IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Time-series Data-driven Three Dimensional Shape Control of Deformable Linear Objects using a Dual-arm Robot with Dynamic Model Updating

Jiyoung Choi, Micheale Haileslassie Gebrezgiher, *Student Member, IEEE*, Donggun Lee, *Member, IEEE*, and Ayoung Hong, *Member, IEEE*

***Abstract*—Deformable objects (DOs) are prevalent in everyday environments and represent important targets for robotic manipulation. However, their high degrees of freedom and complex nonlinear deformations make them more challenging to model and control than rigid objects when relying on traditional analytical approaches. To address this, we propose a data-driven method to model the dynamics of deformable objects. Our method utilizes time-series data to predict future states without relying on complex dynamics. We employ model predictive control (MPC) for robot manipulation and improve its performance through online updates of the data-driven model. To handle cables with varying configurations, interpolation is applied to align model input structures. In this study, we focus on manipulating deformable linear objects (DLOs) with different mechanical properties and configurations using a dual-arm robotic system, both in simulation and in real-world environments.**

***Index Terms*—Dual Arm Manipulation, AI-based Methods, Optimization and Optimal Control**

I. Introduction

DEFORMABLE linear objects (DLOs) are characterized by having one dimension significantly longer than the other two, such as cables, ropes, and wires. Effective manipulation of DLOs offers significant advantages in robotics. For example, cable handling is essential in assembly manufacturing, while sutures are critical for closing wounds in surgical procedures. However, the shapes or structures of these objects change easily in response to external forces, making them difficult to control using robots.

Unlike rigid bodies, which typically have six degrees of freedom (DoF) and exhibit minimal deformation under external forces, DLOs possess significantly more DoFs, resulting in greater uncertainty in deformation and complex, nonlinear dynamics. As a result, effective manipulation of DLOs requires predicting their future states, as their shape evolves in response to applied forces. Dynamic modeling that predicts the next state based on current observations is therefore essential for guiding a DLO toward a desired shape.

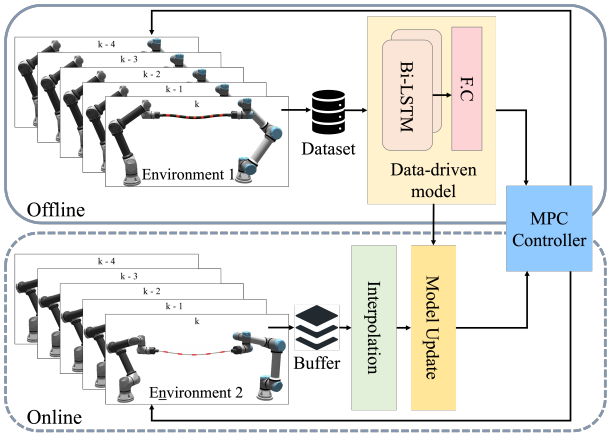


Fig. 1. Overview of the proposed DLO manipulation framework using a time-series data-driven model. This framework enables adaptation across diverse DLO environments by combining offline training with online learning.

The DLO modeling is classified into two main categories: analytical and data-driven methods. Modeling with analytical methods, such as mass-spring systems, position-based dynamics, and continuum mechanics, requires parameters that are difficult to obtain from the real environment. Data-driven approaches have an advantage over analytical methods, as they do not require complex physical dynamics and can also accurately learn complex dynamics [1]–[3]. Jacobian-matrix-based models use a Jacobian matrix to locally approximate the relationship between the state of end effectors and the features of the DLOs, which requires minimal data and can be computed in real-time. Neural network-based models can learn complex and global dynamics, which enhances accuracy and robustness compared to Jacobian-matrix-based models. However, these approaches had limitations in adapting to environmental changes and modeling inaccuracies. Recently, online learning has introduced the ability to adapt to environmental changes and object variations. This adaptability improves performance by ensuring accuracy and robustness in dynamic models during manipulation and decreases the Sim-to-Real gap even in changed environments [4], [5].

Manipulation methods for controlling DLOs have been proposed based on these models. Approaches that did not update the model are introduced, such as methods based on

Manuscript received: August 2 2025; revised: November 28 2025; accepted December 30 2025. This paper was recommended for publication by Editor Clement Gosselin upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2024-00340040). *(Corresponding author: Ayoung Hong.)*

Jiyoung Choi, Micheale H. Gebrezgiher, and Ayoung Hong are with the Department of Mechanical Engineering, Chonnam National University, Gwangju 61186, Republic of Korea jiyoung3347@gmail.com, Michealehaileslassie@gmail.com, ahong@jnu.ac.kr

Donggun Lee is with the Department of Mechanical and Aerospace Engineering, North Carolina State University, United States dlee48@ncsu.edu

Digital Object Identifier (DOI): see top of this page.

model predictive path integral (MPPI) [6] and model predictive control (MPC) [7], [8]. Adaptive control methods have also been introduced, combining online learning to update models and adjusting control gains dynamically, enhancing control performance [9], [10]. Additionally, a reinforcement learning-based approach was employed to update the model, thereby enhancing adaptability in dynamic environments [11].

In this paper, we focus on DLO manipulation in 3D space using two heterogeneous robotic arms, as shown in Fig. 1. We employ a time-series data-based recurrent neural network capable of learning both temporal and physical characteristics of the system. Additionally, we implement an interpolation-based online-offline learning strategy to enable control of DLOs that were not included in the original training dataset. Our main contribution is listed as follows:

- Employing time-series-based input data for a bi-LSTM network enables accurate future state prediction of DLOs without complex dynamic information. We do not need to collect complex dynamic parameters from DLOs.
- In data-driven models, once trained, the control of various configurations becomes challenging. By combining online residual learning and interpolation, we can effectively manipulate diverse DLOs that have different numbers of nodes and mechanical properties.

II. Related Works

A. Dynamic modeling of deformable linear objects

Dynamic modeling methods that estimate the future behavior or physical state of the DLO based on current observation are required for manipulation toward desired shapes. Here, we provide an overview of DLO modeling methods.

Analytical methods based on Newton's second law can be applied to model DLOs, encompassing mass-spring-damper systems [12], position-based dynamics [13], continuum mechanics [14], and elastic-rod systems [15]. However, analytical methods rely on parameters that are often difficult to obtain accurately, whereas data-driven methods learn deformations directly from data without requiring detailed physical modeling. Jacobian-based models simplify the neural network-based mapping between the state of end effectors and the DLO's feature state. They utilize data more efficiently for robotic manipulation and are helpful for cases involving small deformations and requiring real-time control.

Sequence-to-sequence models, such as LSTM and Transformer, and graph neural networks (GNNs), are capable of capturing complex dynamics given sufficient data and exhibit robustness to large deformations. A bi-LSTM network that matches the DLO structure of a chain-like mass-spring system has been introduced for modeling DLO dynamics [6]. It propagates the DLO states in both directions and predicts the next state. Moreover, the combination of a bi-LSTM network with an interaction network (IN) [16] that learns local interaction dynamics between nodes has been proposed for modeling DLO dynamics [8]. GNNs have been applied to learn the graph dynamics of DLOs [4], [5], [17]. They utilize message-passing mechanisms to aggregate and update node features based on the information they possess. The network is modeled based on interactions between nodes, with the DLO represented as the graph structure. Transformer-based approaches have also been explored, where the deformation of DLOs is formulated as a sequence-to-sequence task to more effectively capture temporal and contextual dependencies [10].

B. Online model learning

In augmentation of NN models, online learning has been introduced to address the challenges of Sim-to-Real transfer. Simulations simplify physical properties, leading to discrepancies when the model is applied in real-world scenarios. Online learning reduces uncertainties by enabling the continuous updating of pre-trained models in real-time based on residuals, which capture the differences between predicted and ground-truth states. This dynamic adaptation enables the model to handle unmodeled dynamics, including material variability and environmental disturbances. Recent studies have demonstrated the effectiveness of online learning in DLO manipulation. For example, online learning was applied to adapt a pre-trained deformation model for large-scale deformations [4], [5], [17]. Attention-based architectures have also been adopted to capture global deformation patterns in DLOs better, with some approaches incorporating online learning to adapt the model during execution [10]. Integrating online learning ensures accurate and robust manipulation of DLOs in diverse and dynamic environments.

C. Model-based control methods

Control strategies for manipulating deformable objects using robots have been studied. This section discusses the integration of MPC in DLO dynamic modeling and the adaptive control, which enhances control performance while improving the model's accuracy. MPC is an advanced control strategy that uses a system model to predict future behavior over a specified time horizon [18]. At each time step, the MPC algorithm calculates the optimal control inputs that minimize a cost function, considering future states while respecting the system's constraints. Given a deformable dynamic model that can predict the future states of DLOs iteratively, the MPC algorithm can optimize the control inputs to achieve the desired shape. Yan *et al.* performed shape control using MPC optimization through the shooting method [6], and Cao *et al.* adopted the interior point solver IPOPT for optimization [5]. Additionally, Tang *et al.* incorporated a control barrier function (CBF)-based safety filter into the MPC pipeline to enforce safety constraints in real time [10]. Adaptive control is a field of control theory that handles situations with unknown uncertainties in the system model being controlled. It adjusts the system or the controller parameters to satisfy performance requirements. Recently, DLO manipulation tasks employing adaptive control have been conducted to achieve improved shape-control performance. Li *et al.* and Yu *et al.* performed adaptive control combined with online learning [9], [19]. They achieved enhanced control performance by updating the model through online learning while adjusting control gains.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

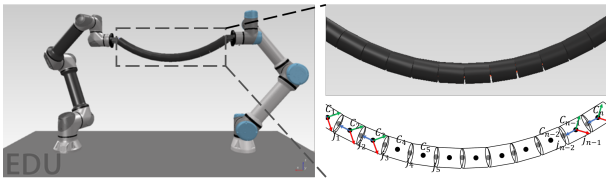


Fig. 2. Simulation environments with two robotic arms (UR5e and RB5-850) holding both ends of a DLO using Coppeliasim (V-REP) [20].

III. Dynamic Model of DLO

We model the DLO from time-series data instead of using its chain-like structure as input [6], [8], reducing state variables and easing measurement. We compare the proposed dynamic model with prior methods using simulation data.

A. Time-series data-based dynamic model

To develop a dynamic model of the DLO, we utilize the past or current state of the DLO as input to predict its future states. Rather than imposing a predefined structural framework for the propagation of effects between segments, this approach models the interactions among DLO segments based on their temporal dynamic behavior.

The dynamic model of the DLO, denoted as f, takes the prior states of DLO and the control input to predict the current state $\mathbf{X}_k$ as

$$\mathbf{X}_k = f(\mathbf{X}_{k-m}, \ldots, \mathbf{X}_{k-1}, \mathbf{U}_{k-m}, \ldots, \mathbf{U}_{k-1}) \tag{1}$$

where $\mathbf{X}$ is the state of the DLO and $\mathbf{U}$ is the control input for two robotic manipulators at a certain time step. Here, m represents the number of time-steps used to predict the current DLO state. The DLO is represented as a series of n linear cylindrical segments as shown in Fig. 2, and the state $\mathbf{X}$ consists of each segment's information $\mathbf{x}$ as

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \end{bmatrix}^T. \tag{2}$$

We considered two types of state variables: the first incorporates both position and velocity of each node, while the second includes only positional information, as follows:

$$\text{Type 1:} \quad \mathbf{x}_i = \begin{bmatrix} x_i & y_i & z_i & v_{xi} & v_{yi} & v_{zi} \end{bmatrix}^T \tag{3}$$

$$\text{Type 2:} \quad \mathbf{x}_i = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^T \tag{4}$$

Given that the robot is controlled via velocity commands, the control input to the system is defined as follows:

$$\mathbf{U} = \begin{bmatrix} \mathbf{u}_l & \mathbf{u}_r \end{bmatrix}^T \tag{5}$$

where $\mathbf{u}_l = \begin{bmatrix} \dot{q}_{l1} & \dot{q}_{l2} & \dot{q}_{l3} & \dot{q}_{l4} & \dot{q}_{l5} & \dot{q}_{l6} \end{bmatrix}^T$ and $\mathbf{u}_r = \begin{bmatrix} \dot{q}_{r1} & \dot{q}_{r2} & \dot{q}_{r3} & \dot{q}_{r4} & \dot{q}_{r5} & \dot{q}_{r6} \end{bmatrix}^T$ represent the velocity control inputs for the manipulators of the left and right arms, respectively.

B. Dynamic model training

To train the proposed time-series data-based model of the DLO and compare its performance with previous works, we utilized the simulation environment shown in Fig. 2. Two different robotic arms, each holding one end of a DLO with 24 segments, were sequentially positioned in random configurations. The DLO's position and velocity, as well as the robots' joint velocities, were recorded at 20 fps. We collected 1,000 trajectories, each for 9 s.

We trained the proposed time-series data-based bi-LSTM model and other chain-like models from [6], [8], [10] using the collected simulation data. We reimplemented the baseline models and trained using the same dataset and training schedules for fair comparison. The network models were implemented in the PyTorch library and trained on an Ubuntu machine equipped with an Intel i7-10900K processor and an NVIDIA RTX 3070 GPU. The root mean square error (RMSE) was used as the loss function, and the Adam optimizer [21] with a fixed learning rate of 0.0001 was used. The LSTM internal gates used sigmoid and tanh as the activation functions. Dropout, early stopping, and data augmentation were not applied, and reproducibility was ensured by fixing the random seed. All models were trained for 1000 epochs with a batch size of 512, and the collected dataset was divided into training and validation sets in a 7:3 ratio.

C. Performance of dynamic model

The performance of the proposed time-series data-driven dynamic model was evaluated based on prediction accuracy, measured by position RMSE, and model efficiency, assessed by inference time. Its prediction performance was compared to the performance of previous chain-like models in [6], [8], [10], as summarized in Table I.

The proposed model achieved higher prediction accuracy and faster inference speed compared to the other models. Given that chain-like models utilize a fixed single-step input, the comparison was made under the condition $m = 1$. At horizon one ($h = 1$), the proposed model achieved a position RMSE of 0.71 cm and 0.77 cm for DLO state of Type 1 and Type 2, respectively, which is slightly better than the performance of the other models. To evaluate long-term prediction capabilities, we increased the prediction horizon and performed recursive forecasting. While the rollout prediction error increased for all models as the horizon extended, the proposed method consistently demonstrated superior performance when using Type 2 states. In terms of inference time, the proposed model was at least three times faster than the other models, clearly highlighting its advantage for real-time applications.

When we analyzed the prediction performance of the proposed dynamic model for the parameter m, we observed that the accuracy improved as m increased. This indicates that incorporating information from more previous time steps enhances the proposed model's ability to capture temporal dependencies. Both Type 1 (position and velocity) and Type 2 (position only) states led to improved performances as m increased; however, Type 2 achieved better results for short-term predictions, reaching a position RMSE of 0.23 cm at $m = 7$. Since Type 2 states involve fewer variables and velocity information is typically more difficult to acquire, we use dynamic models for DLO using Type 2 states. Considering

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

TABLE I
PERFORMANCE OF THE PROPOSED TIME-SERIES DATA-DRIVEN MODEL

Model	DLO state [a]	m [b]	Position RMSE[c] [cm]				Inference time [ms]	Params [M]	FLOPs / sample [M]
			$h=1$	$h=2$	$h=5$	$h=10$			
bi-LSTM (time-series model)	Type 1	1	0.71	2.03	5.79	11.3	0.22	0.96	1.91
		2	0.52	0.86	2.14	4.95	0.26	1.00	3.81
		3	0.36	0.58	1.62	3.85	0.29	1.04	5.72
		4	0.36	0.55	1.50	3.63	0.32	1.08	7.62
		5	0.37	0.52	1.32	3.19	0.36	1.13	9.53
		6	0.32	0.45	1.20	3.18	0.38	1.17	11.4
		7	0.34	0.49	1.27	3.15	0.40	1.21	13.3
	Type 2	1	0.77	1.46	3.15	5.28	0.21	0.85	1.7
		2	0.51	1.04	2.73	6.47	0.25	0.87	3.4
		3	0.39	0.77	2.23	5.69	0.29	0.89	5.1
		4	0.31	0.65	1.97	4.69	0.32	0.91	6.8
		5	0.27	0.55	1.72	4.54	0.34	0.93	8.5
		6	0.25	0.50	1.56	4.08	0.37	0.96	10.1
		7	0.23	0.48	1.61	4.49	0.40	0.98	11.8
bi-LSTM [6] (chain-like model)	Type 1	1	0.72	1.90	4.68	7.92	0.88	1.78	37.8
	Type 2	1	0.80	1.46	3.41	5.94	0.85	1.26	36.6
INbiLSTM [8] (chain-like model)	Type 1	1	0.81	2.06	6.00	15.4	0.65	1.16	58.8
	Type 2	1	1.63	3.17	7.94	15.7	0.64	1.16	58.6
Transformer [10] (chain-like model)	Type 1	1	0.71	1.20	5.92	12.4	0.92	2.39	38.9
	Type 2	1	0.78	1.61	3.81	6.89	0.91	2.38	38.9

All baseline models were retrained on identical data and training schedules for fair comparison.
[a] Type 1 contains both the DLO position and velocity states, while Type 2 includes only the DLO position state, as in Eq. (3),(4).
[b] The parameter m represents the number of time steps used to predict the current DLO state, as in Eq. (1).
[c] The position RMSE is represented as a function of parameter h, which assesses the model's performance through multi-step rollout predictions.

TABLE II
PREDICTION ERROR FOR DIFFERENT NUMBER OF SEGMENTS

n	4	6	8	12	24
RMSE [cm]	0.789	0.708	0.593	0.597	0.570

the model's prediction performance and efficiency, we selected $m = 5$ for further experiments.

We also analyzed how the number of segments n affects prediction error using the selected model as shown in Table II. The error decreased notably as n increased from 4 to 8, but improvements became marginal beyond $n = 8$. Based on these results, we used $n = 7$ in the real-world experiments to balance accuracy with practical constraints such as marker placement, noting that this value is near the saturation point around $n \approx 8$.

IV. CONTROL FRAMEWORK FOR DLO MANIPULATION

A. MPC formulation

An MPC controller is designed to find the optimal actions for robots, minimizing the positional errors, based on the dynamic model of the system as follows:

$$\mathbf{U}^* = \arg\min_{\mathbf{U}_{k:k+h-1}} \sum_{i=1}^{h} \texttt{RMSE}(\mathbf{X_d}, \hat{\mathbf{X}}_{\mathbf{k+i}}) + c_{\text{dist}} \tag{6}$$

$$\text{subject to} \quad \hat{\mathbf{X}}_k = f(\mathbf{X}_{k-m}, ..., \mathbf{X}_{k-1}, \mathbf{U}_{k-m}, ..., \mathbf{U}_{k-1})$$
$$|\mathbf{U}_k| \leq \epsilon$$

where $\mathbf{X_d}$ is the desired shape and $\hat{\mathbf{X}}$ is the predicted shape using the trained dynamic model f. The control bound ϵ limits commanded joint velocity magnitude and therefore influences the observed physical manipulation regime. Small ϵ yields slow motions that resemble a sequence of quasi-static configurations, while larger ϵ permits higher velocities and more dynamic behavior with stronger inertial effects. Since the DLO is assumed to be inextensible, we include a soft constraint term, c_{dist}, in the MPC formulation to prevent excessive stretching. This term adds a fixed cost λ when the distance between the first and last nodes exceeds a certain threshold d_L, as follows:

$$c_{\text{dist}} = \begin{cases} \lambda, & \text{if } \|\mathbf{x}_1 - \mathbf{x}_n\| \geq d_{\text{L}} \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

B. Interpolation for DLOs with different configurations

When manipulating DLOs with a different number of nodes than the configuration used during training, one-dimensional interpolation is applied to adjust the input structure. This enables the model to handle DLOs with different configurations without retraining the entire model.

The positions of the original and new nodes are first normalized to the range $[0, 1]$. For the original DLO with n_{orig} nodes, normalized positions are defined as $\mathbf{x}_i = \frac{(i-1)}{(n_{\text{orig}}-1)}$, and for the new DLO with n_{new} nodes, as $\mathbf{x}'_{i'} = \frac{(i'-1)}{(n_{\text{new}}-1)}$. A linear interpolation function is constructed using the original positions and corresponding node values, and new node values

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Algorithm 1 MPC with Dynamic Model Update

Input: Data-driven model f, horizon h, cost function J, error threshold e

Output: Optimal control input $\mathbf{U}^*$

Initialize f, J
while $\texttt{RMSE}(\mathbf{X}_d, \mathbf{X}_k) \geq e$ **do**
  $\mathbf{X}_{k-1} \leftarrow \texttt{Interpolate}(\mathbf{X}_{k-1})$
  Predict next state: $\hat{\mathbf{X}}_k = f(\mathbf{X}_{k-m:k-1}, \mathbf{U}_{k-m:k-1})$
  Apply control constraint: $|\mathbf{U}_k| \leq \epsilon$
  Solve optimization to minimize J using DE:
  for $i = 1$ **to** h **do**
    Predict: $\hat{\mathbf{X}}_{k+i} = f(\mathbf{X}_{k+i-m:k+i-1}, \mathbf{U}_{k+i-m:k+i-1})$
    Compute cost J
  Apply control: $\mathbf{U}_k^*$
  Obtain measured state: $\mathbf{X}_k \leftarrow \texttt{Interpolate}(\mathbf{X}_k)$
  Update model: $f \leftarrow \texttt{Update}(f, \hat{\mathbf{X}}_k, \mathbf{X}_k)$

are obtained by evaluating this function at the normalized positions of the new nodes as follows:

$$\mathbf{X}^{\text{in}} = \texttt{interpolate}(\mathbf{x}_i, \mathbf{X}^{\text{orig}})(\mathbf{x}'_{i'}) \tag{8}$$

The interpolated components are then combined to reconstruct the full 3D coordinates of the DLO.

C. Dynamic model update

To handle diverse DLO configurations not included in the training dataset, we employ an integrated control method that combines online learning and the interpolation method described in Section IV-B. The interpolation enables the robust manipulation of DLOs with varying numbers of nodes without requiring the retraining of the entire model. The online residual transfer learning approach enables the model to adapt to various DLO configurations that have different dynamic properties.

Algorithm 1 presents the MPC framework incorporating dynamic model update. The algorithm begins by initializing the data-driven model f and the cost function J. It then enters a loop if the RMSE between the target state and the current state exceeds the predefined threshold e. The previous state is reconstructed through interpolation, after which the data-driven model f predicts the future state. The DE algorithm is then used to minimize the cost function J over the prediction horizon h and determine the optimal control vector. The resulting control input $\mathbf{U}_k^*$ is applied to the system, and the model f is updated in real-time based on the residual error between the predicted state and the ground truth states. This real-time update enables the model to adapt to changes in DLO characteristics.

V. Experiments

A. Experimental Setup

Experiments were conducted using a dual-arm robotic system to manipulate cables with varying configurations into predefined target shapes, both in simulation and real-world environments.

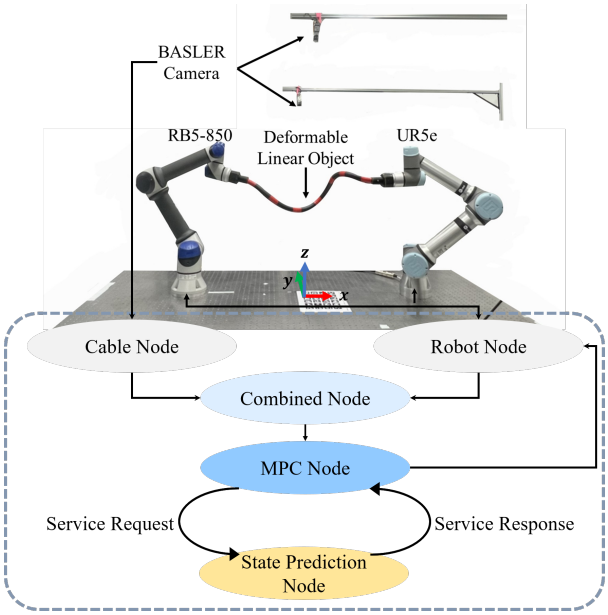


Fig. 3. Experimental environment and software framework. Two Basler cameras capture the cable state; marker 3D positions are estimated using stereo matching.

1) Setup: The experimental setup and the software framework are illustrated in Fig. 3. The system was implemented in a ROS2 workspace, where control optimization was performed in C++, while other components (including dynamic model inference and online updates) were developed in Python. The system operates at a control rate of 5 Hz, corresponding to an end-to-end control loop time of approximately 200 ms. We selected a dynamic model of DLOs with $m = 5$, based on the performance evaluation presented in Section III-C.

We have evaluated three control frameworks as follows: MPC (model predictive control without model updates), MPC+OL (MPC with online model updates), and MPC+OL+I (MPC with online model updates and input interpolation)

For online learning, we used the Adam optimizer with a learning rate of 5×10^{-7} with L_2 regularization and each online update consisted of a single gradient step. During online adaptation, only the final linear decoder layer is fine-tuned, while all LSTM layers remain frozen. The design was chosen to ensure that the model can adapt to the changes while minimizing degradation of previously learned performance.

2) Procedure: A feasible target shape is either randomly selected from the training dataset or determined by positioning the DLO using robot arms within the simulation or real-world environments. This procedure ensures that all target shapes are physically attainable. For every final shape observed in our tests for the original cable, we verified that the final shape is a locally asymptotically stable equilibrium of the closed-loop system (see Appendix).

The control framework then attempts to manipulate the cable to match the target shape within a fixed time limit, T. A trial is considered successful if the cable reaches the target shape with a position RMSE below the threshold e; otherwise, it is deemed a failure. Additionally, a trial is considered a fail-

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

TABLE III
EXPERIMENTAL RESULTS IN SIMULATION

	DLO Configurations				Method[a]	Achieved targets	Success rate	Error [m]	Time [s]
	diameter [cm]	length [cm]	weight [kg]	n					
Original cable	4.35	50	0.03	24	MPC	9/10	25/30	0.03812	11.47
					MPC+OL	8/10	21/30	0.03843	13.14
					MPC+OL+I	8/10	21/30	0.03831	12.11
Cable I (heavier)	4.35	50	0.06	24	MPC	9/10	24/30	0.03804	14.84
					MPC+OL	9/10	27/30	0.03798	12.14
					MPC+OL+I	9/10	26/30	0.03892	13.08
Cable II (longer)	5.00	60	0.05	24	MPC	8/10	18/30	0.03852	26.38
					MPC+OL	9/10	21/30	0.03843	23.79
					MPC+OL+I	9/10	21/30	0.03839	28.47
Cable III (more nodes)	4.35	50	0.03	28	MPC+I	7/10	19/30	0.03842	17.90
					MPC+OL+I	8/10	21/30	0.03847	16.72
Cable IV (shorter)	4.35	25	0.03	24	MPC	5/10	7/30	0.03776	9.64
					MPC+OL	7/10	15/30	0.03870	19.51
					MPC+OL+I	7/10	14/30	0.03851	15.23
Cable V (less nodes)	5.00	60	0.05	12	MPC+I	6/10	8/30	0.03771	44.89
					MPC+OL+I	8/10	19/30	0.03825	31.02

[a] MPC: Model Predictive Control, OL: Online Dynamic Model Update, I: Interpolation

ure if the time limit is exceeded, if either of the robotic arms collides with the table, or if the cable undergoes overextension. Each experiment was repeated three times for the same target shape. For the simulation environment, we set the time limit to $T = 70\,\mathrm{s}$ and the position error threshold to $e = 4\,\mathrm{cm}$. For the real-world environment, we used $T = 85\,\mathrm{s}$ and $e = 3\,\mathrm{cm}$.

B. Simulation Results

1) Dynamic model training: The simulation environment was implemented using Coppeliasim (V-rep) to obtain accurate ground truth data. We used the experimental environment shown in Fig. 2, which was also employed for evaluating the data-driven model evaluation. To construct the training dataset, 1,000 simulation episodes were collected using the original cable in Table III, each representing a unique cable deformation sequence. The dynamic model is implemented as a two-layer bi-LSTM network, with each layer comprising 150 hidden units. A fully connected layer is appended to produce the final output predictions.

2) Results: Table III compares the simulation results across various cable configurations and control methods (MPC, MPC+OL, MPC+OL+I) in terms of the number of achieved targets and the success rate. The number of achieved targets refers to the number of target shapes that were successfully reached at least once, while the success rate denotes the number of successful attempts out of three trials conducted for each target. The final cable shape was compared to the target shape to compute the average error, and the completion time was averaged over only the successful trials. Since cables I, II, and IV have the same number of nodes as the original cable, interpolation is unnecessary. In contrast, interpolation must be applied for cables III and V due to differences in node count.

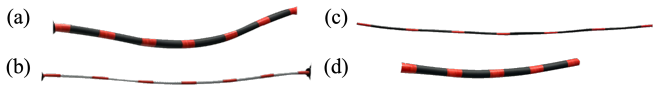


Fig. 4. The cables used in the experiment. (a, d) Hydraulic hoses with different lengths, (b) a coated steel cable, and (c) a multi-core cable. Detailed configurations of the cables are provided in Table IV.

MPC achieved high success rates for the original cable, with nine target shapes. However, when online learning was introduced (MPC+OL, MPC+OL+I), overall performance declined with fewer achievable targets, a reduced success rate, and an increased completion time. For cables I, II, and IV, which have the same number of nodes but with different physical properties, online learning (MPC+OL) resulted in a performance improvement compared to MPC. However, in these cases, interpolation has minimal impact. For cables III and V, which have the different numbers of nodes and physical properties, OL provides a clear advantage. Overall, the addition of online learning consistently improves performance when the DLO configuration deviates from the training setup.

C. Real-world experimental results

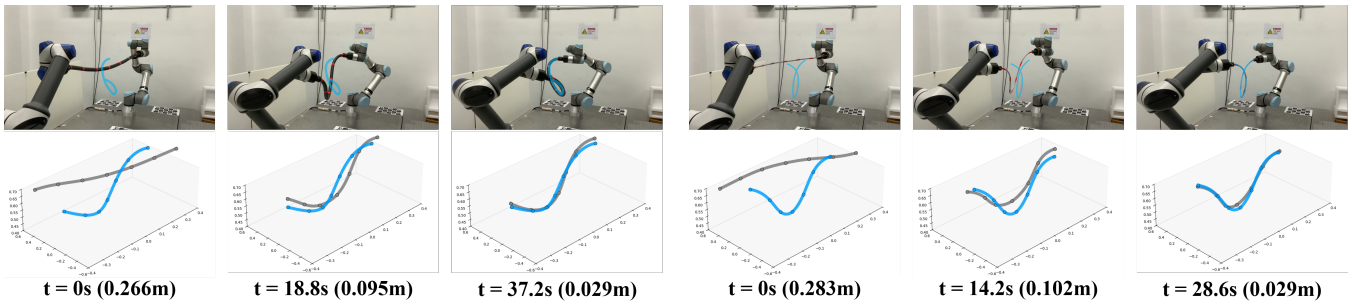
1) Dynamic model training: In the real-world environment, four different types of cables were used, as shown in Fig. 4. To construct the training dataset, 160 episodes were collected using a 98 cm hydraulic hose. Each episode corresponds to 9 seconds of motion recorded at 5 fps, resulting in a total of 7,200 states. The dynamic model is implemented as a three-layer bi-LSTM network, with each layer comprising 128 hidden units. All other training settings remained identical to those used in the simulation.

2) Results: Table IV presents the results across various cable configurations and control methods in real-world ex-

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

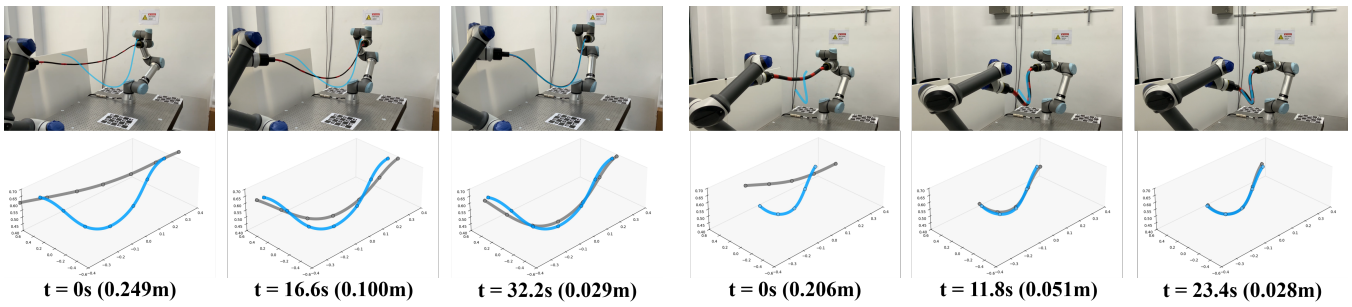
TABLE IV
EXPERIMENTAL RESULTS IN REAL-WORLD ENVIRONMENT

	Cable Configurations				Method	Achieved targets	Success rate	Error [m]	Time [s]
	diameter [cm]	length [cm]	weight [kg]	n					
Original cable	2.75	98.0	0.435	7	MPC MPC+OL	4/5 4/5	12/15 10/15	0.0287 0.0292	29.49 33.45
Cable I (lighter)	1.00	98.0	0.298	7	MPC MPC+OL	3/5 4/5	7/15 11/15	0.0292 0.0286	30.97 31.44
Cable II (longer)	1.00	114.0	0.205	7	MPC MPC+OL	4/5 4/5	10/15 12/15	0.0295 0.0282	27.08 28.70
Cable III (less nodes)	2.75	69.0	0.310	5	MPC+I MPC+OL+I	4/5 4/5	7/15 10/15	0.0295 0.0289	22.12 24.41



(a) Original cable ($n = 7$) - MPC

(b) Cable with different weight ($n = 7$) - MPC + OL



(c) Long cable ($n = 7$) - MPC + OL

(d) Short cable ($n = 5$) - MPC + OL + I

Fig. 5. Real-world experimental results of cable manipulation under various configurations. In the top images of each subfigure, the yellow line represents the target shape. The bottom plots show the cable's shape from the initial to the final configuration. The gray line depicts the cable's motion over time, while the blue line indicates the target shape.

periments. Figure 5 shows representative results from one successful trial for each of the tested cable configurations.

For the original cable, MPC slightly outperformed MPC+OL in terms of success rate, which aligns with the simulation results and suggests that model adaptation may be unnecessary when the system closely matches the trained model. In contrast, for cables I and II, the use of online learning helped adapt to changes in cable properties. Cable III, which had both a reduced number of nodes and a shorter length compared to the original configuration, was tested with MPC+I and MPC+OL+I. In this case, MPC+OL+I yielded better performance, achieving a higher success rate than interpolation alone.

VI. DISCUSSION AND CONCLUSION

The proposed time-series data-driven dynamic model for DLOs processes the input data sequentially along the time axis, focusing on learning the temporal dynamics of the DLO. In contrast, the previous chain-like models interpret the DLO as a structure composed of sequential nodes or segments, emphasizing spatial relationships based on the object's topological configuration. These two approaches reflect fundamentally different strategies for representing and utilizing temporal versus spatial information. This study demonstrates that a temporally structured and computationally lightweight model can achieve reliable prediction performance without explicitly modeling the spatial configuration.

We performed shape control on ten distinct cable configurations using four control methods (MPC, MPC+I, MPC+OL, MPC+OL+I) in simulated and real-world environments. MPC alone provided a strong baseline performance when the cable configuration matched the training conditions. However, adding online learning or interpolation degraded performance.

In contrast, for configurations that deviated from the training conditions, such as those with different lengths, weights, or numbers of nodes, online learning improved the success rates and robustness across trials by compensating for residual modeling errors. We employed simple linear interpolation to map variable node discretizations into the fixed input format expected by the trained model, enabling control without retraining. Interpolation was effective when paired with online learning, however, it can distort geometry and introduce representation error for highly curved shapes. Structure-aware upsampling approaches could be explore to provide higher accuracy and robustness when discretization differs substantially [22], [23].

In this paper, the experiment was configured to terminate once a predefined target error was reached, which defined the success criterion. As a result, the reported control errors may appear larger than those in other studies. The experiments used relatively long cables in a 3D environment, which is more challenging, and allowing longer execution by relaxing the termination threshold would further reduce control error. We also observed that success rates declined when the initial configuration was significantly different from the target shape. To address this limitation, future work may incorporate feasible via-points or path planning strategies to guide the DLO toward the target more effectively.

While the proposed approach demonstrated reliable cable manipulation under controlled, obstacle-free conditions, its applicability to more realistic settings remains limited. External disturbances, table friction, and knot formation were not modeled and may introduce additional complexity that could degrade performance. Addressing these interactions will require enhanced physical modeling or alternative learning architectures, and we consider this an important direction for future work.

APPENDIX

STABILITY ANALYSIS AT THE TARGET SHAPE

Using the trained model, we evaluate the largest magnitude of the eigenvalues of the Jacobian matrix by solving

$$\min_{|\mathbf{U}|\leq\epsilon} |\lambda|_{\max}(J(\mathbf{U})) \tag{9}$$

where the Jacobian matrix $J(\mathbf{U})$ is defined as

$$\begin{aligned} J(\mathbf{U}) = &\frac{\partial f}{\partial \mathbf{X}_{k-m}}(\mathbf{X}_d, \ldots, \mathbf{X}_d, \mathbf{U}, \ldots, \mathbf{U}) \\ &+ \cdots + \frac{\partial f}{\partial \mathbf{X}_{k-1}}(\mathbf{X}_d, \ldots, \mathbf{X}_d, \mathbf{U}, \ldots, \mathbf{U}). \end{aligned} \tag{10}$$

For each target shape listed in Table IV, the resulting largest eigenvalue magnitudes of the Jacobian matrix are 0.7352, 0.7822, 0.8505, 0.7594, and 0.7723, respectively. Since all values are strictly less than 1, the discrete Hartman–Grobman theorem [24] implies that each target shape is locally asymptotically stable.

REFERENCES

[1] H. Yin, A. Varava, and D. Kragic, "Modeling, learning, perception, and control methods for deformable object manipulation," *Sci. Robot.*, vol. 6, no. 54, p. eabd8803, 2021.

[2] J. Sanchez, J.-A. Corrales, B.-C. Bouzgarrou, and Y. Mezouar, "Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey," *Int. J. Robot. Res.*, vol. 37, no. 7, pp. 688–716, 2018.

[3] K. Almaghout, A. Cherubini, and A. Klimchik, "Robotic co-manipulation of deformable linear objects for large deformation tasks," *Robotics and Autonomous Systems*, vol. 175, p. 104652, 2024.

[4] C. Wang, Y. Zhang, X. Zhang, Z. Wu, X. Zhu, S. Jin, T. Tang, and M. Tomizuka, "Offline-online learning of deformation model for cable manipulation with graph neural networks," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 5544–5551, 2022.

[5] B. Cao, X. Zang, X. Zhang, Z. Chen, S. Li, and J. Zhao, "Shape control of elastic deformable linear objects for robotic cable assembly," *Adv. Intell. Syst.*, vol. 6, no. 7, p. 2300835, 2024.

[6] M. Yan, Y. Zhu, N. Jin, and J. Bohg, "Self-supervised learning of state estimation for manipulating deformable linear objects," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2372–2379, 2020.

[7] Y. Li, J. Wu, J.-Y. Zhu, J. B. Tenenbaum, A. Torralba, and R. Tedrake, "Propagation networks for model-based control under partial observation," in *IEEE International Conference on Robotics and Automation*, 2019, pp. 1205–1211.

[8] Y. Yang, J. A. Stork, and T. Stoyanov, "Learning to propagate interaction effects for modeling deformable linear objects dynamics," in *IEEE International Conference on Robotics and Automation*, 2021, pp. 1950–1957.

[9] M. Yu, K. Lv, H. Zhong, S. Song, and X. Li, "Global model learning for large deformation control of elastic deformable linear objects: An efficient and adaptive approach," *IEEE Trans. Robot.*, vol. 39, no. 1, pp. 417–436, 2023.

[10] Y. Tang, X. Chu, J. Huang, and K. W. Samuel Au, "Learning-based mpc with safety filter for constrained deformable linear object manipulation," *IEEE Robot. Autom. Lett.*, vol. 9, no. 3, pp. 2877–2884, 2024.

[11] Y. Yang, J. A. Stork, and T. Stoyanov, "Online model learning for shape control of deformable linear objects," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022, pp. 4056–4062.

[12] N. Lv, J. Liu, X. Ding, J. Liu, H. Lin, and J. Ma, "Physically based real-time interactive assembly simulation of cable harness," *J. Manuf. Syst.*, vol. 43, pp. 385–399, 2017.

[13] M. Macklin, M. Müller, and N. Chentanez, "Xpbd: position-based simulation of compliant constrained dynamics," in *International Conference on Motion in Games*, 2016, pp. 49–54.

[14] A. Koessler, N. R. Filella, B. Bouzgarrou, L. Lequièvre, and J.-A. C. Ramon, "An efficient approach to closed-loop shape control of deformable objects using finite element models," in *IEEE International Conference on Robotics and Automation*, 2021, pp. 1637–1643.

[15] S. Tiburzio, T. Coleman, D. Feliu-Talegon, and C. D. Santina, "Controlling deformable objects with nonnegligible dynamics: A shape-regulation approach to end-point positioning," *IEEE Trans. Robot.*, vol. 41, pp. 6213–6228, 2025.

[16] P. Battaglia, R. Pascanu, M. Lai, D. Jimenez Rezende *et al.*, "Interaction networks for learning about objects, relations and physics," *Advances in neural information processing systems*, vol. 29, 2016.

[17] Y. Huang, C. Xia, X. Wang, and B. Liang, "Learning graph dynamics with external contact for deformable linear objects shape control," *IEEE Robot. Autom. Lett.*, vol. 8, no. 6, pp. 3892–3899, 2023.

[18] J. B. Rawlings, D. Q. Mayne, M. Diehl *et al.*, *Model predictive control: theory, computation, and design*. Nob Hill Publishing Madison, WI, 2017, vol. 2.

[19] X. Li and C. C. Cheah, "Adaptive neural network control of robot based on a unified objective bound," *IEEE Trans. Control Syst. Technol.*, vol. 22, no. 3, pp. 1032–1043, 2014.

[20] E. Rohmer, S. P. N. Singh, and M. Freese, "V-rep: A versatile and scalable robot simulation framework," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 1321–1326.

[21] D. P. Kingma, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations*, 2014.

[22] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. Battaglia, "Learning to simulate complex physics with graph networks," in *International Conference on Machine Learning*, 2020, pp. 8459–8468.

[23] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "Pu-net: Point cloud upsampling network," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2790–2799.

[24] S. Sastry, *Nonlinear systems: analysis, stability, and control*. Springer Science & Business Media, 2013, vol. 10.