Article

The International Journal of Robotics Research
2024, Vol. 43(6) 791–810

DOI: 10.1177/02783649231214553
journals.sagepub.com/home/ijr

Sim2Real Neural Controllers for Physics-Based Robotic Deployment of Deformable Linear Objects

Dezhong Tong[1], Andrew Choi[2], Longhui Qin[1,3], Weicheng Huang[1,3], Jungseock Joo[4,5] and Mohammad Khalid Jawed[1]

Abstract

Deformable linear objects (DLOs), such as rods, cables, and ropes, play important roles in daily life. However, manipulation of DLOs is challenging as large geometrically nonlinear deformations may occur during the manipulation process. This problem is made even more difficult as the different deformation modes (e.g., stretching, bending, and twisting) may result in elastic instabilities during manipulation. In this paper, we formulate a physics-guided data-driven method to solve a challenging manipulation task—accurately deploying a DLO (an elastic rod) onto a rigid substrate along various prescribed patterns. Our framework combines machine learning, scaling analysis, and physical simulations to develop a physics-based neural controller for deployment. We explore the complex interplay between the gravitational and elastic energies of the manipulated DLO and obtain a control method for DLO deployment that is robust against friction and material properties. Out of the numerous geometrical and material properties of the rod and substrate, we show that only three non-dimensional parameters are needed to describe the deployment process with physical analysis. Therefore, the essence of the controlling law for the manipulation task can be constructed with a low-dimensional model, drastically increasing the computation speed. The effectiveness of our optimal control scheme is shown through a comprehensive robotic case study comparing against a heuristic control method for deploying rods for a wide variety of patterns. In addition to this, we also showcase the practicality of our control scheme by having a robot accomplish challenging high-level tasks such as mimicking human handwriting, cable placement, and tying knots.

Keywords

Deformable object manipulation, data-driven models, deep neural networks, rope deployment, knots

Received 4 March 2023; Revised 24 August 2023; Accepted 15 October 2023
Senior Editor: Christian Ott
Associate Editor: Matteo Saveriano

1. Introduction

Intelligent manipulation of deformable objects, such as ropes and cloth, is necessary for beneficial and ubiquitous robots. As most objects in the practical world are non-rigid, endowing robots with proper manipulation skills for deformable objects has enormous humanitarian and economic potential. Some examples include robotic surgical suturing (Sen et al., 2016; Stefanidis et al., 2010), wire management (She et al., 2021), laundry folding (Miller et al., 2012), and caregiving for elderly and disabled communities (Kapusta et al., 2019; Clegg et al., 2018; Yu et al., 2017; Erickson et al., 2018; Pignat and Calinon, 2017). However, given the large and geometrically nonlinear deformations of deformable objects, it is difficult to obtain an obvious mapping from the observations of those manipulated objects to a concrete robotic manipulation scheme. Therefore, developing accurate and effective strategies for manipulating deformable objects is still an open research problem.

[1]Department of Mechanical & Aerospace Engineering, University of California, Los Angeles, CA, USA
[2]Department of Computer Science, University of California, Los Angeles, CA, USA
[3]School of Mechanical Engineering, Southeast University, Nanjing, China
[4]Department of Communication, University of California, Los Angeles, CA, USA
[5]NVIDIA Corporation, Santa Clara, CA, USA

Corresponding authors:
Longhui Qin, School of Mechanical Engineering, Southeast University, Nanjing, China.
Email: lhqin@seu.edu.cn

Mohammad Khalid Jawed, Department of Mechanical & Aerospace Engineering, University of California, 420 Westwood Plaza, Los Angeles, CA 90095, USA.
Email: khalidjm@seas.ucla.edu

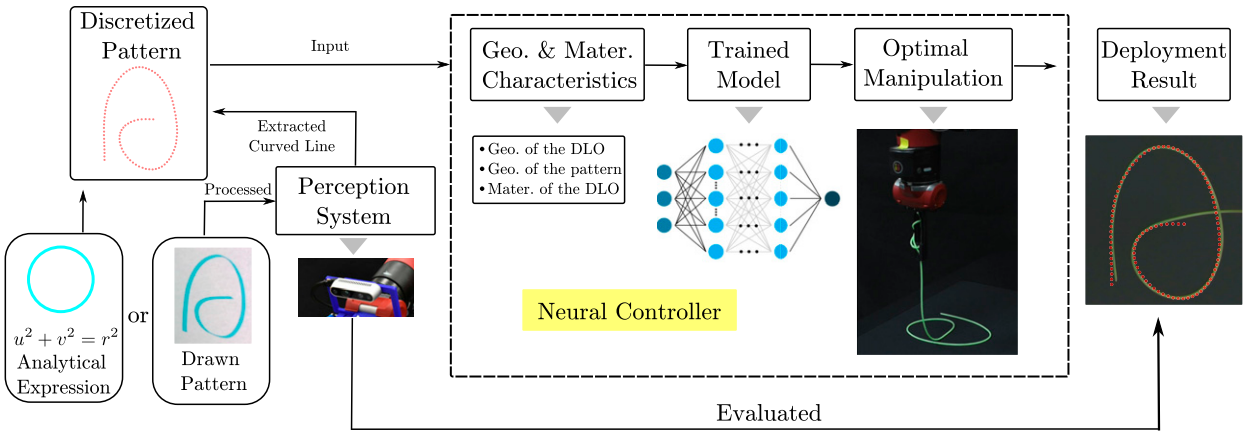

Figure 1. A full end-to-end pipeline for deploying a DLO with a sim2real physics-based deployment scheme. The pipeline begins by discretizing the DLO pattern, which can be obtained through user input via an analytical expression or a hand-drawn pattern scanned by a perception system (Choi et al., 2023a). A neural controller trained entirely from simulation then generates an optimal manipulation path for deploying the pattern, taking into account the shape of the pattern as well as the geometrical and material properties of the DLO. Finally, the deployment result is evaluated using an Intel RealSense camera positioned to provide a top-down view of the pattern to assess the accuracy of the deployment.

Among various deformable objects, deformable linear objects (DLOs), which include elastic rods and the extensions of rods-like structures, for example, cables, ropes, rods, and wires (Sanchez et al., 2018), have attracted significant research interest due to their widespread industrial and domestic applications. In this article, we focus on the category of rod-like structures and adopt the term DLO to refer to those solid elongated objects. DLOs usually possess extremely complicated nonlinearity due to the coupling of their multiple deformation modes: stretching, bending, and twisting. Given the practicality and difficulty of manipulating DLOs, there is a growing need for robust and effective methods to manipulate DLOs.

Prior works on manipulating DLOs can be divided into two categories. The first involves robots attempting to manipulate DLOs to satisfy some high-level conditions without controlling the exact shapes of DLOs. This includes knot tangling/untangling (Wakamatsu et al., 2006; Saha and Isto, 2007), obstacle avoidance (McConachie et al., 2020; Mitrano et al., 2021), and following guidance and insertion (She et al., 2021; Zhu et al., 2019). The second category involves robots attempting to precisely control the exact shape of the DLOs. For this task, a key challenge is formulating a mapping between the robot's motions and the shape of the manipulated DLO (Nair et al., 2017; Takizawa et al., 2015; Lv et al., 2022). In this article, we look into how to design a manipulation scheme for controlling the shape of elastic rods through deployment, which involves manipulating one end of DLO in a way that gradually lays the DLO on a substrate in a desired pattern with superhuman accuracy, sufficient efficiency, and strong robustness. The full end-to-end pipeline of our physics-based deployment scheme is shown in Figure 1. In addition to achieving precise shape control, we show our control method can be used to solve high-level tasks such as reproducing human writing with a deployed DLO, cable placement, and knot tying.

1.1. Deployment of DLOs

Deploying DLOs is instrumental in the practical world, for example, drawing or writing on cakes with icing (Sun et al., 2015), deploying marine cables (Whitcomb, 2000), depositing carbon nanotubes (Geblinger et al., 2008), and melting electrospinning for advanced manufacturing (Teo and Ramakrishna, 2006). Therefore, a concrete and applicable deployment scheme is a perfect solution to the shape control problem of DLOs.

Now a natural question arises: how to deploy a DLO along a prescribed pattern accurately on a substrate? Intuitively, we can assume that during the deployment process, the manipulated end $\mathbf{q}_M$ is directly above the contact point $\mathbf{q}_C$ and that the gripper's decreasing distance along the negative z-axis is equal to the added deployed length on the substrate. However, this deployment strategy does not take into consideration the nonlinear geometric deformations of the manipulated DLO and therefore, results in a poor quality deployment as illustrated by later experimental results. A schematic of the intuitive deployment method inspired by Takizawa et al. (2015) can be observed in Figure 2(a).

In this paper, we propose a framework that combines physically accurate simulation, scaling analysis, and machine learning to generate an optimized control scheme capable of deploying solid rod-like structures, which we refer to as DLOs, along any feasible pattern. Our control scheme does not currently incorporate energy dissipation from manipulations with DLOs such as viscous threads, as our physical-based simulation is based on the rod model. However, the controlling scheme can be adapted by adjusting the physical-based simulation in our combined framework to include these factors. We validate the scheme with various DLOs (e.g., elastic rods, rope, and cable) in robotic experiments. The usage of physically accurate numerical simulations not only allows us to incorporate

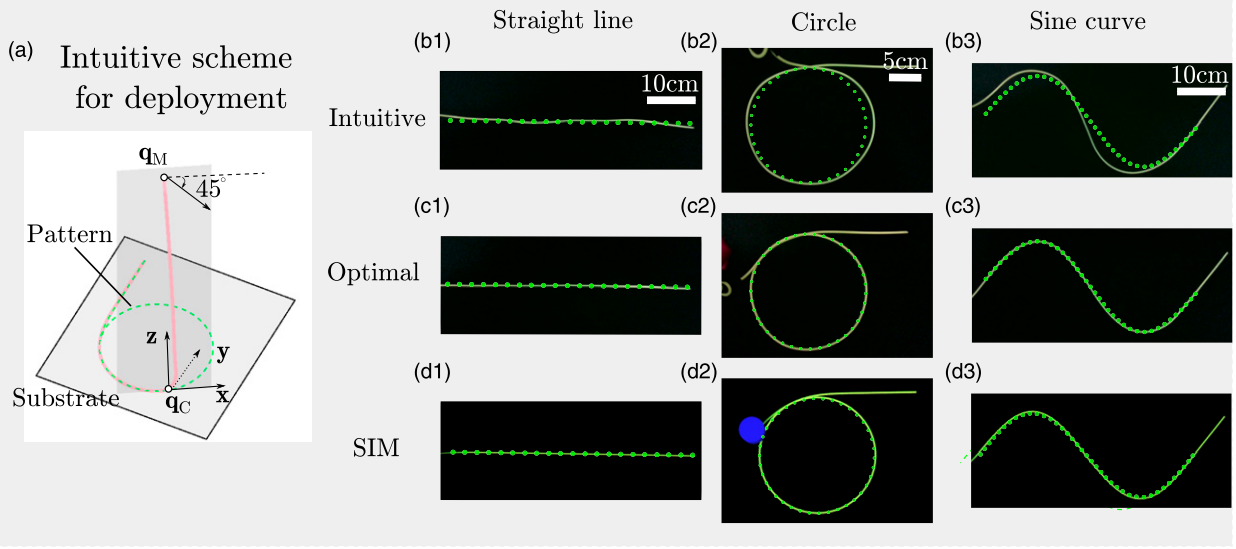

Figure 2. (a) Schematic of the intuitive control method from Takizawa et al. (2015). A DLO is being deployed along a circular pattern shown in dashed green. During the deployment process, the manipulated position $\mathbf{q}_M$ deploys along the tangent of the pattern $\mathbf{x}$ in a downward 45-degree angle with respect to the y-axis. The x-z-plane is shown in opaque gray. In addition, a comparison of experimental results between the (b) intuitive control method, (c) our designed optimal control method, (d) and simulation results using the optimal control method for the patterns of straight line, circle, and sine curve are shown. Note the effects of forgoing the influence of nonlinear geometric deformations in the intuitive deployment scheme's failure to follow simple patterns.

physics into our manipulation scheme but also results in full sim2real realization. Scaling analysis allows us to formulate the problem with generality using non-dimensional parameters, resulting in a control scheme robust against the material properties of the manipulated rods. Finally, machine learning allows us to train a neural network to model the controlling rules of deployment in a data-driven fashion. The high inference speed of our neural controller makes real-time operation feasible.

Our main contributions are as follows: (1) we formulate a solution to the DLO shape control problem through deployment with a physically robust scheme that leverages scaling analysis, resulting in generality against material, geometric, and environmental factors (friction); (2) we train a neural network (NN) with non-dimensional simulation data to serve as a fast and accurate neural controller for optimal manipulations of deployment tasks. The trained mechanics-based NN-solver has remarkable efficiency and sufficient accuracy when compared to a numerical solver; and (3) we demonstrate full sim2real realization through an extensive robotic case study demonstrating our control method's success for various practical deployment patterns with various DLOs on different substrates. In addition, we showcase the utility of our control scheme for complex high-level applications such as mimicking human handwriting, managing cables, and tying different knots.

Moreover, we have released our source codes and supplementary videos.[1]

1.2. Overview

The remainder of the article is organized as follows: we begin with a literature review related to robotic DLO shape control in Section 2. The formulation of the physics-based numerical model is discussed in Section 3, where we also formulate the deployment problem with scaling analysis. In Section 4, we analyze the nonlinearity of the deployment in detail and show how to discover optimal robot manipulation through numerical simulation. In addition, a learning framework is formulated to obtain a fast, generalized motion planning solution. Next, in Section 5, we introduce our overall robotic system, including perception and motion planning modules. Experimental results and analysis for different deployment cases, including writing letters and tying a knot, are given in Section 6. Finally, we provide concluding remarks and discuss future research avenues in Section 7.

2. Related work

Constructing a mapping relationship from observations of a manipulated DLO to the robot's action space is the primary basis of controlling DLOs. To uncover this mapping relationship, prior works usually implemented models to predict or perception systems to observe the deformations of DLOs under various manipulations. Manipulation schemes are then generated based on the predicted or sensed data. Therefore, model-based and perception-based methods can be considered two of the main categories for tackling manipulation problems of deformable objects. Due to the outstanding performance of machine learning algorithms for processing and generalizing data from models and perceptions, learning-based approaches have become another mainstream solution. In fact, many prior works take advantage of a combination of these three methods to

develop hybrid schemes for different manipulation tasks. Here, we carry out a systematic review of prior scholarly contributions that have utilized techniques based on the three delineated categories to manipulate DLOs and other deformable objects.

Perception-based approaches involve utilizing sensors such as tactile sensors (She et al., 2021) and cameras (Tang et al., 2018; Yan et al., 2020; Lee et al., 2014; Maitin-Shepard et al., 2010) to generate motions based on detected deformations. While sensors can capture the deformations as the manipulation proceeds, perception-based methods are usually not robust against the material and geometrical differences of the manipulated objects. In Tang et al. (2018), a learning-based perception framework is presented based on the Coherent Point Drift algorithm, which is able to register states of manipulated DLOs with captured images. Yan et al. (2020) developed state estimation algorithms for DLOs based on images so that a robot can perform pick-and-place manipulation on the detected configuration. However, those perception systems based on cameras fail to extract accurate results when occlusions happen. To overcome this shortcoming, tactile sensors have become prevalent in the robotics community. For example, She et al. (2021) implements GelSight, a force feedback tactile sensor, to perform robotic cable management. Since sensing data by itself cannot predict future deformations of the manipulated objects, pure perception-based methods are typically insufficient for complex deformable material manipulation tasks.

Model-based methods usually construct a physically accurate model to predict the behavior of manipulated DLOs. Multiple methods exist for modeling DLOs (Yin et al., 2021; Sanchez et al., 2018). A simple and widely used model, mass-spring systems, are often used to model deformable objects including ropes (Schulman et al., 2013; Kita et al., 2011; Macklin et al., 2014) and fabrics (Macklin et al., 2016; Guler et al., 2015). However, due to the simplification of mass-spring systems, such models usually suffer from inaccuracies when undergoing large deformations and lack of physical interpretability. Position-based dynamics is another type of modeling method that usually represents DLOs as chains of rigid bodies (Servin and Lacoursiere, 2008; Terzopoulos and Qin, 1994; Müller et al., 2007) and introduces constraints between the positions of those rigid bodies to simulate deformations. Though this method is straightforward and fast, physical interpretability is also lacking.

Finite element methods (FEM) are also popular for modeling deformable objects (Haouchine et al., 2018; Kaufmann et al., 2009; Buckham et al., 2004). However, FEM usually requires considerable computation resources and is hardly suitable for online predictions. More recently, fast simulation tools from the computer graphics community have attracted researchers' attention. For example, Discrete Elastic Rods (DER) (Bergou et al., 2008, 2010) has arisen as a robust and efficient algorithm for simulating flexible rods. Lv et al. (2022) used DER as a predictive modeling tool and achieved promising performance in DLO manipulation tasks. Though various ways to model deformable objects exists, each has their respective strengths and weaknesses and often possesses a trade-off between computational efficiency and accuracy.

Finally, learning-based approaches have become prevalent as they are capable of not only predicting the shape of the deformable object but also higher-level information such as forces (Choi et al., 2023b). Most prior works use human demonstrations or robot explorations to train controlling policies for different tasks. Nair et al. (2017), Sundaresan et al. (2020), and Lee et al. (2021) fed human-made demonstrations to robots for learning control policies for shape control and knot tying. Due to the tedium of constructing manual demonstrations, some researchers take advantage of the robots' automation to learn a policy purely from robotic exploration (Yu et al., 2022; Wang et al., 2019). To acquire training data more efficiently, researchers have also looked into training policies purely from simulation (Matas et al., 2018). Although learning-based methods have shown promising performance for manipulating deformable objects, the trained policies are often only valid for specific tasks whose state distribution matches that of the training set. In other words, learning-based approaches often fail when parameters such as the material and geometrical properties of the manipulated object change.

More relevant to the deployment task itself, Takizawa et al. (2015) implemented the intuitive control method shown in Figure 2(a) for controlling the shape of a rope to make a clove hitch knot. They achieve a success rate of 60% but require empirical hardcoded adjustments to their controlling scheme, indicating the intuitive approach's unsuitability for extreme precision deployment. Additionally, Lv et al. (2022) use a precise physical numerical model to predict the DLO's configuration during deployment. However, they use a trial-and-error method to exhaustively solve the optimal deployment path, which is computationally expensive and slow.

Although the three discussed types of methods are suitable to be combined when solving deformable manipulation problems given the complementariness of their pros and cons, how to develop a combined approach to take advantage of different types of approaches is still an open problem in the robotic community. We find that combining physically accurate simulations and machine learning can endow the learned model with excellent accuracy from the simulations and real-time performance because of the inference speed of the neural network. In addition, scaled physics analysis, which is a vital tool from the mathematical physics community, is valuable for augmenting the model with high generality.

In this article, we show how physical analysis can extract the true contributing factors of the deployment problem and how a learning-based approach can generalize the information from physics to offer real-time computation speed for the manipulation task.

3. Numerical framework and physical analysis

In this section, we first discuss the numerical framework for studying the nonlinear behaviors of the DLO during

deployment. Then, we extract the main controlling parameters for the analyzed system with Buckingham's π theorem.

3.1. Discrete differential geometry -based numerical framework

To simulate DLOs, we use a DDG-based simulator—Discrete Elastic Rods (DER) (Bergou et al., 2008, 2010)—whose physical accuracy has been validated in many various scenarios such as flagella motions (Jawed et al., 2015), knot tying (Choi et al., 2021; Tong et al., 2023a, 2023b), rod coiling (Jawed et al., 2014), and elastic buckling in structures (Tong et al., 2021).

As shown in Figure 3(a), the centerline of a DLO can be discretized into $N+1$ nodes $[\mathbf{q}_0, \mathbf{q}_1, \ldots, \mathbf{q}_N]$ $(\mathbf{q}_i \in \mathbb{R}^3)$ and N edges $[\mathbf{e}^0, \mathbf{e}^1, \ldots, \mathbf{e}^{N-1}]$ $(\mathbf{e}^i = \mathbf{q}_i - \mathbf{q}_{i-1})$. In this section, node-relevant quantities are denoted with subscripts, for example, $\mathbf{q}_i$, while edge-relevant quantities are denoted with superscripts, $\mathbf{e}^i$. Each edge $\mathbf{e}^i$ possesses two orthogonal frames: a reference frame $[\mathbf{d}_1^i, \mathbf{d}_2^i, \mathbf{t}^i]$ and a material frame $[\mathbf{m}_1^i, \mathbf{m}_2^i, \mathbf{t}^i]$. The material frame, which captures the rotation of the centerline of the DLO, can be obtained by rotating the reference frame by a rotation angle θ^i with respect to the shared director $\mathbf{t}^i$. The reference frame is arbitrarily initialized at the initial time $t = 0$s and is updated between time steps using time parallel transport (Bergou et al., 2010). The following DOF vector of size $(4N + 3)$ is constructed to capture all the deformations of the rod

$$\mathbf{q} = \left[\mathbf{q}_0, \theta^0, \mathbf{q}_1, \ldots, \mathbf{q}_{N-1}, \theta^{N-1}, \mathbf{q}_N\right]^T, \tag{1}$$

where T is the transpose operator.

Based on DER (Bergou et al., 2008, 2010), the deformations of a DLO can be divided into three modes, each corresponding to a distinct type of elastic energy: stretching, bending, and twisting. Using the formulations of these elastic energies in DER, we can outline the equations of motion (EOM) we must solve at each time step.

First, we write down the formulation of stretching energy

$$E_s = \frac{k_s}{2}\sum_{i=0}^{N-1}\left(1 - \frac{\|\mathbf{e}^i\|}{\|\tilde{\mathbf{e}}^i\|}\right)^2 \|\tilde{\mathbf{e}}^i\|, \tag{2}$$

where k_s is the stretching stiffness and $\|\tilde{\mathbf{e}}^i\|$ is the undeformed length of the $i-$ th edge. Note that we assume that the manipulated rod is of an isotropic linear elastic material in this manuscript. Hereafter, all quantities with $\tilde{()}$ refer to their resting undeformed value.

Next, the bending energy is outlined as

$$E_b = k_b \sum_{i=1}^{N-1} \frac{(\boldsymbol{\kappa}_i - \tilde{\boldsymbol{\kappa}}_i)^T(\boldsymbol{\kappa}_i - \tilde{\boldsymbol{\kappa}}_i)}{\|\tilde{\mathbf{e}}^i\| + \|\tilde{\mathbf{e}}^{i-1}\|}, \tag{3}$$

where k_b is the bending stiffness, and $\boldsymbol{\kappa}, \tilde{\boldsymbol{\kappa}} \in \mathbb{R}^3$ are the deformed and undeformed curvature vectors, respectively. Here, the relationship between the turning angle ϕ_i and curvature $\boldsymbol{\kappa}_i$ is given as $2\tan(\phi_i/2) = \|\boldsymbol{\kappa}_i\|$. The illustration of

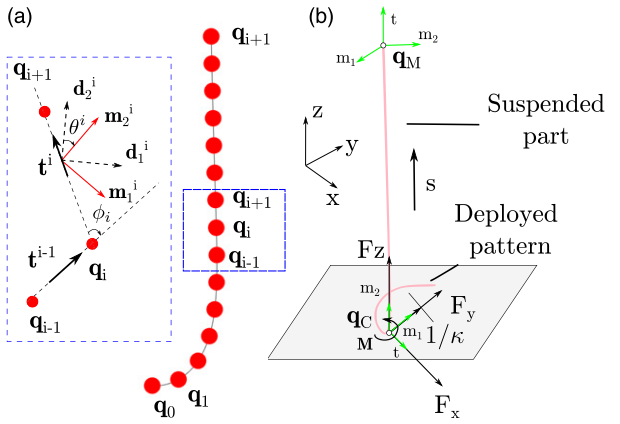

Figure 3. (a) Discrete diagram of the centerline of a DLO and relevant notations; and (b) schematic of deploying a DLO along a prescribed pattern.

turning angle ϕ_i can be seen in Figure 3(a). Note that we also assume that the resting undeformed shape of the rod is straight, that is, $\tilde{\phi}_i = 0$ in our study.

Finally, the twisting energy is

$$E_t = k_t \sum_{i=1}^{N-1} \frac{(\tau_i - \tilde{\tau}_i)^2}{\|\tilde{\mathbf{e}}^i\| + \|\tilde{\mathbf{e}}^{i-1}\|}, \tag{4}$$

where k_t is the twisting stiffness, $\tau_i = \theta^i - \theta^{i-1} + \Delta\tau_i^{\text{ref}}$ is the discrete twist, $\tilde{\tau}_i$ is the natural twist, and $\Delta\tau_i^{\text{ref}}$ is the angular difference between the reference frames on edges $\mathbf{e}^{i-1}$ and $\mathbf{e}^i$. For our DLOs, we presume $\tilde{\tau}_i$ to be zero.

With equations (2)–(4), the internal forces of the rod can be obtained as

$$\mathbf{F}^{\text{int}} = -\frac{\partial(E_s + E_b + E_t)}{\partial \mathbf{q}}. \tag{5}$$

We can then construct the equations of motion implicitly based on Newton's second law

$$\mathcal{R}(\mathbf{q}) \equiv \frac{\mathbb{M}}{\Delta t}\left(\frac{\mathbf{q}(t_{i+1}) - \mathbf{q}(t_i)}{\Delta t} - \dot{\mathbf{q}}(t_i)\right) - \mathbf{F}^{\text{int}} - \mathbf{F}^{\text{ext}} = 0, \tag{6a}$$

$$\dot{\mathbf{q}}(t_{i+1}) = \frac{\mathbf{q}(t_{i+1}) - \mathbf{q}(t_i)}{\Delta t}, \tag{6b}$$

where $\mathbb{M}$ is a square lumped mass matrix of size $4N+3$, $\mathbf{F}^{\text{int}}$ is a $(4N+3)\times 1$ elastic force vector (from equation (5), and $\mathbf{F}^{\text{ext}}$ is a $(4N+3)\times 1$ external force vector. The $\dot{()}$ operator represents the derivative of a quantity with respect to time, that is, $\dot{\mathbf{q}}(t_i)$ is the velocity vector at time t_i. Note that the subscript in equation (6a) and (6b) is the time stamp. By solving equation (6a) with Newton's method, the nonlinear geometric deformation of the manipulated rod over time can be simulated accurately.

3.2. Physical analysis and controlling rule construction

When manipulating DLOs, we should consider their geometrically nonlinear deformations. Moving forward, $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$,

and $\hat{\mathbf{z}}$ refer to the unit directors of the coordinate system defined by the connective node $\mathbf{q}_C$ shown in Figure 2(a) and 3(b).

As shown in Figure 3(b), when a DLO is being deployed along a prescribed pattern on a rigid substrate, it can be divided into two parts: a deployed part on the substrate and a suspended part that does not contact the substrate. Here, we presume the pattern on the substrate is fixed since the DLO should ideally be deployed along the prescribed pattern. Therefore, the unknown deformations only exist in the suspended part.

3.2.1. Solving the suspended part. To capture the deformations of the suspended part, we introduce some quantities to assist our analysis. First, we define $\mathbf{q}(s)$ to describe the position of the manipulated DLO's centerline, where s is the arc length along the DLO's centerline. Then, a material frame $\mathbf{m}(s) = [\mathbf{m}_1, \mathbf{m}_2, \mathbf{t}] \in SO(3)$ is attached along the DLO to capture the DLO's rotation, where $\mathbf{t} = \frac{d\mathbf{q}}{ds}$ is the tangent of the DLO. With the help of $\mathbf{q}(s)$ and $\mathbf{m}(s)$, we can fully describe the deformed configuration of the suspended part.

To solve the configuration of the suspended part, we can treat the suspended part as an independent DLO starting from the connective node $\mathbf{q}_C$ to the manipulated node $\mathbf{q}_M$. Here, $\mathbf{q}_C = \mathbf{q}(0)$ is the connective node connecting the deployed part and the suspended part. Given the continuity of the manipulated DLO, the curvature vector $\boldsymbol{\kappa}$ at $\mathbf{q}_C$ can be obtained from the prescribed pattern, where the magnitude of $\boldsymbol{\kappa}$ is denoted as κ. The manipulated end grasped by the robot is then $\mathbf{q}_M = \mathbf{q}(l_s)$, where l_s is the total curve length of the suspended part. Deployment of the pattern is then carried out purely by controlling $\mathbf{q}_M$. Since equation (6a) and (6b) implies that the DLO's configuration $\mathbf{q}(s)$ and $\mathbf{m}(s)$ can be solved when boundary conditions are determined, we can write down the governing equations for the suspended part as

$$\begin{aligned} &\mathcal{R}(\mathbf{q}) = 0, \\ &\text{s.t. } \mathbf{q}(l_s) = \mathbf{q}_M, \qquad \mathbf{R} = \mathbf{m}(l_s)\mathbf{m}(0)^T, \\ &\mathbf{q}(0) = \mathbf{q}_C, \qquad \frac{d\mathbf{q}(0)}{ds} = \mathbf{t}(0), \qquad \frac{d\mathbf{t}(0)}{ds} = \kappa\hat{\mathbf{y}}, \end{aligned} \tag{7}$$

where $\mathbf{q}_M$ is the position and $\mathbf{R}$ is the orientation of the manipulated end with respect to the connective node $\mathbf{q}_C$. Note that the position of the connective node $\mathbf{q}_C$, tangent $\mathbf{t}(0)$, and curvature vector $\kappa\hat{\mathbf{y}}$ can be determined from the deployed pattern, where $\hat{\mathbf{y}}$ is the unit vector illustrated in Figure 3(b). By solving equation (7), we can obtain the configuration of the suspended part for any predefined pattern and manipulated end pose.

3.2.2. Influence of forces and friction. Once the deformed configuration is known, we can now calculate the forces applied on the suspended part, which is key to hyper-accurate control of the DLO. We denote the external forces $\mathbf{F}^{\text{ext}} = F_x\hat{\mathbf{x}} + F_y\hat{\mathbf{y}} + F_z\hat{\mathbf{z}}$ and twisting moment $\mathbf{M}(0)$ applied on the suspended part from the connective node $\mathbf{q}_C$. Here, the moment $\mathbf{M}$ is a function of arc length s; for example, $\mathbf{M}(s)$ is the twisting moment applied on the manipulated end. The quantities $\mathbf{F}^{\text{ext}}$ and $\mathbf{M}(0)$ are relevant with the friction coefficient μ between the substrate and the rod, and μ is an unknown and uncontrollable environment factor. In addition, the quantities $\mathbf{F}^{\text{ext}}$ and $\mathbf{M}(0)$ also influence the tangent $\mathbf{t}(0)$ at the connective node $\mathbf{q}_C$ because of Newton's third law. Therefore, we must minimize quantities $\mathbf{F}^{\text{ext}}$ and $\mathbf{M}(0)$ to achieve an optimal controlling rule.

Despite the optimal controlling rule minimizing the influence of friction, it is still worth clarifying the significance of friction within this context. Though we make the strong assumption that the deployed pattern remains fixed during deployment, this is only upheld if the following relation is satisfied for the deployed segment

$$k_b\kappa'' \leq \mu_s \rho A g, \tag{8}$$

where k_b is the bending stiffness of the rod, κ'' is the second derivative of κ with respect to the arc length s $\left(\kappa'' = \frac{d^2\kappa}{ds^2}\right)$, μ_s is the static friction coefficient, ρ is the volumetric density of the rod, A is the cross-sectional area, and g is the gravitational acceleration. Equation (8) is derived by analyzing an arbitrary finite element of the deployed pattern with a clamped-end Euler–Bernoulli beam model. Clearly, friction plays a crucial role in the deployment process.

As a result, our designed optimal deployment strategy maintains a reliance on adequate friction for effective execution while the scheme mitigates external tangential forces apart from the essential friction on the substrate. Consequently, the scheme necessitates only a modest static friction coefficient between the substrate and the manipulated DLO.

3.2.3. Computing optimal grasp. In addition to the minimization of the external forces $\mathbf{F}^{\text{ext}}$ and twisting moment $\mathbf{M}(0)$ applied on the suspended part, we set up a rule for the manipulated end: the robot end-effector should induce minimal deformations on the manipulated node $\mathbf{q}_M$ so that the curvature (bending deformations) at the manipulated end should be 0. This results in the following optimization problem to compute the optimal grasp

$$\begin{aligned} &\left(\mathbf{q}_M^*, \mathbf{R}^*\right) = \arg\min \left(\|\mathbf{F}^{\text{ext}}\|^2 + \left(\frac{\|\mathbf{M}(0)\|}{h}\right)^2 \right) \\ &\text{s.t. } \frac{d\mathbf{q}(0)}{ds} = \mathbf{t}(0), \quad \frac{d\mathbf{t}(0)}{ds} = \kappa\hat{\mathbf{y}}, \\ &\frac{d^2\mathbf{q}(l_s)}{ds^2} = 0, \qquad \mathcal{R}(\mathbf{q}) = 0. \end{aligned} \tag{9}$$

By solving equation (9), optimal grasp ($\mathbf{q}_M^*$ and $\mathbf{R}^*$) can be obtained. Physical analysis tells us that a direct mapping relationship exists between the contributing factors and the optimal grasp. Recall from equations (2)–(4), that stretching stiffness k_s, bending stiffness k_b, twisting stiffness k_t, density

ρ, and rod radius h are the primary material and geometric properties of a rod. By adding in additional geometric properties such as suspended length l_s and curvature κ, the mapping relationship

$$\left(\mathbf{q}_M^*, \mathbf{R}^*\right) = f(l_s, \kappa, k_s, k_b, k_t, h, \rho), \tag{10}$$

can be constructed where $f(\cdot)$ is a highly nonlinear (and unknown) function that describes the controlling rule.

Note however the high input dimensionality of equation (10). In other words, to accurately learn the mapping $f(\cdot)$, we would have to exhaustively perform large parameter sweeps for various ranges of material and geometric parameters within simulations. This process of collecting data quickly gets out of hand due to the curse of dimensionality. To circumvent this, we can perform scaling analysis to obtain an equivalent reduced-order mapping.

3.2.4. Scaling analysis via Buckingham's π theorem. In this article, we use Buckingham's π theorem to reduce the dimensions of the mapping $f(\cdot)$. Buckingham's π theorem is a fundamental principle in dimensional analysis, stating that a physically meaningful equation involving n physical parameters can be expressed using a reduced set of $p = n - k$ dimensionless parameters derived from the original parameters. Here, k represents the number of physical dimensions. Using Buckingham's π theorem allows us to obtain a reduced-order non-dimensionalized mapping $\mathcal{F}(\cdot)$ from the original function f

$$\begin{aligned}
\left(\overline{\mathbf{q}}_M^*, \mathbf{R}^*\right) &= \mathcal{F}\left(\overline{l}_s, \overline{\kappa}, \overline{k}_s\right), \\
L_{gb} &= \left(\frac{k_b}{2\pi h^2 \rho g}\right)^{1/3}, \\
\overline{k}_s &= \frac{k_s L_{gb}^2}{k_b}, \\
\overline{\mathbf{q}}_M^* &= \frac{\mathbf{q}_M^* - \mathbf{q}_C}{L_{gb}}, \\
\overline{l}_s &= \frac{l_s}{L_{gb}}, \\
\overline{\kappa} &= \kappa L_{gb}.
\end{aligned} \tag{11}$$

Hereafter, all quantities with $\overline{()}$ indicate normalized quantities. In equation (11), all quantities are unitless so that the mapping relationship $\mathcal{F}(\cdot)$ maps from the unitless groups encapsulated the geometric and material properties to the unitless optimal robotic grasp. The benefit of doing such is that we reduce the dimensions of the mapping function $\mathcal{F}(\cdot)$ in equation (10) and eliminate the dependence of $\mathcal{F}(\cdot)$ on the units. Note that in equation (11), we do not consider the influence of the twisting stiffness k_t in this article since twisting energies are minimal compared to bending and stretching. However, the influence of k_t can also be analyzed with our proposed analysis. In the following article, we will show how to establish the nonlinear mapping function in equation (11).

4. Optimization and deep learning

In this section, we further analyze the optimization of the system to obtain the nonlinear mapping function in equation (11). Given the high nonlinearity of the system, we first solve equation (11) with a numeric optimization solver in a data-driven way. While doing so, we analyze the elastic instability of the system to choose the optimal robotic grasp for the deployment task. Afterward, we reconstruct equation (11) using a neural network to take advantage of its high inference speed. This neural controller is then used by our robotic system as the controlling law to complete various deployment tasks in Section 6.

4.1. Elastic instability in deployment along a straight line

In this section, we first take a look at an intriguing physical phenomenon: elastic instability. Elastic instability occurs when changes in the boundary conditions cause a deformed structure to become unstable. When observed visually, a small geometric perturbation of the system will lead to a substantial change in configuration (Timoshenko and Gere, 2009). An example of this can be observed when a robot employs the intuitive control method to deploy a DLO along a straight line as the rod unexpectedly adopts a curved shape on the substrate. This observation defies our intuition as the intuitive method only manipulates the DLO in the 2D plane (x-z plane) as illustrated in Figure 4(a). Consequently, the suspended part should ideally experience only 2D deformations within that plane, thereby avoiding significant deformations along the y-axis. On the contrary, this observed phenomenon results from the unaccounted elastic instability of the manipulated DLO.

Given this, it is crucial to take elastic instability into consideration when designing an optimal deployment scheme so that the robot's grasp and possible jittering of the manipulator does not introduce large undesired deformations of the DLO. To achieve this, we thoroughly analyze all potential robot grasps for manipulating a DLO in the x-z plane to achieve a straight-line deployment. Our objective is to identify an optimal grasp that satisfies equation (9) while effectively preventing the manipulated DLO from buckling due to elastic instability.

4.1.1. Discovering potential grasp region. Given the suspended part's geometric properties and boundary conditions, we can write down the constraints $\mathcal{C}$ which should be satisfied

$$\begin{aligned}
&\overline{\mathbf{q}}(\overline{s}) \cdot \widehat{\mathbf{z}} \geq 0 \ \forall \ \overline{s} \in \left[0, \overline{l}_s\right], \\
&\overline{\mathbf{F}}^{\text{ext}} \cdot \widehat{\mathbf{z}} \geq 0.
\end{aligned} \tag{12}$$

These constraints enforce that (i) the suspended part should be above the substrate and (ii) external contact responses along the z-axis should always be larger than or equal to 0.

By solving equation (7) with constraints $\mathcal{C}$, we obtain all potential robot grasps of the manipulated end, forming a closed manifold $\mathcal{M}$ for a fixed normalized suspended length

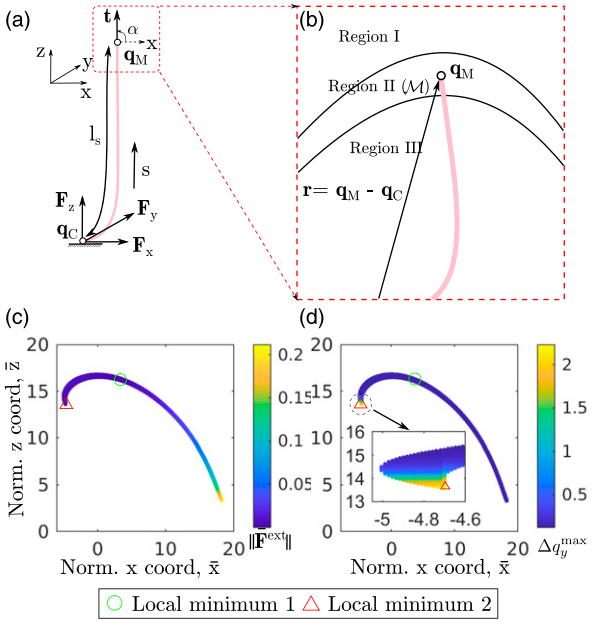

Figure 4. Schematic of a DLO manipulated in a 2D workspace (a) and its corresponding available region denoted by $\mathcal{M}$ (b). Visualization of a specific case with $\bar{l}_s$= 17.68. The force distribution is shown in (c), and (d) displays the maximum geometric deformation of the suspended part under a disturbance of $\Delta\bar{y}$= 0.12 along the y-axis.

$\bar{l}_s$. The boundary condition at the connective node $\bar{\mathbf{q}}_C$ is defined as $\mathbf{t}(0)$ = (1, 0, 0) and $\bar{\kappa}$= 0. Each point in the manifold $\mathcal{M}$ corresponds to a position $\bar{\mathbf{q}}_M$ and rotation $\mathbf{R}$ of the manipulated end. Given that the deformed configuration is located within the 2D x-z plane, we can use a 2 × 1 vector $\bar{\mathbf{q}}_M = (\bar{x}_{\text{Top}}, \bar{z}_{\text{Top}})$ to express the position of $\bar{\mathbf{q}}_M$ and a scalar value α to denote the rotation information. For example, tangent $\mathbf{t}(\bar{l}_s) = (\cos(\alpha), \sin(\alpha))$ is shown in Figure 4(a). Since the manifold $\mathcal{M}$ is a closed set, we only need to obtain the boundary of the manifold $\partial\mathcal{M}$.

To discover the boundary $\partial\mathcal{M}$, we explore along a ray $\mathbf{r}$ from the connective node $\bar{\mathbf{q}}_C$ to the manipulated node $\bar{\mathbf{q}}_M$. The robot grasp along the ray can be divided into three regions as shown in Figure 4(b). When the robot grasp exists in regions I and III, constraints $\mathcal{C}$ are not satisfied. In region I, the external force $\bar{F}_z = F_z h^2/k_b$ is smaller than 0, violating the constraints as stretching occurs, and in region III, the manipulated end is too low, leading to contact between the suspended part and the substrate. Thus, region II, existing between regions I and III, represents the manifold $\mathcal{M}$ area that satisfies the constraints $\mathcal{C}$. In this article, we implement a bisection method to obtain the boundary $\partial\mathcal{M}$ of region II. The pseudocode for the bisection method is given in Algorithm 1.

Algorithm 1: Bisection Method for Obtaining $\partial\mathcal{M}$

Input: $\bar{l}_s, \bar{k}_s, \nu$
Output: $\partial\mathcal{M}$
Func `DiscoverManifoldBoundary` $(\bar{l}_s, \bar{k}_s)$**:**
  $\theta \leftarrow$ a small positive value
  $\beta \leftarrow$ a small positive value
  $\partial M \leftarrow$ initialize an empty list
  $\delta \leftarrow$ a small positive value as tolerance
  $\mathcal{R} \leftarrow$ initialized rod solver with $\bar{l}_s, \bar{k}_s, \nu$
  while $\theta \leq \pi$ **do**
    $\mathbf{r} \leftarrow (\bar{l}_s \cos(\theta), \bar{l}_s \sin(\theta))$
    do
      $\mathbf{r} \leftarrow (1+\beta)\mathbf{r}$
      $\bar{F}_z \leftarrow \mathcal{R}(\mathbf{r})$
    while $\bar{F}_z < 0$
    $\mathbf{r}_c \leftarrow \mathbf{r}$
    while $\mathcal{C}$ is not satisfied **do**
      $\mathbf{r} \leftarrow \mathbf{r} - \delta\hat{\mathbf{r}}$
      $\bar{\mathbf{q}}, \bar{F}_z \leftarrow \mathcal{R}(\mathbf{r})$
      if $\|\mathbf{r}\| < 0$ **then**
        break
    $\mathbf{r}_f \leftarrow \mathbf{r}$
    while $\|\mathbf{r}_c - \mathbf{r}_f\| \geq \delta$ **do**
      $\bar{\mathbf{q}}, \bar{F}_z \leftarrow \mathcal{R}(\mathbf{r})$
      if $\mathcal{C}$ is satisfied **then**
        $\mathbf{r}_f \leftarrow \mathbf{r}$
      else
        $\mathbf{r}_c \leftarrow \mathbf{r}$
      $\mathbf{r} \leftarrow (\mathbf{r}_c + \mathbf{r}_f)/2$
    $\partial\mathcal{M}$.append$((\mathbf{r}\cos\theta, \mathbf{r}\sin\theta))$
    $\mathbf{r}_c \leftarrow \mathbf{r}$
    $\mathbf{r}_f \leftarrow (0, 0)$
    while $\|\mathbf{r}_c - \mathbf{r}_f\| \geq \delta$ **do**
      $\bar{\mathbf{q}}, \bar{F}_z \leftarrow \mathcal{R}(\mathbf{r})$
      if $\mathcal{C}$ is satisfied **then**
        $\mathbf{r}_c \leftarrow \mathbf{r}$
      else
        $\mathbf{r}_f \leftarrow \mathbf{r}$
      $\mathbf{r} \leftarrow (\mathbf{r}_c + \mathbf{r}_f)/2$
    $\partial\mathcal{M}$.append$((\mathbf{r}\cos\theta, \mathbf{r}\sin\theta))$
    $\theta \leftarrow \theta + \delta\theta$
return $\partial\mathcal{M}$

Note that θ in Algorithm 1 is the angle between the x-axis and ray $\mathbf{r}$. A specific case for $\bar{l}_s$= 17.68 is visualized in Figure 4(c). Since deformations only occur in the x-z plane, the twisting moment $\bar{\mathbf{M}}(0) = \mathbf{M}(0)h/k_b$ applied on the connective node $\bar{\mathbf{q}}_C$ is always 0. To achieve the optimal pose of the manipulated end for $\bar{l}_s$= 17.68, we need to find the poses in $\mathcal{M}$ that minimizes $\left\|\bar{\mathbf{F}}^{ext}\right\|$. Two local minima are found in the case shown in Figure 4(c), corresponding to two solutions of equation (9). As stated before, we must select the local minima corresponding to the stable deformed suspended part.

4.1.2. Checking elastic instability via perturbations. To test the elastic stability of these local minima, we apply a disturbance $\Delta\bar{y} = y/L_{gb}$ along the y-axis while the

manipulated end $\overline{\mathbf{q}}_M$ is at each local minimum. Figure 5 illustrates the changes in $\overline{F}_y = F_y h^2/k_b$ and the configurations resulting from these perturbations for each local optimum.

For local minimum 2, we can see a sudden snapping process, where an immediate change can be observed, while the disturbance on local minimum 1 results in a continuous and steady change. Therefore, we can conclude that the optimum for deploying the DLO is at a local minimum 1 since this minimum corresponds to a configuration with more gentle bending deformations of the suspended part.

Here, we also illustrate that the neighboring region around the elastic instability points has a higher tendency for significant deformations when the jittering of the manipulator occurs. In simulation, we introduce a small disturbance of $\Delta\overline{y} = 0.12$ along the y-axis for all potential robot grasps on the manifold $\mathcal{M}$. Figure 4(d) illustrates the maximum displacement of the suspended part along the y-axis $\Delta q_y^{\max} = \max_{0\le s\le l_s}(\mathbf{q}(s)\cdot\hat{\mathbf{y}})$ caused by this small disturbance. It is evident from the results that the neighboring region around local minimum two exhibits a higher tendency for significant deformations along the y-axis. Consequently, robot grasps within this region are more likely to induce instability in the manipulated DLO.

We can now output the optimal deployment rule for a straight line using the method introduced in this section. In the next section, we focus on optimal 3D manipulation, that is, deploying patterns with non-zero curvature. The following section discusses how to use a first-order optimization algorithm to solve equation (9) for deploying any arbitrary prescribed pattern, where the optima for straight-line deployment is used as seeds when searching for the optima of more complex patterns.

4.2. Deployment in 3D workspace

As mentioned in Section 3.2, the mapping relationship $\mathcal{F}(\cdot)$ in equation (11) must be constructed to achieve optimal deployment in the 3D workspace. For the connective node of any prescribed pattern, since the deformations of the pattern are only in the x-y plane, we can ensure that the twisting moment $\mathbf{M}(0)$ can always be 0. Therefore, the optimal pose of the manipulated end can be obtained by minimizing $\left\|\overline{\mathbf{F}}^{\text{ext}}\right\|$ by solving

$$\nabla_{\overline{\mathbf{q}}_M}\left\|\overline{\mathbf{F}}^{\text{ext}}\right\| = \frac{\partial \overline{\mathbf{F}}^{\text{ext}}}{\partial \overline{\mathbf{q}}_M}\overline{\mathbf{F}}^{\text{ext}} = 0. \tag{13}$$

As the deploying rod is a continuous system, $\overline{\mathbf{F}}^{\text{ext}}$ must change when $\overline{\mathbf{q}}_M$ changes. Therefore, we can convert equation (13) to be a root-finding problem

$$\overline{\mathbf{F}}^{\text{ext}} = 0. \tag{14}$$

As discussed before, solving the configurations of the deploying DLO is a boundary value problem. Since the pattern's shape determines the boundary conditions on the connective end, the external forces $\overline{\mathbf{F}}^{\text{ext}}$ are influenced solely by the manipulated end pose $\overline{\mathbf{q}}_M$, with a unique corresponding $\mathbf{R}$ for describing the rotation of the manipulated end.

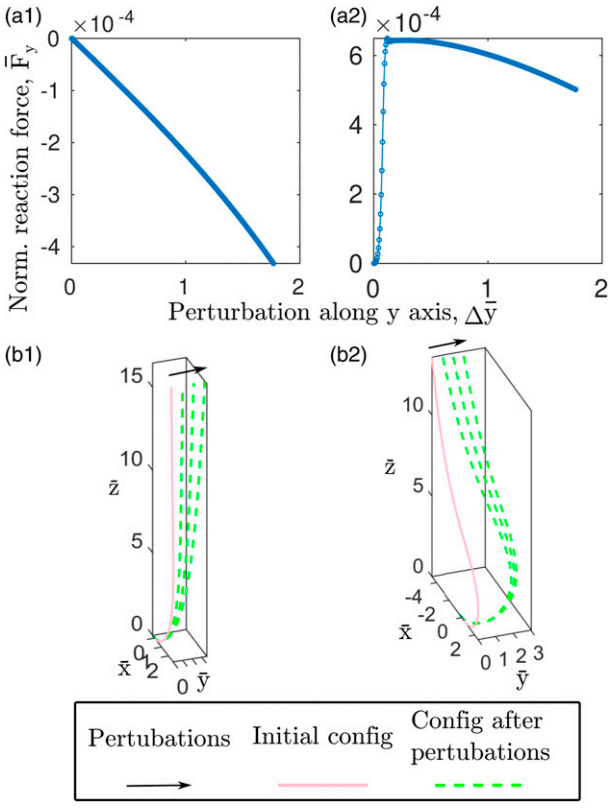

Figure 5. Change of the magnitude of normalized force $\overline{F}_y$ when adding a perturbation along the y-axis at local minimum 1 (a1) and local minimum 2 (a2), and change of the configurations of the rod when adding the perturbations at local minimum 1 (b1) and local minimum 2 (b2) for $\overline{l}_s = 17.68$.

Given the high nonlinearity of the DLO, it is nontrivial to solve the root-finding problem in equation (14) analytically. Therefore, we employ a finite difference approach to calculate the numerical Jacobian of $\overline{\mathbf{F}}^{\text{ext}}$. We perturb the manipulated end along x, y, and z-axes with a small distance δ and use the finite difference to compute the numerical Jacobian

$$\mathbf{J}^{\text{ext}} = \frac{1}{\delta}\begin{bmatrix} \overline{\mathbf{F}}^{\text{ext}}(\overline{\mathbf{q}}_M + \delta\hat{\mathbf{x}}) - \overline{\mathbf{F}}^{\text{ext}}(\overline{\mathbf{q}}_M), \\ \overline{\mathbf{F}}^{\text{ext}}(\overline{\mathbf{q}}_M + \delta\hat{\mathbf{y}}) - \overline{\mathbf{F}}^{\text{ext}}(\overline{\mathbf{q}}_M), \\ \overline{\mathbf{F}}^{\text{ext}}(\overline{\mathbf{q}}_M + \delta\hat{\mathbf{z}}) - \overline{\mathbf{F}}^{\text{ext}}(\overline{\mathbf{q}}_M) \end{bmatrix}^T, \tag{15}$$

where T is the transpose operator and $\delta\hat{\mathbf{x}}$, $\delta\hat{\mathbf{y}}$ and $\delta\hat{\mathbf{z}}$ are small perturbations along x, y, and z-axes, respectively, that is, $\delta\hat{\mathbf{x}} = [\delta, 0, 0]^T$.

Here, $\mathbf{J}^{\text{ext}}$ is a 3×3 matrix and can be used to calculate the Newton search step so that equation (14) can be solved with a gradient descent method. Further details of this solving process are stated in Algorithm 2. Additionally, we also implement a line search algorithm to help determine the appropriate step size for the Newton search step $\Delta\overline{\mathbf{q}}$ as shown in Algorithm 3.

Algorithm 2: Gradient Descent for Optimal Grasp

Input: $\bar{l}_s, \bar{\kappa}\hat{\mathbf{y}}, \bar{k}_s, \nu$
Output: $\bar{\mathbf{q}}_M^\star$
Func `OptimalGrasp` $(\bar{l}_s, \bar{\kappa}, \bar{k}_s)$ **:**
  $k \leftarrow 0$
  $\delta \leftarrow$ a small value as tolerance
  $\bar{\mathbf{q}}_M^{(k)} \leftarrow$ initialize a random pose of end-effector
  $\mathcal{R}(\cdot) \leftarrow$ initialize the rod solver with $\bar{l}_s, \bar{\kappa}, \bar{k}_s$
  do
    $\bar{\mathbf{F}}^{\text{ext}} \leftarrow \mathcal{R}(\bar{\mathbf{q}}_M^{(k)})$
    $\mathbf{J}^{\text{ext}} \leftarrow$ equation (15)
    $\Delta\bar{\mathbf{q}} \leftarrow (\mathbf{J}^{\text{ext}})^{-1}\bar{\mathbf{F}}^{\text{ext}}$
    $\alpha \leftarrow$ `LineSearch` $(\bar{\mathbf{q}}_M^{(k)}, \Delta\mathbf{q}, \|\bar{\mathbf{F}}^{ext}\|, \mathcal{R})$
    $\bar{\mathbf{q}}_M^{(k+1)} \leftarrow \bar{\mathbf{q}}_M^{(k)} - \alpha\Delta\bar{\mathbf{q}}$
    $k \leftarrow k + 1$
  while $\|\bar{\mathbf{F}}^{ext}\| \geq \delta$
  $\bar{\mathbf{q}}_M^* \leftarrow \bar{\mathbf{q}}_M^{(k)}$
  return $\bar{\mathbf{q}}_M^*$

Algorithm 3: Line Search Algorithm

Input: $\bar{\mathbf{q}}_M, \Delta\mathbf{q}, f_0, \mathcal{R}$
Output: α
Func
  `LineSearch` $(\bar{\mathbf{q}}_M, \Delta\mathbf{q}, f_0, \mathcal{R}, \alpha_0 = 1, m = 0.5)$ **:**
  $\alpha \leftarrow \alpha_0$
  $k \leftarrow 0$
  success $\leftarrow$ False
  do
    $\bar{\mathbf{q}}_M^{(k)} \leftarrow \bar{\mathbf{q}}_M - \alpha\Delta\mathbf{q}$
    $\bar{\mathbf{F}}^{\text{ext}} \leftarrow \mathcal{R}(\bar{\mathbf{q}}_M^{(k)})$
    $f^{(k)} \leftarrow \|\bar{\mathbf{F}}^{\text{ext}}\|$
    if $f^{(k)} \geq f_0$ **then**
      $\alpha \leftarrow m\alpha$
      $k \leftarrow k + 1$
    else
      success $\leftarrow$ True
  while not success
  return α

In this article, both position $\bar{\mathbf{q}}_M$ and rotation $\mathbf{e}$ of the manipulated end are represented as 3×1 vectors: $\bar{\mathbf{q}}_M = (\bar{x}^{\text{Top}}, \bar{y}^{\text{Top}}, \bar{z}^{\text{Top}})$ and $\mathbf{e} = (e_x, e_y, e_z)$. The rotation vector $\mathbf{e}$ can be translated to a rotation matrix through an axis-angle representation $(\hat{\mathbf{e}}, \|\mathbf{e}\|)$, where $\|\mathbf{e}\|$ is the rotation angle along the rotation axis $\hat{\mathbf{e}} = \mathbf{e}/\|\mathbf{e}\|$. For an input tuple $(\bar{l}_s, \bar{\kappa}, \bar{k}_s)$, we can now solve for the optimal pose of the manipulated end $(\bar{\mathbf{q}}_M^*, \mathbf{e}^*)$. Visualizations of the discretely solved optimal poses obtained from simulation are shown as red hollow circles in Figure 6.

We now know how to obtain the optimal manipulation pose given the input $(\bar{l}_s, \bar{\kappa}, \bar{k}_s)$ with simulations. A numeric solver based on simulations for generating the optimal trajectory for various prescribed patterns is released (see [1]). However, solving for the optimal poses with the numeric solver makes real-time manipulation infeasible as trajectory generation can take several hours. Instead, the following section introduces using a neural network to learn the optimal controlling rule for fast real-time inference.

4.3. Training the neural controller

Rather than obtaining the optimal grasp through the numerical solver detailed in the previous section, we train a neural network to learn an analytical approximation of $\mathcal{F}(\cdot)$ similar to the approach in Choi et al. (2023b). We use a simple fully connected feed-forward nonlinear regression network consisting of 4 hidden layers, each with 392 nodes, as the network architecture. Aside from the output, each layer is followed by a rectified linear unit (ReLU) activation.

We frame the neural controller to have an input $\mathbf{i} \in \mathbb{R}^3$ and an output $\mathbf{o} \in \mathbb{R}^6$, where the input consists of the three non-dimensional values $\bar{l}_s$, $\bar{\kappa}$, and $\bar{k}_s$ and the output consists of two concatenated 3×1 vectors: the optimal position $\bar{\mathbf{q}}_M^*$ and rotation $\mathbf{e}^*$ of the manipulated end. Using our simulation framework, we construct a dataset $\mathcal{D}$ consisting of 6358 training samples.

When training the neural controller, we first preprocess all inputs $\mathbf{i}$ through the standardization

$$\mathbf{i}' = \frac{\mathbf{i} - \bar{\mathbf{i}}_{\mathcal{D}}}{\boldsymbol{\sigma}_{\mathcal{D}}},$$

where $\bar{\mathbf{i}}_{\mathcal{D}}$ and $\boldsymbol{\sigma}_{\mathcal{D}}$ are the mean and standard deviation of the input portion of the dataset $\mathcal{D}$. Afterward, we use an initial 80–20 train-val split on the dataset $\mathcal{D}$ with a batch size of 128. We use mean absolute error (MAE) as our loss and use a training strategy that alternates between stochastic gradient descent (SGD) and Adam whenever training stalls. In addition, the batch size is gradually increased up to a max size of 2048, and the entire dataset is used to train the controller once MAE reaches < 0.003. With this scheme, we achieve a final MAE of < 0.0015. The neural network's approximation of $\mathcal{F}(\cdot)$ can be seen visualized in Figure 6.

5. Robotic system

5.1. Perception system

To obtain the Cartesian centerline coordinates of the deployed DLO (or drawn patterns), we use the DLO perception algorithm mBEST (Choi et al., 2023a). This algorithm obtains the centerline coordinates of DLOs within an image by traversing their skeleton pixel representations. The ambiguity of path traversal at intersections is handled by an optimization objective that minimizes the cumulative bending energy of the DLOs during the pixel traversal. One case of extracting discretized patterns from the handwriting pattern is shown in Figure 7. RGB images of the deployed DLO are obtained through an Intel RealSense D435 camera as shown in Figure 9. Further details of the perception algorithm itself can be found in the referenced paper.

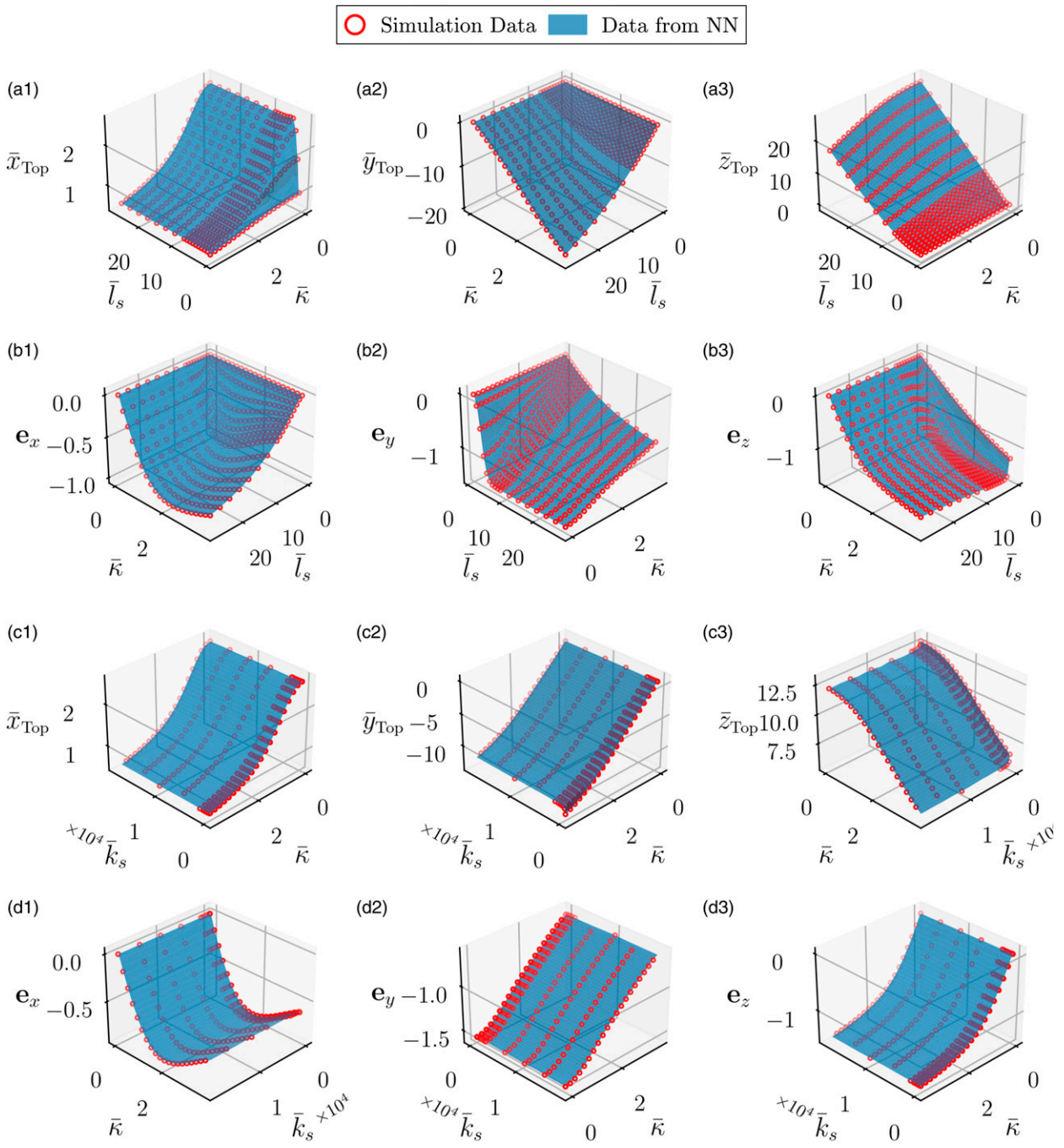

Figure 6. Visualization of the influence from curvature $\bar{\kappa}$ and suspended length $\bar{l}_s$ on the (a1–a3) manipulated end position and (b1–b3) manipulated end orientation for fixed values of $\bar{k}_s = 2087$; visualization of the influence from stretching stiffness $\bar{k}_s$ and curvature $\bar{\kappa}$ on the (c1–c3) manipulated end position and (d1–d3) manipulated end orientation with fixed values of $\bar{l}_s = 13.68$.

5.2. Motion planning with the neural controller

In Figure 1, we showcase the full end-to-end pipeline of our proposed deployment scheme. Here, we give a full description of how to integrate the trained neural controller into a robot motion planner.

The first step of the deployment process is to specify the desired pattern. This pattern can be defined by either an analytical function or detected as a drawn curve as shown in Figure 1. Note that the pattern $\mathbf{P}(s)$ is treated as a function of the curve length s. Based on the configuration of the pattern, we can compute the required inputs for the neural controller when the connective node $\mathbf{q}_C$ achieves each point in the pattern $\mathbf{P}(s)$. The details of generating an optimal trajectory based on the pattern $\mathbf{P}(s)$ and the properties of the manipulated rod are given in Algorithm 4.

Algorithm 4: Optimal Deployment Trajectory

Input: $\mathbf{P}, L, L_{gb}, \bar{k}_s$
Note that material parameters $L_{gb}, \bar{k}_s$ must be measured in advance (Figure 8 and equation (16)).
Output: τ

Func OPT $(L, \mathbf{P}, L_{gb}, \bar{k}_s)$ **:**
  $S, \kappa, \mathbf{T} \leftarrow$ ProcessPattern $(\mathbf{P})$
  $\Delta s \leftarrow$ step size of deployment
  $\tau \leftarrow$ initialize an empty list
  $\hat{\mathbf{z}} \leftarrow$ director along vertical direction
  $s \leftarrow 0$
  while $s \leq S$ **do**
    $\mathbf{q}_C \leftarrow \mathbf{P}(s)$
    $\hat{\mathbf{x}} \leftarrow \mathbf{T}(s)$
    $\bar{\kappa} \leftarrow \kappa(s) L_{gb}$
    $\bar{l}_s \leftarrow (L - s)/L_{gb}$
    $(\bar{\mathbf{q}}^*_M, \mathbf{e}^*) \leftarrow \mathcal{F}(\bar{l}_s, \bar{\kappa}, \bar{k}_s)$
    $\mathbf{R} \leftarrow$ AxangtoRot $(\hat{\mathbf{e}}^*, \|\mathbf{e}^*\|)$
    $\mathbf{R}_t \leftarrow (\hat{\mathbf{x}}, \hat{\mathbf{z}} \times \hat{\mathbf{x}}, \hat{\mathbf{z}})$
    $\mathbf{q}^*_M \leftarrow \mathbf{q}_C + \mathbf{R}_t \bar{\mathbf{q}}^*_M L_{gb}$
    $\mathbf{R}^* \leftarrow \mathbf{R}_t \mathbf{R}$
    Append $(\mathbf{q}^*_M, \mathbf{R}^*)$ to τ
    $s \leftarrow s + \Delta s$
  return τ

In Algorithm 4, κ and $\mathbf{T}$ are all functions of the arc length s of the pattern, where $\mathbf{T}$ is the tangent along the pattern. With Algorithm 4, we obtain the optimal grasp trajectory τ and then use OMPL (Sucan et al., 2012) to generate a valid motion planning sequence on a real robot system.

One highlight of our overall robotic system is its real-time capability. The real-time efficiency of the perception algorithm has been validated by Choi et al. (2023a), while the average end-to-end time to generate a full optimal deployment motion plan is less than 1 s. Therefore, our approach is also efficient enough for sensorimotor closed-loop control. However, as offline control has achieved excellent deployment accuracy in our experiments, online control is not carried out in this work.

6. Experiments and analysis

6.1. Measurement of material parameters

To carry out deployment with our proposed scheme, we must validate its efficacy with comprehensive experiments. In this article, we choose to deploy various DLOs on different substrates for multiple tasks so that we can look into the robustness of the proposed scheme against the material difference and friction.

First, we need to find the geometric and material properties of the manipulated DLO. The geometry of the manipulated rod, for example, total length L and rod radius h, is trivial to measure. Measuring the material properties of the DLO is less clear. Overall, we need to develop a way to find the following material properties: gravito-bending length L_{gb} and normalized stretching stiffness $\bar{k}_s$.

Here, we presume the material is linearly elastic and incompressible. The incompressible material means the volume of the rod will not change when deformations happen. Therefore, Poisson's ratio is set as $\nu = 0.5$. In addition, bending stiffness is $k_b = E\pi h^4/4$, where E is

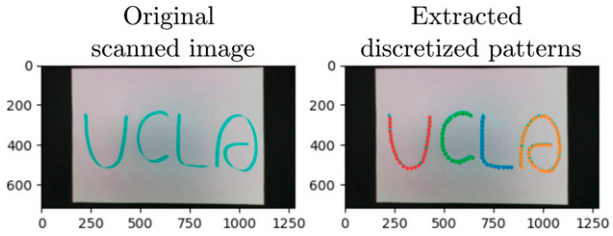

Figure 7. Handwritten letters and the corresponding extracted discretized patterns using mBEST Choi et al. (2023a).

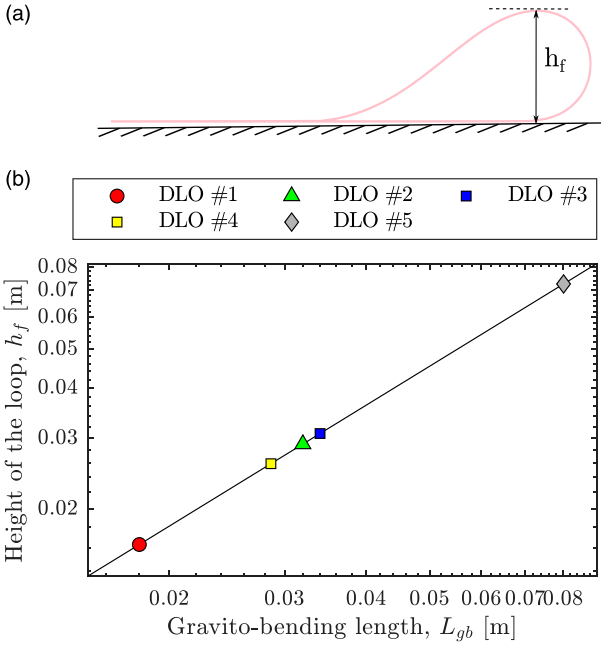

Figure 8. (a) Deformed configurations of a DLO under gravity in 2D plane; and (b) relationship between the height of the loop h_f and the gravito-bending length L_{gb}.

Young's modulus, and the expression for gravito-bending length L_{gb} and normalized stretching stiffness $\bar{k}_s$ is

$$L_{gb} = \left(\frac{Eh^2}{8\rho g}\right)^{1/3}, \qquad \bar{k}_s = \frac{k_s L_{gb}^2}{k_b} = \frac{4L_{gb}^2}{h^2}. \tag{16}$$

When observing equation (16), we find that the only parameter we must obtain is L_{gb}. It is still unclear how to compute this as L_{gb} is relevant to Young's Modulus E and the density ρ of the rod, which is usually hard to measure. Here, we propose a simple method that is able to measure L_{gb} by observing the geometry of the rod. When we form a loop in a rod naturally using gravity in a 2D plane, we can observe the geometry of the rod becomes what is shown in Figure 8(a). Indeed, the height h_f of the loop has a linear relationship with L_{gb}. Therefore, we can obtain L_{gb} for different rods by simply measuring h_f. According to prior

work (Pan et al., 2020) and our validation shown in Figure 8(b), $h_f = 0.9066 L_{gb}$.

6.2. Experiment setup

6.2.1. Materials and robot hardware. In this article, we conducted experiments involving five distinct types of DLOs. Among these, three are silicone-based rubber fabricated by vinyl polysiloxane (VPS); the fourth is a commercially available rope; and the fifth is a stiff USB cable. Note that we also validate the robustness of the deployment scheme against different substrates. The friction between the DLOs and substrates is also qualitatively measured. Comprehensive details regarding the parameters for each of these DLOs can be found in Table 1.

For our experiments, we used two Rethink Robotics' Sawyer manipulators as shown in Figure 9. One arm is attached with a gripper for manipulating the rod. The other arm holds an Intel RealSense D435 camera which is used to scan drawn patterns as well as obtain a top-down view of the deployment result for evaluations. A workstation with an AMD Ryzen 7 3700X CPU and an NVIDIA RTX 2070 SUPER GPU was used for all experiments.

6.2.2. Experiment tasks. We implement our proposed deployment scheme across four distinct tasks. First, we deploy a rod along some canonical cases obtainable through analytical expressions such as a line, circle, and sine curve. The rod is deployed using the robotic arm with the gripper. Once the deployment is finished, the other arm with the camera moves to scan the deployment result.

The second task involves deploying patterns drawn on paper. Users draw patterns, subsequently scanned by the camera to obtain ordered discretized pattern coordinates. The robot then manipulates the rod to replicate the drawn pattern. This article showcases deployment results for the letters "U," "C," "L," and "A" with the precise shapes detailed in Figure 10(a). The third task is geared towards validating the deployment scheme's application in cable placement, a vital aspect of cable management. The scheme's efficacy is demonstrated by placing cables along constrained paths with the help of pre-installed fixtures on the substrate. Lastly, the deployment scheme's application for tying knots is verified. Both robotic arms are equipped with grippers for this task.

For the first two tasks, patterns are evaluated using both intuitive and optimal control methods. Additionally, three different rods (DLOs #1, #2, and #3) are deployed on substrates of various materials (fabric, steel, and foam) to assess the method's robustness against material disparities and friction. In the third task, DLO #5 (USB cable) is employed for cable placement using both algorithms. Finally, DLOs #2 and #4 are used to tie distinct knots for the fourth task. Each experimental case is subjected to 10 trials for each control method, culminating in a total of 1340 experimental trials.

6.3. Metrics

We now formulate the metrics used to evaluate the performance of the deployment scheme. When deploying a pattern $\mathbf{P}$, we need to assess the accuracy of the deployment result. We first discretize the pattern $\mathbf{P}$ into N points and denote the i-th point of the prescribed pattern as $\mathbf{P}^i$. The actual deployment pattern obtained from perception is denoted as $\mathbf{P}_{\text{exp}}$. With this discretization scheme, we compute the average error e_{mean} and standard deviation σ as

$$e_{\text{mean}} = \frac{1}{N}\sum_{i-1}^{N}\left\|\mathbf{P}^i_{\text{exp}} - \mathbf{P}^i\right\|,$$
$$\sigma = \sqrt{\frac{\sum_{i=1}^{N}\left(\left\|\mathbf{P}^i_{\text{exp}} - \mathbf{P}^i\right\| - e_{\text{mean}}\right)}{N}}, \tag{17}$$

for both the intuitive and optimal control results.

The accuracy evaluation is not applicable for the two application tasks: cable placement and knot tying as they are high-level tasks. Therefore, we simply use the success rate of those application tasks to evaluate the performance of the deployment scheme. In addition to accuracy, we also report a detailed comparison of runtimes and errors between the numerical and NN-based solvers. Details of the relevant results and analysis are discussed in the next section.

6.4. Results and analysis

6.4.1. Accuracy. All experimental results can be seen in Table 2. To compute the error metrics in equation (17), we used a discretization of $N = 50$. From all results, we can observe a noticeable improvement in our optimal control method over the intuitive method for various geometrical, material, and environmental parameters.

Table 1. Material and geometric properties of the DLOs used in the experiments.

	Material and geometric parameters							
DLO	Material	L_{gb} (cm)	h (mm)	L (m)	v	μ_{fabric}	μ_{steel}	μ_{foam}
#1	Pink VPS	1.8	1.6	0.875	0.5	Low	Medium	High
#2	Green VPS	3.2	1.6	0.885	0.5	Low	Medium	High
#3	Rope	3.4	2.0	0.89	0.5	Medium	Low	High
#4	Pink VPS	2.86	3.2	0.84	0.5	Low	Medium	High
#5	Cable	8.01	1.8	0.87	0.5	Medium	Low	High

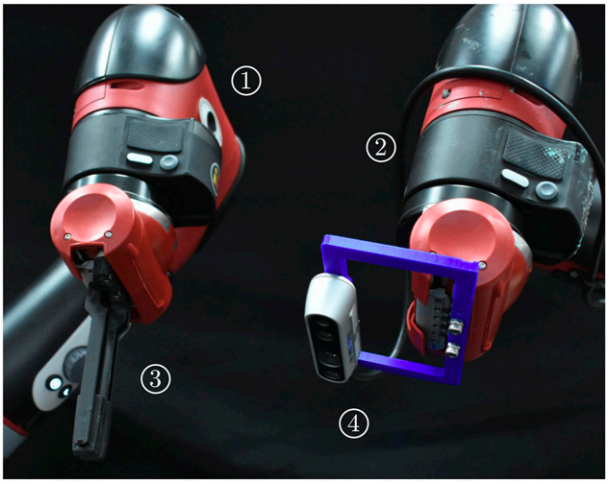

Figure 9. Experimental apparatus: Two robot manipulators, one for manipulation of the deploying rod (1) and the other for holding the camera for perception (2). A gripper (3) is used for grabbing the manipulated end of the rod. A camera (4) is used for extracting patterns from the drawn patterns and evaluating the deployment results.

To better visualize our method's generality, we visually depict deployment outcomes across different DLOs on the fabric surface in Figure 10. In addition, a comparative visual representation of deployment results for a single DLO (#2) on varying substrates is shown in Figure 11. Readers seeking comprehensive visual comparisons of all deployment outcomes can refer to the supplementary video for detailed insights (see Footnote [1]).

Among the seven deployed patterns, the first three (straight line, circle, and sine curve) are canonical cases, that is, their shapes have explicit analytical expressions. Note that when deploying the circle and sine curve patterns, a small "remainder" section is first deployed. This is necessary as the circle and sine curve patterns have a non-zero curvature at the start of their pattern. We compensate for this by deploying a remainder part whose curvature gradually evolves from a straight line with 0 curvature to the prescribed curvature of the pattern's first point. The remainder can improve the deployment task's accuracy as the deployed pattern will require slight friction based on equation (8).

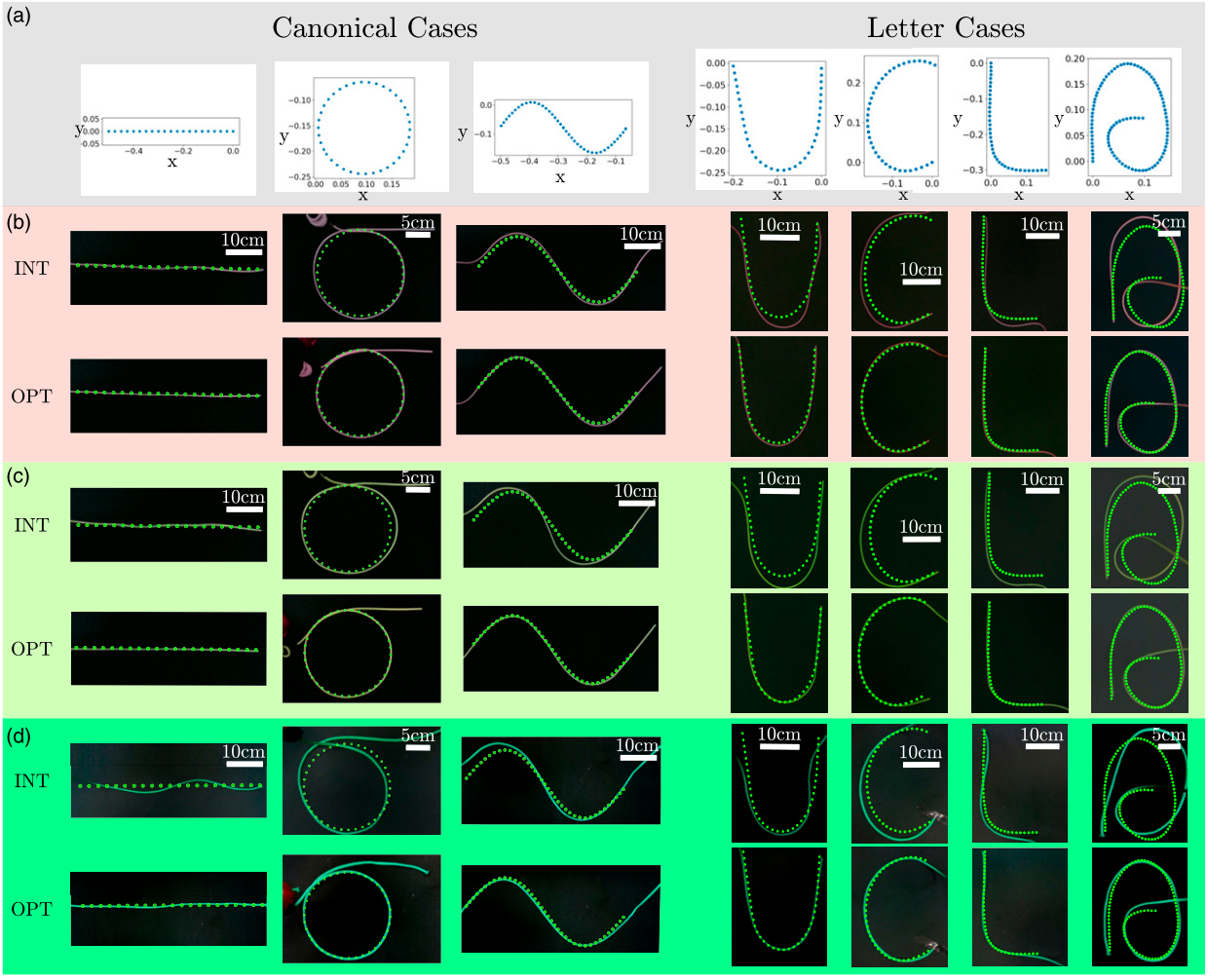

Figure 10. Experiment results of deployment along various patterns. (a) All used prescribed patterns are discretized and plotted. Deployment results for (b) DLO #1 (pink VPS), (c) DLO #2 (green VPS), and (d) DLO #3 (rope) are shown for each prescribed pattern. Results for the intuitive control method and optimal control method are shown for each rod.

Table 2. Evaluation of deployment accuracy for various patterns, DLOs, and substrates.

			Pattern type and accuracy $e_{\text{mean}} \pm \sigma$ (cm) (equation (17))						
DLO	SUB	Control scheme	Line	Circle	Sine Curve	Letter "U"	Letter "C"	Letter "L"	Letter "A"
#1	Fabric	INT	0.40 ± 0.22	0.61 ± 0.36	1.66 ± 0.74	1.39 ± 0.63	2.21 ± 0.92	1.00 ± 0.59	4.81 ± 2.27
		OPT	0.14 ± 0.09	0.15 ± 0.07	0.27 ± 0.10	0.22 ± 0.07	0.22 ± 0.10	0.35 ± 0.18	0.47 ± 0.23
	Steel	INT	1.42 ± 0.66	2.34 ± 1.24	2.69 ± 1.69	3.59 ± 2.39	3.67 ± 1.93	0.87 ± 0.55	3.64 ± 2.09
		OPT	0.22 ± 0.12	0.22 ± 0.08	0.27 ± 0.10	0.24 ± 0.13	0.27 ± 0.09	0.42 ± 0.16	0.58 ± 0.37
	Foam	INT	1.03 ± 0.21	1.23 ± 0.45	2.84 ± 1.52	3.33 ± 1.93	3.89 ± 1.29	1.13 ± 0.74	4.09 ± 2.19
		OPT	0.25 ± 0.15	0.18 ± 0.06	0.29 ± 0.16	0.24 ± 0.15	0.41 ± 0.20	0.35 ± 0.12	0.54 ± 0.24
#2	Fabric	INT	0.52 ± 0.13	1.64 ± 0.95	1.60 ± 0.83	3.74 ± 2.89	4.58 ± 1.15	1.74 ± 1.11	4.95 ± 2.63
		OPT	0.13 ± 0.07	0.16 ± 0.07	0.20 ± 0.09	0.17 ± 0.11	0.19 ± 0.22	0.29 ± 0.11	0.32 ± 0.18
	Steel	INT	1.72 ± 0.63	2.52 ± 1.02	3.30 ± 2.08	4.78 ± 4.15	6.66 ± 2.53	2.14 ± 1.26	5.23 ± 3.38
		OPT	0.17 ± 0.08	0.22 ± 0.09	0.54 ± 0.20	0.21 ± 0.09	0.74 ± 0.31	0.66 ± 0.24	0.36 ± 0.17
	Foam	INT	1.38 ± 0.60	2.24 ± 0.97	4.17 ± 2.57	5.42 ± 4.47	6.14 ± 3.08	1.70 ± 1.32	5.09 ± 3.39
		OPT	0.27 ± 0.13	0.20 ± 0.09	0.37 ± 0.14	0.17 ± 0.08	0.39 ± 0.18	0.37 ± 0.15	0.43 ± 0.19
#3	Fabric	INT	1.56 ± 0.81	1.13 ± 0.53	5.09 ± 1.35	4.22 ± 3.10	3.36 ± 1.58	2.37 ± 1.56	4.59 ± 2.54
		OPT	0.49 ± 0.28	0.29 ± 0.15	0.47 ± 0.23	0.36 ± 0.18	0.35 ± 0.19	0.50 ± 0.24	0.56 ± 0.29
	Steel	INT	4.53 ± 2.80	1.85 ± 0.45	4.43 ± 2.82	4.53 ± 2.80	3.35 ± 1.55	2.57 ± 1.62	4.30 ± 1.73
		OPT	0.47 ± 0.20	0.29 ± 0.13	0.46 ± 0.20	0.47 ± 0.20	0.56 ± 0.20	0.51 ± 0.24	0.81 ± 0.30
	Foam	INT	2.00 ± 0.88	1.94 ± 0.84	3.80 ± 1.96	3.67 ± 2.46	6.03 ± 3.11	3.32 ± 1.80	4.47 ± 2.50
		OPT	0.78 ± 0.34	0.27 ± 0.15	0.46 ± 0.20	0.32 ± 0.16	0.56 ± 0.26	0.33 ± 0.14	0.52 ± 0.20

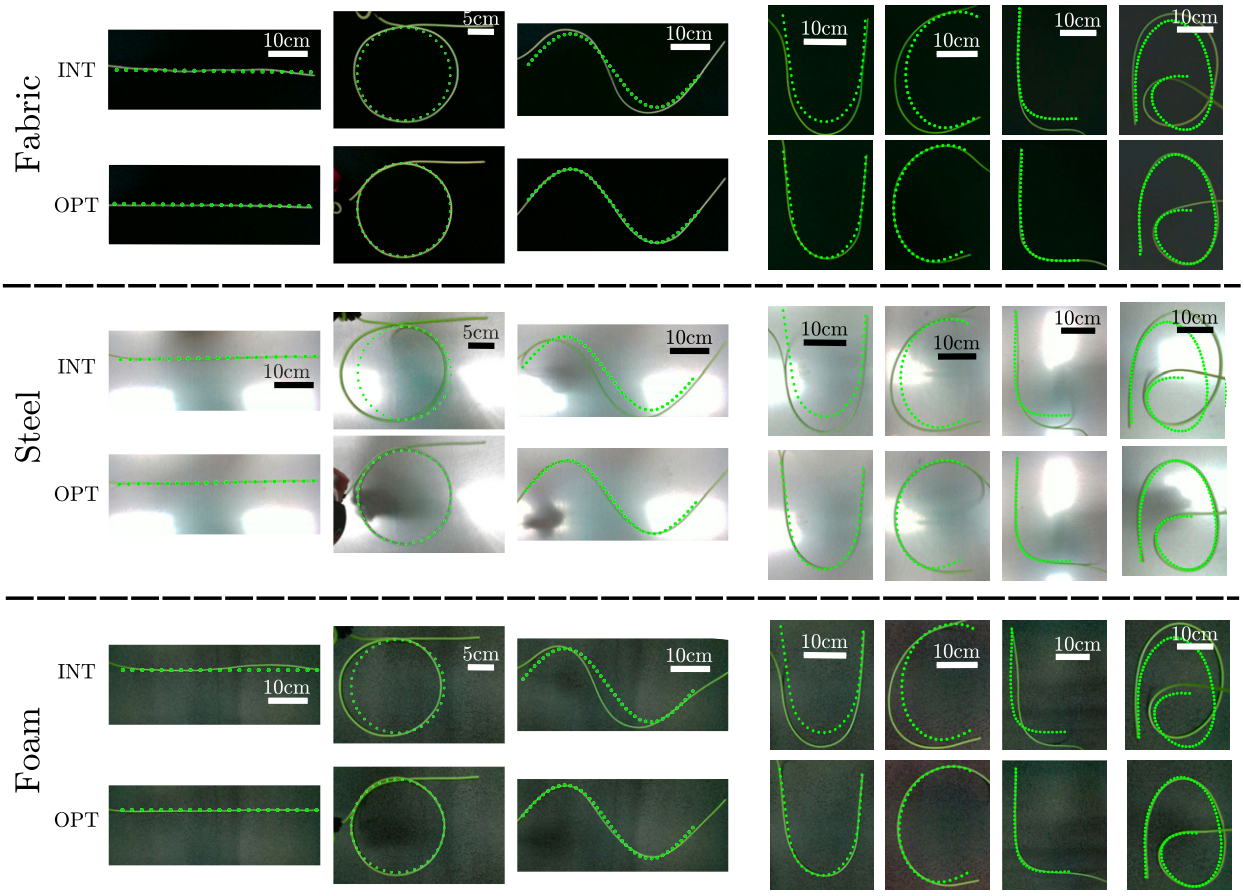

Figure 11. Experiment results of deployment with DLO #2 (green VPS) along various patterns on different substrates.

We have omitted the designed remainder for the four remaining patterns denoted by the letters "U," "C," "L," and "A" for better visualization. Among these, patterns "U," "L," and "A" exhibit a relatively low κ'' value during the beginning stage of the deployment, resulting in the deployment accuracy being minimally affected by surface friction.

Conversely, the "C" pattern demonstrates a comparatively higher κ'' value initially, leading to a possible noticeable mismatch between the deployed DLO and the intended pattern in the beginning. The impact of friction becomes more pronounced during the rope deployment corresponding to DLO #3 since the rope has higher bending stiffness k_b and experiences lower friction with the substrate. Fixing the free end is essential to precisely replicate the "C" pattern with the rope as shown in Figure 10(d). Despite this limitation, our optimized deployment strategy consistently outperforms the intuitive approach.

6.4.2. Computational efficiency. Next, we also evaluated the computational efficiency of our neural controller. Table 3 compares time costs between the neural network solver (NN-solver) and the numeric solver based on simulations. When calculating a single optimal robot grasp for a given parameter tuple ($\bar{l}_s$, $\bar{\kappa}$, $\bar{k}_s$), the numeric solver takes approximately 10 to 20 s, while our NN-solver takes roughly 0.4 s.

The difference of time costs becomes more significant when generating a series of optimal robot grasps for a discretized pattern. Note that a discretized pattern typically consists of 100 to 200 nodes and that the numeric solver needs to compute the robot trajectory in sequence as the optimal grasp for the previous step is needed as the seed for computing the next optimal grasp. Therefore, the time costs quickly accumulate for the numeric solver, which substantially elongates the overall computation time. In contrast, the NN-solver leverages vectorization to solve multiple robot grasps simultaneously, resulting in a speed advantage of several orders of magnitude compared to the numeric solver when generating optimal deployment trajectories.

6.4.3. Precision of the neural controller. Finally, Table 3 also presents the precision of the NN-solver. The solutions from the numeric solver serve as the ground truth. Mean Absolute Error (MAE) is employed to evaluate the optimal trajectories the NN-solver generates against the ground truth. Remarkably, the MAE consistently remains below 0.003 m for position error and 0.009 for differences in rotation quaternions. Importantly, it's noteworthy that none of the solved trajectories in this analysis were part of the training dataset. Thus, we can confidently assert that our NN-solver exhibits robustness, efficiency, and accuracy, rendering it well-suited for real-time control applications.

6.5. Application #1: Cable placement

In this section, we showcase the application of the deployment scheme for cable placement. The importance of cable management has surged, particularly in engineering contexts involving tasks like wire harnessing, infrastructure development, and office organization (Sanchez et al., 2018; Lattanzi and Miller, 2017). Given cables' inherent high bending stiffness, shaping them to specific forms can be challenging, often necessitating external fixtures to maintain the desired configuration. When humans perform cable management manually, meticulous placement along the designated pattern is essential, coupled with the use of fixtures to secure the cable in place. However, a robotic system can autonomously execute cable placement with our designed optimal deployment strategy.

In our experimental setup, we pre-installed external fixtures into the stainless steel breadboard to delineate the intended patterns. These fixtures also counteract the cable's rigid nature, preventing it from reverting to its original shape. The deployment results can be visualized in

Table 3. Evaluation of computation times of various patterns for the numerical and NN-solvers with error metrics.

		Patterns with number of nodes						
		Line	Circle	Sine curve	Letter "U"	Letter "C"	Letter "L"	Letter "A"
DLO	Solver times (s) and MAEs	101 nodes	156 nodes	138 nodes	190 nodes	190 nodes	190 nodes	194 nodes
#1	Numeric solver	1572.68	2036.11	2897.17	3954.12	4015.24	4777.30	4666.55
	NN-solver	0.402	0.393	0.395	0.431	0.431	0.400	0.417
	Position error (m)	0.0008	0.0007	0.0009	0.0008	0.0007	0.0008	0.0008
	Orientation error	0.0012	0.0010	0.0032	0.0025	0.0020	0.0020	0.0021
#2	Numeric solver	776.56	1213.14	1769.66	2286.66	2226.73	2720.08	2933.90
	NN-solver	0.397	0.391	0.396	0.419	0.408	0.404	0.406
	Position error (m)	0.0016	0.0016	0.0019	0.0018	0.0016	0.0020	0.0017
	Orientation error	0.0012	0.0078	0.0050	0.0042	0.0020	0.0058	0.0030
#3	Numericsolver	666.01	1041.71	1561.12	1984.63	1972.39	2405.71	2639.44
	NN-solver	0.400	0.407	0.395	0.407	0.420	0.405	0.411
	Position error (m)	0.0016	0.0017	0.0020	0.0020	0.0016	0.0021	0.0018
	Orientation error	0.0010	0.0087	0.0052	0.0055	0.0023	0.0054	0.0032

Figure 12. Compared to the failure placement results with the intuitive scheme, our optimal deployment scheme can place the cable along the prescribed pattern "U" and "S" on the substrate. We did 10 experimental trials for each deployment task illustrated in Figure 12. Notably, the optimal deployment approach achieved an impressive 90% (9/10) success rate for both patterns, whereas the intuitive method failed in all trials (0/10) as shown in Table 4.

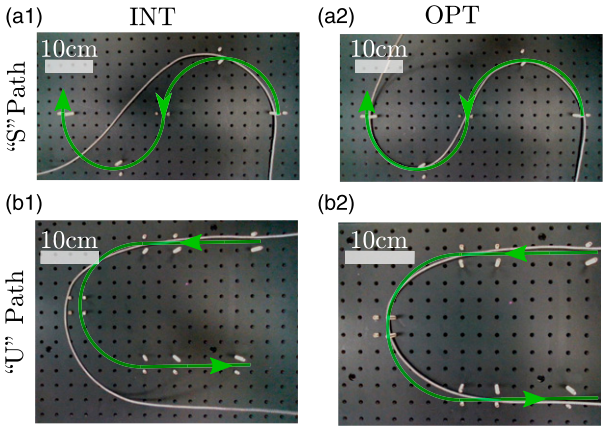

Figure 12. A demonstration of cable placement along different prescribed patterns with both intuitive and optimal control schemes.

Table 4. Real-world application experiment results.

Experiment type	Scheme	Success rate
"S" cable placement	INT	0/10
	OPT	9/10
"U" cable placement	INT	0/10
	OPT	9/10
Trefoil knot	INT	0/10
	OPT	9/10
Reef knot	INT	0/10
	OPT	7/10

6.6. Application #2: Knot tying

Since our optimal deployment scheme can control the shape of various DLOs with excellent accuracy, we can use this scheme to tie knots. First, the manipulated rod is deployed along a predesigned pattern on the substrate. Users can draw the predesigned pattern so that only a few extra manipulations are required. Then, the camera will scan the drawn pattern and send it as input to our designed scheme. The deployed pattern is designed in a way that only a few simple pick-and-place operations on certain knot segments is required to complete the tying sequence. Since the prescribed pattern's shape is known in advance, we can let the robot execute the pick-and-place procedure without perception feedback. So long as the initial deployment is accurate and repeatable, the subsequent pick-and-place procedure should succeed most of the time.

We showcase two knot-tying sequences in Figure 13. The top row showcases a trefoil knot, one of the most fundamental knots in engineering (Crowell and Fox, 2012). For this knot, we used DLO #4. Another case is a reef knot, a prevalent knot widely used in for various applications including shoelaces, packaging, sewing, etc. When tying the reef knot, we used DLOs #2 and #4. Although these two DLOs have totally different material properties, our generalizable neural controller allows two robots to deploy both DLOs accurately along the designed patterns. With the help of the deployed patterns, reef knots can be tied with simple pick-and-place procedures. Such knot-tying cases strongly support the potential of our deployment scheme in various engineering applications.

We show the results of the knot-tying tasks in Table 4. The successful rate of knot tying is remarkable. We achieve a success rate of 90% (9 successful trials out of 10) for tying a trefoil knot and a success rate of 70% (7 successful trials out of 10) with the optimal control method. Based on our observations, all the failure cases were caused by the rod slipping out of the gripper. In contrast, the intuitive control method achieves a success rate of 0% for both cases as the initially deployed pattern does not match the intended pattern.

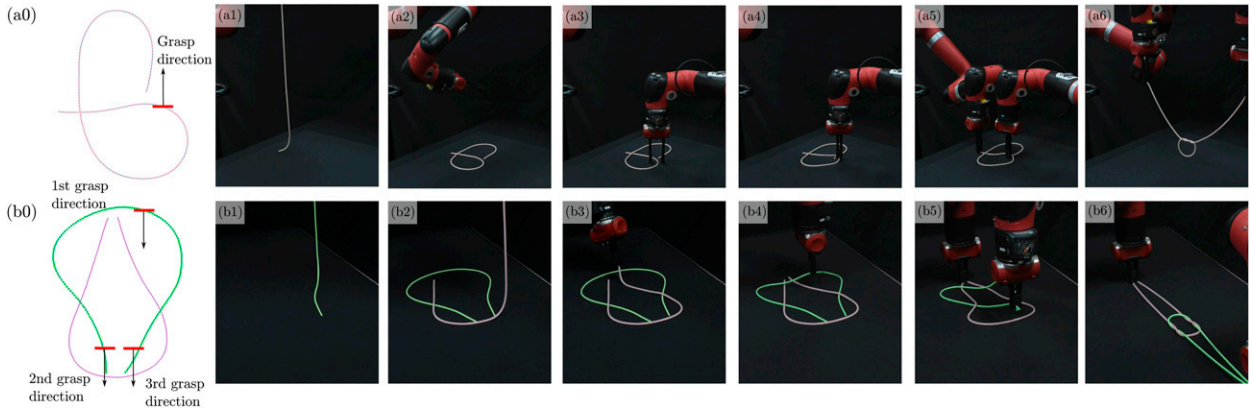

Figure 13. A demonstration of two knot-tying cases using the DLO deployment scheme. (a0) and (b0) are designed patterns for the trefoil knot and the reef knot, respectively. Time marches for trefoil knot from (a1) to (a6) and reef knot from (b1) to (b6).

Therefore, the intuitive control method would require some visual feedback to choose the pick-and-place motion adaptively for the trefoil knot case. As for the reef knot case, due to the deployment results being totally wrong, even though the visual feedback is applied, it is still hard to achieve a complete reef knot with intuitive method.

Therein, we can see the potential of the deployment scheme in high-level robotic tasks like knot tying. In future work, the optimal deployment scheme will be incorporated with the perception system to automatically tie any prescribed knots with the robotics system.

7. Conclusion

In this article, we have introduced a novel deployment scheme that allows for robust and accurate control of the shape of DLOs using a single manipulator. Our framework integrates techniques from various disciplines, including physical simulation, machine learning, and scaling analysis, and has been demonstrated to be highly effective in real robotic experiments. Our results highlight the advantages of incorporating physics into robotic manipulation schemes and showcase impressive performance on complex tasks such as writing letters with elastic rods, cable placement, and tying knots.

Looking to the future, we plan to leverage the precision and efficiency of our deployment scheme to tackle some high-level robotic tasks systematically, for example, robotic knot tying. While exact shape control is not strictly required during such manipulations, our deployment scheme offers sufficient accuracy and efficiency to design the configurations of the middle states of a manipulated DLO, which is essential for robots to successfully tie complex knots. We also aim to explore the use of generalized problem formulations and data-driven control schemes, such as reinforcement learning, to develop more flexible and adaptive solutions to the challenges of robotic manipulation. By continuing to push the boundaries of robotic manipulation, we hope to advance the state-of-the-art in this field and enable new and exciting applications of robotic technology.

Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This research was funded in part by the National Science Foundation under award numbers OAC-2209782, CMMI-2101751, CAREER-2047663, and IIS-1925360.

ORCID iD

Mohammad Khalid Jawed https://orcid.org/0000-0003-4661-1408

Supplemental Material

Supplemental material for this article is available online.

Note

1. See https://github.com/StructuresComp/rod-deployment.

References

Bergou M, Wardetzky M, Robinson S, et al. (2008) Discrete elastic rods. In: *ACM SIGGRAPH 2008 Papers, SIGGRAPH '08*. New York, NY, USA: Association for Computing Machinery, 1–12. DOI: 10.1145/1399504.1360662.

Bergou M, Audoly B, Vouga E, et al. (2010) Discrete viscous threads. *ACM Transactions on Graphics* 29(4): 1–10.

Buckham B, Driscoll FR and Nahon M (2004) Development of a finite element cable model for use in low-tension dynamics simulation. *Journal of Applied Mechanics* 71(4): 476–485.

Choi A, Tong D, Jawed MK, et al. (2021) Implicit contact model for discrete elastic rods in knot tying. *Journal of Applied Mechanics* 88(5): 1–13.

Choi A, Tong D, Park B, et al. (2023a) mbest: realtime deformable linear object detection through minimal bending energy skeleton pixel traversals. *IEEE Robotics and Automation Letters* 8(8): 4863–4870.

Choi A, Tong D, Terzopoulos D, et al. (2023b) *Deep Learning of Force Manifolds from the Simulated Physics of Robotic Paper Folding*. Ithaca, NY: arXiv.

Clegg A, Yu W, Tan J, et al. (2018) Learning to dress: synthesizing human dressing motion via deep reinforcement learning. *ACM Transactions on Graphics* 37(6): 1–10.

Crowell RH and Fox RH (2012) *Introduction to Knot Theory*. Berlin, Germany: Springer Science & Business Media, vol. 57.

Erickson Z, Clever HM, Turk G, et al. (2018) Deep haptic model predictive control for robot-assisted dressing. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. New York, NY: IEEE, 4437–4444.

Geblinger N, Ismach A and Joselevich E (2008) Self-organized nanotube serpentines. *Nature Nanotechnology* 3(4): 195–200.

Guler P, Pauwels K, Pieropan A, et al. (2015) Estimating the deformability of elastic materials using optical flow and position-based dynamics. In: *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. New York, NY: IEEE, 965–971.

Haouchine N, Kuang W, Cotin S, et al. (2018) Vision-based force feedback estimation for robot-assisted surgery using instrument-constrained biomechanical three-dimensional maps. *IEEE Robotics and Automation Letters* 3(3): 2160–2165.

Jawed MK, Da F, Joo J, et al. (2014) Coiling of elastic rods on rigid substrates. *Proceedings of the National Academy of Sciences* 111(41): 14663–14668.

Jawed MK, Khouri NK, Da F, et al. (2015) Propulsion and instability of a flexible helical rod rotating in a viscous fluid. *Physical Review Letters* 115(16): 168101.

Kapusta A, Erickson Z, Clever HM, et al. (2019) Personalized collaborative plans for robot-assisted dressing via optimization and simulation. *Autonomous Robots* 43(8): 2183–2207.

Kaufmann P, Martin S, Botsch M, et al. (2009) Flexible simulation of deformable models using discontinuous galerkin fem. *Graphical Models* 71(4): 153–167.

Kita Y, Kanehiro F, Ueshiba T, et al. (2011) Clothes handling based on recognition by strategic observation. In: *2011 11th IEEE-RAS International Conference on Humanoid Robots*. New York, NY: IEEE, 53–58.

Lattanzi D and Miller G (2017) Review of robotic infrastructure inspection systems. *Journal of Infrastructure Systems* 23(3): 04017004.

Lee AX, Huang SH, Hadfield-Menell D, et al. (2014) Unifying scene registration and trajectory optimization for learning from demonstrations with application to manipulation of deformable objects. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. New York, NY: IEEE, 4402–4407.

Lee R, Hamaya M, Murooka T, et al. (2021) Sample-efficient learning of deformable linear object manipulation in the real world through self-supervision. *IEEE Robotics and Automation Letters* 7(1): 573–580.

Lv N, Liu J and Jia Y (2022) Dynamic modeling and control of deformable linear objects for single-arm and dual-arm robot manipulations. In: *IEEE Transactions on Robotics*. New York, NY: IEEE.

Macklin M, Müller M, Chentanez N, et al. (2014) Unified particle physics for real-time applications. *ACM Transactions on Graphics* 33(4): 1–12.

Macklin M, Müller M and Chentanez N (2016) Xpbd: position-based simulation of compliant constrained dynamics. In: *Proceedings of the 9th International Conference on Motion in Games*. New York, NY: Association for Computing Machinery, 49–54.

Maitin-Shepard J, Cusumano-Towner M, Lei J, et al. (2010) Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding. In: *2010 IEEE International Conference on Robotics and Automation*. New York, NY: IEEE, 2308–2315.

Matas J, James S and Davison AJ (2018) Sim-to-real reinforcement learning for deformable object manipulation. *Conference on Robot Learning*. Zurich, Switzerland: PMLR, 734–743.

McConachie D, Dobson A, Ruan M, et al. (2020) Manipulating deformable objects by interleaving prediction, planning, and control. *The International Journal of Robotics Research* 39(8): 957–982.

Miller S, Van Den Berg J, Fritz M, et al. (2012) A geometric approach to robotic laundry folding. *The International Journal of Robotics Research* 31(2): 249–267.

Mitrano P, McConachie D and Berenson D (2021) Learning where to trust unreliable models in an unstructured world for deformable object manipulation. *Science Robotics* 6(54): eabd8170.

Müller M, Heidelberger B, Hennix M, et al. (2007) Position based dynamics. *Journal of Visual Communication and Image Representation* 18(2): 109–118.

Nair A, Chen D, Agrawal P, et al. (2017) Combining self-supervised learning and imitation for vision-based rope manipulation. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. New York, NY: IEEE, 2146–2153.

Pan K, Phani AS and Green S (2020) Periodic folding of a falling viscoelastic sheet. *Physical Review E* 101(1): 013002.

Pignat E and Calinon S (2017) Learning adaptive dressing assistance from human demonstration. *Robotics and Autonomous Systems* 93: 61–75.

Saha M and Isto P (2007) Manipulation planning for deformable linear objects. *IEEE Transactions on Robotics* 23(6): 1141–1150.

Sanchez J, Corrales JA, Bouzgarrou BC, et al. (2018) Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey. *The International Journal of Robotics Research* 37(7): 688–716.

Schulman J, Gupta A, Venkatesan S, et al. (2013) A case study of trajectory transfer through non-rigid registration for a simplified suturing scenario. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. New York, NY: IEEE, 4111–4117.

Sen S, Garg A, Gealy DV, et al. (2016) Automating multi-throw multilateral surgical suturing with a mechanical needle guide and sequential convex optimization. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. New York, NY: IEEE, 4178–4185.

Servin M and Lacoursiere C (2008) Rigid body cable for virtual environments. *IEEE Transactions on Visualization and Computer Graphics* 14(4): 783–796.

She Y, Wang S, Dong S, et al. (2021) Cable manipulation with a tactile-reactive gripper. *The International Journal of Robotics Research* 40(12–14): 1385–1401.

Stefanidis D, Wang F, Korndorffer JR, et al. (2010) Robotic assistance improves intracorporeal suturing performance and safety in the operating room while decreasing operator workload. *Surgical Endoscopy* 24(2): 377–382.

Sucan IA, Moll M and Kavraki LE (2012) The open motion planning library. *IEEE Robotics and Automation Magazine* 19(4): 72–82.

Sun J, Peng Z, Zhou W, et al. (2015) A review on 3d printing for customized food fabrication. *Procedia Manufacturing* 1: 308–319.

Sundaresan P, Grannen J, Thananjeyan B, et al. (2020) Learning rope manipulation policies using dense object descriptors trained on synthetic depth data. *2020 IEEE International Conference on Robotics and Automation (ICRA)*. New York, NY: IEEE, 9411–9418.

Takizawa M, Kudoh S and Suehiro T (2015) Method for placing a rope in a target shape and its application to a clove hitch. In: *2015 24th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*. New York, NY: IEEE, 646–651.

Tang T, Wang C and Tomizuka M (2018) A framework for manipulating deformable linear objects by coherent point drift. *IEEE Robotics and Automation Letters* 3(4): 3426–3433.

Teo WE and Ramakrishna S (2006) A review on electrospinning design and nanofibre assemblies. *Nanotechnology* 17(14): R89.

Terzopoulos D and Qin H (1994) Dynamic nurbs with geometric constraints for interactive sculpting. *ACM Transactions on Graphics* 13(2): 103–136.

Timoshenko SP and Gere JM (2009) *Theory of Elastic Stability*. Chelmsford, MA: Courier Corporation.

Tong D, Borum A and Jawed MK (2021) Automated stability testing of elastic rods with helical centerlines using a robotic system. *IEEE Robotics and Automation Letters* 7(2): 1126–1133.

Tong D, Choi A, Joo J, et al. (2023a) Snap buckling in overhand knots. *Journal of Applied Mechanics* 90(4): 041008.

Tong D, Choi A, Joo J, et al. (2023b) A fully implicit method for robust frictional contact handling in elastic rods. *Extreme Mechanics Letters* 58: 101924.

Wakamatsu H, Arai E and Hirai S (2006) Knotting/unknotting manipulation of deformable linear objects. *The International Journal of Robotics Research* 25(4): 371–395.

Wang A, Kurutach T, Liu K, et al. (2019) *Learning Robotic Manipulation Through Visual Planning and Acting*. Ithaca, NY: arXiv preprint arXiv:1905.04411.

Whitcomb LL (2000) Underwater robotics: out of the research laboratory and into the field. In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*. New York, NY: IEEE, vol. 1, 709–716.

Yan M, Zhu Y, Jin N, et al. (2020) Self-supervised learning of state estimation for manipulating deformable linear objects. *IEEE Robotics and Automation Letters* 5(2): 2372–2379.

Yin H, Varava A and Kragic D (2021) Modeling, learning, perception, and control methods for deformable object manipulation. *Science Robotics* 6(54): eabd8803.

Yu W, Kapusta A, Tan J, et al. (2017) Haptic simulation for robot-assisted dressing. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. New York, NY: IEEE, 6044–6051.

Yu M, Lv K, Zhong H, et al. (2022) Global model learning for large deformation control of elastic deformable linear objects: an efficient and adaptive approach. In: *IEEE Transactions on Robotics*. New York, NY: IEEE.

Zhu J, Navarro B, Passama R, et al. (2019) Robotic manipulation planning for shaping deformable linear objects with-environmental contacts. *IEEE Robotics and Automation Letters* 5(1): 16–23.