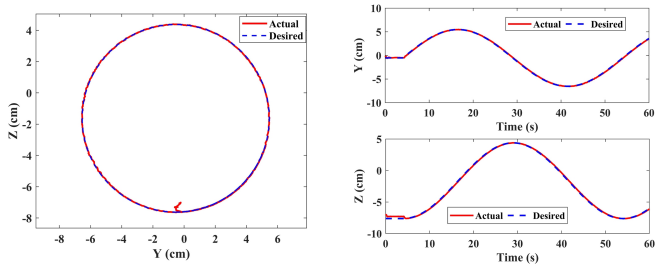

Fig. 7. Circular time-varying trajectory tracking in Y-Z plane by applying the proposed global controller.

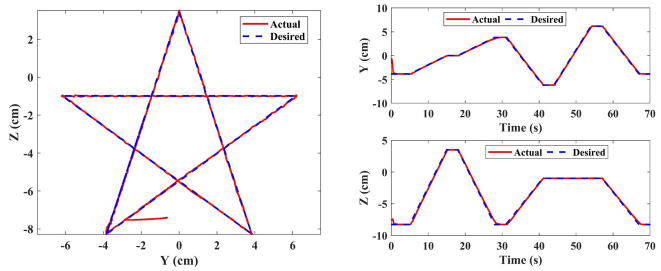

Fig. 8. Star-shaped time-varying trajectory tracking in Y-Z plane by using the proposed global controller.

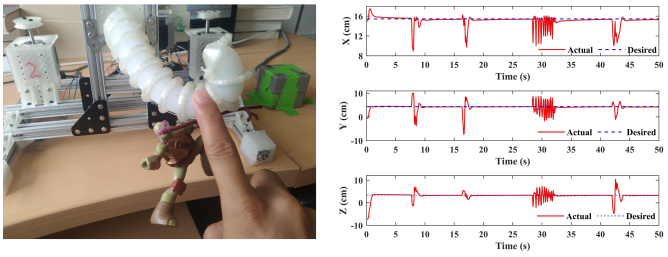

Fig. 9. The robustness validation of the soft manipulator from one point to another one under the combination of the temporary and permanent disturbances.

the temporary disturbances are exerted on the soft manipulator with hand. The manipulator reaches the desired point $\boldsymbol{u}_{ref} = \begin{bmatrix}15.44 & 4.33 & 3.40\end{bmatrix}^\top$ from the initial equilibrium point $\boldsymbol{u}_0 = \begin{bmatrix}14.83 & 0.00 & -7.53\end{bmatrix}^\top$. It can be clearly seen from the experimental results depicted in Fig. 9 that the proposed controller is robust in the sense that end-effector of the soft manipulator still returns to the desired position after a small fluctuation.

C. Comparison Results and Discussion

It can be observed that the size of circle and star-shaped trajectories for the global controller tracking are slightly smaller than those for the local controller tracking introduced in [24], which may not fully highlight the strengths of the proposed control scheme despite its good tracking performances. Therefore, to intuitively demonstrate the advantages of the proposed controller, we have implemented a comparison of the tracking performance for the slowly time-varying spatial curve trajectory by using the proposed control scheme and

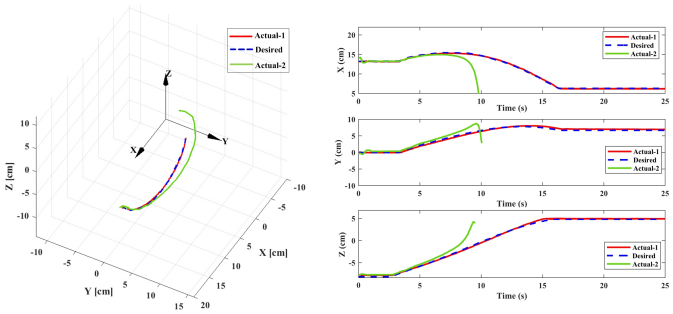

Fig. 10. Tracking effect comparison of the slowly time-varying curve trajectory in 3D space between the local controller (green line) with a constant Jacobian matrix and the proposed controller (red line).

the controller with the constant Jacobian matrix. The result indicates that the controller with a fixed Jacobian matrix for the manipulator to track a slowly time-varying curve trajectory around the whole workspace results in the divergence of the system, as shown by the green line in Fig. 10. However, the tracking effect for the global controller is excellent, as the Jacobian matrix $\boldsymbol{\Gamma}$ of any end-effector position within the whole workspace can be obtained. Thus, we can conclude that global tracking performance has been achieved by unifying RBFNN and PLS Cosserat.

To further showcase the superiority of the global control method, a quantitative analysis of the tracking performance of the studied soft manipulator given by the proposed global controller and gain-scheduling controller in terms of maximum error (ME) and root mean square error (RMSE) in task-space trajectory of the end-effector has been performed and can be found in Table I. From this table, it is confirmed that the proposed global control scheme gives minimum tracking errors (ME and RMSE) in X, Y and Z directions of different trajectories (circle, star, and spatial curve) as compared to the gain-scheduling controller. The used gains of the gain-scheduling controller are displayed in Table II.

Through the tracking effect over the period of operation of the system, we have found that the fine motion control possible with the proposed global controller is extremely useful for the positioning tasks, as it enables all the capabilities of the soft manipulator to be exploited.

V. Conclusion

This contribution opens, for the first time, the possibility of controlling the soft manipulator via a hybrid approach of both PLS Cosserat static model and RBFNN to approximate the Jacobian matrix between the task space and the actuator space of the soft manipulator in its whole workspace, based on which we designed a robust closed-loop controller to control the end-effector position of the soft manipulator. Experimental results for the soft manipulator following the fixed point in the presence of external disturbances or moving along slowly time-varying trajectories are presented to demonstrate the performance (i.e., feasibility and robustness) provided by the proposed global control technique. Additionally, the global controller is independent of any specific robot's cross-section,

TABLE I
QUANTITATIVE ANALYSIS OF END-EFFECTOR POSITION TRACKING CONTROL ALONG X, Y AND Z DIRECTIONS FOR DIFFERENT CONTROLLERS

Control schemes	Trajectory types	u_x (Unit:cm)		u_y (Unit:cm)		u_z (Unit:cm)	
		ME	RMSE	ME	RMSE	ME	RMSE
Global controller	Circle	0.106	0.027	0.100	0.033	0.011	0.062
	Star	0.091	0.021	0.096	0.011	0.098	0.042
	Curve	0.128	0.103	0.117	0.091	0.104	0.085
Gain-scheduling controller [43]	Circle	0.209	0.142	0.214	0.161	0.257	0.194
	Star	0.255	0.149	0.269	0.157	0.230	0.135
	Curve	0.232	0.163	0.218	0.134	0.245	0.151

TABLE II
PARAMETER SETUPS OF THE GAIN-SCHEDULING CONTROLLER

Trajectory types	$\boldsymbol{\lambda}_1$	$\boldsymbol{\lambda}_2$
Circle	$\begin{bmatrix} 15 & & \\ & 15 & \\ & & 3.5 \end{bmatrix}$	$\begin{bmatrix} 14 & & \\ & 14 & \\ & & 2.5 \end{bmatrix}$
Star	$\begin{bmatrix} 20 & & \\ & 20 & \\ & & 7.0 \end{bmatrix}$	$\begin{bmatrix} 16 & & \\ & 16 & \\ & & 6.5 \end{bmatrix}$
Curve	$\begin{bmatrix} 18 & & \\ & 18 & \\ & & 8.5 \end{bmatrix}$	$\begin{bmatrix} 12 & & \\ & 12 & \\ & & 2.5 \end{bmatrix}$

and thus it can be easily adapted to controlling a wide range of soft manipulators.

In the near future, the PLS Cosserat dynamic model-based controllers will be designed to achieve the strain and position control of the soft manipulator, and experiments on the soft prototype will be carried out to validate the tracking performances of the controllers.

APPENDIX

A. Derivation of $\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{q}}$ and $\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{T}}$

1) $\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{q}}$: According to the definition of the velocity twist $\boldsymbol{\eta}$, the linear velocity $\boldsymbol{V}$ can be formulated as

$$\boldsymbol{V} = \boldsymbol{R}^{-1}(\boldsymbol{q})\dot{\boldsymbol{u}} = \boldsymbol{R}^{-1}(\boldsymbol{q})\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{q}}\,\dot{\boldsymbol{q}} \tag{16}$$

Combining the definition of velocity twist with differential model (1c), the linear velocity $\boldsymbol{V}$ can also be written as

$$\boldsymbol{V} = \begin{bmatrix}\mathbf{0} & \mathbf{I}_3\end{bmatrix}\boldsymbol{\eta} = \begin{bmatrix}\mathbf{0} & \mathbf{I}_3\end{bmatrix}\boldsymbol{J}(\boldsymbol{q})\,\dot{\boldsymbol{q}} \tag{17}$$

Using the expressions (16) and (17), the partial derivative of $\boldsymbol{u}$ with respect to $\boldsymbol{q}$ can be deduced

$$\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{q}} = \boldsymbol{R}(\boldsymbol{q})\,[\mathbf{0}_3,\ \mathbf{I}_3]\boldsymbol{J}(\boldsymbol{q}) = [\mathbf{0}_3, \boldsymbol{R}(\boldsymbol{q})]\boldsymbol{J}(\boldsymbol{q})$$

2) $\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{T}}$: Taking the derivative of (1a) with respect to $\boldsymbol{T}$, we have

$$\left(\boldsymbol{K} + \frac{\partial[\boldsymbol{G}(\boldsymbol{q})]}{\partial \boldsymbol{q}}\right)\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{T}} = \boldsymbol{H}$$

It is then supposed that the cables are installed in a manner such that the matrix $\left(\boldsymbol{K} + \frac{\partial[\boldsymbol{G}(\boldsymbol{q})]}{\partial \boldsymbol{q}}\right)$ deduced is invertible (if it is not, we then use pseudo-inverse), then the partial derivative of $\boldsymbol{q}$ with respect to $\boldsymbol{T}$ can be obtained

$$\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{T}} = \left(\boldsymbol{K} + \frac{\partial[\boldsymbol{G}(\boldsymbol{q})]}{\partial \boldsymbol{q}}\right)^{\dagger}\boldsymbol{H}$$

REFERENCES

[1] H. Banerjee, Z. T. H. Tse, and H. Ren, "Soft robotics with compliance and adaptation for biomedical applications and forthcoming challenges," *Int. J. Robot. Autom.*, vol. 33, no. 1, pp. 69–80, 2018.

[2] F. Boyer, M. Porez, and W. Khalil, "Macro-continuous computed torque algorithm for a three-dimensional eel-like robot," *IEEE Trans. Robot.*, vol. 22, no. 4, pp. 763–775, 2006.

[3] R. J. Webster III and B. A. Jones, "Design and kinematic modeling of constant curvature continuum robots: A review," *Int. J. Robot. Res.*, vol. 29, no. 13, pp. 1661–1683, 2010.

[4] I. S. Godage, G. A. Medrano-Cerda, D. T. Branson, E. Guglielmino, and D. G. Caldwell, "Dynamics for variable length multisection continuum arms," *Int. J. Robot. Res.*, vol. 35, no. 6, pp. 695–722, 2016.

[5] B. A. Jones and I. D. Walker, "Kinematics for multisection continuum robots," *IEEE Trans. Robot.*, vol. 22, no. 1, pp. 43–55, 2006.

[6] A. D. Marchese and D. Rus, "Design, kinematics, and control of a soft spatial fluidic elastomer manipulator," *Int. J. Robot. Res.*, vol. 35, no. 7, pp. 840–869, 2016.

[7] R. K. Katzschmann, C. Della Santina, Y. Toshimitsu, A. Bicchi, and D. Rus, "Dynamic motion control of multi-segment soft robots using piecewise constant curvature matched with an augmented rigid body model," in *Proc. IEEE Int. Conf. Soft Robot. (RoboSoft)*, 2019, pp. 454–461.

[8] C. Della Santina, R. K. Katzschmann, A. Bicchi, and D. Rus, "Model-based dynamic feedback control of a planar soft robot: trajectory tracking and interaction with the environment," *Int. J. Robot. Res.*, vol. 39, no. 4, pp. 490–513, 2020.

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

[10] V. Falkenhahn, A. Hildebrandt, R. Neumann, and O. Sawodny, "Model-based feedforward position control of constant curvature continuum robots using feedback linearization," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 762–767.

[11] B. Caasenbrood, A. Pogromsky, and H. Nijmeijer, "Control-oriented models for hyperelastic soft robots through differential geometry of curves," *Soft Robot.*, 2022.

[12] A. Ataka, T. Abrar, F. Putzu, H. Godaba, and K. Althoefer, "Model-based pose control of inflatable eversion robot with variable stiffness," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 3398–3405, 2020.

[13] C. Duriez, "Control of elastic soft robots based on real-time finite element method," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2013, pp. 3982–3987.

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

[15] Z. Zhang, J. Dequidt, and C. Duriez, "Vision-based sensing of external forces acting on soft robots using finite element method," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 1529–1536, 2018.

[16] M. Thieffry, A. Kruszewski, C. Duriez, and T.-M. Guerra, "Control design for soft robots based on reduced-order model," *IEEE Robot. Autom. Lett.*, vol. 4, no. 1, pp. 25–32, 2018.

[17] M. Tummers, V. Lebastard, F. Boyer, J. Troccaz, B. Rosa, and M. T. Chikhaoui, "Cosserat rod modeling of continuum robots from newtonian and lagrangian perspectives," *IEEE Trans. Robot.*, vol. 39, no. 3, pp. 2360–2378, 2023.

[18] A. A. Alqumsan, S. Khoo, and M. Norton, "Robust control of continuum robots using cosserat rod theory," *Mech. Mach. Theory.*, vol. 131, pp. 48–61, 2019.

[19] F. Campisano, S. Caló, A. A. Remirez, J. H. Chandler, K. L. Obstein, R. J. Webster III, and P. Valdastri, "Closed-loop control of soft continuum manipulators under tip follower actuation," *Int. J. Robot. Res.*, vol. 40, no. 6-7, pp. 923–938, 2021.

[20] T. George Thuruthel, F. Renda, and F. Iida, "First-order dynamic modeling and control of soft robots," *Front. Robot. AI.*, vol. 7, p. 95, 2020.

[21] F. Renda, C. Armanini, V. Lebastard, F. Candelier, and F. Boyer, "A geometric variable-strain approach for static modeling of soft manipulators with tendon and fluidic actuation," *IEEE Robot. Autom. Lett.*, vol. 5, no. 3, pp. 4006–4013, 2020.

[22] F. Renda, C. Armanini, A. Mathew, and F. Boyer, "Geometrically-exact inverse kinematic control of soft manipulators with general threadlike actuators' routing," *IEEE Robot. Autom. Lett.*, 2022.

[23] H. Li, L. Xun, and G. Zheng, "Piecewise linear strain cosserat model for soft slender manipulator," *IEEE Trans. Robot.*, vol. 39, no. 3, pp. 2342–2359, 2023.

[24] H. Li, L. Xun, G. Zheng, and F. Renda, "Discrete cosserat static model-based control of soft manipulator," *IEEE Robot. Autom. Lett.*, vol. 8, no. 3, pp. 1739–1746, 2023.

[25] T. George Thuruthel, E. Falotico, M. Manti, A. Pratesi, M. Cianchetti, and C. Laschi, "Learning closed loop kinematic controllers for continuum manipulators in unstructured environments," *Soft robot.*, vol. 4, no. 3, pp. 285–296, 2017.

[26] A. Melingui, O. Lakhal, B. Daachi, J. B. Mbede, and R. Merzouki, "Adaptive neural network control of a compact bionic handling arm," *IEEE/ASME Trans. Mechatronics*, vol. 20, no. 6, pp. 2862–2875, 2015.

[27] F. Piqué, H. T. Kalidindi, L. Fruzzetti, C. Laschi, A. Menciassi, and E. Falotico, "Controlling soft robotic arms using continual learning," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 5469–5476, 2022.

[28] G. Zheng, Y. Zhou, and M. Ju, "Robust control of a silicone soft robot using neural networks," *ISA Trans.*, vol. 100, pp. 38–45, 2020.

[29] T. G. Thuruthel, E. Falotico, F. Renda, and C. Laschi, "Learning dynamic models for open loop predictive control of soft robotic manipulators," *Bioinspiration biomimetics*, vol. 12, no. 6, p. 066003, 2017.

[30] G. Li, J. Shintake, and M. Hayashibe, "Deep reinforcement learning framework for underwater locomotion of soft robot," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2021, pp. 12 033–12 039.

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

[32] J. M. Bern, Y. Schnider, P. Banzet, N. Kumar, and S. Coros, "Soft robot control with a learned differentiable model," in *Proc. IEEE Int. Conf. Soft Robot.*, 2020, pp. 417–423.

[33] G. Fang, Y. Tian, Z.-X. Yang, J. M. Geraedts, and C. C. Wang, "Efficient jacobian-based inverse kinematics with sim-to-real transfer of soft robots by learning," *IEEE/ASME Trans. Mechatronics*, 2022.

[34] T. George Thuruthel, Y. Ansari, E. Falotico, and C. Laschi, "Control strategies for soft robotic manipulators: A survey," *Soft robot.*, vol. 5, no. 2, pp. 149–163, 2018.

[35] M. Giorelli, F. Renda, M. Calisti, A. Arienti, G. Ferri, and C. Laschi, "Neural network and jacobian method for solving the inverse statics of a cable-driven soft arm with nonconstant curvature," *IEEE Trans. Robot.*, vol. 31, no. 4, pp. 823–834, 2015.

[36] T. G. Thuruthel, E. Falotico, F. Renda, and C. Laschi, "Model-based reinforcement learning for closed-loop dynamic control of soft robotic manipulators," *IEEE Trans. Robot.*, vol. 35, no. 1, pp. 124–134, 2018.

[37] C. C. Johnson, T. Quackenbush, T. Sorensen, D. Wingate, and M. D. Killpack, "Using first principles for deep learning and model-based control of soft robots," *Front. Robot. AI.*, vol. 8, p. 654398, 2021.

[38] M. Wiese, G. Runge-Borchert, B.-H. Cao, and A. Raatz, "Transfer learning for accurate modeling and control of soft actuators," in *Proc. IEEE Int. Conf. Soft Robot. (RoboSoft)*, 2021, pp. 51–57.

[39] Z. Q. Tang, H. L. Heung, K. Y. Tong, and Z. Li, "Model-based online learning and adaptive control for a "human-wearable soft robot" integrated system," *Int. J. Robot. Res.*, vol. 40, no. 1, pp. 256–276, 2021.

[40] A. Walid, G. Zheng, A. Kruszewski, and F. Renda, "Discrete cosserat method for soft manipulators workspace estimation: An optimization-based approach," *J. Mechan. Robot.*, vol. 14, no. 1, p. 011012, 2021.

[41] A. El-Bakry, R. A. Tapia, T. Tsuchiya, and Y. Zhang, "On the formulation and theory of the newton interior-point method for nonlinear programming," *J. Optimiz. theory. App.*, vol. 89, no. 3, pp. 507–541, 1996.

[42] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Neural networks for perception*, 1992, pp. 65–93.

[43] K. Wu and G. Zheng, "Fem-based gain-scheduling control of a soft trunk robot," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3081–3088, 2021.