GraphDLO: Graph-Based Neural Dynamics for Deformable Linear Object Trajectory Prediction

Holly Dinkel[1], Muhammad Zahid[2], Bhumsitt Pramuanpornsatid[1], Brian Coltin[3], Trey Smith[3], Florian Pokorny[2], and Timothy Bretl[1]

***Abstract*—This work introduces GraphDLO, a graph-based learning framework for predicting the future trajectories of Deformable Linear Objects (DLOs) under both prehensile (grasping) and non-prehensile (pushing) interactions. A data set collected over 300 hours of interactions among each of three distinct rope objects is collected remotely using a cloud robotics platform and automatically labeled with rope and gripper state information from perception. A Graph Neural Network (GNN) is trained on this dataset to take as input the current rope state and a gripper trajectory to predict a trajectory of future rope states. The GraphDLO-predicted trajectories exhibit close qualitative agreement with ground truth trajectories across a prediction horizon of up to ten steps, demonstrating its potential for accurate long-horizon prediction in deformable object manipulation.**

I. Introduction

Robots interact with increasingly diversified objects in the real world. Manipulation robots–including humanoids, manipulation arms, manipulation drones, and other dexterous devices–will soon be ubiquitous, and the types of objects with which they interact must become better understood for the automation to be worthwhile. One category of manipulable objects present in nearly every human environment is Deformable Linear Objects (DLOs). Future autonomous manipulation systems may manipulate DLOs into desired shapes to route cables [1], suture wounds [2], install wires [3], knit or braid string [4], or tie knots [5], [6]. One open problem specific to DLO shape control with dexterous robots is trajectory prediction. This problem is difficult because the shape of the entire object may change as the robot grasps and moves only one part of it, where the magnitude of the change in shape at points along the object typically depends on their distances to the rigidly grasped point [7], [8]. To address this problem, this work makes the following listed contributions:

1) A graph-based dynamics model that explicitly encodes the grasped region of the deformable linear object (DLO), enabling accurate prediction of future DLO states given the current configuration and a planned gripper trajectory.
2) A scalable framework for automatic labeling of planar rope configurations during real-world robotic manipulation, implemented on a cloud robotics platform to facilitate large-scale data collection for learning deformable object dynamics.
3) A set of ablation studies highlight the trade-offs between model performance and prediction horizon, providing insights into the temporal limits of dynamics-based forecasting for DLOs.

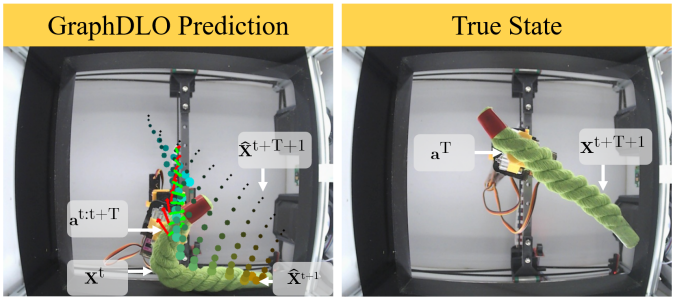

Fig. 1. **Predicting DLO Trajectories with GraphDLO.** GraphDLO predicts the future trajectory of a DLO, $\hat{\mathbf{X}}^{t+1:t+T+1}$, given its current state $\mathbf{X}^t$ and a sequence of actions $\mathbf{a}^{t:T}$. Predicted states are visualized as filled circles, where larger and brighter circles denote states closer in time, and smaller, darker circles represent states further in the future. The color gradient of DLO states is scaled from teal to gold based on the node order, indicating the structure in the representation of the DLO. Gripper actions are visualized as planar Cartesian axes at each step, with the red x-axis and the green y-axis indicating the gripper orientation.

II. Related Work

Graph-based representations enable local state information sharing through connected edges, making them well-suited for modeling the dynamics of non-rigid and granular objects [9]–[12]. Prior work on learning object dynamics has primarily focused on revealing properties such as inertial parameters or friction through interaction [13], [14], or applying analytical techniques like mass-spring systems and position-based dynamics (PBD) to evolve object shape over time [15]–[18]. In contrast, GraphDLO learns to directly predict rope trajectories—tasks that traditionally relied on numerically integrating classical dynamics models, often with trade-offs between accuracy and computational cost. Accurate prediction of DLO trajectories supports a range of downstream applications, including shape control and planning [12], [19]–[22], digital twin development [23], [24], and physics-informed state estimation [11], [25]–[28]. Predicting DLO trajectories also enables longer-horizon planning in environments with obstacles or interaction constraints [29], [30].

Although recent works have demonstrated success in learning deformable dynamics, they predominantly rely on synthetic datasets generated in physics-based simulators [16], [31]–[33]. The models trained on synthetic data are deployed

[1]Holly Dinkel, Bhumsitt Pramuanpornsatid, and Timothy Bretl are with the University of Illinois Urbana-Champaign, Urbana, IL, USA. e-mail: {hdinkel2, bp17, tbretl}@illinois.edu.

[2]Muhammad Zahid and Florian Pokorny are with the KTH Royal Institute of Technology, Stockholm, Sweden. e-mail: {mzmi, fpokorny}@kth.se

[3]Brian Coltin and Trey Smith are with the NASA Ames Research Center, Moffett Field, CA, USA. e-mail: {brian.coltin, trey.smith}@nasa.gov.

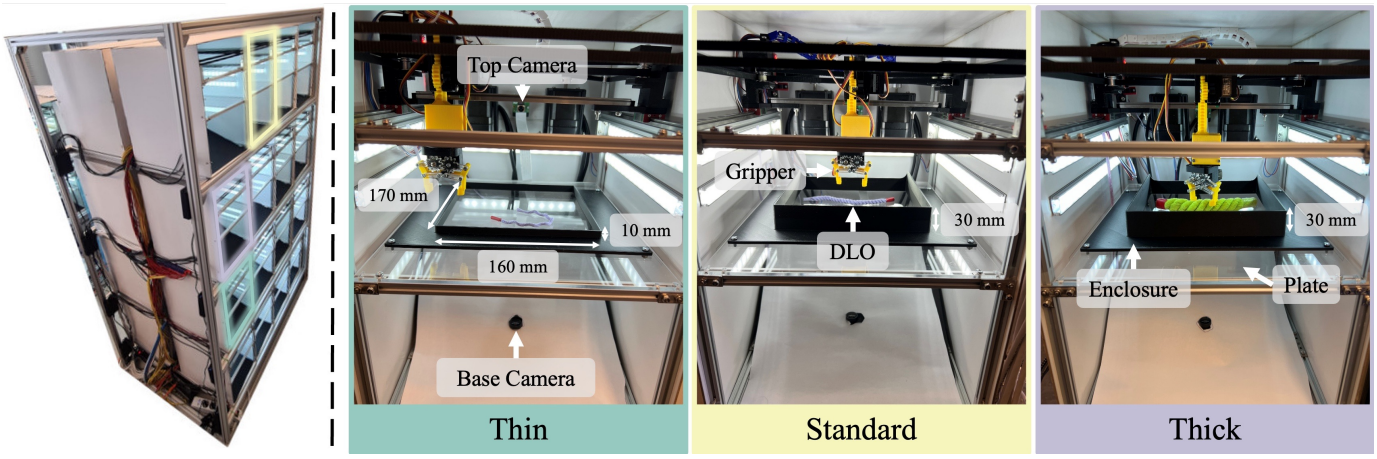

Fig. 2. **Collecting Data at Scale with CloudGripper.** Three grippers from the CloudGripper cloud robotics platform are used to collect interaction data for three different DLOs. In each workcell, a base-mounted camera captures an occlusion-free bottom-up view of the DLO resting on a transparent plexiglass plate. A 3D-printed enclosure surrounds the manipulation area to constrain the rope within the gripper's workspace during interaction.

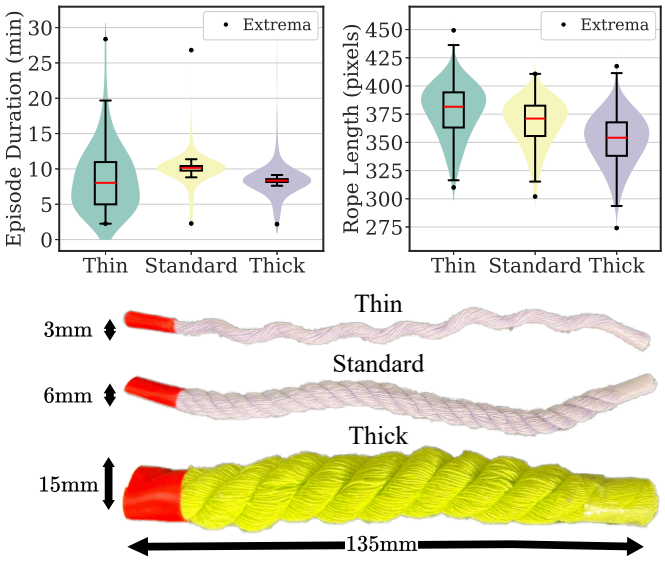

Fig. 3. **Estimating Large-Scale Data Statistics.** (Top left) Each interaction episode spans approximately 10 minutes and consists of up to 100 commanded gripper waypoints. (Top right) During automated data collection, rope length is estimated in each image as a proxy for rope labeling accuracy. The resulting distribution of measured rope lengths across all episodes remains tightly concentrated, indicating consistent and reliable labeling. (Bottom) Three ropes of approximately equal length but varying thickness are used throughout data collection and model training.

directly on data acquired in the real world for inference, and it is still unclear how transferrable this approach is when more complex environment effects such as non-prehensile interaction with environment obstacles are introduced. Advances in cloud robotics provide a promising alternative by enabling large-scale, real-world data collection through distributed fleets of identical robots operating in synchronized environments [34]. This capability is especially valuable for gathering datasets involving rich contact dynamics or complex deformable behaviors [35]–[38]. Additionally, recent topology-grounded developments in DLO perception and tracking enable automatic labeling of such data at scale [39], [40].

While many robot learning approaches for deformables have focused on non-prehensile pushing interactions [10], [34], [41], [42], key challenges remain unaddressed. One such gap is preserving topological consistency, which is essential for many DLOs that are visually or functionally asymmetric along their length. Representing a DLO as an unordered set of points can lead to ambiguities where point order inverts between time steps despite the object itself remaining unchanged, particularly under rotation. Furthermore, non-prehensile manipulation does not capture rotational dynamics within a grasp. This work directly addresses these limitations by incorporating grasp-aware modeling, ordered graph structures, and real-world physical interactions into deformable object prediction.

III. The GraphDLO Method

The GraphDLO model predicts a trajectory of future states of a DLO based on its current state, the grasp location, and the planned gripper trajectory as shown in Figure 1. In contrast to prior approaches that use on past object states and gripper motions to forecast the next state, GraphDLO relies on the Markov assumption and models the future states as dependent solely on the current state [10], [42]. This simplifies the input representation and reduces the dimensionality of the model.

A. Graph Model

For a planar DLO represented as a graph of N nodes and $N-1$ edges in two dimension, the GraphDLO algorithm learns to predict the 2D trajectories of nodes as

$$\hat{\mathbf{X}}^{t+1:t+T+1} = f(\mathbf{X}^t, \mathbf{g}^t, \mathbf{a}^{t:t+T}), \tag{1}$$

where $\hat{\mathbf{X}}^{t+1:T+1} \subset [0,1]^{T\times N\times 2}$ is the predicted trajectory, $\mathbf{X}^t \subset [0,1]^{N\times 2}$ is the estimated DLO state, $\mathbf{g}^t \subset \{0,1\}^N$ encodes which node is grasped such that $\sum_i \mathbf{g}_i^t = 1$, and $\mathbf{a}^t = \{x_g^t, y_g^t, \theta_g^t\} \subset [0,1]^{1\times 3}$ is the gripper state. All model inputs and outputs are normalized to prevent features with larger scales from dominating in backpropagation to improve convergence speed and stability. The superscript $t : t+T$

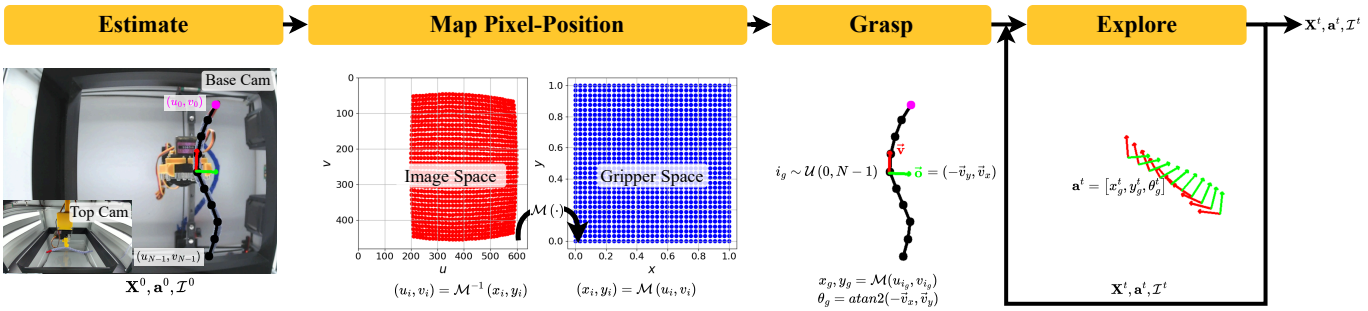

Fig. 4. **Automating DLO Prehensile Interactions.** The data generation process with CloudGripper automates prehensile manipulation of a rope across hundreds of interaction episodes. Dataset collection for a single episode proceeds as follows: the configuration of the DLO is first estimated in image space as an ordered sequence of pixel coordinates, captured from a base-mounted camera that provides an unobstructed bottom-up view [39], [40]. These pixel coordinates are mapped to the gripper frame using extrinsic calibration from pixels in the image space of the base camera to positions in the workspace of the gripper. A planar grasp is planned by uniformly sampling a point along the rope and computing a grasp pose based on the local geometry around the selected point. Once the rope is grasped, the robot executes a sequence of randomized planar translations and rotations, enabling the generation of diverse rope configurations under realistic manipulation. After each executed gripper waypoint, the system records the estimated DLO state in both image and gripper space, the base camera image, and the position of the gripper in both image and gripper frames.

indicates a temporally consecutive sequence indexed at start time t and with horizon T, so

$$\begin{aligned} \hat{\mathbf{X}}^{t+1:t+T+1} &= \left[\hat{\mathbf{X}}^{t+1}, \ldots, \hat{\mathbf{X}}^{t+T+1}\right]^\intercal \\ \mathbf{a}^{t:t+T} &= \left[\mathbf{a}^t, \ldots, \mathbf{a}^{t+T}\right]^\intercal \end{aligned}. \tag{2}$$

This work uses K-hop message passing to model interactions in the graph. First, the degree matrix D is computed from the adjacency matrix $\mathbf{A}$ and $\mathcal{N}(i)$, the set of neighbors for node i, where each diagonal element D_{ii} represents the number of neighbors for node i, computed as

$$D_{ii} = \sum_{j \in \mathcal{N}(i)} A_{ij}. \tag{3}$$

The adjacency matrix $\mathbf{A}$ can cause large variations in node features after aggregation, and nodes with many connections (high D_{ii}) can dominate. To stabilize aggregation, symmetric normalization is applied to the adjacency matrix as

$$\tilde{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}} \tag{4}$$

where $\mathbf{D}^{-\frac{1}{2}}$ is the inverse square root of the degree matrix and is computed as

$$D_{ii}^{-\frac{1}{2}} = \frac{1}{\sqrt{D_{ii}}}, \tag{5}$$

where if $D_{ii} = 0$, $D_{ii}^{-\frac{1}{2}}$ is set to 0 to avoid division by zero. If an edge exists between node i and node j, its weight is scaled by $D_{ii}^{-\frac{1}{2}} D_{jj}^{-\frac{1}{2}}$ to distribute information evenly across all nodes. After computing $\tilde{\mathbf{A}}$, the node states are updated for $k = 1, \ldots, K$ as

$$\mathbf{x}_i^{k+1} = \sum_{j \in \mathcal{N}(i)} \tilde{\mathbf{A}}_{ij} \mathbf{x}_j^k. \tag{6}$$

After message passing, the feature vector $\mathbf{F}^t = [\mathbf{X}^t, \mathbf{g}^t, \mathbf{a}^{t:t+T}]$ is passed through a neural network with hidden layers $\mathbf{h}_1 \subset [0,1]^{T, N\times 2+3\times T}$ with structure

$$\begin{aligned} \mathbf{h}_1 &= ReLU(\mathbf{W}_1 \mathbf{F}^t + \mathbf{b}_1) \\ \mathbf{h}_2 &= ReLU(\mathbf{W}_2 \mathbf{h}_1 + \mathbf{b}_2) \\ \hat{\mathbf{X}}^{t+1:t+T+1} &= \mathbf{W}_3 \mathbf{h}_2 + \mathbf{b}_3 \end{aligned}. \tag{7}$$

B. Loss Function

The loss function is selected to balance the desire for the model to learn DLO states that are low in distance to the target states as well as shapes that are similar in geometry to the target shapes. This is achieved by combining the Mean-Squared Error (MSE) with the contrastive Cosine Embedding (CE) loss functions. The MSE loss is

$$\mathcal{L}_{MSE}(\hat{\mathbf{X}}^t, \mathbf{X}^t) = \frac{1}{N} \sum_{i=1}^{N} \left(\hat{\mathbf{x}}_i^t, \mathbf{x}_i^t\right)^2. \tag{8}$$

The CE loss is given for margin λ by

$$\mathcal{L}_{CE}(\hat{\mathbf{X}}^t, \mathbf{X}^t, \mathbf{l}^t) = \begin{cases} 1 - S_C(\hat{\mathbf{X}}^t, \mathbf{X}^t) \mid l^t = 1 \\ \max(0, S_C(\hat{\mathbf{X}}^t, \mathbf{X}^t) - \lambda) \mid l^t = -1 \end{cases} \tag{9}$$

where the label, $l^t \in \mathbf{l}^t \subset \{-1, 1\}^N$, encourages the predicted and target vectors to be as similar as possible for $l = 1$ and penalizes the proximity of the prediction and target for $l = -1$, and the cosine similarity, S_C is

$$S_C(\hat{\mathbf{X}}^t, \mathbf{X}^t) := cos(\mathbf{\Theta}^t) = \frac{\hat{\mathbf{X}}^t \cdot \mathbf{X}^t}{\|\hat{\mathbf{X}}^t\| \|\mathbf{X}^t\|}. \tag{10}$$

The final loss function is

$$\mathcal{L} = \alpha \mathcal{L}_{MSE} + (1 - \alpha) \mathcal{L}_{CE}, \tag{11}$$

for hyperparameter α weighting each loss component.

IV. Data Collection

Over 300 hours of prehensile and non-prehensile interactions between a gripper and each of three cotton ropes were collected using the CloudGripper cloud robotics platform shown in Figure 2. While all the ropes share similar lengths, they differ in thickness and stiffness as shown in Figure 3. To break object symmetry, one tip of each rope was marked with red tape. The data collection process was automated to minimize human supervision. Initial object and tip masks were obtained using color and contour segmentation with object-specific thresholding, followed by refinement with the the

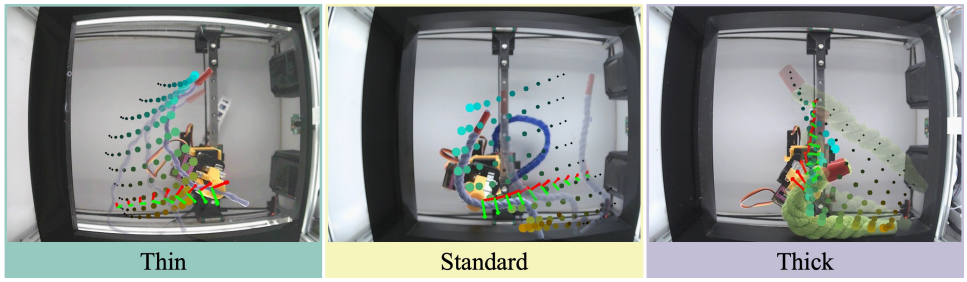

Fig. 5. **Visualizing GraphDLO Predictions.** Each panel shows an initial rope configuration (opaque rope) at t. The GraphDLO-predicted trajectory for $t+1:t+T+1$ is overlaid as dots, and the ground truth state at $t+1+5$ and $t+1+10$ (for $T=10$) is overlaid as semi-transparent. These predictions predictions demonstrate GraphDLO's ability to closely align with real-world trajectories for three different ropes.

Segment Anything Model (SAM) 2 predictor in instances where the length of the skeletonized mask fell outside an object-specific threshold length [43]. Given a binary object segmentation mask $\mathcal{M}^t \in \{0,1\}^{H\times W}$, a deformable one-dimensional object routing algorithm was used to skeletonize the mask, extract connected chains from the skeleton, and sample N evenly-distributed nodes. Each node is represented as $\mathbf{x}_i^t \in \mathbf{X}^t \subset [0,1]^{N\times 2}$ in Cartesian coordinates and $(v_i^t, u_i^t) \in \mathcal{I}^t\,(v_i^t, u_i^t) \subset \{0,\ldots,H-1\}\times\{0,\ldots,W-1\}^N$ in pixel space along the skeleton [39], [40]. The length of the rope serves as a proxy for segmentation quality and was used to constrain rope state labels during data collection. The distributions of rope lengths in pixel coordinates for the three objects are shown in Figure 3.

To enable pixel-to-position transformations, planar hand-eye calibration was performed using a red calibration cube translated through a dense grid of gripper positions. Each Cartesian gripper location was mapped to the corresponding pixel centroid of the cube, forming lookup tables for bidirectional interpolation between pixel and world coordinates. This calibration is limited to planar mappings, so all rope configurations were constrained to lie in the same plane. During interaction, the gripper grasp point is selected by sampling a node index $i_g \sim \mathcal{U}(0, N-1)$, where the pixel coordinate of the grasp node is $\left(u_{i_g}, v_{i_g}\right)$. The grasp orientation is computed as $\theta_g = \text{mod}(\theta_{i_g}, \pi)$, where θ_{i_g} is the local rope orientation at node i_g, ensuring valid orientations for the CloudGripper hardware. The gripper executes the grasp by rotating to θ_{i_g} and moving to $\left(x_{i_g}, y_{i_g}\right) = \mathcal{M}^{-1}\left(u_{i_g}, v_{i_g}\right)$ before closing. It then follows a sequence of randomly sampled waypoints, interpolating each transition into 10 intermediate poses. At each waypoint, rope and gripper states (in both pixel and Cartesian space) along with synchronized images and timestamps are recorded. The full data generation process for one episode is summarized in Figure 4.

V. Demonstration and Limitations

The performance of the GraphDLO algorithm is demonstrated on trajectory predictions for the thin, standard, and thick ropes. As shown in the qualitative results shared in Figure 5, GraphDLO accurately predicts the rope trajectories sampled from the validation data.

However, the model occasionally exhibits uncertainty in predicting DLO shapes, particularly near where the rope contacts the fixed enclosure. This may stem from the fact that the enclosure is not explicitly represented in the training data or modeled within GraphDLO. Future work could explore incorporating the enclosure as part of the action or environmental state to assess its effect on training efficiency and prediction accuracy. Additionally, the validity of the Markov assumption warrants further investigation for objects that accumulate internal energy—such as stiff deformable materials or granular media—where system dynamics may exhibit temporal dependencies. In the quasi-static demonstrations considered here, the Markov assumption holds reasonably well, but it may not generalize to more dynamic manipulation tasks.

A further limitation of the current method is the absence of 3D shape information. Future work could enhance spatial perception by stereo-matching the top and base cameras of the CloudGripper platform, incorporating foundation models such as Depth Anything or FoundationStereo for depth estimation [44]–[46]. This would enable the development of a 3D state estimator capable of capturing non-planar object configurations, facilitating downstream tasks such as knot tying or winding around a winch.

VI. Conclusion

This work introduced GraphDLO, a graph-based learning framework for predicting the future trajectories of a DLO given its current state, where it was grasped, and a gripper trajectory. By using this Markov assumption, GraphDLO simplifies the prediction problem while maintaining accuracy in quasi-static manipulation. The GraphDLO model was trained using over 300 hours of diverse prehensile and non-prehensile planar interactions collected autonomously with the CloudGripper platform, featuring three ropes with varying physical properties. Current limitations highlight opportunities for future work, including explicit modeling of workspace constraints, investigating the limits of the Markov assumption in more dynamic settings, and incorporating 3D shape estimation through stereo depth models. These directions will move us closer to generalizable, structured robot learning for real-world deformable object manipulation tasks such as knot tying, cable routing, and textile handling.

ACKNOWLEDGMENTS

The authors thank João Marcos Correia Marques for feedback on the manuscript, the teams developing the open-source software used in this project [47]–[51], and the members of the Representing and Manipulating Deformable Linear Objects project (github.com/RMDLO) for their support. Holly Dinkel was supported by NASA Space Technology Graduate Research Opportunity award 80NSSC21K1292, a P.E.O. Scholar Award, and the Zonta International Amelia Earhart Fellowship. This work was also supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

REFERENCES

[1] H. Zhou, S. Li, Q. Lu, and J. Qian, "A Practical Solution to Deformable Linear Object Manipulation: A Case Study on Cable Harness Connection," in *IEEE Int. Conf. Adv. Robot. Mech. (ICARM)*, 2020, pp. 329–333. 1

[2] S. A. Pedram, C. Shin, P. W. Ferguson, J. Ma, E. P. Dutson, and J. Rosen, "Autonomous Suturing Framework and Quantification Using a Cable-Driven Surgical Robot," *IEEE Trans. Robot.*, vol. 37, pp. 404–417, 2021. 1

[3] R. Lagneau, A. Krupa, and M. Marchal, "Automatic Shape Control of Deformable Wires Based on Model-Free Visual Servoing," in *IEEE Robot. Autom. Lett.*, vol. 5, Oct. 2020, pp. 5252–5259. 1

[4] J. Xiang and H. Dinkel, "Simultaneous Shape Tracking of Multiple Deformable Linear Objects with Global-Local Topology Preservation," in *IEEE Int. Conf. Robot. Autom. (ICRA) Workshop on Representing and Manipulating Deformable Objects*, May 2023. 1

[5] B. Lu, H. K. Chu, and L. Cheng, "Dynamic Trajectory Planning for Robotic Knot Tying," in *IEEE Int. Conf. Real-Time Comput. Robot. (RCAR)*, 2016, pp. 180–185. 1

[6] H. Dinkel, R. Navaratna, J. Xiang, B. Coltin, T. Smith, and T. Bretl, "KnotDLO: Toward Interpretable Knot Tying," *IEEE ICRA Workshop on 3D Visual Representations for Manipulation*, 2024. 1

[7] D. Berenson, "Manipulation of Deformable Objects Without Modeling and Simulating Deformation," in *IEEE/RSJ Int. Conf. Intell. Robot. Sys. (IROS)*, 2013, pp. 4525–4532. 1

[8] M. Ruan, D. M^{c}Conachie, and D. Berenson, "Accounting for Directional Rigidity and Constraints in Control for Manipulation of Deformable Objects without Physical Simulation," in *IEEE/RSJ Int. Conf. Intell. Robot. Sys. (IROS)*, 2018, pp. 512–519. 1

[9] Y. Li, H. He, J. Wu, D. Katabi, and A. Torralba, "Learning Compositional Koopman Operators for Model-Based Control," in *Int. Conf. Learn. Represent.*, 2019. 1

[10] K. Zhang, B. Li, K. Hauser, and Y. Li, "AdaptiGraph: Material-Adaptive Graph-Based Neural Dynamics for Robotic Manipulation," in *Robot. Sci. Syst. (RSS)*, 2024. 1, 2

[11] A. Longhini, M. Büsching, B. P. Duisterhof, J. Lundell, J. Ichnowski, M. Björkman, and D. Kragic, "Cloth-Splatting: 3D Cloth State Estimation from RGB Supervision," in *Int. Conf. Robot Learn. (CoRL)*, 2024. 1

[12] F. Gu, H. Sang, Y. Zhou, J. Ma, R. Jiang, Z. Wang, and B. He, "Learning Graph Dynamics with Interaction Effects Propagation for Deformable Linear Objects Shape Control," *IEEE Trans. Autom. Sci. Eng.*, pp. 1–12, 2025. 1

[13] Z. Xu, J. Wu, A. Zeng, J. B. Tenenbaum, and S. Song, "DensePhysNet: Learning Dense Physical Object Representations via Multi-Step Dynamic Interactions," in *Robot. Sci. Syst. (RSS)*, 2019. 1

[14] B. Bianchini, M. Halm, and M. Posa, "Simultaneous Learning of Contact and Continuous Dynamics," in *Int. Conf. Robot Learn. (CoRL)*, 2023. 1

[15] B. Lloyd, G. Székely, and M. Harders, "Identification of Spring Parameters for Deformable Object Simulation," *IEEE Trans. Vis. Comput. Graphics*, vol. 13, no. 5, pp. 1081–1094, 2007. 1

[16] M. Macklin, M. Müller, N. Chentanez, and T.-Y. Kim, "Unified Particle Physics for Real-Time Applications," *ACM Trans. Graph.*, vol. 33, no. 4, Jul. 2014. 1

[17] J. Bender, M. Müller, and M. Macklin, "Position-Based Simulation Methods in Computer Graphics." in *Eurographics (Tutorials)*, 2015, pp. 1–32. 1

[18] H. Yin, A. Varava, and D. Kragic, "Modeling, Learning, Perception, and Control Methods for Deformable Object Manipulation," in *Sci. Rob.*, vol. 6, May 2021, pp. 1–16. 1

[19] M. Moll and L. E. Kavraki, "Path Planning for Deformable Linear Objects," *IEEE Trans. Robot.*, vol. 22, no. 4, pp. 625–636, 2006. 1

[20] M. Yu, K. Lv, C. Wang, M. Tomizuka, and X. Li, "A Coarse-to-Fine Framework for Dual-Arm Manipulation of Deformable Linear Objects with Whole-Body Obstacle Avoidance," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2023. 1

[21] M. Yu, H. Zhong, and X. Li, "Shape Control of Deformable Linear Objects with Offline and Online Learning of Local Linear Deformation Models," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 1337–1343. 1

[22] M. Yu, K. Lv, C. Wang, Y. Jiang, M. Tomizuka, and X. Li, "Generalizable Whole-Body Global Manipulation of Deformable Linear Objects by Dual-Arm Robot in 3-D Constrained Environments," *Int. J. Robot. Res.*, vol. 0, 2023. 1

[23] H. Jiang, H.-Y. Hsu, K. Zhang, H.-N. Yu, S. Wang, and Y. Li, "PhysTwin: Physics-Informed Reconstruction and Simulation of Deformable Objects from Videos," *arXiv preprint arXiv:2503.17973*, 2025. 1

[24] J. Abou-Chakra, L. Sun, K. Rana, B. May, K. Schmeckpeper, M. V. Minniti, and L. Herlant, "Real-is-Sim: Bridging the Sim-to-Real Gap with a Dynamic Digital Twin for Real-World Robot Policy Evaluation," *arXiv preprint arXiv:2504.03597*, 2025. 1

[25] J. Schulman, A. Lee, J. Ho, and P. Abbeel, "Tracking Deformable Objects with Point Clouds," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2013, pp. 1130–1137. 1

[26] T. Tang, Y. Fan, H.-C. Lin, and M. Tomizuka, "State Estimation for Deformable Objects by Point Registration and Dynamic Simulation," in *IEEE/RSJ Int. Conf. Intell. Robot. Sys. (IROS)*, 2017, pp. 2427–2433. 1

[27] J. Abou-Chakra, K. Rana, F. Dayoub, and N. Suenderhauf, "Physically Embodied Gaussian Splatting: A Visually Learnt and Physically Grounded 3D Representation for Robotics," in *Int. Conf. Robot Learn. (CoRL)*, 2024. 1

[28] M. Zhang, K. Zhang, and Y. Li, "Dynamic 3D Gaussian Tracking for Graph-Based Neural Dynamics Modeling," in *Int. Conf. Robot Learn. (CoRL)*, 2024. 1

[29] P. Mitrano, D. McConachie, and D. Berenson, "Learning Where to Trust Unreliable Models in an Unstructured World for Deformable Object Manipulation," *Sci. Rob.*, vol. 6, no. 54, pp. 1–12, 2021. 1

[30] J. Huang, X. Chu, X. Ma, and K. W. S. Au, "Deformable Object Manipulation with Constraints using Path Set Planning and Tracking," *IEEE Trans. Robot.*, vol. 39, no. 6, pp. 4671–4690, 2023. 1

[31] Y. Hu, T.-M. Li, L. Anderson, J. Ragan-Kelley, and F. Durand, "Taichi: A Language for High-Performance Computation on Spatially Sparse Data Structures," *ACM Trans. Graph.*, vol. 38, no. 6, p. 201, 2019. 1

[32] Y. Hu, L. Anderson, T.-M. Li, Q. Sun, N. Carr, J. Ragan-Kelley, and F. Durand, "DiffTaichi: Differentiable Programming for Physical Simulation," *Int. Conf. Learn. Represent.*, 2020. 1

[33] M. Macklin, "Warp: A High-performance Python Framework for GPU Simulation and Graphics," March 2022, nVIDIA GPU Technology Conference (GTC). 1

[34] M. Zahid and F. T. Pokorny, "CloudGripper: An Open Source Cloud Robotics Testbed for Robotic Manipulation Research, Benchmarking and Data Collection at Scale," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2024, pp. 12 076–12 082. 2

[35] R. Hoque, K. Shivakumar, S. Aeron, G. Deza, A. Ganapathi, A. Wong, J. Lee, A. Zeng, V. Vanhoucke, and K. Goldberg, "Learning to Fold Real Garments with One Arm: A Case Study in Cloud-Based Robotics Research," in *IEEE/RSJ Int. Conf. Intell. Robot. Sys. (IROS)*, 2022, pp. 251–257. 2

[36] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu *et al.*, "RT-1: Robotics Transformer for Real-World Control at Scale," *arXiv preprint arXiv:2212.06817*, 2022. 2

[37] S. Jin, R. Wang, M. Zahid, and F. T. Pokorny, "How Physics and Background Attributes Impact Video Transformers in Robotic Manipulation: A Case Study on Planar Pushing," in *IEEE/RSJ Int. Conf. Intell. Robot. Sys. (IROS)*, 2024, pp. 7391–7398. 2

[38] M. Ahn, D. Dwibedi, C. Finn, M. G. Arenas, K. Gopalakrishnan, K. Hausman, B. Ichter, A. Irpan, N. Joshi, R. Julian, S. Kirmani, I. Leal, E. Lee, S. Levine, Y. Lu, I. Leal, S. Maddineni, K. Rao, D. Sadigh, P. Sanketi, P. Sermanet, Q. Vuong, S. Welker, F. Xia, T. Xiao, P. Xu, S. Xu, and Z. Xu, "AutoRT: Embodied Foundation Models for Large Scale Orchestration of Robotic Agents," *arXiv preprint arXiv:2401.12963*, 2024. 2

[39] A. Keipour, M. Bandari, and S. Schaal, "Efficient Spatial Representation and Routing of Deformable One-Dimensional Objects for Manipulation," *IEEE/RSJ Int. Conf. Intell. Robot. Sys. (IROS)*, pp. 211–216, 2022. 2, 3, 4

[40] J. Xiang, H. Dinkel, H. Zhao, N. Gao, B. Coltin, T. Smith, and T. Bretl, "TrackDLO: Tracking Deformable Linear Objects Under Occlusion With Motion Coherence," *IEEE Robot. Autom. Lett.*, vol. 8, no. 10, pp. 6179–6186, 2023. 2, 3, 4

[41] Y. Wang, Y. Li, K. Driggs-Campbell, L. Fei-Fei, and J. Wu, "Dynamic-Resolution Model Learning for Object Pile Manipulation," in *Robot. Sci. Syst. (RSS)*, 2023. 2

[42] M. Zhang, K. Zhang, and Y. Li, "Dynamic 3D Gaussian Tracking for Graph-Based Neural Dynamics Modeling," in *Int. Conf. Robot Learn. (CoRL)*, 2024. 2

[43] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson *et al.*, "SAM 2: Segment Anything in Images and Videos," *arXiv preprint arXiv:2408.00714*, 2024. 4

[44] L. Yang, B. Kang, Z. Huang, X. Xu, J. Feng, and H. Zhao, "Depth Anything: Unleashing the Power of Large-Scale Unlabeled Data," in *IEEE/CVF Int. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2024. 4

[45] L. Yang, B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao, "Depth Anything V2," in *arXiv: 2406.09414*, 2024. 4

[46] B. Wen, M. Trepte, J. Aribido, J. Kautz, O. Gallo, and S. Birchfield, "FoundationStereo: Zero-Shot Stereo Matching," in *arXiv: 2501.09898*, 2025. 4

[47] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000. 5

[48] C. R. Harris, J. Millman, S. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. Fernández del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array Programming with NumPy," *Nature*, vol. 585, pp. 357–362, 2020. 5

[49] J. D. Hunter, "Matplotlib: A 2D Graphics Environment," *Comput. Sci. Eng.*, vol. 9, no. 3, pp. 90–95, 2007. 5

[50] P. Virtan, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright *et al.*, "Scipy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nat. Methods*, vol. 17, pp. 261—-272, 2020. 5

[51] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," *Adv. Neur. Inf. Proc. (NeurIPS)*, vol. 32, 2019. 5