Robotic Manipulation of Deformable Linear Objects via Multiview Model-Based Visual Tracking

Alessio Caporali and Gianluca Palli, *Senior Member, IEEE*

***Abstract*—The manipulation of deformable linear objects (DLOs)—such as electrical cables, wires, and hoses—is common in various industrial processes. Currently, this task is typically performed by human operators, making it labor-intensive, inefficient, and prone to errors. Consequently, developing robotic solutions for DLOs manipulation is crucial for improving both industrial efficiency and quality. This article addresses the challenge of tracking the state of DLOs during robotic manipulation, a key requirement for achieving accurate and reliable control. To this end, we propose a novel model-based multiview visual tracking algorithm. The algorithm integrates a predictive model of DLO behavior based on the Cosserat rod formulation and employs an NN-based approximation to enable efficient evaluation of DLO shapes. By guiding visual perception with the predictive model, the algorithm effectively manages occlusions and estimates the 3-D shape of the manipulated DLO in cluttered environments. This is accomplished by triangulating simple 2-D images, enabling seamless integration into existing robotic systems without the need for costly and often unreliable 3-D sensors. The proposed method is evaluated in both simulated and real-world scenarios, demonstrating its effectiveness in reliably tracking thin DLOs in 3-D environments.**

***Index Terms*—Deformable linear objects (DLOs), multiview triangulation, robotic manipulation, visual tracking.**

I. Introduction

Electrical cables, wires, and hoses are examples of deformable linear objects (DLOs), a class of deformable objects characterized by a circular cross-section and a length significantly greater than their radius [1].

The manipulation of electrical cables and wires is a common and essential task in various manufacturing and assembly processes, particularly in industries such as automotive and aerospace. In these sectors, the assembly and routing of wires and wire harnesses remain critical tasks that are still predominantly performed manually [2]. Similarly, in switchgear assembly, routing cables within cabinets is labor-intensive and prone to errors, relying heavily on manual labor [3].

Automating these tasks presents significant challenges due to the deformable nature of DLOs and their typically small sizes. Indeed, numerous challenges arise when addressing the manipulation of DLOs, including the need for accurate perception and tracking of the DLO's shape, handling occlusions, and controlling the DLO's shape during manipulation [2], [4], [5].

Researchers have recently proposed several solutions to enhance the robotic perception and manipulation of DLOs. Learning-based models for manipulating DLOs have gained popularity, as they can provide accurate real-time predictions of DLO behavior while addressing the computational efficiency limitations of analytical models [6], [7]. Regarding the perception of DLOs, the literature includes several methods for estimating their shape and position, such as deep segmentation networks [8] and stereo-based approaches [9]. Point cloud-based methods have also been developed for tracking DLOs [10], [11], [12], often incorporating learning algorithms to enhance performance [13], [14].

However, existing methods exhibit several limitations when applied to real-world DLO manipulation tasks [3]. Stereo-based perception methods are often limited to static scenes, while tracking algorithms typically assume that the scene's point cloud is already segmented. Moreover, the tracking performance can degrade significantly in the presence of long-term occlusions. Similarly, shape estimation techniques focus on detecting and segmenting DLOs in individual frames without considering temporal continuity across frames. Furthermore, manipulation approaches are generally tested in controlled environments, often within the shape control framework, and tend to overlook the challenges posed by occlusions, which are common in real-world scenarios.

In this work, illustrated in Fig. 1, we propose a novel approach for the visual tracking of DLOs during robotic manipulation, leveraging a multiview, model-based strategy.

Received 11 September 2024; revised 22 January 2025 and 11 March 2025; accepted 16 April 2025. Date of publication 8 May 2025; date of current version 17 October 2025. Recommended by Technical Editor K. Bai and Senior Editor H. Wang. This work was supported by the SiMOD project —Multi-robot Mobile Systems for Deformable Objects Manipulation—PR FESR 21-27 Action 1.1.2—Call for strategic industrial research projects aimed at the priority areas of the Smart Specialization Strategy—CUP J47G22000830003. The work of Alessio Caporali was supported by FSE+ 2021-2027 under a research contract per Law 240/2010, Art. 24(3)(a), and D.G.R. 693/2023 (REF. PA: 2023-20090/RER—CUP: J19J23000730002). *(Corresponding author: Alessio Caporali.)*

The authors are with the DEI—Department of Electrical, Electronic and Information Engineering, University of Bologna, 40136 Bologna, Italy (e-mail: alessio.caporali@unibo.it).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TMECH.2025.3562295.

Digital Object Identifier 10.1109/TMECH.2025.3562295

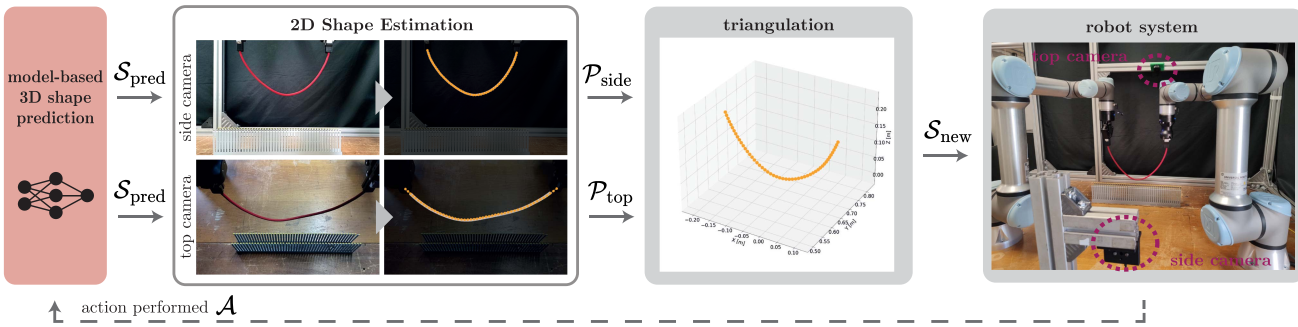

Fig. 1. Overview of the proposed multiview model-based tracking approach. While the figure illustrates a two-camera setup, the method is scalable to include additional cameras, further minimizing the likelihood of occlusions.

Our method estimates and tracks the 3-D shape of the manipulated DLO using simple 2-D images. This is achieved through a multiview triangulation approach, where the DLO's 3-D shape is estimated from each camera viewpoint and then fused. This sensing method offers significant advantages for DLO perception compared to current 3-D sensing technologies, providing greater flexibility in camera positioning, field of view, resolution, and frame rate [15].

The proposed approach utilizes a model of DLO behavior built on Cosserat rod theory, which predicts the DLO's shape changes given its initial state and the actions of the robots. This predictive model, based on a fast neural network (NN) approximation of the Cosserat rod model, enables real-time predictions of the DLO's shape, overcoming the computational inefficiency of the analytical model. In each tracking iteration, the predictive model guides the vision system, simplifying the perception of the current DLO shape and providing approximated shapes for occluded areas not visible to the cameras.

The proposed method, evaluated both in simulated and real-world scenarios, is capable of handling occlusions and can effectively track a manipulated DLO in dynamic settings. In summary, the main contributions of this work are as follows.

1) The adoption of a model-based approach for tracking, which facilitates the handling of occlusions and provides an initial estimate of the DLO state.
2) A method for estimating the 2-D shape of DLOs in cluttered images starting from the predicted DLO state.
3) The use of a multiview triangulation approach to estimate the 3-D shape of DLOs from 2-D images in dynamic scenes, eliminating the need for unreliable, bulky, and expensive 3-D sensors.
4) Evaluation in both simulated and real-world scenarios, demonstrating the method's effectiveness in tracking thin DLOs in cluttered, real-world environments.

The implementation of the proposed method is available on https://github.com/lar-unibo/DLO_MultiView_Tracking.

The rest of this article is organized as follows. Section II reviews related works, while Section III presents the problem formulation. Sections IV and V describe the DLO predictive model and vision perception modules, respectively. Section VI presents the experimental results. Finally, Section VII concludes this article.

II. Related Works

The proposed method is closely related to several research areas at the intersection of robotics and DLOs. In this section, we provide an overview of the most relevant literature on DLOs, focusing on three key topics: 3-D shape estimation, tracking, and modeling for robotic manipulation.

A. 3-D Shape Estimation

For robotic manipulation tasks, the accurate estimation of the 3-D shape of DLOs is crucial. Direct 3-D shape estimation of DLOs is less explored compared to general shape estimation. According to [15], a possible reason is that only certain high-end cameras can reliably detect DLOs. These cameras are large, have limited resolution, and a small field of view, making them hard to integrate into robotic systems. Consequently, research has predominantly focused on 2-D shape estimation, treating 3-D shape estimation as a secondary task, e.g., by leveraging depth data to transform the estimated 2-D shape into Cartesian space [16]. However, this approach is applicable only in specific restrictive scenarios, such as when the DLO diameter is sufficiently large, otherwise the depth data may not be reliable [15]. To address potential sensor noise in the depth measurements across the DLO shape, a smoothness strategy is implemented in [17], utilizing a discrete elastic rod model of the DLO [18].

The reliable 3-D detection of DLOs is also investigated in [9] exploiting a multiview stereo approach. In particular, a simple 2-D camera is used to capture images of the DLO from different stereo-like viewpoints. Subsequently, the 3-D shape of the DLO is reconstructed by combining the 2-D images with a triangulation procedure. Although effective in reconstructing the 3-D shape of a DLO for grasping and manipulation, the method is limited to static, occlusion-free scenes.

In this work, we employ a similar triangulation procedure, enhanced by a physics-based DLO model that accounts for the DLO's deformability and occlusions, allowing the method to function in dynamic and occluded environments.

B. Tracking

Tracking involves estimating the 3-D position of a DLO across frames while accounting for occlusions and physical constraints. DLO tracking methods can be broadly classified into registration-based and learning-based approaches.

1) Registration-Based Methods: A common approach treats DLO tracking as a point-set registration task by leveraging nonrigid registration algorithms such as coherent point drift (CPD) [19] and global-local topology preservation (GLTP) [20]. In the CPD algorithm, the alignment of two point sets is treated as a probability density estimation problem. One point set corresponds to the Gaussian mixture model (GMM) centroids, typically representing the current estimates of key nodes in the DLO, while the other set represents the data points, usually comprising all the points in the current unregistered camera frame. An essential aspect of the CPD algorithm is enforcing the coherent movement of GMM centroids as a collective entity, known as motion coherence theory [21]. The GLTP algorithm builds on top of the CPD method, combining CPD's global regularization with a local regularization.

Both CPD and GLPT methods neglect physical constraints and object knowledge. Consequently, research in DLO tracking has focused on integrating these algorithms with physics-based simulators or regularization techniques to improve performance. Methods such as [11], [22], [23] introduce a specific DLO-based regularization technique on top of the nonrigid registration pipeline, the authors in [11] and [22] leveraging a simulator, while Wang and Yamakawa [23] employed a finite element method model. Alternatively, constraints such as stretching [24] or geometric considerations [10], [12] can be applied.

2) Learning-Based Methods: Recently, data-driven approaches have gained traction in DLO tracking. In [14], a particle filter within a learned, lower dimensional latent space is proposed for DLO tracking. This learning-based approach enables tracking in a reduced-dimensional state space by leveraging implicit knowledge of physically plausible DLO states. In [13], a PointNet++ encoder extracts feature points from the input point cloud, addressing potential occlusions. A fusion approach combining regression and voting branches is then applied, along with a CPD-based registration method. While the work in [13] is not strictly a tracking method (since it processes frames independently), it can handle occlusions, unlike the 3-D shape estimation methods discussed in Section II-A.

Importantly, both approaches do not require the initial DLO state, a physical simulator, or regularization during deployment, unlike registration-based methods. Instead, they incorporate a physical prior of the DLO's behavior into their training data. In [14], the model is trained on synthetic data with specific DLO properties, leading to performance drops when real DLOs deviate from the training data. On the other hand, Lv et al. [13] used domain randomization, varying DLO attributes like length, radius, and stiffness during training to bridge the gap between simulation and real-world scenarios.

3) Limitations of Existing Tracking Methods: Existing tracking methods have several limitations. First, they often require high-quality depth or point cloud data, which is not always available in practice due to the typically small size of DLOs [15]. Second, most methods rely on an initial tracked state, which may not always be feasible to have. This limitation is closely related to the third, which is the need for pre-segmented data—the point cloud must contain only the DLO being tracked. Achieving this is particularly challenging outside controlled laboratory environments.

On the contrary, the proposed method avoids these limitations by exploiting 2-D images and a physics-based DLO model to estimate the 3-D shape of a tracked DLO in realistic cluttered scenarios.

C. DLO Modeling for Robotic Manipulation

The intrinsic deformability of DLOs leads to a high-dimensional state space with complex and nonlinear dynamics [18]. Therefore, modeling the DLO's behavior has been a key focus in the literature, intending to enable more effective manipulation strategies for a robotic system.

Different physical models of DLOs have been proposed in the past [25], ranging from simpler mass-spring to more accurate but computationally demanding elastic-rod [26] or dynamic splines [27]. Alternatively, in the context of position-based dynamic simulation, a differentiable model of DLOs has been proposed in [28]. However, the application of these models in robotic systems is usually limited due to the requirements of real-time predictions. Learning-based methods are starting to become popular to address the computation efficiency, where the DLO model is learned from data, as in [6],[7], [29], [30], [31]. A similar trend is also observed in the field of continuum robots, where learning approaches are employed to model the robot's behavior [32]. For example, recent work combines physics-informed NNs with Cosserat rod theory to capture soft robots' complex behavior [33].

In this article, we exploit a similar strategy to [6] by learning a fast predictive model of the DLO behavior. As opposed to the mass-spring planar model of [6], the proposed model exploits the 3-D state of the DLO and it is based on Cosserat rod theory, providing a more accurate estimate of the DLO state.

III. Problem Formulation

This work addresses the challenge of reliably tracking DLOs, such as ropes or cables, during manipulation tasks with potential occlusions from obstacles or the robots. The approach assumes a dual-arm robotic system manipulates the DLO along a precomputed trajectory, a common setup in DLO tasks, but can be readily extended to single-arm systems.

Therefore, the primary goal is to ensure accurate tracking of the DLO, despite potential occlusions and deformations, in complex environments where perception is inherently challenging. Specific to the perception problem, straightforward segmentation of the DLO from the background is infeasible due to its thin, flexible nature and cluttered surroundings. To address this, the proposed tracking system leverages a multiview visual approach combined with a predictive model of the DLO's behavior.

The visual tracking task addressed in this article is shown in Fig. 1 and formalized in Algorithm 1. The DLO's initial state, $\mathcal{S}_{\text{in}}$, is provided as input. In particular, $\mathcal{S}_{\text{in}}$ can be obtained from the DLO model by considering the robots' end-effector configurations. The algorithm then iterates over the task, where the robots' most recent action is denoted as $\mathcal{A}$. A predictive model

Algorithm 1: Multi-View Visual Tracking Algorithm.

Input: initial DLO state $\mathcal{S}_{\text{in}}$
Output: online updated DLO state $\mathcal{S}_{\text{new}}$

```
while task is true do
    A ← GetRobotsAction()
    S_pred ← DloPredictiveModel(S_in, A)
    P_side, P_top ← MultiCamerasEstimates(S_pred)
    S_new ← TriangulateViews(P_side, P_top)
    S_in ← S_new
```

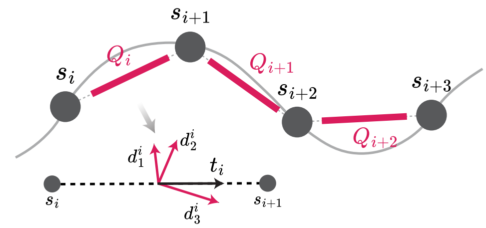

Fig. 2. Discrete Cosserat rod model with vertices s_i and material frames Q_i.

of the DLO behavior, referred to as `DloPredictiveModel`, is employed to estimate the DLO's state $\mathcal{S}_{\text{pred}}$ based on $\mathcal{S}_{\text{in}}$ and $\mathcal{A}$. The actual shape of the DLO is then obtained from each view by the multiview visual system, providing the 2-D shapes $\mathcal{P}_{\text{side}}$ and $\mathcal{P}_{\text{top}}$ (to clarify, in this work a *side-view* and a *top-view* camera are assumed). A triangulation algorithm, `TriangulateViews`, is then applied to obtain the 3-D shape $\mathcal{S}_{\text{new}}$ of the DLO. Finally, the DLO's initial state, $\mathcal{S}_{\text{in}}$, is updated to $\mathcal{S}_{\text{new}}$, and the process is repeated for the next iteration.

Importantly, the proposed method is designed to handle dynamic and occluded scenarios: the DLO's is actively manipulated, the DLO shape may not be fully observable, and obstacle locations are not known and may move. To address this, the DLO's behavior is modeled using Cosserat rod theory and approximated with a fast NN model, which guides the vision system and effectively manages occlusions (see Section IV for details). The vision system uses simple 2-D cameras, selected over a single 3-D camera due to the latter's limitations in detecting thin objects and its restricted field of view and working space. In contrast, the multiview visual approach provides a more comprehensive and reliable solution for such scenarios. Furthermore, the cameras can be positioned to capture the scene from diverse viewpoints, effectively mitigating potential occlusions. The vision system is detailed in Section V.

IV. DLO Model

The DLO model employed in this work is derived from an elastic rod formulation, which describes the DLO as a thin, flexible, and extensible rod. Specifically, Cosserat rod theory (see Section IV-A) is exploited, providing a mathematical framework that captures all six degrees of freedom of the rod, including bending, twisting, stretching, and shearing [26]. While the Cosserat rod model is accurate and realistic, it is computationally intensive, making it unsuitable for online robotic applications. To overcome this, an NN (see Section IV-B) is trained to approximate the predictions of the (analytical) Cosserat rod model.

A. Analytical Model: Cosserat Rod

For DLO modeling, the Cosserat rod model is favored over simpler models like mass-spring or multibody systems because it offers a more precise and realistic representation of the DLO's behavior [26].

A Cosserat rod is described by its centerline $s(z, t)$ (where z is the arc length of the rod and t is time) and a material frame. The material frame is defined by the versors d_1, d_2, and d_3, which are mutually orthogonal and form a right-handed system. The versors d_1 and d_2 span the cross-section of the rod, while d_3 is the normal vector to the cross-section. The versors are defined in the material frame and are functions of the arc length z and time t. Due to the rod's flexibility, the cross-section of the rod is not necessarily circular, and d_3 is not necessarily aligned with the tangent vector of the centerline.

The dynamics of the Cosserat rod are governed by the conservation laws of linear and angular momentum applied at each cross-section. These governing equations are solved numerically to track the deformation of the rod over time [26].

Therefore, the rod is discretized into a series of centerline vertices $s_i(t)$, with $i = 1 \ldots n$, connected by line segments that correspond to the reference material frames $Q_i(t)$, as depicted in Fig. 2. The full state of the system at any given time is defined by the positions of these vertices $s_i(t)$ and the orientations of the material frames $Q_i(t)$. The numerical solution of the conservation laws then yields the time evolution of the rod's deformation.

The robot's manipulation actions are applied to the rod's extremities. Each extremity is associated with a pose vector, defined by the vertex position s and the material frame Q. The action is described as a displacement and a rotation applied to the current poses of the extremities. The action set is represented as $\mathcal{A} = \{a_1, a_n\}$, where a_1 and a_n refer to the action on the first and last DLO ends. Considering for reference the action $a_1 \in \mathbb{R}^7$, it is defined as $a_1 = [\delta_x, \delta_y, \delta_z, q_x, q_y, q_z, q_w]^\top$, where δ_x, δ_y, and δ_z are the linear displacements applied to vertex s_1 and q_x, q_y, q_z, and q_w are the quaternion components representing the rotation applied to the material frame Q_1. A similar definition holds for a_n.

B. NN Model: Cosserat Rod Approximation

An NN approximates the Cosserat rod model, offering a computationally efficient predictive model. Indeed, trained on a dataset of DLO movements generated offline by the analytical model (see Section IV-B2), the compact NN architecture (see Section IV-B1) achieves small inference times, several orders of magnitude faster than the analytical model (see Section IV-A). To simplify the learning process, the DLO state is reduced and represented as a sequence of 3-D points, each corresponding to a vertex s_i of the rod discretization $\mathcal{S} = \{s_1, \ldots s_n\}$. The NN is trained to predict state changes caused by a given action $\mathcal{A}$.

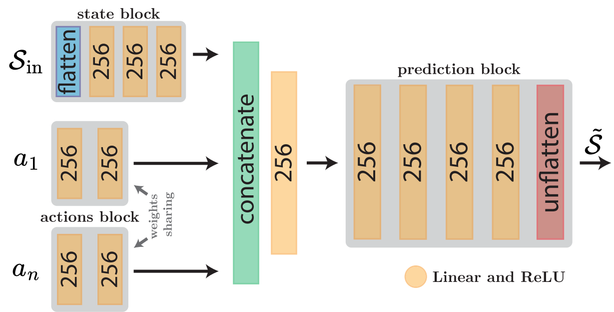

Fig. 3. Neural network architecture.

1) Neural Network Architecture: The architecture of the employed NN is detailed in Fig. 3. It is derived from [6] with several modifications to accommodate the 3-D nature of the DLO state and the differences in action parameters. The network consists of a series of linear layers, each followed by a ReLU activation function. The architecture is organized into three main blocks: the state block, the actions block, and the prediction block. The network takes as input the initial state of the DLO $\mathcal{S}_{\text{in}}$ and the action set $\mathcal{A}$. The state block embeds the DLO state into a latent space by first flattening the 3-D points and then passing them through a series of linear layers. Similarly, the actions block processes the actions using a single set of linear layers for both actions, i.e., by sharing the layer weights. The embedded state and actions are then concatenated and passed through a series of linear layers to predict the changes in the DLO state. The network's output, denoted as $\tilde{\mathcal{S}}$, represents the predicted changes in the 3-D coordinates of the DLO from the initial state. Predicting these changes, rather than the final state, enables the network to learn the DLO's behavior more effectively, as also discussed in [6]. The final DLO state $\mathcal{S}_{\text{pred}}$ is then obtained by adding $\tilde{\mathcal{S}}$ to $\mathcal{S}_{\text{in}}$, i.e.:

$$\mathcal{S}_{\text{pred}} = \texttt{DloPredictiveModel}(\mathcal{S}_{\text{in}}, \mathcal{A}) = \tilde{\mathcal{S}} + \mathcal{S}_{\text{in}}.$$

The network is trained to minimize the mean squared error between the predicted $\mathcal{S}_{\text{pred}}$ and the expected $\mathcal{S}_{\text{out}}$ final states.

2) Dataset Generation: The dataset is generated by simulating the analytical DLO model described in Section IV-A, subjected to a series of random actions.

To generate a dataset spanning DLOs with varying configurations, first the simulated DLO is initialized in a random configuration. Then, a set of k_1 actions is sampled, and the DLO's behavior is simulated as these actions are applied sequentially. Each action is discretized into k_2 steps so to record a smooth transition of states between the initial and final configurations. This approach ensures the creation of a diverse dataset, capturing a wide range of DLO configurations with varying levels of complexity and range of motions. After each action, a dataset sample is saved, being described by the rollout trajectory of the DLO and the applied action. As an example, Fig. 4 shows four samples of the generated dataset, illustrating the DLO's initial, intermediate, and final states.

To learn an invariant NN model in terms of DLO length, the dataset is generated by considering DLOs with different lengths ranging within a given interval. Furthermore, the initial and final configurations of the DLO are mean-centered to enhance the training efficiency and generalization capabilities of the NN model. In addition, to account for potential noise in real-world DLO estimations, Gaussian noise is added to the initial state during training.

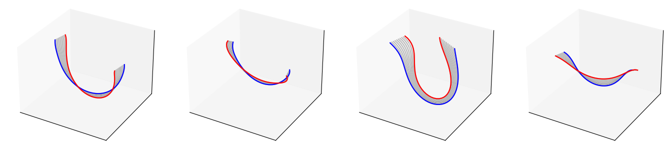

Fig. 4. Example dataset samples generated by the Cosserat rod model, showing the initial, intermediate, and final states in blue, black, and red, respectively.

V. Multiview Visual Perception

The vision system employed in this work is based on two 2-D cameras, a *side-view* and a *top-view* camera, which provide images of the manipulated DLO from different perspectives. First, the vision system estimates the DLO's 2-D shape from the images captured by the cameras (see Section V-A). Then, the detected 2-D shapes are triangulated to reconstruct the 3-D shape of the DLO (see Section V-B). Due to the triangulation approach, the system can be easily extended to incorporate additional input cameras.

A. 2-D Shape Estimation

The 2-D shape estimation is performed independently for each camera. The process begins with detecting the target manipulated DLO in the image, utilizing a graph-based approach inspired from *RT-DLO* [34], a deep learning-based instance segmentation method tailored for DLOs. Similar to RT-DLO, a semantic segmentation network is first employed to remove the background from the image, even in cluttered environments, generating a mask M_b. This network is trained on a mix of synthetic and real data, enabling it to generalize effectively across various scenarios [8]. Next, a specialized pipeline, different from RT-DLO, is applied to extract the target DLO. Indeed, while RT-DLO is designed to detect all DLOs in the scene without employing prior knowledge, the proposed pipeline leverages an initial guess based on the predicted DLO state $\mathcal{S}_{\text{pred}}$ provided by the NN model of Section IV-B.

The pipeline consists of the following three key steps:

1) projecting $\mathcal{S}_{\text{pred}}$ onto each camera's view obtaining $\mathcal{P}_{\text{pred}}$;
2) generating a graph representation of the DLO as observed in the scene;
3) associating the nodes of the graph with $\mathcal{P}_{\text{pred}}$.

The process is illustrated in Fig. 5 and detailed in the following sections.

1) DLO Predicted State Projection: The DLO state $\mathcal{S}_{\text{pred}}$, predicted by the NN model of Section IV-B, is projected onto the image plane of the camera, resulting in $\mathcal{P}_{\text{pred}} = \{p_1 \ldots p_n\}$ where p_i is a point representing the projection of the ith DLO node in the image. This projection serves as an initial estimate of the DLO's position in the image, thereby simplifying the subsequent segmentation task. The projection is based on the camera's

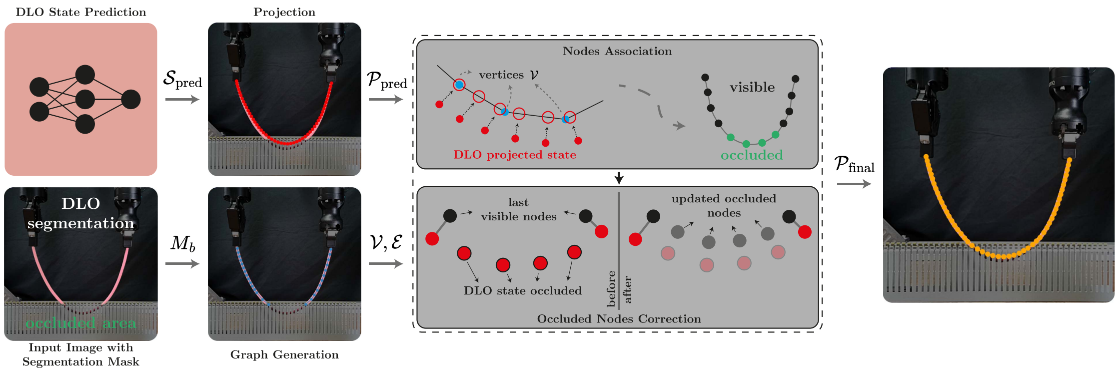

Fig. 5. DLO 2-D shape estimation pipeline leveraging the predicted DLO state $\mathcal{S}_{\text{pred}}$, as seen from the side-view camera.

intrinsic and extrinsic parameters. An occupancy binary mask is then created from $\mathcal{P}_{\text{pred}}$, with a value of 1 near the projected DLO and 0 elsewhere. This mask refines the segmentation network's output by focusing the processing only on the expected DLO area.

2) Graph Generation: From the segmentation mask M_b, a graph representation of the scene is constructed, consisting of nodes $\mathcal{V}$ and edges $\mathcal{E}$.

For the nodes, a distance transform is applied to the binary mask M_b producing M_{dist}. This transform assigns intensity values to pixels based on their distance from the nearest boundary. Following this, M_{dist} is dilated using a 3×3 kernel to create M_{dil}. Local maxima are then identified as pixels where the values in M_{dist} match those in M_{dil} [34]. The resulting mask, $M_{\max}$, identifies the maximal points in M_b and is used to determine the DLO's centerline. These maximal points are treated as a point set, from which a refined set of nodes is sampled using the farthest point sampling (FPS) algorithm [35]. The FPS algorithm iteratively selects the point farthest from the current set of nodes, ensuring a mostly uniform distribution along the DLO's centerline. The resulting set of nodes $\mathcal{V}$ is used to construct the graph.

The edges $\mathcal{E}$ of the graph are formed by establishing connections among the nodes $\mathcal{V}$ according to a policy based on distance and angle scores. This connection strategy, based on the edge sampling procedure outlined in [34], ensures that the graph maintains a coherent and connected representation.

3) Nodes Association: Given the graph representation of the scene, the next step is to associate the projected predicted DLO state $\mathcal{P}_{\text{pred}}$ with the graph nodes $\mathcal{V}$.

The association is performed by matching the projected points p_i with the nodes v_j of the graph. Importantly, the association is guided by the predicted DLO state, such that erroneous or missing nodes do not affect the matching process. Considering a point p_i, the process begins by identifying its closest edge (v_1, v_2) in the graph. The point p_i is then projected onto this edge, yielding a projected point p_i'. The projection is computed as follows:

$$p_i' = v_1 + \left(\frac{(p_i - v_1) \cdot (v_2 - v_1)}{\|v_2 - v_1\|^2} \right) (v_2 - v_1).$$

This *edge projection* ensures a more accurate association compared to using the closest vertex only. Indeed, the DLO is expected to lie along the edges of the graph, and the number and distribution of the vertices sampled with FPS may differ significantly from $\mathcal{P}_{\text{pred}}$. An illustration of the *edge projection* phase is shown in the mid-section of Fig. 5.

By repeating this process for all points p_i, the association between $\mathcal{P}_{\text{pred}}$ and $\mathcal{V}$ is established. Simultaneously, the segmentation mask M_b is used to assign a *visible* or *occluded* attribute to each point p_i in $\mathcal{P}_{\text{pred}}$ by evaluating the corresponding pixel value of p_i' in M_b.

For the *occluded* points, a reliable association is therefore not possible, and the original predicted DLO state $\mathcal{S}_{\text{pred}}$ is used as a fallback. However, a gap (or step) is usually introduced between the $\mathcal{P}_{\text{pred}}$ and the associated points, as the predicted DLO state is not perfectly aligned with the graph. To provide as output a smooth DLO state, the gap is addressed by translating the occluded points over the expected centerline of the occluded area. This translation is performed by interpolating between the *edge projection* distances, i.e., the distance between p_i and p_i', of the associated points at the extremity of the occluded areas. This occluded nodes correction process is illustrated in Fig. 5. This approach ensures that the vision system can handle occlusions effectively, as the NN model provides an estimate of the DLO's state in these cases.

Thus, the final output is a set of 2-D points $\mathcal{P}_{\text{final}}$ representing the DLO's shape in the image, where each point originates from the predicted DLO state $\mathcal{P}_{\text{pred}}$ and is modified according to the graph vertices $\mathcal{V}$. Notably, the entire state of the DLO in the image is provided, as occluded areas are addressed as outlined above.

B. Multiview Triangulation

Triangulation is the process of determining the 3-D position of a point by intersecting the rays passing through it from different points of view. In the context of the DLO, the 2-D shapes obtained from the *side-view* and *top-view* cameras are triangulated to reconstruct the 3-D shape of the DLO. While the approach is demonstrated with two cameras, it can easily accommodate additional cameras if needed. For clarity, the algorithm is initially explained for a single point triangulation (see Section V-B1), and then extended to the full DLO shape (see Section V-B2).

1) Single Point Triangulation Algorithm: Given a pixel point p in the 2-D image plane of the camera, the unit ray ν passing through the image reference frame origin and p can be expressed in the camera frame as

$$\nu' = \begin{bmatrix} p_x - c_x \\ p_y - c_y \\ f \end{bmatrix}, \quad \nu = \frac{\nu'}{\|\nu'\|}$$

where c_x and c_y are the pixel coordinates of the image center and f is the camera focal distance.

The unit ray ν can be expressed in a reference world frame by ${}^w\nu = {}^wT_c v$ where ${}^wT_c$ is the transformation matrix from the camera frame to the world frame.

Provided that m distinguished points of view are available, the estimation $\tilde{p}$ of the unknown point p can be obtained by looking for the point having the minimum distance from all the rays. By defining the symmetric matrix V_i as

$$V_i = I - {}^w\nu_i {}^w\nu_i^T \tag{1}$$

providing the seminorm on the ray distance, the point location estimate $\tilde{p}$ is provided by the nearest point search algorithm, i.e.,

$$\tilde{p} = \left(\sum_{i=1}^{m} V_i\right)^{-1} \left(\sum_{i=1}^{m} V_i \, {}^wt_{c_i}\right)$$

where ${}^wt_{c_i}$ is the translation vector of the camera i in the world frame, $i = 1, \ldots, m$.

2) Multiview Procedure: The single-point procedure is, thus, applied to each DLO node. Considering m viewpoints and n DLO points, with V_{ij} is denoted the matrix computed according to (1) for the jth point provided by the ith image. The final 3-D DLO shape is then applied by the following formula:

$$\tilde{P} = V_l^{\#} V_r$$

where $^{\#}$ represents the matrix pseudoinverse and

$$V_l = \begin{bmatrix} (\sum_{i=1}^m V_{i1}) \\ (\sum_{i=1}^m V_{i2}) \\ \vdots \\ (\sum_{i=1}^m V_{in}) \end{bmatrix}, V_r = \begin{bmatrix} (\sum_{i=1}^m V_{i1} {}^wt_{c_i}) \\ (\sum_{i=1}^m V_{i2} {}^wt_{c_i}) \\ \vdots \\ (\sum_{i=1}^m V_{in} {}^wt_{c_i}) \end{bmatrix}.$$

A critical aspect of every triangulation process is accurately associating points across different views, the so-called *correspondence problem*. Misalignments during this association can result in suboptimal triangulation. Thus, the association procedure for DLO points detailed in Section V-A3 is essential for ensuring precise matching between the views.

VI. Experiments

The experimental setup (see Fig. 1) includes two UR5e robots with two-fingered grippers (Robotiq Hand-E, 2F-85) and two static 2-D cameras (Luxonis OAK-1, 1920 × 1080 pixels) providing *side* and *top* views. Both cameras are intrinsically and extrinsically calibrated to the robot bases. ROS2 handles communication, and the algorithm, implemented in Python 3.10 with PyTorch 2.0, runs on a workstation with an Intel Core i9-9900K CPU (3.60GHz) and an NVIDIA GTX 2080 Ti GPU.

The tracking algorithm's performance is evaluated using the following metrics.

1) *Point-to-point (PTP):* Defined as the norm of the point-wise difference between the predicted and ground truth (GT) DLO states. In Cartesian space, the error is expressed in meters and denoted as PTP_{3D}. In image space, the error is normalized by the image width, expressed as a percentage, and denoted as PTP_{2D}.
2) *Occlusion ratio:* Defined as the percentage of DLO state points that are obscured in image space.
3) *Distance-to-centerline (DTC):* Defined in the image space as the shortest distance between the predicted projected state and the GT centerline in image space. The error is normalized by the image width and expressed as a percentage.

First, the DLO predictive model's approximation capabilities are assessed (see Section VI-A). Next, the tracking algorithm is evaluated in simulation and compared to a baseline (see Section VI-B), followed by real-world testing (see Section VI-C). Timings and limitations are analyzed in Sections VI-D and VI-E.

A. DLO Predictive Model Evaluation

1) Optimization Details: The NN model (see Section IV-B) is trained on a dataset of DLO manipulation samples, obtained by the analytical model (see Section IV-A), comprising about 2500 rollout trajectories. The time required for dataset generation is less than 1 h on the employed workstation by leveraging parallel processing. The DLO shape is discretized into $n = 51$ elements, and the simulation is performed with a time step of 10^{-4} s. For dataset generation, the DLO is simulated with a length varying between 0.45 and 0.6 m, while other physical parameters are held constant: the Young's modulus at 10^5 Pa, the radius at 3.0 mm, and the density at 10^3 kg/m^3. The action values are sampled from a uniform distribution, ranging from ±0.05 m for the linear components and $\pm\pi/8$ radians for the angular components. Rotations around the DLO's neutral axis are excluded from the actions, as they introduce twisting effects that are difficult to detect reliably using vision sensors. Then, each action is discretized into $k_2 = 10$ "steps" (see Fig. 4). Starting from the initial random state, $k_1 = 5$ random actions are applied in sequence. During training, each dataset sample is selected by the following:

1) loading a rollout trajectory;
2) sampling a random "step" within the trajectory;
3) sampling a random final state up to three steps ahead;
4) extracting the corresponding action.

In this way, different states and action magnitudes are considered, providing a diverse dataset for training the NN model.

The dataset is divided into training and validation sets using a 90%–10% split. The network is optimized for 20 000 steps with a batch size of 256 and a learning rate of 5×10^{-4}. The final model weights are selected based on the epoch with the lowest validation loss.

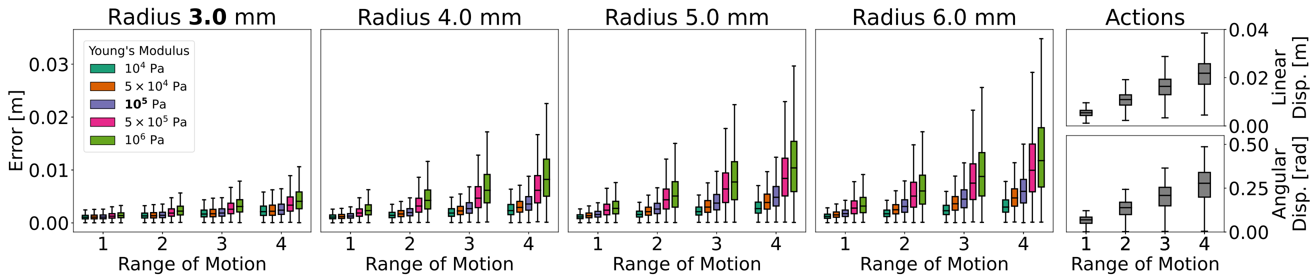

Fig. 6. Error distribution of the NN model predictions on different physical parameter values (Young's modulus and radius) across the *test set*. The reference physical parameter values are denoted in bold. The magnitude of actions is represented by motion steps (1 to 4), with corresponding linear and angular displacement characteristics shown on the right.

2) Approximation Capabilities: The performance of the learned NN DLO predictive model is assessed by comparing its predictions to a *test set* of samples generated by the analytical model. This *test set*, which was not exposed to the NN during the training process, is created using a procedure similar to that used for generating the training and validation sets. Specifically, 500 rollout trajectories are generated for each parameter set, with the Young's modulus and radius assigned values different from those specified in Section VI-A1. The Young's modulus takes on values of 10^4, 5×10^4, 10^5, 5×10^5, or 10^6 Pa, while the radius is varied to be 3 mm, 4 mm, 5 mm, or 6 mm. The main objective of this evaluation is to assess the approximation capabilities of the NN model against physical parameters not seen during training. As metric for the evaluation, the $\mathrm{PTP}_{3\mathrm{D}}$ error is used.

Fig. 6 illustrates the error distribution across test set samples for each configuration. The magnitude of applied actions is categorized into motion steps (1 to 4), with corresponding linear and angular displacements detailed on the right. Overall, the NN model effectively approximates the analytical model, even for physical parameters not encountered during training, with deviations growing as the physical parameters diverge more from the reference values.

B. Tracking Algorithm Evaluation in Simulation

To quantitatively evaluate the tracking performance of our method, we employ synthetic data generated via the analytical model of Section IV-A. RGB-D observations of the scene are rendered in Blender and stored alongside the GT DLO state. The proposed tracking algorithm is applied to this data, and its performance is evaluated using the $\mathrm{PTP}_{3\mathrm{D}}$ metric.

1) Comparison With Particle Filter-Based (PF) Tracker: The method is compared against a PF tracker proposed in [14]. The PF method is chosen for comparison due to its shared key feature with our approach: leveraging a learning-based strategy to model DLO physics. To ensure a fair comparison, it is trained on the same dataset as our method (see Section VI-A), and both trackers use the same DLO state initialization procedure.

The PF method uses a presegmented 3-D point cloud and requires prior knowledge of occluding objects via an image mask. In contrast, our approach is more versatile, requiring only two or more RGB images and eliminating the need for 3-D sensor-based segmentation or prior scene segmentation. While this article focuses on tracking, our method introduces a predictive DLO model that can be adapted for control tasks. By comparison, the PF tracker employs a simpler autoencoder to compress the DLO state. A limitation of our approach is its reliance on action information, unlike [14], though this is often available in DLO manipulation scenarios.

TABLE I
AVERAGE $\mathrm{PTP}_{3\mathrm{D}}$ ERROR DURING DLO TRACKING IN THE SIMULATED SCENARIOS (VALUES IN MM)

Method	Exp 1 Our	Exp 1 PF	Exp 2 Our	Exp 2 PF	Exp 3 Our	Exp 3 PF
No occlusion	1.09	10.30	0.95	14.74	1.21	11.39
With occlusion (0–10%)	1.28	13.81	1.22	15.26	2.37	15.42
With occlusion (10–30%)	1.98	14.41	2.76	10.77	3.84	18.07
With occlusion (30–50%)	3.17	25.10	6.07	17.31	6.77	27.57

PF is from [14].

The comparative evaluation is conducted on three experiments with increasing occlusion levels and a simulated DLO with reference physical parameters (see Section VI-A2). The quantitative and qualitative results are presented in Table I and Fig. 7, respectively. The results demonstrate that our method consistently outperforms the PF tracker, achieving a mean PTP_{3D} error of just a few millimeters, even under challenging occlusion scenarios. In contrast, the PF tracker exhibits significantly higher errors (whose values are aligned with the findings reported in [14]).

To thoroughly evaluate our method's performance during motion execution, we analyze the reprojection error (calculated using the $\mathrm{PTP}_{2\mathrm{D}}$ metric) and the occlusion ratio, as shown in Fig. 8. The $\mathrm{PTP}_{2\mathrm{D}}$ error, expressed as a percentage of the image width, stays below 1% across all experiments. These results underscore the robustness of the tracking algorithm, even in scenarios with significant occlusion.

It is important to note that occlusion is simulated for both image views to assess the worst-case scenario. However, in a real-world setup, it is unlikely that both cameras will be occluded simultaneously since quite different point of views can be employed.

2) Sensitivity to Camera Calibration Error: The utilization of the simulated environment enables a detailed evaluation of the tracking algorithm's sensitivity to camera calibration errors. The same simulation experiments of Section VI-B1 are used, with

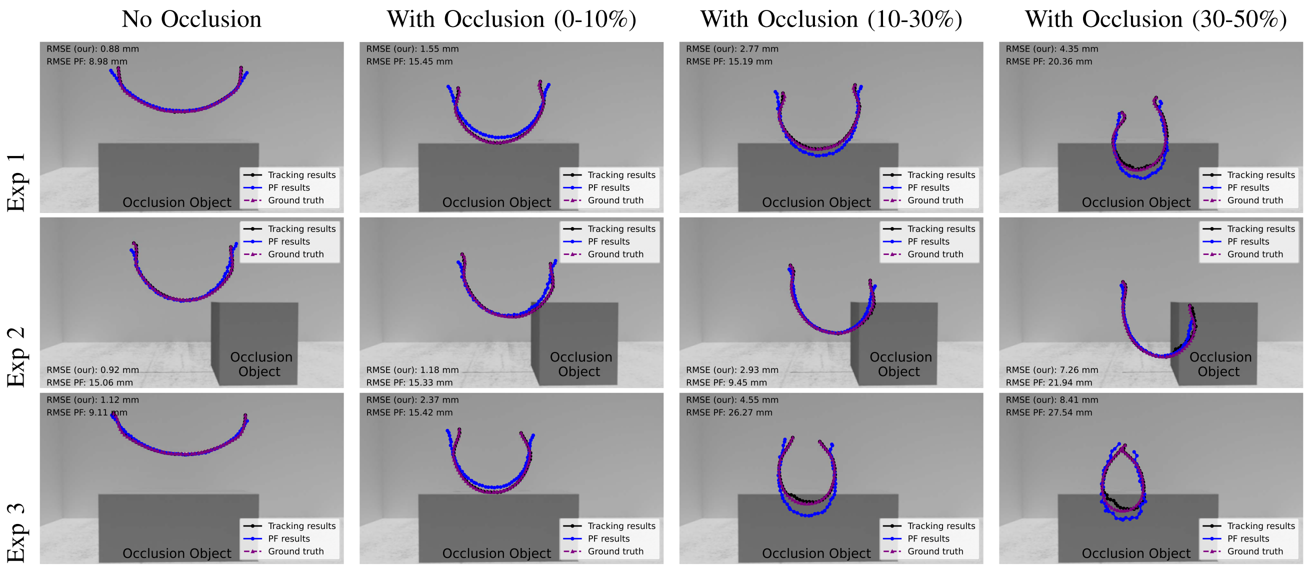

Fig. 7. Qualitative comparison of the tracking performance between our method and PF [14] in a simulated environment (Exp 1 and Exp 2 with an action magnitude of 1 "step," Exp 3 with an action magnitude of 2 "steps").

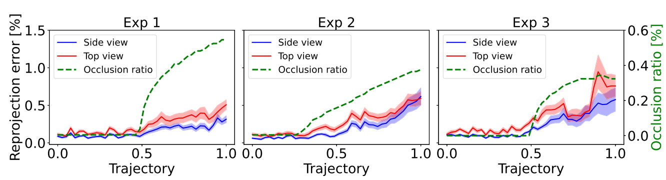

Fig. 8. Reprojection error (calculated using the PTP_{2D} metric, showing the mean and 95% confidence interval) and occlusion ratio (dashed line) during the simulation experiments.

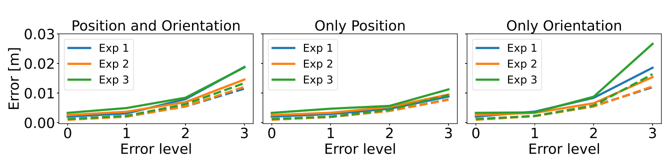

Fig. 9. Sensitivity of the tracking algorithm to camera calibration errors, with the x-axis showing perturbation severity. Dashed lines represent errors for nonoccluded frames only.

the camera extrinsic parameters perturbed by varying levels of noise. The PTP_{3D} error is computed for each noise level, and the results are presented in Fig. 9.

The perturbation levels, ranging from 0 to 3, correspond to the following offsets applied to all linear and angular components of the camera pose:

1) 0 mm/0°;
2) 1.0 mm/0.1°;
3) 2.5 mm/0.25°;
4) 5.0 mm/0.5°.

As expected, the error increases with the severity of the perturbations, with the algorithm exhibiting higher sensitivity to rotational noise. Nonetheless, the method maintains accurate tracking performance, even under calibration errors involving several millimeters and fractional degrees of misalignment. In addition, the fixed camera setup simplifies the calibration process, reducing the likelihood of such errors in practice.

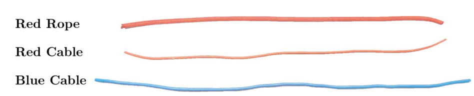

Fig. 10. Set of DLOs used in the real-world experiments.

C. Real-World Tracking Algorithm Evaluation

The tracking algorithm is evaluated in real-world DLO manipulation tasks. The experiments follow the same methodology as in Section VI-B1, with the main difference being the lack of GT 3-D data. Instead, DLO masks are manually annotated across frames, and the DLO centerline is extracted as the reference state for the visible portion via skeletonization. Tracking performance is then quantitatively assessed by computing the closest distance between each estimated projected state node and the GT centerline with the DTC metric. Notice that a quantitative evaluation is possible only for nonoccluded sections of the DLO.

Three different real-world DLOs are used: red rope (length 0.53 m, diameter 6.0 mm); blue cable (length 0.60 m, diameter 4.8 mm); and red cable (length 0.52 m, diameter 3.6 mm). The selection of these DLOs is based on their varying flexible behaviors. The red rope and red cable share similar flexibility characteristics but differ significantly in diameter and material. The blue cable is included to test the tracking algorithm with a stiffer DLO that exhibits pronounced plastic deformation. The three DLOs are illustrated in Fig. 10.

Four distinct manipulation trajectories are used (Real Exp 1 to 4). These trajectories are designed to emphasize the algorithm's ability to handle occlusions, with the DLOs undergoing varying motions and levels of occlusion. Notably, Real Exp 4 features a wrapping motion around a metal bar, introducing also a mild contact between the DLO and the object.

Two scenarios are considered: *S1*, where only the manipulated DLO is present, and *S2*, which involves a more complex setup with multiple DLOs and industrial components within the camera's field of view.

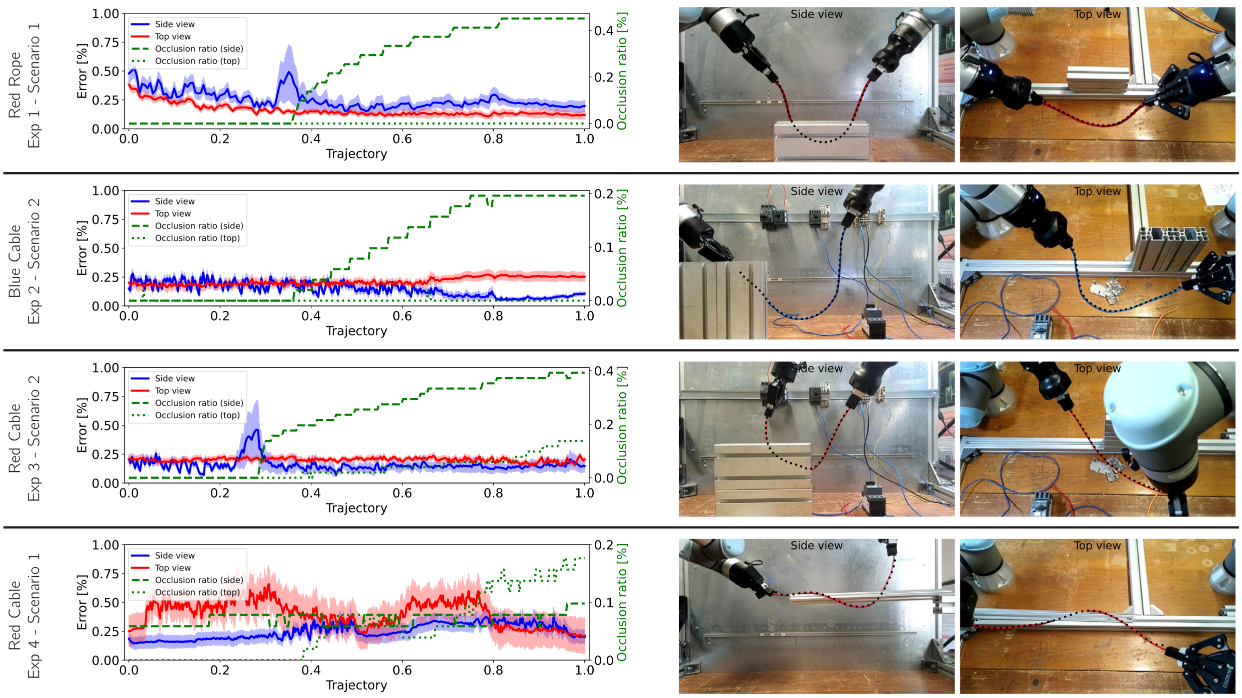

Fig. 11. Tracking performance in real-world scenarios for different experiment trajectories, DLOs, and scenarios. On the right, a snapshot of the predicted DLO state at the end of each experiment is displayed.

TABLE II
AVERAGE DTC ERROR FOR REAL-WORLD TRACKING ACROSS DLOS, TRAJECTORIES AND SCENARIOS (VALUES IN PERCENTAGE)

Exp Setup	Red Rope S1	Red Rope S2	Red Cable S1	Red Cable S2	Blue Cable S1	Blue Cable S2
Real Exp 1	0.217	0.256	0.217	0.205	0.234	0.299
Real Exp 2	0.200	0.204	0.187	0.186	0.203	0.178
Real Exp 3	0.213	0.205	0.172	0.180	0.231	0.226
Real Exp 4	0.317	0.315	0.324	0.334	0.379	0.391

1) Tracking Results: Quantitative results of the real-world experiments are provided in Table II. The DTC error is computed for each experiment, demonstrating the algorithm's ability to track the DLO state in real-world scenarios accurately. The tracking performance remains consistent across the diverse set of test DLOs and scenarios, with the average DTC error staying below 1% in all cases. This is illustrated in the plots of Fig. 11, which display the error progression during trajectory execution for various combinations of DLO, manipulation trajectory, and scenario. The figure also includes snapshots of the various trajectories and occlusion setups involved. Notably, the same NN predictive model is employed across all test DLOs without any DLO-specific fine-tuning.

2) DLO State Prediction Comparison: Additional experiments are conducted, employing the same trajectories but using a modified setup without occlusions, to emphasize the role of $\mathcal{S}_{\text{pred}}$ in extracting the DLO state from images (see Section V-A). As a baseline, we used RTDLO [34], where all DLO instances within the camera frame were extracted, and the correct instance was identified by leveraging the projected gripper poses.

The success rate metric is employed to evaluate performance. A state prediction is considered successful if it is closely aligned with the GT centerline and reaches the two grippers within a predefined tolerance of 50 pixels (or 2.5%). The results of this evaluation are presented in Fig. 12.

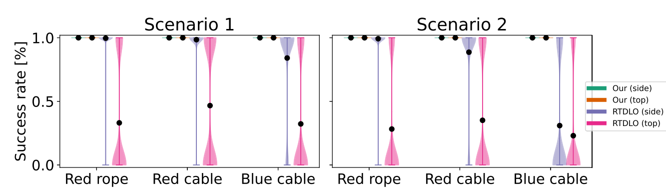

Fig. 12. Distribution of success rate for DLO state prediction between RTDLO [34] and the proposed method in real-world, nonoccluded scenarios.

The proposed method demonstrated robust performance in extracting the DLO state from both camera views, even in the more challenging S2 scenario. In contrast, RTDLO shows difficulties in some frames, particularly when using the *top* camera view, due to the more challenging perspective. Notably, the results are not sensitive to the chosen tolerance values, since RTDLO failures primarily originate from predicting multiple disconnected DLO instances for a single object, rather than minor inaccuracies near the grippers. When comparing the two methods using the DTC metric against the centerline, both maintain errors consistently below 1%.

D. Timings

The average execution times for the DLO predictive model and visual tracking components are presented in Table III. The NN responsible for predicting the DLO state operates very quickly. However, the visual tracking components, particularly

TABLE III
TIMINGS OF THE ALGORITHM'S MAIN COMPONENTS

Component		Time (ms)
DLO model	NN inference	1.06 ± 0.62
Visual tracking	Mask segmentation	11.51 ± 0.54
	Graph generation	16.22 ± 5.45
	Nodes association	0.96 ± 0.20
	Triangulation	0.99 ± 0.29

the mask segmentation and graph generation stages, are the most time-consuming procedures. Notably, these stages must be executed for each camera in the system. Mask segmentation is handled by a deep convolutional neural network deployed on the GPU, while the remaining components run on the CPU. To reduce computation time, batch inference and multiprocessing are exploited. Despite these optimizations, overhead from sources such as CPU-GPU memory transfers limits the algorithm's overall frequency to approximately 15 Hz when using two cameras.

E. Limitations

The method has limitations, primarily due to simplifications in the NN predictive model. In situations involving occlusions, collisions, or complex interactions with the environment, the model may struggle to accurately predict the DLO's state. In addition, the NN predictive model exploits only a subset of the full state provided by the Cosserat rod model. Without visual feedback, these inaccuracies cannot be corrected, raising the risk of failures in complex manipulation tasks.

To address these issues, a parameter-aware predictive model could be introduced to estimate the optimal parameters for the manipulated DLO, in a similar fashion to [6]. Incorporating collisions and interactions with the environment into the model formulation could further enhance prediction accuracy, building on previous works such as [31]. Concepts from physics-informed NN could be exploited to incorporate the full state of the DLO in the predictive model (e.g., expanding on [33]), ensuring a more accurate prediction of the DLO's behavior.

VII. CONCLUSION

This article presents a novel approach to the visual tracking of DLOs. The proposed method employs a predictive model of DLO behavior, powered by a fast neural network, to enhance the vision system's ability to track the DLO state, even in the presence of occlusions. The visual tracking algorithm refines the initial predicted DLO state by utilizing a multiview camera setup, where the DLO state is estimated from each viewpoint and integrated through a triangulation algorithm.

The accuracy of the DLO predictive model is evaluated against an analytical Cosserat rod model, demonstrating effective approximation capabilities. The tracking algorithm is tested in simulated and real-world scenarios involving varying levels of occlusions. To validate the method, real-world experiments are conducted with three DLOs of different material properties and dimensions. The visual tracking algorithm operates at approximately 15 Hz, enabling effective online visual feedback during DLO manipulation tasks.

Future work will focus on enhancing the DLO predictive model with physics-based knowledge, addressing collision handling, and utilizing the visual tracking algorithm for DLO shape control in complex manipulation tasks.

REFERENCES

[1] 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, pp. 688–716, 2018.

[2] J. Trommnau, J. Kühnle, J. Siegert, R. Inderka, and T. Bauernhansl, "Overview of the State of the Art in the production process of automotive wire harnesses, current research and future trends," *Procedia CIRP*, vol. 81, pp. 387–392, 2019.

[3] G. Palli, S. Pirozzi, M. Indovini, D. De Gregorio, R. Zanella, and C. Melchiorri, "Automatized switchgear wiring: An outline of the wires experiment results," in *Advances in Robotics Research: From Lab to Market: ECHORD++: Robotic Science Supporting Innovation*. Berlin, Germany: Springer, 2020, pp. 107–123.

[4] Y. Gao, Z. Chen, Y. Ling, J. Yang, Y.-H. Liu, and X. Li, "A hierarchical manipulation scheme for robotic sorting of multiwire cables with hybrid vision," *IEEE/ASME Trans. Mechatron.*, vol. 28, no. 2, pp. 860–872, Apr. 2023.

[5] W. Ma, B. Zhang, L. Han, S. Huo, H. Wang, and D. Navarro-Alarcon, "Action planning for packing long linear elastic objects into compact boxes with bimanual robotic manipulation," *IEEE/ASME Trans. Mechatron.*, vol. 28, no. 3, pp. 1718–1729, Jun. 2022.

[6] A. Caporali, P. Kicki, K. Galassi, R. Zanella, K. Walas, and G. Palli, "Deformable linear objects manipulation with online model parameters estimation," *IEEE Robot. Autom. Lett.*, vol. 9, no. 3, pp. 2598–2605, Mar. 2024.

[7] 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, Feb. 2023.

[8] A. Caporali, M. Pantano, L. Janisch, D. Regulin, G. Palli, and D. Lee, "A weakly supervised semi-automatic image labeling approach for deformable linear objects," *IEEE Robot. Autom. Lett.*, vol. 8, no. 2, pp. 1013–1020, Feb. 2023.

[9] A. Caporali, K. Galassi, and G. Palli, "Deformable linear objects 3D shape estimation and tracking from multiple 2D views," *IEEE Robot. Autom. Lett.*, vol. 8, no. 6, pp. 3852–3859, Jun. 2023.

[10] Y. Wang, D. McConachie, and D. Berenson, "Tracking partially-occluded deformable objects while enforcing geometric constraints," in *Proc. IEEE Int. Conf. Robot. Autom*, 2021, pp. 14199–14205.

[11] T. Tang and M. Tomizuka, "Track deformable objects from point clouds with structure preserved registration," *Int. J. Robot. Res.*, vol. 41, pp. 599–614, 2022.

[12] J. Xiang et al., "TrackDLO: Tracking deformable linear objects under occlusion with motion coherence," *IEEE Robot. Autom. Lett.*, vol. 8, no. 10, pp. 6179–6186, Oct. 2023.

[13] K. Lv, M. Yu, Y. Pu, X. Jiang, G. Huang, and X. Li, "Learning to estimate 3-D states of deformable linear objects from single-frame occluded point clouds," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 7119–7125.

[14] Y. Yang, J. A. Stork, and T. Stoyanov, "Particle filters in latent space for robust deformable linear object tracking," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 12577–12584, Oct. 2022.

[15] K. P. Cop, A. Peters, B. L. Žagar, D. Hettegger, and A. C. Knoll, "New metrics for industrial depth sensors evaluation for precise robotic applications," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 5350–5356.

[16] P. Kicki, A. Szymko, and K. Walas, "DLOFTBs—Fast tracking of deformable linear objects with B-splines," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 7104–7110.

[17] Z. Sun, H. Zhou, L. Nanbo, L. Chen, J. Zhu, and R. B. Fisher, "A robust deformable linear object perception pipeline in 3D: From segmentation to reconstruction," *IEEE Robot. Autom. Lett.*, vol. 9, no. 1, pp. 843–850, Jan. 2024.

[18] N. Lv, J. Liu, and Y. Jia, "Dynamic modeling and control of deformable linear objects for single-arm and dual-arm robot manipulations," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2341–2353, Aug. 2022.

[19] A. Myronenko and X. Song, "Point set registration: Coherent point drift," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 12, pp. 2262–2275, Dec. 2010.

[20] S. Ge, G. Fan, and M. Ding, "Non-rigid point set registration with global-local topology preservation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2014, pp. 245–251.

[21] A. L. Yuille and N. M. Grzywacz, "A mathematical analysis of the motion coherence theory," *Int. J. Comput. Vis.*, vol. 3, pp. 155–175, 1989.

[22] T. Tang, Y. Fan, H.-C. Lin, and M. Tomizuka, "State estimation for deformable objects by point registration and dynamic simulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*. IEEE, 2017, pp. 2427–2433.

[23] T. Wang and Y. Yamakawa, "Real-time occlusion-robust deformable linear object tracking with model-based Gaussian mixture model," *Front. Neurorobot.*, vol. 16, 2022, Art. no. 886068.

[24] C. Chi and D. Berenson, "Occlusion-robust deformable object tracking without physics simulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 6443–6450.

[25] N. Lv, J. Liu, H. Xia, J. Ma, and X. Yang, "A review of techniques for modeling flexible cables," *Comput.-Aided Des.*, vol. 122, 2020, Art. no. 102826.

[26] M. Gazzola, L. Dudte, A. McCormick, and L. Mahadevan, "Forward and inverse problems in the mechanics of soft filaments," *Roy. Soc. Open Sci.*, vol. 5, 2018, Art. no. 171628, doi: 10.1098/rsos.171628.

[27] G. Palli, "Model-based manipulation of deformable linear objects by multivariate dynamic splines," in *Proc. 2020 IEEE Conf. Ind. Cyberphysical Syst.*, 2020, pp. 520–525.

[28] F. Liu, E. Su, J. Lu, M. Li, and M. C. Yip, "Robotic manipulation of deformable rope-like objects using differentiable compliant position-based dynamics," *IEEE Robot. Autom. Lett.*, vol. 8, no. 7, pp. 3964–3971, Jul. 2023.

[29] Y. Yang, J. A. Stork, and T. Stoyanov, "Learning differentiable dynamics models for shape control of deformable linear objects," *Robot. Auton. Syst.*, vol. 158, 2022, Art. no. 104258.

[30] C. Wang et al., "Offline-online learning of deformation model for cable manipulation with graph neural networks," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 5544–5551, Apr. 2022.

[31] 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, Jun. 2023.

[32] C. Armanini, F. Boyer, A. T. Mathew, C. Duriez, and F. Renda, "Soft robots modeling: A structured overview," *IEEE Trans. Robot.*, vol. 39, no. 3, pp. 1728–1748, Jun. 2023.

[33] M. Bensch, T.-D. Job, T.-L. Habich, T. Seel, and M. Schappler, "Physics-informed neural networks for continuum robots: Towards fast approximation of static Cosserat rod theory," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2024, pp. 17293–17299.

[34] A. Caporali, K. Galassi, B. L. Žagar, R. Zanella, G. Palli, and A. C. Knoll, "RT-DLO: Real-time deformable linear objects instance segmentation," *IEEE Trans. Ind. Informat.*, vol. 19, no. 11, pp. 11333–11342, Nov. 2023.

[35] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5105–5114.

Alessio Caporali received the M.Sc. and Ph.D. degrees in automation engineering from the University of Bologna, Bologna, Italy, in 2019 and 2024, respectively.

He is currently a junior Assistant Professor (fixed-term) with the University of Bologna, working at the intersection of robotics and machine learning. His research interests include computer vision and tactile sensing, deformable object modeling, robotic manipulation of deformable objects, and deep learning for physical systems.

Gianluca Palli (Senior Member, IEEE) received the Laurea and Ph.D. degrees in automation engineering from the University of Bologna, Bologna, Italy, in 2003 and 2007, respectively.

He is currently a Full Professor with the University of Bologna. He has authored or coauthored more than 180 scientific articles presented at conferences or published in journals. His research interests include design and control of manipulation devices, mobile manipulation, manipulation of deformable objects and soft robotics.