Learning to Occlusion-Robustly Estimate 3-D States of Deformable Linear Objects from Single-Frame Point Clouds

Kangchen Lv, Mingrui Yu, Yifan Pu, and Xiang Li

Abstract—Accurately and robustly estimating the state of deformable linear objects (DLOs), such as ropes and wires, is crucial for DLO manipulation and other applications. However, it remains a challenging open issue due to the high dimensionality of the state space, frequent occlusion, and noises. This paper focuses on learning to robustly estimate the states of DLOs from single-frame point clouds in the presence of occlusions using a data-driven method. We propose a novel two-branch network architecture to exploit global and local information of input point cloud respectively and design a fusion module to effectively leverage both the advantages. Simulation and real-world experimental results demonstrate that our method can generate globally smooth and locally precise DLO state estimation results even with heavily occluded point clouds, which can be directly applied to real-world robotic manipulation of DLOs in 3-D space.

I. INTRODUCTION

Robotic manipulation of deformable linear objects (DLOs), such as ropes and wires, has a wide variety of applications in industrial, service, and health-care sectors [1], [2]. An accurate and robust state estimator for DLOs is obviously the prerequisite for subsequent manipulations. Compared to rigid objects, the infinite dimensional DLO state space makes it very challenging to perceive deformations. Besides, occlusions and noises occur frequently in unstructured environments, resulting in higher requirements for robust DLO state estimation.

Frequently used representations to describe DLO states include Fourier-based parameterization [3], implicit latent descriptors learned by neural networks [4], [5], a chain of uniformly distributed nodes [6]–[8], etc. Among these methods, representing a DLO as a chain of 3-D nodes (see Fig. 1) is general and commonly used in various manipulation tasks and will be adopted in this work.

A complete processing stream to estimate the DLO state can be roughly divided into three procedures: segmentation (i.e., segmenting the DLO from environment), detection (i.e., estimating the DLO state in a single frame), and tracking (i.e., tracking the deformation across several frames). As sensors are RGB or RGB-D cameras in most cases, segmenting the DLO region in image space is the essential first step for consecutive processing. [9]–[12] focus on how to obtain pixel-level DLO masks of high quality using traditional image processing or data-driven methods. As for detection, this step aims at estimating the positions of nodes along the DLO in one frame with the cleaned sensory data as input. For example, [13], [14] use neural networks to encode the DLO into several sequential key-points in the 2-D image space; [15] estimates a skeleton line and 3-D joint positions on it from point cloud to represent the DLO, but not robust for occlusion and different DLO types. As for tracking, various works have also been proposed to track the correspondence of point cloud across video frames in the presence of occlusion and self-intersection [6], [16]–[21]. These works model DLO tracking task as a GMM-based non-rigid point registration problem with some geometric constraints. However, these pure tracking-based methods rely on an accurate initial state which requires manual setting or specific initial conditions. Besides, there are few effective ways to rectify the accumulated drift errors or re-initialize for tracking failure. Therefore, it is necessary to develop an accurate and robust 3-D state estimation method for DLOs from a single frame, which can be independently applied to estimate the DLO state in each frame or combined with tracking methods above to utilize temporal information.

In this paper, we focus on occlusion-robustly estimating a sequence of ordered and uniformly distributed nodes from single-frame point cloud to represent the state of DLO, as shown in Fig. 1. Note that we only use point cloud as our input without any auxiliary physical simulation and robot configurations. The challenges of this task are as follows: 1) there are few distinguishable features in the point cloud of DLOs; 2) occlusions and noises are common in the environment; 3) good generalization ability for different DLOs is required. To deal with challenges above, we propose a novel two-branch network architecture to leverage both the global geometry information for guaranteeing smooth and occlusion-robust shape, and local geometry information for precise estimations. To the best of our knowledge, we are
the first to realize accurate and robust 3-D state estimation of DLOs from single-frame point cloud input even with heavy occlusions. Specifically, we first exploit a PointNet++ encoder [22] to extract deep features of the input point cloud and then feed the features into two branches: End-to-End Regression and Point-to-Point Voting. We encourage these two branches to focus on global and local geometry information respectively and fuse their estimations to combine their advantages. The whole framework is trained on synthetic dataset generated in simulation without collecting real-world data. Experimental results suggest that our method achieves high performance on occlusion-robust state estimation of DLOs and can be directly applied in real-world scenarios.

II. PROBLEM STATEMENT

The goal of our method is to estimate the 3-D states of DLOs from point cloud obtained by an RGBD camera. In this work, we focus on the state estimation problem and assume that the point cloud of the DLO has already been segmented out of the raw full point cloud by RGB image segmentation. We represent the DLO state as a sequence of $M$ nodes uniformly distributed, where $M$ is a pre-defined number of nodes that can sufficiently describe the DLO state. The problem is to estimate the coordinates of the nodes $Y = [y_1, y_2, \ldots, y_M]^T \in \mathbb{R}^{M \times 3}$ from the input point cloud $X = [x_1, x_2, \ldots, x_N]^T \in \mathbb{R}^{N \times 3}$ where $N$ is the number of the points in the segmented point cloud. Note that the input point cloud $X$ is unordered, while the order of estimated nodes in $Y$ from one end to another end has been represented by the index $1, 2, \ldots, M$. In addition, the point cloud of the DLO may be fragmentary and noisy because of the occlusions, imperfect segmentation, and depth images of low quality.

III. METHOD

As shown in Fig. 2, our proposed method contains two branches: an End-to-End Regression branch and a Point-to-Point Voting branch, which focuses on the global and the local geometry information, respectively. Then, a deformable registration module is designed to leverage the advantages of both branches and fuse the two predictions to output the final estimated node sequence.

A. End-to-End Regression

The most straightforward approach is to train an end-to-end network with the point cloud $X \in \mathbb{R}^{N \times 3}$ as input and the node sequence $Y \in \mathbb{R}^{M \times 3}$ as output, which is indicated as End-to-End Regression. We exploit a PointNet++ [22] encoder denoted as $F(\cdot)$ to extract deep latent features $F(X) \in \mathbb{R}^{N \times C_{out}}$ of input point cloud $X$, which means that each point in input point cloud has a $C_{out}$-dimensional feature vector. A max pooling layer is then applied to get the global feature $\text{MaxPool}(F(X)) \in \mathbb{R}^{C_{out}}$ which is irrelevant to the input point order. Finally, a fully-connected decoder predicts the node sequence $Y^{\text{pred}}$. The whole regression network is defined as:

$$Y^{\text{pred}}_{reg} = FC_1(\text{MaxPool}(F(X))). \quad (1)$$

With the ground-truth node coordinates $Y^{gt}$, the training loss function for each sample is:

$$L_{reg} = \|Y^{\text{pred}} - Y^{gt}\|^2. \quad (2)$$

It is experimentally found that such an end-to-end network can ensure that the estimated DLO shapes are smooth and look like real DLOs even using heavily occluded point cloud input, which suggests that this network can well learn the key global characteristic of DLOs. However, the predictions are often slightly different from the actual states such that they are not sufficiently accurate for applications (see Fig. 5). We believe this phenomenon is brought about by the feature max pooling operation, which neglects crucial local information for precise estimation.

B. Point-to-Point Voting

To make up for the shortcomings of the end-to-end regression method, we design a point-to-point voting framework to utilize local geometry information, which is inspired by early works [23], [24]. Instead of using max-pooling layers for direct regression, this method generates pointwise predictions $Y^{\text{pred},1}, Y^{\text{pred},2}, \ldots, Y^{\text{pred},N}$ from each input point $x_1, x_2, \ldots, x_N$ and then uses a point-to-point voting scheme to get the final estimation. Specifically, we can regress an offset vector $O_{ij}$ which predicts the vector beginning from input point $x_i$ and ending at node $y_j$. During inference, the $y_j^{\text{pred},i}$ can be calculated by adding
the estimated $O_{ij}^{\text{pred}}$ to $x_i$ and then we can apply a voting scheme among the predictions $Y^{\text{pred},i}$ of all input points to get the final $Y^{\text{pred}}$.

Similar to [23], we decompose the point-wise offset vector $O_{ij}$ into a heatmap value $H_{ij}$ for the distance from $x_i$ to $y_j$ and a unit offset vector $U_{ij}$ for the direction, as illustrated in Fig. 3, which makes network training easier. We also restrict the ground truth value of such a point-wise estimation inside the neighborhood of the desired node to exclude noisy estimations of points far away. Given a neighborhood radius $r$, the ground-truth heatmap value $H_{ij}$ is defined as

$$H_{ij} = \begin{cases} 1 - \frac{\|x_i - y_j\|}{r}, & \|x_i - y_j\| < r, \\ 0, & \|x_i - y_j\| \geq r, \end{cases} \quad (3)$$

and the ground-truth unit offset vector $U_{ij}$ is defined as

$$U_{ij} = \begin{cases} (y_j - x_i)/\|x_i - y_j\|, & \|x_i - y_j\| < r, \\ 0, & \|x_i - y_j\| \geq r, \end{cases} \quad (4)$$

We regress the point-wise heatmap $H \in \mathbb{R}^{N \times M}$ and offset vector $U \in \mathbb{R}^{N \times M \times 3}$ from the encoded latent feature $F(X) \in \mathbb{R}^{N \times \text{Cout}}$ using shared point-wise fully-connected layers $FC_2$ and $FC_3$, respectively. The prediction of the heatmap and unit offset is denoted as:

$$H^{\text{pred}} = \text{Sigmoid}(FC_2(F(X))), \quad (5)$$
$$U^{\text{pred}} = \text{Normalized}(FC_3(F(X))), \quad (6)$$

where the Sigmoid activation function and vector normalization are used for regulating the predicted heatmap value $\in [0, 1]$ and the unit offset, respectively. For the point-to-point voting method, the training loss is:

$$L_{\text{vot}} = \sum_{i=1}^{N} \sum_{j=1}^{M} \left[ (H_{ij}^{\text{pred}} - H_{ij}^{\text{gt}})^2 + \|U_{ij}^{\text{pred}} - U_{ij}^{\text{gt}}\|^2 \right]. \quad (7)$$

The overall training loss for the whole network is a weighted sum of the $L_{\text{reg}}$ and $L_{\text{vot}}$.

As for the inference, the point-wise estimation for node $y_j$ from input point $x_i$ is obtained as

$$y_j^{\text{pred},i} = r(1 - H_{ij}^{\text{pred}})U_{ij}^{\text{pred}} + x_i. \quad (8)$$

We assume that the input point closer to the desired node predicts more accurate heatmap and unit offset, so we also use the heatmap value $H_{ij}^{\text{pred}}$ as the confidence of the prediction $y_j^{\text{pred},i}$. To eliminate the influence of unconfident predictions, we choose input points with the highest $K$ heatmap value for the $j^{th}$ node to calculate the final estimation as

$$y_j^{\text{pred}} = \left( \sum_{i \in K} H_{ij}^{\text{pred}} y_j^{\text{pred},i} \right) / \sum_{i \in K} H_{ij}^{\text{pred}}, \quad (9)$$

where the indexes of the $K$ chosen points form the set $K$.

Experiment results show that the point-to-point voting scheme can produce precise state estimations, suggesting that it mainly focuses on local regions and well learns the local characteristic for precise estimation. However, its architecture determines that if no enough point cloud exists in the local neighborhood of a node because of heavy occlusions, the prediction of the occluded part will be significantly inaccurate (also shown in Fig. 5).

C. Fusion of the Two Branches

As mentioned before, the max pooling operation in the end-to-end regression method enables the network to learn the global shape at the cost of losing rich local geometry information. Thus, the estimation of the node sequence is not very precise, but always keeps smooth and uniform with strong robustness to heavy occlusions. In contrast, the point-to-point voting method fully exploits the information in the local region of each node using point-wise estimations and voting mechanism. Owing to focusing on rich local information, the voting method shows accurate prediction performance for nodes outside occluded parts; however, for lack of learning the global characteristics, the estimation results for heavily occluded parts is significantly unreliable.

To leverage both the global and local geometry evidence and achieve occlusion-robust state estimation, we introduce a non-rigid registration-based fusion module for the two branches. According to the known correspondence between the regression and voting results, we estimate a non-rigid transformation from the smooth but imprecise regression results to the accurate unoccluded voting results, as shown in Fig. 4. Then, we transform all nodes in regression results using the estimated transformation to obtain the final estimations whose unoccluded parts are as accurate as the voting results and occluded parts are filled up by the transformed smooth regression results.

Firstly, we have to recognize which node lies in the missing part of the input point cloud. We define an occlusion possibility $p_j$ of the node $y_j$ as the max value of the heatmap among all input points:

$$p_j = 1 - \max_i H_{ij}^{\text{pred}}, \quad (10)$$

where a higher value means that there are fewer input points close to $y_j$.

Then, we regard the nodes whose $p_j$ is greater than a threshold $T$ as visible nodes. Denoting the occlusion possibility of all nodes $Y$ as $P$, the voting and regression
results for non-rigid registration (simplified as nrr) are:

\[
\begin{align*}
Y_{nrr} &= Y_{pred}^{rmvot} [P > T], \\
Y_{nrr}^{reg} &= Y_{pred}^{reg} [P > T].
\end{align*}
\]

We utilize a modified Coherent Point Drift (CPD) algorithm [25] to estimate the non-rigid transformation with known correspondence. The classical CPD formulates registration as a GMM problem and regards source points (\(Y_{nrr}^{reg}\) for us) as the centroids of Gaussian model from which target points (\(Y_{nrr}^{nrr}\)) are sampled. For non-rigid registration, CPD ensures the coherent motion of these centroids by representing the non-linear spatial transformation as

\[
T(Y_{nrr}^{reg}) = Y_{nrr}^{reg} + G(Y_{nrr}^{reg}) \cdot W,
\]

where \(G(\cdot)W\) represents a displacement function using a Gaussian Radius Basis Function Network. The Gaussian kernel matrix \(G\) with a parameter \(\beta\) is

\[
G_{ij}(Z) = \exp(-\|z_i - z_j\|^2/2\beta^2),
\]

and \(W \in \mathbb{R}^{N_z \times D}\) (\(N_z\) as the number of points in \(Z\), and \(D = 3\) for us) is the coefficient matrix.

For general-defined non-rigid registration problem, estimating the correspondence between two point sets is very challenging, and CPD uses the EM algorithm to iteratively update the correspondence probability matrix, \(\sigma^2\) of the Gaussian model, and the coefficient matrix \(M\) until convergence. In E-step, CPD fixes the current estimation of \(\sigma^2\) and \(W\) to estimate more accurate correspondences while \(\sigma^2\) and \(W\) are later optimized in M-step. However, the correspondence of our target \(Y_{nrr}^{nrr}\) and source \(Y_{nrr}^{reg}\) has been given naturally by the order of nodes in the sequence. Thus, there is no need to execute the E-step and we can directly update \(\sigma^2\) and \(W\) iteratively.

Following Equa. (22) and Equa. (23) in [25], we can fix the correspondence probability matrix as an identity matrix and solve \(W\) and \(\sigma^2\) using

\[
(G(Y_{nrr}^{reg}) + \lambda \sigma^2 I) W = Y_{nrr}^{nrr} - Y_{nrr}^{reg},
\]

where \(\lambda\) is a parameter reflecting the amount of smoothness regularization and \(D = 3\) for our 3-D state estimation task. The non-linear spatial transformation we need is determined by the coefficient \(W\). With all regression nodes \(Y_{nrr}^{reg}\) and our chosen subset \(Y_{nrr}^{pred}\), we reconstruct the Gaussian kernel matrix \(G(Y_{nrr}^{reg}, Y_{nrr}^{reg})\) as:

\[
G_{ij}(Z, Y) = \exp(-\|z_i - y_j\|^2/2\beta^2).
\]

Finally, transforming the whole regression node sequence using the non-rigid transformation represented by \(G(Y_{nrr}^{reg}, Y_{nrr}^{nrr})\) and \(W\), we get the fused results as our final estimated node sequence:

\[
Y_{fuse}^{pred} = T(Y_{nrr}^{pred}) = Y_{nrr}^{pred} + G(Y_{nrr}^{pred}, Y_{nrr}^{nrr}) \cdot W.
\]

IV. RESULTS

A. Data Collection and Model Training

All the training data is generated in simulations for the convenience of getting the ground-truth node positions. We use the Unity3D [26] as the simulator and the Obi Rope package [27] for simulating DLOs, which is a unified particle physics that models DLOs as chains of oriented particles with stretch, shear, bend, and twist constraints.

During the data collection, RGB and depth images captured by a camera and 3-D positions of the DLO particles in the camera frame are recorded to generate corresponding point cloud data and ground-truth node positions. The two ends of the DLO are randomly moved using the same strategy in our previous work [7] which is efficient in covering as many different shapes as possible. The DLO properties (lengths, radius, and stiffness) and camera poses are also randomized to increase the richness of data and then improve the generalization ability of the model.

We record one frame per simulation second and totally collect 500 \(\times\) 50 frames of data (500 sequences, 50 frames in each sequence). These collected frames are randomly separated with a ratio of 0.8 and 0.2 for training and validation, respectively. During the training, the input point cloud is augmented by random jittering with Gaussian noises, random rotation across each axis, and random occlusions. For batch training, we sample \(N = 1024\) points from the initial point cloud with the farthest point sampling (FPS) method and predict \(M = 50\) nodes from it.

In our implementation, we build up the PointNet++ encoder as the architecture for segmentation tasks in [22], which has 4 point set abstraction levels and 4 feature propagation levels to learn and aggregate features hierarchically. We set the number of local regions in each point set abstraction level as \(N_1 = 1024\), \(N_2 = 256\), \(N_3 = 64\), \(N_4 = 16\), while the output feature dimensions of each feature propagation level are \(C_1 = 256\), \(C_2 = 256\), \(C_3 = 512\), \(C_{out} = 1024\). As for hyper-parameters, we use the Adam optimizer with a initial learning rate 0.01, a weight decay of 0.0005 and a batch size 32 to train the network.
TABLE I
QUANTITATIVE RESULTS OF THREE METHODS WITH DIFFERENT DEGREE OF OCCLUSION.

| Degree of occlusion | No occlusion | 10% occluded | 20% occluded | 40% occluded |
|---------------------|--------------|--------------|--------------|--------------|
|                      | Regression  | Voting       | Fusion       | Regression  | Voting       | Fusion       | Regression  | Voting       | Fusion       |
| Errors of all nodes (mm) ↓ | 25.6 2.7 2.7 | 25.8 3.7 3.8 | 26.6 9.1 9.8 | 29.0 17.6 20.0 |
| Errors of unoccluded nodes (mm) ↓ | 26.6 39.6 | 19.4 1.7 | 18.3 1.1 1.1 | 35.2 112.3 33.4 |
| Errors of occluded nodes (mm) ↓ | - - - | 25.6 40% occluded | 2.7 0.9 | 2.7 0.9 |
| Uniformity (mm) ↓ | 1.1 1.1 0.9 | 1.10 17.5 1.0 | 27.9 101.1 35.2 |

TABLE II
ABLATION STUDIES OF DIFFERENT FUSION METHODS.

| Degree of occlusion | Fusion method | Errors of all nodes (mm) ↓ | Uniformity (mm) ↓ |
|---------------------|---------------|----------------------------|-------------------|
| No occlusion        | Point Concate. | 15.9 1.2                  | 1.2              |
|                     | Latent Concate. | 18.0 1.2 | 1.2 |
|                     | CPD           | 16.5 1.2                     | 2.7 0.9 |
|                     | Ours          | 20.7 1.2                     | 20.7 1.2 |
| 20% occluded        | Point Concate. | 18.3 1.2                  | 1.2              |
|                     | Latent Concate. | 19.4 1.7 | 1.7 |
|                     | CPD           | 19.4 1.7                     | 11.5 1.7 |
|                     | Ours          | 19.4 1.7                     | 11.5 1.7 |

Fig. 5. Visualization of the regression, voting and fusion results of (a) unoccluded point cloud and (b) occluded point cloud.

Fig. 6. Estimation results of the same DLO under different occlusion ratio.

for 200 epochs. The learning rate decays with a ratio 0.5 every 20 epochs. The radius of the neighborhood for point-to-point voting is \( r = 0.02 \) and the number of the points to vote is \( K = 64 \). For the registration module, we set \( T = 0.5 \), \( \lambda = 0.25 \) and \( \beta = 0.5 \).

B. Simulation Results

We adopt two metrics to evaluate the performance: a) errors of nodes, i.e., the average Euclidean distance between the estimated and ground-truth node positions; and b) uniformity, i.e., the standard deviation of the distances between every adjacent node which illustrates whether the estimated nodes are uniformly distributed.

1) Self-comparisons: We quantitatively evaluate the performance of End-to-End Regression, Point-to-Point Voting and Fusion with different degrees of occlusion, in which we further decompose the distance error of all nodes into those of unoccluded and occluded nodes to suggest the specific properties of these methods. As shown in Table I, the error of regression results is relatively large compared to other methods in unoccluded cases but increases very slowly as the degree of occlusion rises. Besides, the regression results keep uniform under all settings, suggesting that it always estimates smooth global DLO shapes even with heavy occlusions. In contrast, although the voting results are precise for complete point cloud input (with an error of 2.7mm), the voting method has so bare robustness that the error of occluded parts is huge (with an error of 112.3mm for 40% occluded point cloud) and the uniformity is not guaranteed. Under all settings, our proposed fusion method is the most accurate and robust one, which owes to its ability to combine both advantages of the two branches to predict accurate coordinates close to the unoccluded voting results and maintain a smooth shape for occluded parts. The uniformity is also well kept in heavily-occluded scenarios.

Two visualized examples of the three methods are shown in Fig. 5. Note that the estimated nodes are connected one by one to illustrate the order in all visualizations. It can be seen that the regression results are robust against occlusion but imprecise, while the voting results are accurate but without much robustness. The final estimation results using our fusion module can accurately and robustly estimate the DLO state in both unoccluded and occluded scenarios.

2) Robustness and sensitivity analysis: First, we test the relationship between the performance of our method and the occlusion ratio. As shown in Fig. 6, the occlusion ratio of point cloud is 20%, 40%, and 60%, respectively. When the most of the DLO is occluded (such as Fig. 6(c)), the...
In addition, we investigate the sensitivity of our fusion module to the threshold $T$, which is for determining the unoccluded parts of the point cloud in Eq. 11, as shown in Fig. 7(a). If $T$ is close to zero, almost all voting results will be used as source points of the registration step so unreliable estimations for occluded parts also participate in. As $T$ increases, more reliable estimated nodes will be chosen to get a more accurate registration result so that the distance error will decrease. However, too many valid nodes will be excluded and the performance will be hurt severely with a high $T$ value.

We also conduct a test of the model robustness for Gaussian noises on the input point cloud. As demonstrated in Fig. 7(b), despite that the standard deviation of noise is increasing, our performance still keeps stable.

3) Ablation on fusion approaches: Several fusion methods are also compared with the proposed modified CPD-based fusion module. We design the baseline fusion methods as follows: a) Point Concate.: concatenating the voting and regression predictions and learning a refinement mapping to combine them with an MLP; b) Latent Concate.: concatenating the voting results and the global feature in regression branch to learn the refinement; c) CPD: the classical CPD algorithm [25] with unknown correspondences. Experimental results in Table II suggest that our fusion method outperforms all baselines in both occluded or unoccluded scenarios. Note that the baseline Point Concate. and Latent Concate. perform similar to the regression module so that their uniformity under occlusion is good.

C. Real-World Experiments

We choose three DLOs of different lengths (0.5, 0.7, and 0.3m, respectively) and different materials to examine the generalization ability of our method in real-world applications. The two ends of DLOs are rigidly grasped by dual robot arms and deformed to various complex shapes. The point cloud of DLOs may be fragmentary owing to self-occlusions or occlusions by obstacles. The results in Fig. 8 illustrates that our method can be directly applied to estimate the real-world DLO state with small sim-to-real gaps.

An experiment of estimating the state of a moving DLO is also conducted (see Fig. 9). Even with heavy occlusions, the estimated shape of our method is still smooth and reasonable in its current frame. However, as our model only generates the estimations from single-frame point clouds without utilizing temporal information, our predicted shape might not be so continuous across adjacent frames, which we will deal with in future work.

V. Conclusions

In this work, we propose a learning-based method to robustly estimate the 3-D states of DLOs from single-frame point clouds even with heavy occlusions. We use a sequence of ordered nodes as the state representation of DLOs and design a two-branch architecture to estimate it. The two branches are encouraged to utilize the global or local geometry information respectively and their estimations are combined by a fusion module later to get the final output. The simulation and real-world experimental results demonstrate our method can guarantee an accurate and smooth shape of DLOs in occluded or unoccluded cases with high generalization capability for real-world applications. In the future, we will be devoted to integrating the temporal information into the framework to get smoother estimations across a long-term point cloud sequence.
REFERENCES

[1] J. Zhu, A. Cherubini, C. Dune, D. Navarro-Alarcon, F. Alambeigi, D. Berenson, P. Ficuciello, K. Harada, J. Kober, X. Li, et al., “Challenges and outlook in robotic manipulation of deformable objects,” *IEEE Robotics and Automation Magazine*, 2021.

[2] H. Yin, A. Varava, and D. Kragic, “Modeling, learning, perception, and control methods for deformable object manipulation,” *Science Robotics*, vol. 6, no. 54, p. eabd8803, 2021.

[3] D. Navarro-Alarcon and Y.-H. Liu, “Fourier-based shape servoing: A new feedback method to actively deform soft objects into desired 2-d image contours,” *IEEE Transactions on Robotics*, vol. 34, no. 1, pp. 272–279, 2017.

[4] P. Sundaresan, J. Grannen, B. Thananjeyan, A. Balakrishna, M. Laskey, K. Stone, J. E. Gonzalez, and K. Goldberg, “Learning rope manipulation policies using dense object descriptors trained on synthetic depth data,” in 2020 *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 9411–9418.

[5] P. Zhou, J. Zhu, S. Huo, and D. Navarro-Alarcon, “Lasemos: A latent and semantic representation framework for soft object manipulation,” *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5381–5388, 2021.

[6] T. Tang and M. Tomizuka, “Track deformable objects from point clouds with structure preserved registration,” *The International Journal of Robotics Research*, p. 0278364919844131, 2018.

[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 Transactions on Robotics*, 2022.

[8] S. Jin, C. Wang, and M. Tomizuka, “Robust deformation model approximation for robotic cable manipulation,” in 2019 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 6586–6593.

[9] H. Dinkel, J. Xiang, H. Zhao, B. Coltin, T. Smith, and T. Bretl, “Wire point cloud instance segmentation from rgbd imagery with mask r-cnn.”

[10] A. Caporali, K. Galassi, R. Zanella, and G. Palli, “Fastdlo: Fast deformable linear objects instance segmentation,” *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9075–9082, 2022.

[11] D. D. Gregorio, G. Palli, and L. D. Stefano, “Let’s take a walk on superpixels graphs: Deformable linear objects segmentation and model estimation,” in *Asian Conference on Computer Vision*. Springer, 2018, pp. 662–677.

[12] R. Zanella, A. Caporali, K. Tadaka, D. De Gregorio, and G. Palli, “Auto-generated wires dataset for semantic segmentation with domain-independence,” in 2021 *International Conference on Computer, Control and Robotics (ICCCR)*. IEEE, 2021, pp. 292–298.

[13] S. Huo, A. Duan, C. Li, P. Zhou, W. Ma, H. Wang, and D. Navarro-Alarcon, “Keypoint-based planar bimanual shaping of deformable linear objects under environmental constraints with hierarchical action framework,” *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5222–5229, 2022.

[14] M. Yan, Y. Zhu, N. Jin, and J. Bohg, “Self-supervised learning of state estimation for manipulating deformable linear objects,” *IEEE robotics and automation letters*, vol. 5, no. 2, pp. 2372–2379, 2020.

[15] M. Wnuk, C. Hinze, A. Lechler, and A. Verl, “Kinematic multibody model generation of deformable linear objects from point clouds,” in 2020 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 9545–9552.

[16] J. Schulman, A. Lee, J. Ho, and P. Abbeel, “Tracking deformable objects with point clouds,” in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 1130–1137.

[17] T. Tang, Y. Fan, H.-C. Lin, and M. Tomizuka, “State estimation for deformable objects by point registration and dynamic simulation,” in 2017 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 2427–2433.

[18] T. Tang, C. Wang, and M. Tomizuka, “A framework for manipulating deformable linear objects by coherent point drift,” *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3426–3433, 2018.

[19] Y. Chi and D. Berenson, “Occlusion-robust deformable object tracking without physics simulation,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 6443–6450.

[20] Y. Wang, D. McConachie, and D. Berenson, “Tracking partially-occluded deformable objects while enforcing geometric constraints,” in 2021 *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 14199–14205.

[21] S. Jin, W. Lian, C. Wang, M. Tomizuka, and S. Schaal, “Robotic cable routing with spatial representation,” *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5687–5694, 2022.

[22] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “Pointnet++: Deep hierarchical feature learning on point sets in a metric space,” *Advances in neural information processing systems*, vol. 30, 2017.

[23] L. Ge, Z. Ren, and J. Yuan, “Point-to-point regression pointnet for 3d hand pose estimation,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 475–491.

[24] C. Wan, T. Probst, L. Van Gool, and A. Yao, “Dense 3d regression for hand pose estimation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5147–5156.

[25] A. Myronenko and X. Song, “Point set registration: Coherent point drift,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 12, pp. 2262–2275, 2010.

[26] U. Technologies. (2022) Unity real-time development platform. [Online]. Available: https://unity.com/

[27] V. M. Studio. (2022) Obi - Unified particle physics for Unity 3D. [Online]. Available: http://obi.virtualmethodstudio.com/