Particle Filters in Latent Space for Robust Deformable Linear Object Tracking

Yuxuan Yang, Johannes A. Stork, and Todor Stoyanov

***Abstract*—Tracking of deformable linear objects (DLOs) is important for many robotic applications. However, achieving robust and accurate tracking is challenging due to the lack of distinctive features or appearance on the DLO, the object's high-dimensional state space, and the presence of occlusion. In this letter, we propose a method for tracking the state of a DLO by applying a particle filter approach within a lower-dimensional state embedding learned by an autoencoder. The dimensionality reduction preserves state variation, while simultaneously enabling a particle filter to accurately track DLO state evolution with a practically feasible number of particles. Compared to previous works, our method requires neither running a high-fidelity physics simulation, nor manual designs of constraints and regularization. Without the assumption of knowing the initial DLO state, our method can achieve accurate tracking even under complex DLO motions and in the presence of severe occlusions.**

***Index Terms*—Deep learning for visual perception, perception for grasping and manipulation, RGB-D perception.**

I. Introduction

DEFORMABLE Linear Objects (DLOs) are a class of objects, such as cables and ropes, one dimension of which is prevalent over the other two. Robustly tracking the state of a DLO given sensor observations is a necessity for robot manipulation in numerous relevant scenarios [1], [2]. One common manipulation task is to control the shape of a DLO to achieve a certain goal state, such as wires routing [3], [4], [5] and kinematic control for soft robots [6]. In these applications, reliable DLO state tracking is important for feedback in closed-loop classical control and data collection in training data-driven model-based controller [7], [8]. However, tracking DLOs is still an open research question compared to the achievements in rigid object tracking [2]. The dimension of a DLO's state space is one of the factors that makes the tracking task challenging. While, in theory, the state of a DLO is continuous and infinitely dimensional, in practice the state space is usually approximated by the positions of discrete key nodes along the length of the object. Therefore, unlike the state space of a rigid object which only spans six dimensions—three for the position and three for the orientation—the corresponding DLO state space can comprise hundreds or even thousands of dimensions.

The high state space dimensionality precludes the direct application of methods in rigid object tracking. Particle filters (PFs) are one of these methods, which are widely used in target tracking [9], with regard to either a single target [10] or multiple targets [11] in the presence of false or missing data. The number of particles required for accurate state tracking grows exponentially with the dimension of the state space, which makes PFs impractical because of a corresponding dramatic increase in the computational requirements. To address this issue, Lagneau et al. [12] use a geometric model, B-splines [13], to approximate a DLO and use a particle filter to track the position of spline control points over time, but they fail to handle situations with occlusion.

Alternatively, solving a DLO state tracking task via non-rigid registration [14] has been comprehensively explored [15], [16], [17], [18], [19], [20]. While this class of approaches results in statistically accurate estimates, the obtained DLO states are often physically implausible when parts of the tracked object are occluded [16]. Recent works [15], [16], [17], [18], [19], [20] address this issue by using physical simulation which enforces physical constraints in registration. These tracking methods require expert knowledge of the physical properties and their corresponding translations to suitable simulation model parameters. [21], [22] impose geometric constraints by designing complex regularization terms in the expectation-maximization process to avoid using physics simulation because the simulation model parameters are usually difficult to obtain. However, these methods all assume the initial state of the DLO is known and they are sensitive to occlusion.

In this letter, we propose a tracking method for reliable DLO state tracking under occlusion. We use a Sequential Importance Resampling (SIR) particle filter [23] based on a low-dimensional latent state embedding learned via an autoencoder [24]. An overview of our method is shown in Fig. 1. The autoencoder learns a low-dimensional state space for DLOs. The learned space encodes a set of physically plausible DLO states from data, which removes the need for manual modeling or regularization. Specifically, our contributions are twofold—firstly, our approach makes the particle filter tractable for DLO state tracking by learning a low-dimensional latent state representation for DLOs via the autoencoder; secondly, compared to the state-of-the-art

Manuscript received 2 May 2022; accepted 17 September 2022. Date of publication 25 October 2022; date of current version 14 November 2022. This letter was recommended for publication by Associate Editor V. Krueger and Editor C. Cadena Lerma upon evaluation of the reviewers' comments. This work was supported in part by Vinnova/SIP-STRIM under Grant 2020-04467 and in part by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. *(Corresponding author: Yuxuan Yang.)*

The authors are with the Autonomous Mobile Manipulation Lab, Center for Applied Autonomous Sensor Systems (AASS), Örebro University, 701 82 Örebro, Sweden (e-mail: yuxuan.yang@oru.se; j-a-stork@online.de; todor.stoyanov@oru.se).

Digital Object Identifier 10.1109/LRA.2022.3216985

Our implementation is available at https://amm.aass.oru.se/dlo-pf-tracking/.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2023, London, UK. Cite as RA-L paper.

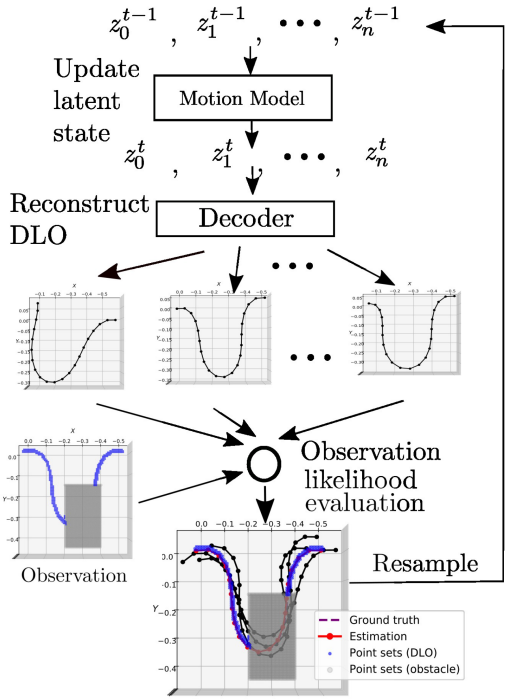

Fig. 1. Overview of the tracking process. $\boldsymbol{z}$ is the latent state (particles). Particles propagate through a motion model and then reconstruct DLOs by a trained decoder. The reconstructed DLOs are aligned to an observation. Then each particle is assigned a weight according to the observation. Last, particles are resampled according to the weights for the next iteration. We calculate the weighted average of particles and decode it to represent the observed DLO state.

methods [12] and [22], our approach generates more robust and accurate tracking results in the presence of occlusions and relaxes the assumption of knowing the initial state.

The remainder of this paper is organized as follows. Section II introduces related works on DLO state tracking, tracking by particle filters, and DLO latent representations. Section III describes our method for DLO state tracking via latent space particle filtering. Section IV demonstrates the performance of the proposed method. Section V concludes the paper and discusses the limitations and future work.

II. Related Work

A. Tracking by Particle Filters

Particle filters can incorporate different observation models and dynamics priors, making them a widely implemented method in various tracking applications [10], [11], [25]. Particle filters are also used for tracking deformable objects. However, these relate to tracking bounding boxes of deformable objects [26], [27] or non-rigid 3D surface tracking [28], which are not suitable for tracking deformable linear object (DLO) state. One of the challenges in tracking the DLO is its large state space while particle filters suffer from "the curse of dimensionality". The required number of particles can scale exponentially with the state dimensionality, leading to computational inefficiency and high memory requirements, even though the particle filter is embarrassingly parallelizable. One direct way to address this issue is to reduce the state space dimensions. For example, Lagneau et al. [12] use a particle filter to track a DLO by tracking a few control points of a B-spline which approximates the DLO. We propose a different approach that learns a low-dimension latent state space through an autoencoder for DLO state estimation and tracking.

B. DLO State Tracking by Non-Rigid Registration

To solve a DLO state tracking problem, some approaches [29], [30], [31] rely on hand-engineered features. Recently, a series of approaches show promising results in tracking a DLO from point clouds via non-rigid registration with the objective of establishing the correspondence between observed points and a chain of nodes which represents a DLO [15]. One solution to non-rigid registration is using a Gaussian Mixture Model to describe a DLO with each centroid of a Gaussian component representing each node's position of the DLO [14]. Expectation-Maximization (EM) algorithm can then be used to optimize the mean of each Gaussian component by maximizing the likelihood of point registration, which does guarantee statistically correct results but may result in physically wrong estimations [16].

To enforce physical constraints, Schulman et al. [15] propose a modified EM algorithm using a physics simulation to find the maximum a posteriori estimation, which accelerates the optimization and automatically ensures that the estimated states satisfy physical constraints. In a series of works, [16], [17], [18], Tang et al. improve DLO registration by introducing regularization terms forming a modified likelihood function to preserve both global topology and local structure. Besides, they enforce physics constraints by formulating a closed-loop that drives a virtual model in simulation to approach the point registration results. However, these methods which utilize physics simulation to help the tracking requires that a model of the deformable object is known beforehand: a condition that may be difficult to satisfy in practice. These methods are also sensitive to occlusion. Jin et al. [20] introduce a point cloud recovery to handle occlusion, but they cannot track a moving DLO under partial occlusion. Chi and Berenson [21] achieve occlusion-robust deformable object tracking without physics simulation while Wang et al. [22] afterward introduce a motion model and more regularization terms to improve the tracking performance in scenarios under occlusion and clutter. However, the tracking results sometimes fail to satisfy physical constraints and are incorrect under occlusion. All of these methods assume that the initial DLO state is known because the tracking significantly relies on the state from the previous frame. In contrast, our method, which tracks the DLO state via particle filtering, does not require an initial state. Our method also gives more physically plausible tracking results in the presence of occlusion compared to the aforementioned methods, because the particle filter samples in a latent state space which encodes a set of possible DLO states learned via an autoencoder as we test in Section IV-A.

C. DLO Latent Representations

DLOs are often modeled directly as the 3D positions of a number of N nodes or segments spaced along the object's length. This results in a state vector of length $3 \times N$, which, in practice, is often large. In general, the high dimensional state poses a challenge for tracking while tracking in a lower dimensional

representation from principal component analysis or Gaussian process latent variable models [32] is a feasible solution. Instead of employing a data-driven dimension reduction, Lai et al. [33] rely on an assumption of a planar scenario to transform the DLO to a sequence of angular offsets. However, their strategy is not feasible for a 3D scenario. Recently, latent-space dynamics has been introduced for improving the performance and robustness of deformable object dynamics simulation [34]. An autoencoder is implemented to learn a nonlinear reduced space for deformation dynamics. Zhang et al. [35] map the DLO dynamics from a non-linear space to a linear space which allows for easier learning and more efficient prediction. Yan et al. [36] use the latent dynamics models from offline learning to solve deformable object manipulation tasks such as spreading ropes and cloths. Similarly, Qi et al. [37] use an autoencoder to learn a compact representation of DLOs shape which is used in a vision-based controller for manipulation. Ma et al. [38] propose latent graph dynamics for deformable object manipulation which abstracts the deformable object state as a low-dimensional keypoint-based graph with learned latent features. Learning latent state shows promising results in manipulation, however, it has not been applied to the DLO tracking problem. In this letter, we map a DLO's high-dimensional state space to a low-dimensional latent space to make particle filtering possible for DLO tracking. We mainly utilize the decoder to provide potential DLO states by reconstructing from sampled latent states.

III. Method

In this section, we describe our method for DLO state tracking via an SIR particle filter. First, we state the problem of DLO state tracking (Section III-A). Then, we introduce learning a low-dimensional latent representation for the DLO state (Section III-B). Last, we formulate the tracking problem in a particle filtering framework and describe in detail how to track efficiently and robustly by the learned latent state (Section III-C).

A. Problem Statement

We discretize a DLO into N nodes, $\mathcal{X}^t$, where $\mathcal{X}^t = \{\boldsymbol{x}_1^t, \boldsymbol{x}_2^t, \ldots, \boldsymbol{x}_N^t\}$, and describe the DLO state at time t as a sequence of these nodes' positions, $\boldsymbol{X}^t = [\boldsymbol{x}_1^{t\intercal}, \boldsymbol{x}_2^{t\intercal}, \ldots, \boldsymbol{x}_N^{t\intercal}]^\intercal$, where $\boldsymbol{x}_i^t \in \mathbb{R}^3$ is the position of the ith node at the time step t. The DLO state is the output of the tracking algorithm given a sequence of RGB-D images $\mathcal{I}^{1:t}$. We assume a point cloud is generated from the RGB-D image and objects are segmented out from the point clouds. In this work, since we focus on tracking rather than segmentation, we assume that each RGB-D image has an associated object mask for DLO, and all points which do not belong to the object are obstacles. We denote the deformable linear object mask as $\mathcal{M}_D^t$, point sets of DLO are $\mathcal{P}^t$, and obstacle masks as $\mathcal{M}_O^t$. Our goal in tracking is to minimize $\|\hat{\boldsymbol{X}}^t - \boldsymbol{X}^t\|_2$, where $\hat{\boldsymbol{X}}^t$ is the tracking result and $\boldsymbol{X}^t$ is the ground truth state, by finding the best match between non-occluded nodes in $\hat{\mathcal{X}}^t$ and $\mathcal{P}^t$ given known obstacle masks $\mathcal{M}_O^t$.

B. DLO State Dimension Reduction

We take a data-driven approach to construct a low-dimensional embedding space, along with two mappings to translate to and from the learned manifold. Positions of the nodes along a DLO is highly correlated and our goal is to find a low-dimensional latent state space that can represent the DLO state, $\boldsymbol{X}^t = [\boldsymbol{x}_1^{t\intercal}, \boldsymbol{x}_2^{t\intercal}, \ldots, \boldsymbol{x}_N^{t\intercal}]^\intercal$, which forms a manifold within $\mathbb{R}^{3N}$. Then it becomes practical to use a particle filter in the low-dimensional state space instead of the high-dimensional one. To this end, we employ an autoencoder that consists of two parametric functions—an encoder, $\phi : \mathbb{R}^{3N} \to \mathbb{R}^m$, and a decoder, $\psi : \mathbb{R}^m \to \mathbb{R}^{3N}$, where m is the dimension of the latent state. The autoencoder is trained through unsupervised learning. With trained parameters, for a given state, $\boldsymbol{X}$, we have

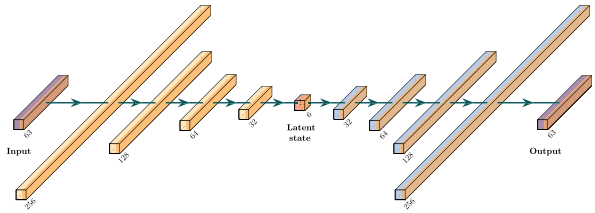

Fig. 2. The structure of the autoencoder network.

$$\boldsymbol{z} = \phi(\boldsymbol{X}) \tag{1}$$

$$\boldsymbol{X}' = \psi(\boldsymbol{z}) \tag{2}$$

where $\boldsymbol{z}$ is the latent state for the particle-filter-based tracking and $\boldsymbol{X}'$ is the reconstructed state. The goal is to reconstruct a $\boldsymbol{X}'$ that is as close as possible to the original $\boldsymbol{X}$.

In this letter, we use multi-layer perceptrons to implement the encoder and the decoder. The structure of the autoencoder is symmetric as shown in Fig. 2. The dimension m of the latent state $\boldsymbol{z}$ is a parameter where a higher dimension usually leads to a lower reconstruction error but increases the computational burden for the particle-filter-based tracking at the same time. The effect of different latent space dimensions is discussed and presented in Section IV-A.

C. Tracking via Latent Space Particle Filtering

A particle filter uses a collection of particles to represent the posterior distribution of a stochastic process given noisy and/or partial observations [39]. At each time step, particles evolve according to a motion model, or transition probability density $p(\boldsymbol{z}_i^t|\boldsymbol{z}_i^{t-1})$. Each particle is assigned a weight, ω_i, according to the observation likelihood—i.e., the likelihood of the current observation being generated by a given particle. Then the particles are resampled based on the importance weights, where particles with higher weights are more likely to be preserved for the next time step. The filtering probability density is estimated by a weighted sum of all particles.

Implementing a particle filter directly for DLO state tracking requires us to estimate a posterior distribution of a DLO state, given the current observation — $p(\boldsymbol{X}^t|\mathcal{I}^t)$. However, the particle filter cannot handle this high-dimensional state space. To address this issue, we propose a particle filter estimating a posterior distribution of the DLO latent state, $p(\boldsymbol{z}^t|\mathcal{I}^t)$. Thus each particle represents a latent state hypothesis. The tracking process is described in Algorithm 1. In this letter, we relax the assumption of a given initial DLO state compared to some related works [20], [21], [22]. If the initial state is unknown, we randomly sample

Algorithm 1: Tracking DLO State by a Particle Filter.

Input : $\mathcal{I}^t$,$\boldsymbol{z}_{1:K}^{t-1}$, α, $\boldsymbol{\Sigma}$
Output: $\boldsymbol{z}_{1:K}^{t}$, $\omega_{1:K}^{t}$

$\hat{\boldsymbol{z}}_{1:K}^{t} \leftarrow \mathrm{N}(\boldsymbol{z}_{1:K}^{t-1} + \alpha(\boldsymbol{z}_{1:K}^{t-1} - \boldsymbol{z}_{1:K}^{t-2}), \boldsymbol{\Sigma})$
for $i \leftarrow 1$ **to** K **do**
  $\omega_i^t = p(\boldsymbol{z}_i^t|\mathcal{I}^t)$; # Algorithm 2
end
$\omega_{1:K} = \frac{\hat{\omega}_{1:K}}{\sum \hat{\omega}_{1:K}}$
$\boldsymbol{z}_{1:K}^{t} \leftarrow \texttt{resample}(\boldsymbol{z}_{1:K}^{t}, \omega_{1:K}^{t})$ # Equation 3 and 4

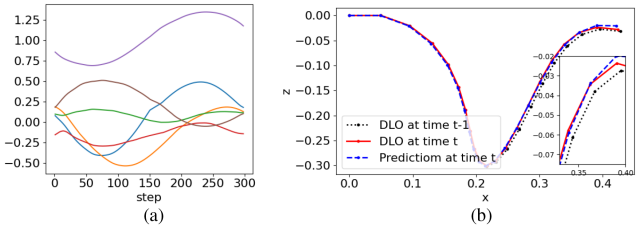

Fig. 3. (a): Changes of 6-dimension latent state when a DLO moves; (b): The DLO state prediction (projected in x-z plane) via the motion model in the latent state.

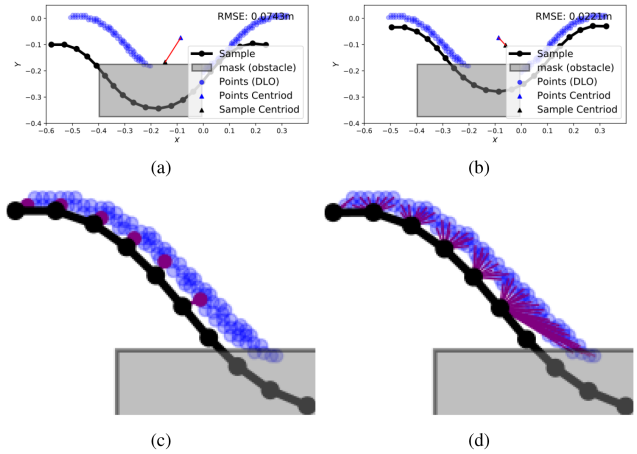

Fig. 4. Alignment of the coordinate frame of one reconstructed DLO and that of the point observations $\mathcal{P}^t$. (a) calculate centroids of non-occluded sample nodes and point sets, and align them; (b) Non-occlude nodes could change due to the movement, therefore we calculate centroids of non-occluded sample nodes and point sets, and align them again; (c) find the nearest neighbor in DLO point sets to each non-occluded sample node and calculate the distance; (d) find the nearest neighbor in non-occluded sample nodes to each point in DLO point sets and calculate the distance.

particles in the latent space. The particles converge to a good estimation of the posterior distribution of the DLO latent state after several time steps. If the initial state is known, we pass it through the encoder and use its latent state to initialize all particles. After initialization, we first propagate the K particles according to the motion model. Then, we compute observation likelihoods for each particle, $p(\boldsymbol{z}_i^t|\mathcal{I}_t)$, and update the weights, ω_i. In the end, we resample particles by systematic resampling to avoid the degeneracy problem of the algorithm. Systematic resampling generates K ordered numbers u_k and uses them to select samples $\boldsymbol{z}_{1:K}^t$ by the multinomial distribution.

$$u_k = \frac{(k-1) + \tilde{u}}{K}, \text{with } \tilde{u} \sim \mathrm{U}[0,1) \tag{3}$$

$$\boldsymbol{z}_k^t = \boldsymbol{z}_i^{t-1} \text{ with } i \text{ s.t.} u_k \in \left[\sum_{s=1}^{i-1} \omega_s, \sum_{s=1}^{i} \omega_s\right] \tag{4}$$

Next, we explain the two main components of our particle filter—the motion model and the observation likelihood.

A motion model propagates the distribution of the state from the previous time step $t-1$ to the current time step t. Empirically we note that under nominal motion of the DLO, the latent encoding changes smoothly, as shown in Fig. 3(a). Therefore, we use a constant velocity model to approximate the probability distribution propagation of the latent state.

$$p\left(\boldsymbol{z}_i^t|\boldsymbol{z}_i^{t-1}, \boldsymbol{z}_i^{t-2}\right) = \mathrm{N}\left(\boldsymbol{\mu}, \boldsymbol{\Sigma}\right) \tag{5}$$

where $\mathrm{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the multivariate normal distribution with mean $\boldsymbol{\mu} = \boldsymbol{z}_i^{t-1} + \alpha(\boldsymbol{z}_i^{t-1} - \boldsymbol{z}_i^{t-2})$ and covariance matrix $\boldsymbol{\Sigma}$. Both $\boldsymbol{\Sigma}$ and α are hyperparameters. The motion model helps approximate the transition in the latent state and an example is shown in Fig. 3(b). We assume $\boldsymbol{\Sigma} = \beta^2 \boldsymbol{I}_n$, where $\boldsymbol{I}_n$ is an n-dimension identity matrix. When $\alpha = 0$, the distribution propagation is only driven by injecting isotropic Gaussian noises with variance β^2.

An observation likelihood, $p(\boldsymbol{z}_i^t|\mathcal{I}_t)$, measures how likely a latent state, $\boldsymbol{z}_i^t$, is given the observation, $\mathcal{I}_t$. As mentioned in Section III-A, we assume the mask $\mathcal{M}_D^t$ for DLO and the mask $\mathcal{M}_O^t$ for obstacles from RGB-D image $\mathcal{I}^t$ are available. Particles are sampled in the latent state space, but we cannot directly evaluate the likelihood there. Therefore, we decode $\boldsymbol{z}_i^t$ to the original state in Cartesian coordinates by the trained decoder, $\boldsymbol{X}_i'^t = \psi(\boldsymbol{z}_i^t)$. Then we register $\boldsymbol{X}_i'^t$ to point sets of the DLO, $\mathcal{P}^t$, and evaluate the likelihood by iterative closest point registration fitness scores [40].

We assume the latent states encode all relevant DLO shapes in Cartesian space. The reconstructed nodes are in a certain space while a DLO point set in the camera frame may not be in the same space. The translation between the reconstructed nodes and the point set is unknown given that we do not assume knowing the position of either end of the DLO. Therefore, we need to align the coordinate frame of the reconstructed DLO and that of the point observations $\mathcal{P}^t$.

We take inspiration from the Iterative Closest Point algorithm [40] and the alignment process is illustrated in Fig. 4. First, we preprocess the generated DLO nodes $\mathcal{X}'^t$. We filter out the occluded nodes by projecting the nodes into a 2D image and comparing them with the obstacle masks $\mathcal{M}_O^t$. The remaining nodes are $\bar{\mathcal{X}}_i'^t$. To find the translation, we directly find the centroid $\boldsymbol{\mu}_{\bar{\mathcal{X}}_i'}$ of $\bar{\mathcal{X}}_i'^t$ and the centroid $\boldsymbol{\mu}_{\mathcal{P}}$ of target point sets $\mathcal{P}$. We then calculate a translation $\boldsymbol{\mu}_{\mathcal{P}} - \boldsymbol{\mu}_{\bar{\mathcal{X}}_i'}$ and apply it to align the two centroids. Due to the alignment, the occlusion of the generated DLO might change, as shown in Fig. 4(a) and (b). In theory, we should solve this process iteratively until the centroid of the sample's non-occluded nodes converges to the centroid of the DLO point sets, but in practice, we iterate this process at most two times as a trade-off of performance and efficiency. After the alignment, for every point in $\mathcal{P}$, we search for the closest point x_j' in the non-occluded generated state $\bar{\mathcal{X}}_i'$ and calculate an inlier RMSE, $d_{P,i}$. Similarly, for each non-occluded generated state

Algorithm 2: Observation Likelihood Calculation.

Input : $\mathcal{I}$, $\boldsymbol{z}_i$
Output: $\hat{\omega}_i$

1: Generate a DLO mask, $\mathcal{M}_D$, and an obstacle mask, $\mathcal{M}_O$, from RGB image in $\mathcal{I}$ through segmentation
2: Generate $\mathcal{P}$ which belong to the DLO from RGB-D image in $\mathcal{I}$
3: $\boldsymbol{X}'_i = \psi(\boldsymbol{z}_i^t)$
4: $\bar{\boldsymbol{X}}'_i \leftarrow$ `Filter_Nonoccluded_nodes`$(\boldsymbol{X}'_i, \mathcal{M}_O)$
5: $d_i \leftarrow$ `Calculate_inlier_RMSE`$(\bar{\boldsymbol{X}}'_i, \mathcal{P}, \mathcal{M}_O)$
6: $\hat{\omega}_i = \frac{1}{d_i}$

$\bar{\mathcal{X}}'_i$, we search for the closest point in $\mathcal{P}$ and calculate an inlier RMSE, $d_{X,i}$. We use the average of these two inlier RMSEs to describe the similarity between two point sets.

$$d_i = \frac{1}{2}\left(d_{P,i} + d_{X,i}\right), \tag{6}$$

$$\hat{\omega}_i = \frac{1}{d_i}, \quad \omega_i = \frac{\hat{\omega}_i}{\sum \hat{\omega}_i} \tag{7}$$

where the weights ω_i, which are proportional to the observation likelihood, are normalized to the sum of one. The calculation of observation likelihood is shown in Algorithm 2. In the algorithm, we assume masks for the DLOs and obstacles are available (we employ color-based image segmentation in our experiments similar to most related works [18], [19], [20], [21], [22]). This segmentation, which is not always available in real-world applications when the background of the image is complicated, constitutes a possible limitation of our method.

IV. Experiments and Results

In this section, we evaluate our method in DLO state tracking problems in simulated and real-world environments. First, we test the dimension reduction and state reconstruction accuracy by comparing the autoencoder network and principal components analysis (PCA) with different latent space dimensions. Then, we conduct experiments in simulation comparing the tracking results of our method with CDCPD2 [22] and a particle filter with B-spline [12]. Next, we test the generalization of our method for tracking DLOs with different properties. Last, we test our method and obtain qualitative results on a real DLO.

A. Dimension Reduction

We implement dimension reduction to map a high-dimensional DLO state to a low-dimensional latent state because a particle filter for tracking requires a low-dimensional state space. We test and compare autoencoder with different latent state dimensions to principal component analysis (PCA) with different principal component dimensions. We use a physics simulator to randomly manipulate a DLO by fixing one end of the object while moving and rotating the other end. The DLO is one-meter long with 21 nodes and Young's module of $1 \times 10^8\ \mathrm{N/m^2}$ in bending and twisting directions. The manipulation repeats 3000 episodes and each of which lasts six seconds with 300 discrete steps. We record the state (i.e. the positions of nodes) at each time step which is then used in PCA and to train an autoencoder. Encoder and decoder are symmetrical and implemented as 4-layer fully connected neural networks with hidden layers of width $\{258, 128, 64, 32\}$. We use a leaky rectified linear unit for the activation function with a negative slope of 0.2. The autoencoder is optimized by Adam [41] with a fixed learning rate 1×10^{-3}. The batch size is 512 and the dataset for training and validation is a 90-10 split. Reconstruction error is measured by the RMSE between the original states and reconstructed states from the corresponding latent state.

TABLE I
Average Reconstruction Error Comparison Between Autoencoder and PCA Among 1000 DLO Shapes in the Simulation

	RMSE ($\times 10^{-3}$m)					
Latent state dimension	8	7	6	5	4	3
PCA	4.0	5.2	7.8	11.7	17.1	24.8
Auto encoder	0.7	1.0	1.5	3.6	7.7	15.7

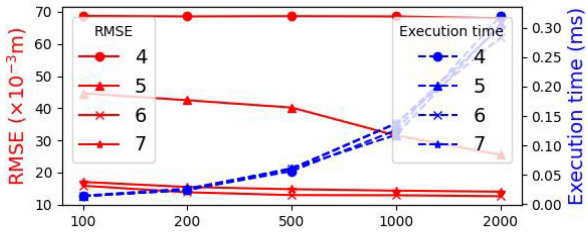

Fig. 5. Tracking performance of the particle filters with different learned latent state dimensions and particle numbers.

The average reconstruction error comparison is shown in Table I. From the table, it is clear that autoencoders generally have lower reconstruction errors. This is because they are nonlinear dimension reduction method that preserve the DLO state's variation better than linear dimension reduction methods PCA. Higher dimensions in the latent state preserve the variation better, leading to better reconstruction. However, as the higher dimensionality would require much more particles used in the tracking, it is important to choose a latent space of sufficient reconstruction performance, without unnecessarily inflating the number of latent dimensions. In the remaining sections, we evaluate our approach using autoencoders with between 4 and 7 latent dimensions.

B. Tracking DLOs in Simulation

To evaluate the tracking performance of our method quantitatively, we render RGB-D observation and obtained the ground truth of a DLO from a simulator. Here, we use the same DLO which is used in training the autoencoder.

To decide the dimension of latent space and the number of particles in tracking, we first test the influence of the learned latent space dimension on the performance of particle filtering. The results are shown in Fig. 5. Both parameters affect the tracking performance. PFs with a lower dimension of latent state perform worse because of the higher reconstruction error. For example, the PF with 4-dimension latent space cannot track the DLO, regardless of the number of particles. The reconstruction error decreases when latent state dimensions increase, but the particle filter suffers from insufficient sampling in the larger

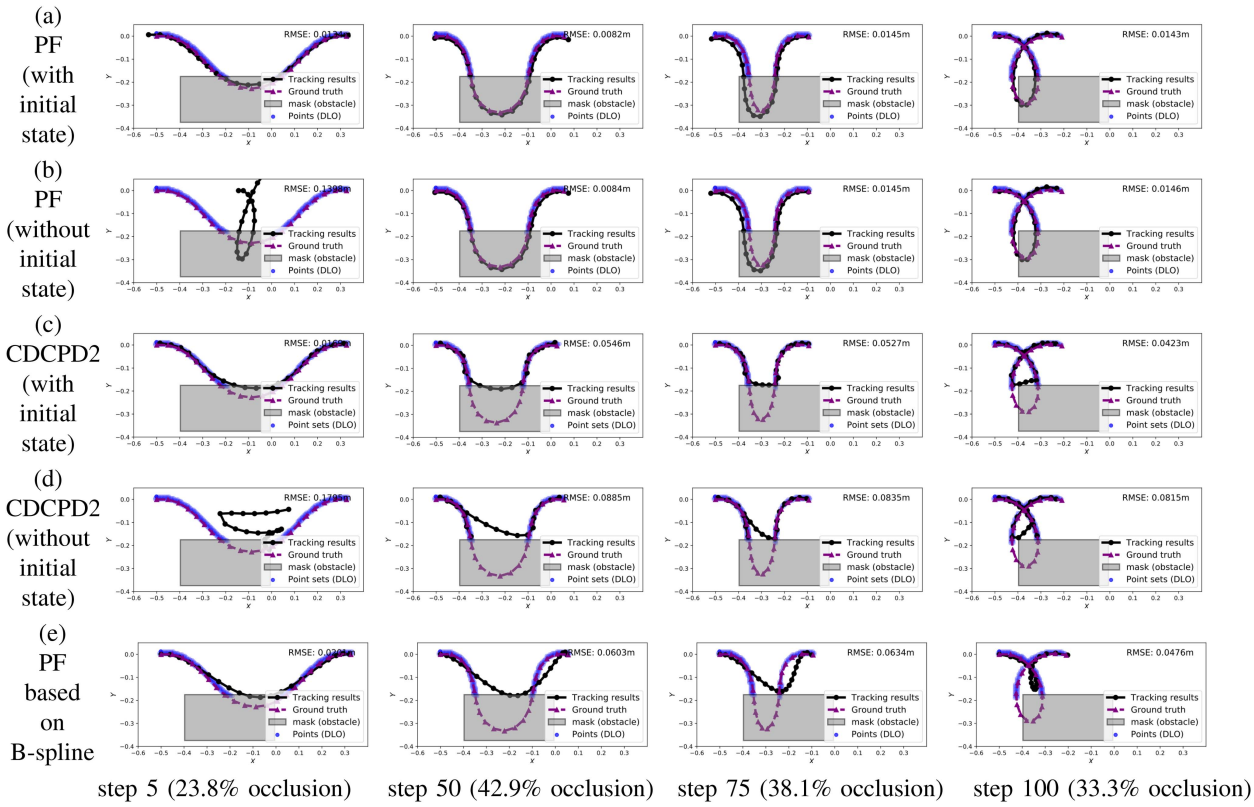

Fig. 6. Qualitative results of tracking performance with occlusion. The percentages of occlusion is calculated by the percentages of the occluded nodes. Blue points belong to the DLO while gray points represent the obstacle. Black lines are the tracking results while purple dashed lines are the ground truth. (a), (b) tracking by a particle filter based on learned latent space performs well even if initial states are unknown and randomly sampled in the latent space; (c) CDCPD2 method fails when occlusion happens; (d) CDCPD2 could fail if DLO's initial state is unknown; (e) PF based on B-spline fails if occlusion happens, the initial control points of which are from approximating the DLO's point set by a B-spline curve.

state space. In our results, the PF with 6-dimensional latent state space outperforms the one with 7-dimensional. More particles result in better tracking performance (except for 4-dimension latent state space) and meanwhile, they cause longer execution time. Therefore, we use a 6-dimension latent space and 500 particles for the following experiments so that we have the best tracking performance which can run at 10 Hz.

The second experiment demonstrates the ability of our method to provide a good guess of the occluded nodes' positions under severe occlusion while CDCPD2 [22] and the PF based on B-splines tend to generate larger errors. The quantitative comparisons are shown in Table II, while the qualitative results are shown in Fig. 6. Our method can robustly track a DLO under occlusion, while CDCPD2 can track the DLO with small occlusion but fails to do so as the occluded area increases. If there is no occlusion, our method performs on par with CDCPD2. We choose five control points for the B-spline in the PF based on B-splines [12]. More control points allow the B-spline curve to fit the shape of a DLO more accurately, however, they also make the state space larger which makes the sampling more difficult. The tracking performance of PF based on B-splines decreases when the DLO is under occlusion or large deformation.

Some tracking methods assume an initial state is known, however, this is not always the case. For example, an obstacle exists that causes occlusion when the tracking starts. Another

TABLE II
QUANTITATIVE COMPARISONS OF DLO TRACKING PERFORMANCE WITHOUT OR WITH SEVERE OCCLUSION

Method	RMSE ($\times 10^{-3}$m)		
	PF(latent state)	CDCPD2	PF(B-Splines)
No occlusion	**12.1**	**10.7**	32.6
With occlusion (0-15%)	**12.8**	18.5	35.0
With occlusion (15-30%)	**13.3**	27.9	40.4
With occlusion (30-45%)	**14.0**	42.1	44.8

Average occlusion percentages, which are the average percentage of the occluded nodes in one trial, are shown in the table. PF(latent state) represents tracking by a PF with 6-dimension latent state and 500 particles. PF(B-Splines) represents tracking by a PF based on B-Spline approximation with 5 control points and 500 particles.

experiment demonstrates that our method is less affected by the initial state. Our method directly samples in the latent state space and the particle filter can gradually converge to an accurate estimation of the posterior distribution of the DLO latent state after several time steps. We set an RMSE of 0.02 m as the threshold of the particle filter converges to an accurate estimation. Quantitative results are shown in Table III and an example of tracking without knowing the initial state is shown in Fig. 6(b). The particle filter has a better convergence with more particles and the covariance matrix Σ in the motion model has a significant influence. However, when we use 2500 particles, the computation time increases hugely and is not suitable for tracking applications.

(a) Tracking a hydraulic hose

(b) Tracking a hydraulic hose under occlusion

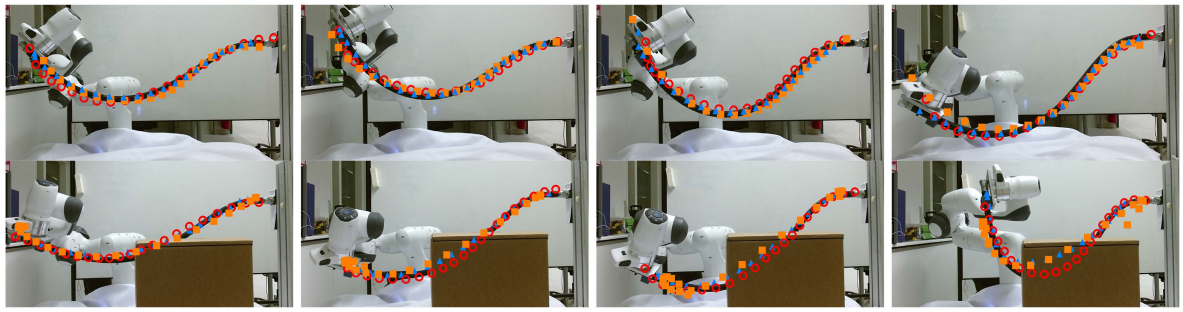

Fig. 7. Qualitative results of tracking a real DLO (a) tracking without occlusion (b) tracking under occlusion. The red circles, blue triangles, and orange squares represent the tracking results from the particle filter based on latent state (our method), CDCPD2, and the particle filter based on B-spline, respectively. We segment the DLO based on grayscale value and the obstacle based on its color.

TABLE III
CONVERGENCE OF THE PARTICLE FILTER IN TRACKING WITHOUT KNOWING THE INITIAL STATE OF THE DLO

Steps N \ β	0.01	0.02	0.04	0.08
100	-	-	47	-
200	-	46	40	-
500	-	28	**24**	-
1000	49	25	21	31

N: number of the particles; β^2: the variance. The covariance matrix is $\mathbf{\Sigma} = \beta^2 \boldsymbol{I}_n$. The time interval is 0.02s, we only report runs where the PF converges within 50 steps.

TABLE IV
QUANTITATIVE RESULTS OF TRACKING PERFORMANCE ON DLOS WITH DIFFERENT YOUNG'S MODULUS AND LENGTHS

	Property	RMSE ($\times 10^{-3}$m) No occlusion	With occlusion
Length	0.5 m	11.748	12.695
	1 m	12.555	13.420
	2 m	28.642	30.170
Young's modulus	1×10^7 N/m^2	15.243	16.712
	3×10^7 N/m^2	14.296	16.141
	1×10^8 N/m^2	12.555	13.420
	3×10^8 N/m^2	18.105	20.764
	1×10^9 N/m^2	28.090	30.764

We test DLOs with Young's modulus of 1×10^8 N/m^2 and different lengths, and DLOs with a length of 1 m and different Young's modulus. Tracking is performed 10 times for each DLO with similar manipulation, obstacles, and obstacles' positions.

C. Generalization Tests

An autoencoder is a learning method and thus its performance relies on data. We only train the autoencoder using data of trajectories collected by manipulating a DLO with a length of 1 m and Young's modulus of 1×10^8 N/m^2. However, tracking targets can be DLOs with different properties. To test the generalization of our method, we use the same autoencoder but track DLOs with different lengths and Young's modulus. The tracking errors, which are shown in Table IV, suggest that the tracking performance degrades gradually as the properties of the DLO vary. Finally, we note that for practical applications performance can be greatly improved via domain randomization of the training data.

D. Tracking DLOs in Real World

We test our method in a real-world DLO tracking experiment, as shown in Fig. 7. We directly use the autoencoder trained for DLO tracking experiments in simulation to track the state of a real-world hydraulic hose as observed by a Microsoft Kinect v1 RGB-D camera. The experiments demonstrate the qualitative performance of our method in a real-world tracking application with/without occlusion. Due to the noise in the segmentation, the tracking is not perfect. This reliance on a known segmentation of the DLO and obstacles is a limitation both to our method and the state-of-the-art baselines [12], [21] we compare with. The particle filter based on B-spline has the worst performance. This is mainly because we do not assume the motion of the gripper to be known which is required in the motion model of the particle filter based on B-spline but not required by the other two methods. CDCPD2 performs well when no occlusions are present, as shown in Fig. 7(a). When there is an occlusion, our method outperforms CDCPD2 because the latent space the particle filter samples from encodes physics priors. This allows our method to better estimate the occluded parts of the DLO in agreement with previously seen configurations in the training data, which helps the method have a better estimation for the occluded part of the DLO. Compared to tracking experiments in simulation, noisy points in point cloud segmentation and the sim-to-real gap in the trained autoencoder result in larger errors. However, point cloud segmentation is not the main focus of the paper and we can narrow the sim-to-real gap by collecting real-world datasets to fine tune the autoencoder [8].

V. CONCLUSION

We proposed a method that tracks a deformable linear object (DLO) using a particle filter. Traditionally it has been difficult and impractical to use a particle filter to track the DLO state because of the high-dimensional state space. We address this issue by learning an appropriate lower-dimensional embedding space. The learned latent state space is a good encoding for the original state space, which offers two advantages: first, this enables the use of a particle filter to estimate the posterior distribution of the DLO; and second, it allows us to reconstruct

the latent state back to a DLO's original state in Cartesian coordinates. This makes calculating observation likelihoods possible and allows the estimation of the posterior distribution of the DLO latent state to be converted to the original DLO state space.

Our experiments suggest that the proposed method provides robust tracking results in the presence of large occlusion compared to the CDCPD2 algorithm and particle filtering based on B-splines. Our method does not rely on the assumption of knowing the initial DLO state. Still, our method has some limitations. The color-based image segmentation limits our method in applications where the background is complicated. The alignment relies on accurate segmentation of DLOs and obstacles. Our method cannot handle the situation when a DLO interacts with an obstacle or itself. These can be addressed by employing a well-designed network structure for autoencoder, a more accurate motion model, and a better posterior distribution approximation. Our future work aims to alleviate these limitations.

REFERENCES

[1] P. Jiménez, "Survey on model-based manipulation planning of deformable objects," *Robot. Comput.- Integr. Manuf.*, vol. 28, no. 2, pp. 154–163, 2012.

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

[3] J. Zhu, B. Navarro, R. Passama, P. Fraisse, A. Crosnier, and A. Cherubini, "Robotic manipulation planning for shaping deformable linear objects WithEnvironmental contacts," *IEEE Robot. Automat. Lett.*, vol. 5, no. 1, pp. 16–23, Jan. 2020.

[4] S. Jin, W. Lian, C. Wang, M. Tomizuka, and S. Schaal, "Robotic cable routing with spatial representation," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 5687–5694, Apr. 2022.

[5] S. Huo et al., "Keypoint-based planar bimanual shaping of deformable linear objects under environmental constraints with hierarchical action framework," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 5222–5229, Apr. 2022.

[6] G. Fang et al., "Vision-based online learning kinematic control for soft robots using local Gaussian process regression," *IEEE Robot. Automat. Lett.*, vol. 4, no. 2, pp. 1194–1201, Apr. 2019.

[7] Y. Yang, J. A. Stork, and T. Stoyanov, "Learning to propagate interaction effects for modeling deformable linear objects dynamics," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 1950–1957.

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

[9] X. Wang, T. Li, S. Sun, and J. M. Corchado, "A survey of recent advances in particle filters and remaining challenges for multitarget tracking," *Sensors*, vol. 17, no. 12, 2017, Art. no. 2707.

[10] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox, "PoseRBPF: A Rao–Blackwellized particle filter for 6-D object pose tracking," *IEEE Trans. Robot.*, vol. 37, no. 5, pp. 1328–1342, Oct. 2021.

[11] Z. Khan, T. Balch, and F. Dellaert, "MCMC-based particle filtering for tracking a variable number of interacting targets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 11, pp. 1805–1819, Nov. 2005.

[12] R. Lagneau, A. Krupa, and M. Marchal, "Automatic shape control of deformable wires based on model-free visual servoing," *IEEE Robot. Automat. Lett.*, vol. 5, no. 4, pp. 5252–5259, Oct. 2020.

[13] C. De Boor, "On calculating with B-splines," *J. Approximation Theory*, vol. 6, no. 1, pp. 50–62, 1972.

[14] H. Chui and A. Rangarajan, "A feature registration framework using mixture models," in *Proc. IEEE Workshop Math. Methods Biomed. Image Anal.*, 2000, pp. 190–197.

[15] J. Schulman, A. Lee, J. Ho, and P. Abbeel, "Tracking deformable objects with point clouds," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 1130–1137.

[16] 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.*, 2017, pp. 2427–2433.

[17] T. Tang, C. Wang, and M. Tomizuka, "A framework for manipulating deformable linear objects by coherent point drift," *IEEE Robot. Automat. Lett.*, vol. 3, no. 4, pp. 3426–3433, Oct. 2018.

[18] T. Tang and M. Tomizuka, "Track deformable objects from point clouds with structure preserved registration," *Int. J. Robot. Res.*, vol. 41, 2018, Art. no. 0278364919841431.

[19] S. Jin, C. Wang, and M. Tomizuka, "Robust deformation model approximation for robotic cable manipulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 6586–6593.

[20] S. Jin, C. Wang, X. Zhu, T. Tang, and M. Tomizuka, "Real-time state estimation of deformable objects with dynamical simulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., Workshop Robot. Manipulation Deformable Objects*, 2020.

[21] 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.

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

[23] N. J. Gordon, D. J. Salmond, and A. F. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proc. F Radar Signal Process.*, vol. 140, no. 2, pp. 107–113, 1993.

[24] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[25] C. Kwok and D. Fox, "Map-based multiple model tracking of a moving object," in *Proc. Robot Soccer World Cup*, 2004, pp. 18–33.

[26] M. A. Rafique, M. Jeon, and M. T. Hassan, "Deformable object tracking using clustering and particle filter," *Comput. Informat.*, vol. 37, no. 3, pp. 717–736, 2018.

[27] D. Y. Kim, E. Yang, M. Jeon, and V. Shin, "Robust auxiliary particle filter with an adaptive appearance model for visual tracking," in *Proc. Asian Conf. Comput. Vis.*, 2010, pp. 718–731.

[28] I. Leizea, H. Álvarez, and D. Borro, "Real time non-rigid 3D surface tracking using particle filter," *Comput. Vis. Image Understanding*, vol. 133, pp. 51–65, 2015.

[29] T. Matsuno and T. Fukuda, "Manipulation of flexible rope using topological model based on sensor information," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2006, pp. 2638–2643.

[30] H. Wakamatsu, E. Arai, and S. Hirai, "Knotting/unknotting manipulation of deformable linear objects," *Int. J. Robot. Res.*, vol. 25, no. 4, pp. 371–395, 2006.

[31] W. H. Lui and A. Saxena, "Tangled: Learning to untangle ropes with RGB-D perception," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 837–844.

[32] N. D. Lawrence and J. Quinonero-Candela, "Local distance preservation in the GP-LVM through back constraints," in *Proc. 23rd Int. Conf. Mach. Learn.*, 2006, pp. 513–520.

[33] Y. Lai, J. Poon, G. Paul, H. Han, and T. Matsubara, "Probabilistic pose estimation of deformable linear objects," in *Proc. IEEE 14th Int. Conf. Automat. Sci. Eng.*, 2018, pp. 471–476.

[34] L. Fulton, V. Modi, D. Duvenaud, D. I. Levin, and A. Jacobson, "Latent-space dynamics for reduced deformable simulation," *Comput. Graph. Forum*, vol. 38, no. 2, pp. 379–391, 2019.

[35] W. Zhang, K. Schmeckpeper, P. Chaudhari, and K. Daniilidis, "Deformable linear object prediction using locally linear latent dynamics," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 13503–13509.

[36] W. Yan, A. Vangipuram, P. Abbeel, and L. Pinto, "Learning predictive representations for deformable objects using contrastive estimation," in *Proc. Conf. Robot Learn.*, 2021, pp. 564–574.

[37] J. Qi, G. Ma, P. Zhou, H. Zhang, Y. Lyu, and D. Navarro-Alarcon, "Towards latent space based manipulation of elastic rods using autoencoder models and robust centerline extractions," *Adv. Robot.*, vol. 36, pp. 101–115, 2022.

[38] X. Ma, D. Hsu, and W. S. Lee, "Learning Latent Graph Dynamics for Visual Manipulation of Deformable Objects," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2022, pp. 8266–8273, doi: 10.1109/ICRA46639.2022.9811597.

[39] P. Del Moral, "Nonlinear filtering: Interacting particle resolution," *Markov Process. Relat. Fields*, vol. 2, no. 4, pp. 555–580.

[40] P. J. Besl and N. D. McKay, "A Method for registration of 3-D shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 379–256, Feb. 1992, doi: 10.1109/34.121791.

[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *Proc. Int. Conf. Learn. Representations.*, 2015.