The following publication S. Huo et al., "Keypoint-Based Planar Bimanual Shaping of Deformable Linear Objects Under Environmental Constraints With Hierarchical Action Framework," in IEEE Robotics and Automation Letters, vol. 7, no. 2, pp. 5222-5229, April 2022 is available at https://doi.org/10.1109/LRA.2022.3154842.

Keypoint-Based Planar Bimanual Shaping of Deformable Linear Objects under Environmental Constraints with Hierarchical Action Framework

Shengzeng Huo, Anqing Duan, Chengxi Li, Peng Zhou, Wanyu Ma, Hesheng Wang and David Navarro-Alarcon

***Abstract*—This paper addresses the problem of contact-based manipulation of deformable linear objects (DLOs) towards desired shapes with a dual-arm robotic system. To alleviate the burden of high-dimensional continuous state-action spaces, we model DLOs as kinematic multibody systems via our proposed keypoint encoding network. This novel encoding is trained on a synthetic labeled image dataset without requiring any manual annotations and can be directly transferred to real manipulation scenarios. Our goal-conditioned policy efficiently rearranges the configuration of the DLO based on the keypoints. The proposed hierarchical action framework tackles the manipulation problem in a coarse-to-fine manner (with high-level task planning and low-level motion control) by leveraging two action primitives. The identification of deformation properties is bypassed since the algorithm replans its motion after each bimanual execution. The conducted experimental results reveal that our method achieves high performance in state representation and shaping manipulation of the DLO under environmental constraints.**

***Index Terms*—Deformable Linear Object, Synthetic Learning, Bimanual Manipulation, Hierarchical Framework**

I. Introduction

DEFORMABLE object manipulation has many promising applications in growing fields, such as flexible cable arrangement [1], clothes folding [2], and surgical robots [3]. Among them, the manipulation of deformable linear objects (DLOs) attracts much attraction [4].

Compared with rigid objects, manipulating deformable objects is much more challenging due to their complex physical dynamics and multiple degrees of freedom. Although great progress has been recently achieved in deformable object manipulation (e.g. [5]–[7]), shaping DLOs under environmental constraints remains an open problem. Humans are good at using external assistance (contacts) to manipulate complex objects (underactuated systems) with high dexterity, whereas it is difficult for robots. Our insight is to endow robots with this skill similar to humans' behaviors: 1) perception with kinematic configurations instead of numerical coefficients of mathematical model and 2) hierarchical action in a coarse-to-fine manner. This paper aims to develop a complete algorithm (including perception and action) to tackle the task of contact-based shaping of DLOs with bimanual manipulation.

Perception: Many researchers have worked on the representation of DLOs in vision [8]. [1], [7] develop Fourier-based descriptors; however, they require high computation cost during online contour fitting. Data-driven based shape analysis has gained popularity in feature extraction [9]. [10] proposes an antoencoder-based network for cloth manipulation, which needs tremendous data collection. Training on synthetic datasets is useful for avoiding time-consuming data collection [11]. [12] simulates 2D fabrics on a mesh grid-connected by springs, requiring high similarity between the simulation and the environment. [13] forms a rope through twisting meshes along a Bézier curve. However, this model lacks a flexible mathematical representation of the continuous curve and has a strong hypothesis about a node on the end to break out the symmetry. [14] encodes a rope with control points in a self-supervised manner; however, it still needs a real dataset for perception finetuning.

Action: Deformable object manipulation is generally divided into two aspects, including model-based and model-free approaches. [15] proves the configuration space of the quasi-static manipulation on a elastic rod is a manifold, while the assumption is very limited. With the pregrasping hypothesis, [1], [16] approximate the local deformation model with a linear Jacobian matrix, while global convergence is not guaranteed. Formulating the task as a multistep pick-and-place manipulation problem, [13], [14] conduct the tasks with single-arm policy while real data collection is required for sim-to-real transferring or human visual demonstration. [17] assembles DLOs for specified fixtures with dual robots, yet contacts are not taken into consideration. [18] presents a framework interleaving prediction, planning and control for deformable object manipulation. However, they purely consider avoiding the obstacles instead of making use of contacts.

The method in [19] exploits environmental contacts for manipulation of DLOs, which is achieved with some customized mechanical grippers and the assumption of pregrasping. In this work, we contribute to the manipulation of DLOs from arbitrary configurations to the desired states under environmental constraints provided by stable fixtures. This scenario is a typical hybrid system whose continuous dynamics within discrete modes switches correspond to making or breaking

This work is supported in part by the Key-Area Research and Development Program of Guangdong Province 2020 under grant 2020B090928001, in part by the Research Grants Council of Hong Kong under grants 14203917 and 15212721, in part by the Jiangsu Industrial Technology Research Institute Collaborative Research Program Scheme under grant ZG9V, and in part by The Hong Kong Polytechnic University under grant 8B01

S. Huo, A. Duan, P. Zhou, W. Ma and D. Navarro-Alarcon are with The Hong Kong Polytechnic University, Department of Mechanical Engineering, Hung Hom, Kowloon, Hong Kong.

C. Li is with The Hong Kong Polytechnic University, Department of Industrial System and Engineering, Hung Hom, Kowloon, Hong Kong.

H. Wang is with the Shanghai Jiatong University, Department of Automation, Shanghai, China.

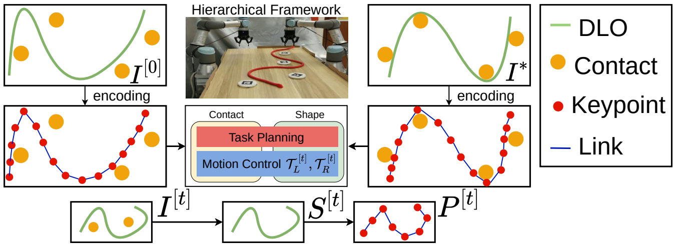

Fig. 1. The overview of the keypoint-based bimanual manipulation for shaping DLOs with contacts. Given the goal I^*, the DLO manipulation is formulated as a goal-directed task from the initial configuration $I^{[0]}$. At each time step t, the perception network encodes the state of the DLO $S^{[t]}$ as sequential keypoints $P^{[t]}$ to form a kinematic model. The hierarchical action framework takes the current $P^{[t]}$ and the goal P^* keypoints as input and outputs the action sequences $(\mathcal{T}_L^{[t]}, \mathcal{T}_R^{[t]})$. The whole algorithm replans based on the new observation after the execution of the robots and iterates until reaching the desired goal.

of contacts [20]. To deal with this issue, the shape of the DLO is characterized by a sequence of ordered keypoints with perception encoding, narrowing the state-action search space robustly and efficiently. Based on the explicit sequential keypoints, we design a hierarchical action framework for this challenging task without requiring any manual data collection and annotations. The original contributions of this work are as follows:

- A novel data-driven keypoint encoding approach for DLOs whose network is trained on a synthetic dataset.
- A hierarchical action framework for contact-based shaping DLOs in a coarse-to-fine manner with two primitives.
- An experimental study to validate our solution for DLOs shaping under real environmental constraints.

The remainder of this paper is organized as follows. Sec. II states the task's formulation. Sec. III explains the perception. Sec. IV reports the hierarchical action framework. Sec. V reports the results and Sec. VI gives the conclusions.

II. PROBLEM FORMULATION

The architecture of our vision-based manipulation system is depicted in Fig. 1. Given a goal observation I^*, our task is to manipulate the DLO from an initial configuration $I^{[0]}$ to match it. Following assumptions are made about the task: 1) the state of the DLO $S^{[t]}$ can be extracted from the raw observation $I^{[t]}$ with a color filter; 2) the information (size, sequence, and position) of the circular fixtures $C = \{c_1, \cdots, c_k, \cdots, c_K\}$ are known; 3) the DLO achieves and keeps the goal state S^* only if the completion of necessary contacts with all fixtures.

Formulating deformable object manipulation as a multistep decision-making process, our aim is to obtain a long-horizon series of action sequences $\mathcal{A} = (A^{[1]}, \cdots, A^{[t]}, \cdots, A^{[H]})$ within H steps, such that the last state $S^{[H+1]}$ reaches the goal state S^*. To deal with this task, we adopt planar bimanual manipulation to shape DLOs on a table. The action sequences $A^{[t]}$ at each time step t is divided into dual arms in the robotic system defined as $A^{[t]} = (\mathcal{T}_L^{[t]}, \mathcal{T}_R^{[t]})$. The action variety of the individual sequences $\mathcal{T}^{[t]}$ contains motion, grasping, and releasing. The state $S^{[t]}$ of the DLO is depicted as $S^{[t]} = \{s_1^{[t]}, \cdots, s_i^{[t]}, \cdots, s_N^{[t]}\}$, where N is the number of the positive values in the binary masked representation $S^{[t]}$.

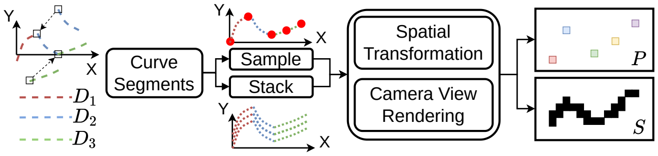

Fig. 2. Synthetic pipeline of our automatically annotated dataset. Multiple curve segments translate for end-to-end connection. This raw data undergoes sampling and stacking for labels and inputs for the dataset. At last, we render the curve as a binary image $S^{[t]}$ and its corresponding keypoints $P^{[t]}$.

Based on the kinematic multibody model [21], our perception representation model $G(\cdot)$ maps the sensory image $S^{[t]}$ to sequential keypoints $P^{[t]} = \{p_1^{[t]}, \cdots, p_j^{[t]}, \cdots, p_M^{[t]}\}$ $(M \ll N)$, which is mathematically described as $P^{[t]} = G(S^{[t]})$. This representation $P^{[t]}$ allows us to narrow down the search space and obtain an informative descriptor for bimanual manipulation. We consider the end of the DLO close to the left robot as the first keypoint in the encoding. According to this description, our hierarchical framework consists of task planning and motion control. Taking $P^{[t]}$ and $P^* = G(S^*)$ as input, the high-level model plans a local sub-goal, while the low-level model controls motion to achieve it.

III. PERCEPTION

In this section, we render an annotated synthetic image dataset (Sec. III-A) to train the keypoint encoding $P^{[t]} = G(S^{[t]})$ with supervised learning and finetune the output of the network through the geometric constraints (Sec. III-B).

A. Synthetic Dataset Generation

Current methods of synthetic DLOs (e.g. [13], [22]) are based on cylindrical meshes, which need great effort and are still far from the real situations. Our synthetic method describes DLOs with a continuous curve and renders it to a camera view since it allows us to 1) define customized sequential keypoints as labels automatically and 2) simulate the raw visual input in the real environment with high similarity. We follow the truncated Fourier series model in [7] to describe a contour. It is worth highlighting that the goal here is not to fit the existing curves, but rather to generate realistic DLOs with a known mathematical model to access the sequential keypoints. Illustrated in Fig. 2, we render a DLO consisting of several curve segments $\mathcal{D} = (D_1, \cdots D_q, \cdots D_Q)$, where $D_q = \{(d_{1x}^q, d_{1y}^q), \cdots, (d_{ex}^q, d_{ey}^q), \cdots, (d_{Ex}^q, d_{Ey}^q)\}$ is extended along the X-axis $(d_{(e+1)x}^q > d_{ex}^q)$. The Fourier-based model $\mathcal{C}(\cdot)$ maps the value in X-axis to Y-axis, denoted as $d_{ey}^q = \mathcal{C}(d_{ex}^q)$. The detailed mathematical model is:

$$d_{ey}^q = \mathcal{C}(d_{ex}^q) = \sum_{r=1}^{R} \left[\zeta_r^1 \cos(\zeta_r^2 d_{ex}^q) + \zeta_r^3 \sin(\zeta_r^4 d_{ex}^q)\right] \quad (1)$$

where $\zeta_r^1, \zeta_r^2, \zeta_r^3, \zeta_r^4$ are random numbers and R is the number of harmonics under consideration. For the adjacent line segments (D_q, D_{q+1}), we translate the first point $(d_{1x}^{q+1}, d_{1y}^{q+1})$

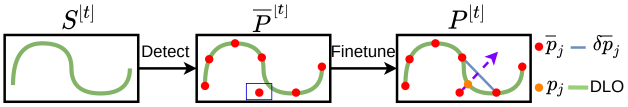

Fig. 3. Illustration of the geometric finetuning. The point locating on the background area is revised along the direction vertical to its tangent.

of D_{q+1} to the last point (d^q_{Ex}, d^q_{Ey}) of D_q, achieving a continuous end-to-end connection. After the connection, we obtain a continuous curve $\mathcal{D} = (o_1, \cdots, o_m, \cdots)$ whose elements are ordered from one end to another end. We define our customized M keypoints from $\mathcal{D}$ in two steps. Firstly, M candidates are sampled uniformly at regular intervals along the length of the rope. Since the points with high curvature are more representative to describe the contour of the DLO, we prefer to consider them as keypoints. We quantify the curvature of a point o_m with the angle α_m between its surrounding vectors, defined as:

$$\alpha_m = \left\langle f'_-(o_m), f'_+(o_m) \right\rangle \tag{2}$$

where $f'_-(o_m) = \vec{o}_m - \vec{o}_{m-1}$ and $f'_+(o_m) = \vec{o}_{m+1} - \vec{o}_m$. Here, $\left\langle \vec{a}, \vec{b} \right\rangle$ denotes the function about computing the angle between two vectors $(\vec{a}, \vec{b})$. We substitute the points whose angles α_m are larger than a threshold τ_u for their nearest candidates in the coarse sampling. In addition, we stack $\mathcal{D}$ along Y-axis to simulate the cross-section of the DLO $S^{[t]}$. Next, both the sampled keypoints and the stacked layers enter into spatial transformation for data augmentation and camera view rendering for image processing. Spatial transformation, including translation and rotation, is significant for balancing the distribution of samples. Camera view rendering consists of resizing the curve into the region of interest and reorder of the points into an image format. Since we adopt a binary image $S^{[t]}$ to represent the DLO, the pixel at $S(u, v)$ is positive if any element locates within its surroundings:

$$S(u, v) = \begin{cases} 1, & \exists o_m \in \mathcal{I}(u, v) \\ 0, & \texttt{otherwise} \end{cases} \tag{3}$$

where $o_m \in \mathcal{I}(u, v) \iff \{u - \epsilon < o_{mx} < u + \epsilon\} \cap \{v - \epsilon < o_{my} < v + \epsilon\}$, u and v are the horizontal and vertical position of the pixel in the image, and ϵ is the half of the size of a pixel. For the labeled keypoints, we transform them from Cartesian frame to image frame, represented as $p_j(u_j, v_j)$ in sequence. In conclusion, we render binary images about DLOs $S^{[t]}$ and their corresponding keypoints $P^{[t]}$.

B. Keypoint Detection

We design a neural network for the mapping $G(\cdot)$. More details about the structure and the training process are discussed in Sec V.

While the network is generalizable across different shapes of DLOs, errors are still unavoidable. As illustrated in Fig. 3, some outputs are visually located on the area of the background, which conflicts with the prior knowledge that the keypoints locate within the DLO. Hence, we implement a finetuning according to this geometric constraint. For an output $\overline{p}_j = (\overline{u}_j, \overline{v}_j)$ that fails, namely $S^{[t]}(\overline{u}_j, \overline{v}_j) = 0$, we utilize the adjacent pixels $(\overline{p}_{j-1}, \overline{p}_{j+1})$ to correct it, which is divided into two cases: (1) the ends are adjusted to the nearest pixels in the area of DLO and (2) the intermediate keypoints are revised through searching along the direction vertical to its tangent space $\delta\overline{p}_j$:

$$\begin{aligned} \texttt{find} \quad & s_i(u_i, v_i) \\ \texttt{s.t.} \quad & S^{[t]}(u_i, v_i) = 1 \\ & \overrightarrow{s_i\overline{p}_j} \cdot \delta\overline{p}_j = 0 \end{aligned} \tag{4}$$

where its tangent space $\delta\overline{p}_j$ is defined as $\delta\overline{p}_j = \vec{\overline{p}}_{j+1} - \vec{\overline{p}}_{j-1}$. Notably, we denote $P^{[t]}$ as the finetuning result of the raw output $\overline{P}^{[t]}$.

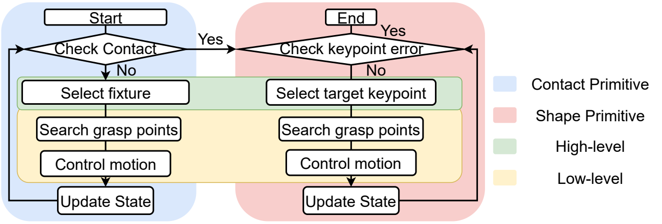

Fig. 4. Flow chart of the proposed hierarchical action framework.

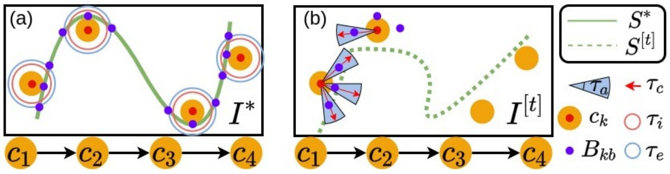

Fig. 5. Graphical explanation of the function **SelectFixture**. (a) The distance thresholds τ_i and τ_e describe the support area of the fixture to the DLO. (b) The thresholds τ_c and τ_a constraint the triangular search region for each benchmark B_{kb}.

IV. Hierarchical Action

In this section, we propose two multistep action primitives, the contact primitive and the shape primitive to achieve the task in a coarse-to-fine manner. The switch between them depends on the analysis of the contact completion, as illustrated in Fig. 4. Sharing the same classical pick-and-place manipulation configuration, we first detail the contact primitive (Sec. IV-A) and highlight the difference of the shape primitive (Sec. IV-B) afterward.

A. Contact Primitive

The goal of the coarse shaping is to make suitable contacts between the DLO and all fixtures according to the goal configuration I^*. We design the contact primitive to achieve it, including selecting a target fixture c_k in high-level and controlling motion to make corresponding contacts. The whole algorithm of this primitive is shown in Alg. 1.

Fig. 5(a) illustrates the effects of fixtures in shaping a DLO as S^*, in which each fixture c_k supports its adjacent elements of the DLO to constrain its mobility. Our **SelectFixture** function searches the target fixture along the sequence of

```
Algorithm 1: ContactPrimitive($S^{[t]}, P^{[t]}, C, \mathcal{J}, \mathcal{B}, \mathcal{B}'$)
  SelectFixture($S^{[t]}, \mathcal{B}$)$\rightarrow c_k$
  SearchGrasp($P^{[t]}, C, c_k, \mathcal{J}$)$\rightarrow \mathcal{T}_L, \mathcal{T}_R$
  if AssignRole($c_k, B'_k \in \mathcal{B}'$)$\rightarrow \gamma = LEFT$ then
    $\mathcal{T}_L \cup$ ArrangeMotion($c_k, B'_k \in \mathcal{B}', \gamma$) $\rightarrow \mathcal{T}_L$
  else
    $\mathcal{T}_R \cup$ ArrangeMotion($c_k, B'_k \in \mathcal{B}', \gamma$) $\rightarrow \mathcal{T}_R$
```

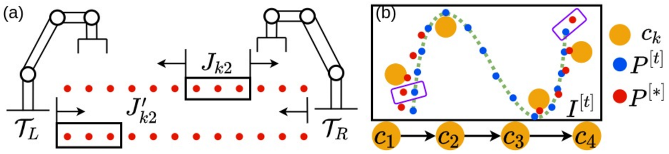

Fig. 6. Graphical explanation of the low-level motion control. (a) The direction of the search scheme about grasping points. (b) The selection of the corresponding pair of individual keypoints in the shape primitive.

$k = 2, \cdots, K, 1$ and stops once the contacts of the corresponding fixture c_k is incomplete. Note that we prioritize the fixtures in the middle since the contacts at the edge are easily affected. To analyze contacts mathematically, we quantify the operation region of the fixture c_k supporting the DLO as the goal state S^* with three benchmarks $\{B_{kb}|b = 1, 2, 3\}$; thus the complete benchmark set for all fixtures is $\mathcal{B} = \{B_{kb}|k = 1, \cdots, K, b = 1, 2, 3\}$. In the following, we first introduce the definition of the benchmark set $\mathcal{B}$ and explain the details about contact evaluation.

The benchmarks (B_{k1}, B_{k2}) are defined as the elements of the goal S^* locating on the edge of a local region around the fixture c_k. We obtain them through a constrained optimization:

$$B_{k1} = \arg\max_{s_i^*} ||s_i^* - c_k||_2, B_{k2} = \arg\max_{s_i^*} ||s_i^* - B_{k1}||_2$$
$$s.t. \quad \tau_i < ||s_i^* - c_k||_2 < \tau_e \tag{5}$$

where $(\tau_i$,$\tau_e)$ are distance thresholds of the local region. Another benchmark B_{k3} is defined as the nearest element s_i^* of the goal S^* to the fixture c_k,

$$B_{k3} = \arg\min_{s_i^*} ||s_i^* - c_k||_2 \tag{6}$$

Fig. 5(a) shows that the benchmark set $\{B_{kb}|b = 1, 2, 3\}$ effectively reflect the completion of contacts. Then, we reorder the benchmark $B_{kb} \in \mathcal{B}$ along the keypoints P^*.

For a fixture c_k, we consider the corresponding contacts are completed only if the DLO $S^{[t]}$ covers the operation region defined by $\{B_{kb}|b = 1, 2, 3\}$. Specifically, at least one element $s_i^{[t]} \in S^{[t]}$ is found that satisfies both the distance and direction conditions for each benchmark in $\{B_{kb}|b = 1, 2, 3\}$. Fig. 5(b) graphically illustrates the conditions around the fixtures. Mathematically, we implement a constrained optimization with respect to each benchmark $\{B_{kb}|b = 1, 2, 3\}$:

$$\begin{aligned} \texttt{find} \quad & s_i^{[t]} \in S^{[t]} \\ \texttt{s.t.} \quad & ||s_i - B_{kb}||_2 < \tau_c \\ & \left\langle \overrightarrow{c_k s_i}, \overrightarrow{c_k B_{kb}} \right\rangle < \tau_a \end{aligned} \tag{7}$$

where $(\tau_c$,$\tau_a)$ are the distance and angle thresholds of the evaluation respectively.

Our low-level controller takes the target fixture c_k (obtained by the **SelectFixture** function) as input and output the action sequences $(\mathcal{T}_L, \mathcal{T}_R)$. The whole procedure of the action sequences $(\mathcal{T}_L, \mathcal{T}_R)$ includes: 1) Search the grasp points; 2) Assign the roles; 3) Arrange the motion to make contacts.

SearchGrasp: We search the grasping points within $P^{[t]}$ according to the benchmark set $\mathcal{B}$. To associate the benchmark to the sequential keypoints, we pair B_{kb} and p_j^* with Euclidean distance, using $\mathcal{J} = \{J_{kb}|k = 1, \cdots, K, b = 1, 2, 3\}$ to mark the corresponding index:

$$J_{kb} = \arg\min_j ||p_j^* - B_{kb}||_2 \tag{8}$$

Fig. 6(a) shows two cases about the grasping points selection based on $\mathcal{J}$ for individual robots. When c_k is in the intermediate (J_{k2} in Fig. 6(a)), we search the keypoints $P^{[t]}$ from the index J_{k2} to the ends, while the direction is reversed for c_k is on the end (J'_{k2} in Fig. 6(a)). This definition allows robots to manipulate a relatively large portion of the DLO and avoid the collision between them. This search paradigm undertakes under the constraints of the robotic system, including the operation range and fixture obstacles.

AssignRole: Two roles, holding and moving, are defined for individual robotic arms respectively, which correspondingly act as limiting the displacement of the unrelated elements of the DLO and making contacts. We assign the left arm as the moving role (namely $\gamma = Left$) if the benchmark B_{k2} is within its reachability and vice versa (namely $\gamma = Right$).

ArrangeMotion: We arrange the local motion based on the potential field [23] to avoid the collision with the fixtures. Specifically, we consider fixtures providing repulsion forces to robots and extend the benchmark $B_{kb} \in \mathcal{B}$ to $B'_{kb} \in \mathcal{B}'$ with a threshold τ_b:

$$B'_{kb} = B_{kb} + \tau_b \cdot \overrightarrow{c_k B_{kb}} / ||B_{kb} - c_k||_2, \tag{9}$$

According to the role mode γ, the sequence of the waypoints in the motion is $(B'_{k3}, B'_{k2}, B'_{k1})$ for $\gamma = Left$ otherwise $(B'_{k1}, B'_{k2}, B'_{k3})$ for $\gamma = Right$. The goal of this motion is to manipulate the relevant elements of the DLO to the operation area of the fixture c_k to make contacts. During the manipulation, the holding arm keeps grasping the selected keypoint and the moving one follows a sequence of actions after grasping: 1) lift; 2) move to the first waypoint; 3) lower down; 4) move to the rest waypoints sequentially.

The 4-DOF pose in a table-top environment is defined as $\pi_j = \{\vec{\chi}_j, \vec{\eta}_j\}$, where $\vec{\chi}_j$ and $\vec{\eta}_j$ are position and direction vectors of 3×1. These two entities are defined by:

$$\vec{\chi}_j = B'_{kb}, \ \vec{\eta}_j \cdot \overrightarrow{c_k B'_{kb}} = 0 \tag{10}$$

B. Shape Primitive

The goal of the shape primitive is to finetune the DLO to match the goal S^*. Fig. 6(b) shows the procedures concerning on the corresponding keypoints, which the shape error ΔP to the goal is defined as:

$$\Delta P = \frac{1}{M} \sum_{j=1}^{M} ||p_j^{[t]} - p_j^*||_2 \tag{11}$$

Intuitively, we select two keypoints whose errors between the current stage $P^{[t]}$ and the goal stage P^* are largest for bimanual manipulation:

$$g \leftarrow \arg\max_{j} ||p_j^{[t]} - p_j^*||_2, g' \leftarrow \arg\max_{j, j \neq g} ||p_j^{[t]} - p_j^*||_2 \tag{12}$$

We reorder (g, g') and reassign it to the dual-arm robot by

$$g_L, g_R \leftarrow \min(g, g'), \max(g, g') \tag{13}$$

where (g, g', g_L, g_R) are indexes of the ordered keypoints. Similar to the contact primitive, we define the search paradigm under the system constraints as g_L to 1 for left arm and g_R to m for the right arm, respectively. Then, we define the target pose with respect to the g-th keypoint p_g^*:

$$\pi_g = \{\vec{p}_g^*, \delta p_g^*\} \tag{14}$$

where δp_g^* is tangent of p_g^*. This shape primitive iterates until the desired goal is reached.

V. RESULTS

A. Hardware Setup

As illustrated in Fig. 1, our bimanual experimental platform consists of two UR3 robotic manipulators equipped with 2-fingered Robotiq grippers. To facilitate the bimanual manipulation, they face each other with an interval of $0.6m$. An Intel Realsense L515 camera is mounted to sense the top-down view of the manipulation space with a resolution of 1280×780. The spatial transformation between the depth camera and dual-arms $(\mathbb{T}^L, \mathbb{T}^R)$ is calibrated through the markers. Each fixture is a cylinder (radius=4cm,height=1cm), localized via ArUco markers. All fixtures are glued on the table, keeping them stable during the whole manipulation process. The fixtures are conventionally ordered according to the detected sequential keypoints P^* concerning the goal shape of DLO S^*. Considering the physical limitations, the operation space of individual robots is constrained to a ring-shaped region.

B. Representation

For perception in real environment, we utilize OpenCV [24] to segment the DLO $S^{[t]}$ from the raw observation $I^{[t]}$ with a morphological operation-based color filter, represented as a binary image. To balance the accuracy and efficiency, we resized $S^{[t]}$ to 128×64 for the following processing.

In this section, we introduce the superiority of our synthetic-based feature extraction without any manual data collection and annotations. To reduce the gap between simulation and reality, the synthetic dataset needs to render the physics. We quantitatively and qualitatively evaluate the robustness and accuracy of the perception model.

Fig. 7 visualizes the synthetic dataset concerning the real data. Note that Fig. 7(a)-(b) is designed manually to act as references to have an intuitive comparison with the simulated Fig. 7(c)-(d). These graphical results validate the visual similarity with the real dataset. Our synthetic dataset includes 7040 labeled images in total, divided into a training dataset and testing dataset with a ratio of 10:1. Each sample is rendered as a binary image, containing a randomly generated curve and $M = 16$ corresponding sorted keypoints in image coordinates. To improve the variation of the dataset, the geometry features of the DLO, including radius, length, and the number of segments, are randomly generated over a wide range.

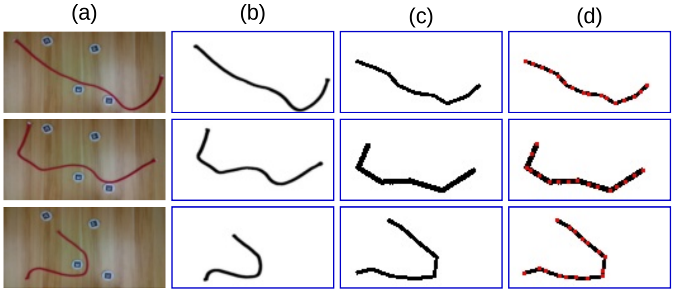

Fig. 7. Visualizations of synthetic dataset and the comparison with the real collected data. (a) Visual observation. (b) Extracted state of the DLO by the color filter. (c) Rendered state of the DLO. (d) Rendered keypoints of the DLO.

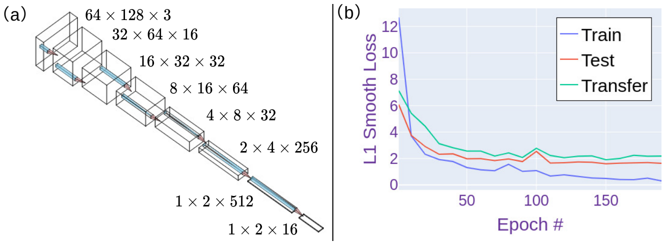

Fig. 8. Details about the perception network. (a) Architecture of the FCN network. (b) Loss convergence of training, validation, testing, and transferring.

Based on the synthetic dataset, we train our supervised keypoint detection network $G(\cdot)$, whose architecture is shown in Fig. 8(a). As a fully convolution network [25], it only involves convolution layers with a similar structure to VGG [26]. In the last layer, we apply 1×1 convolution to regress the dimension of the output as 2×16, where each column represents the position $\bar{p}_j = (\bar{u}_j, \bar{v}_j)$ in the image frame. The training is optimized based on the smooth L1 loss function

$$\mathcal{L}_1(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2 & \text{for } |y - \hat{y}| \leq 1 \\ |y - \hat{y}| - 0.5 & \text{otherwise} \end{cases} \tag{15}$$

where y and $\hat{y}$ denote the ground truth and the output of the training, respectively. Fig. 8(b) shows the corresponding loss trend for training, testing, and transferring. Note that both training and testing are implemented with our synthetic dataset for efficient processing. In addition, the transfer loss is evaluated on the real data collection with manual annotation, which includes fifty samples. Note that this manual collection dataset is only for evaluation and is not used to train the network. The promising results reveal the advantages of our perception method: 1) our synthetic dataset holds a high similarity with the real data to avoid manual collection; 2)

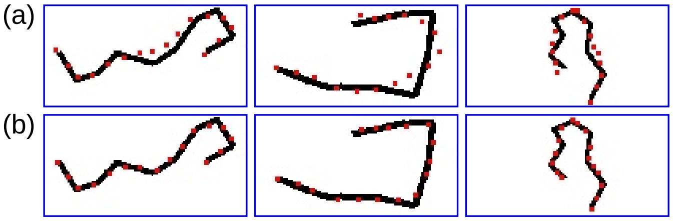

Fig. 9. Visualizations of finetuning the predicted keypoints according to the surrounding geometric features. (a) Raw output of the network. (b) Finetuning results.

TABLE I
COMPARISON OF KEYPOINT DETECTION PERFORMANCE

	Corner Error E_C		Keypoint Error E_P	
	μ_C	σ_C^2	μ_P	σ_P^2
Geo	1.96	29.78	28.21	389.96
Ours	1.71	12.5	3.36	21.44

Geo: Geometric-based method; Our: Our data-driven algorithm. μ_C, σ_C^2 : Mean and variance of corner error E_C. μ_P, σ_P^2 : Mean and variance of keypointd error E_P.

the keypoint detection network converges to minimize the detection error; (3) the perception model is general to unseen samples in testing (simulation) and transferring (real).

As discussed above, geometric finetuning is proposed to account for the residual error. Fig. 9(a) illustrates several failure cases, in which some detected keypoints drop out from the positive region of the DLO, mainly on the steep area of the curve. Comparatively, Fig. 9(b) visualizes the keypoints with finetuning, graphically indicating that this method improves the representation level of the sequential keypoints.

Compared with data-driven learning models, manual designed descriptor is an alternative for keypoint detection due to its intuitiveness and interpretability. Here, we provide a comparison between our method and a traditional geometric-based baseline, whose steps include skeletonizing DLOs via [27] from $S^{[t]}$, searching the corners of DLOs according to the mesh grids, sorting and sampling the keypoints based on nearest neighbor search. Our error metrics include the corner E_C and the keypoint detection error E_P, which are defined as $E_C = \frac{1}{2}(||\hat{p}_1 - p_1||_2 + ||\hat{p}_M - p_M||_2)$ and $E_P = \frac{1}{M}\sum_{j=1}^{j=M} ||\hat{p}_j - p_j||_2$, respectively. We emphasize the corner error E_C here since it is the symbol to order the keypoints. Statistically, we leverage the mean value (μ_C, μ_P) and the variance (σ_C^2, σ_P^2) to evaluate their performance comprehensively. Note that p_j and $\hat{p}_j$ are the ground truth of the dataset and the output of the corresponding algorithm, respectively. The comparison results are shown in Table. I. Due to the substantial diversity of the state space of DLOs, it is very difficult to manually develop a sequential keypoint detection method that is robust to various configurations. Conversely, our perception network is robust with its data-driven manner.

A key issue about descriptors is their representation level versus the original data. Since we only predict keypoints of DLOs based on the link-chain model, we reconstruct the original shape through end-to-end connection. For comparisons, we consider various unsupervised auto-encoders [9], whose goal is also to extract a compact latent code about the high-dimensional data. We choose three baselines to adapt to our case 1) fully connected linear regression (LR), 2) convolutional neural network (CNN), and 3) PointNet [28] (PC). Specifically, the training of LR and CNN autoencoders is conducted based on the binary cross entropy (BCE) loss $\mathcal{L}_{BCE}$, while PC autoencoder is optimized through Chamfer distance d. They are defined as:

TABLE II
COMPARISON OF KEYPOINT DETECTION PERFORMANCE ON SYNTHETIC DATASET

	L1			IoU		
Net	Train	Valid	Test	Train	Valid	Test
FCN-L	0.0074	0.0074	0.0073	0.6645	0.665	0.6648
FCN-R	0.0274	0.0276	0.0272	0.0798	0.0783	0.0804
FCN-F	0.0208	0.0212	0.0205	0.2558	0.2493	0.2573
LR	0.0297	0.0299	0.0325	0.0846	0.0852	0.0643
CNN	0.0294	0.0296	0.0291	0.0996	0.0995	0.0991
PC [28]	0.02	0.0201	0.0199	0.0882	0.0878	0.0852

FCN-L: label of the FCN; FCN-R: raw output of the FCN; FCN-F: finetuning FCN; LR: linear regression; CNN: convolutional neural network; PC [28]: point cloud.

$$\begin{aligned} \mathcal{L}_{BCE} &= -\sum_{i=1}^{n} y_i \log \hat{y}_i + (1 - y_i) \log (1 - \hat{y}_i) \\ d\left(\hat{Y}, Y\right) &= \sum_{\hat{y} \in \hat{Y}} \min_{y \in Y} ||\hat{y} - y||_2^2 + \sum_{y \in Y} \min_{\hat{y} \in \hat{Y}} ||\hat{y} - y||_2^2 \end{aligned} \tag{16}$$

According to the network structure, LR and CNN take the 2D image format as input, while PC utilizes the 3D point cloud with the same size after downsampling.

Since our original state $S^{[t]}$ is a binary image, the shape reconstruction issue here is formulated as a classification concerning each pixel $S^{[t]}(u, v)$. Due to the original output of the above autoencoders in the image format (LR, CNN) is continuous value in $[0, 1]$, we consider it as the probability about the existence of the element of the DLO. In addition, we set a threshold $\tau = 0.5$ to transfer the continuous output into a discrete binary value. Hence, our evaluation metrics are L1 loss $\mathcal{L}_1$ for original continuous output and IoU (Intersection over Union) for thresholding binary values between the reconstructed output and the original information, respectively:

$$\mathcal{L}_1 = \sum_{i=1}^{n} |y_i - \hat{y}_i|, IoU = \frac{\hat{y} \cap y}{\hat{y} \cup y} \tag{17}$$

Table. II shows the comparison results. Note that FCN-L method utilizes the labeled keypoints for reconstruction and acts as ground truth for our data-driven representation. The finetuning output of our perception improves greatly compared with the raw output of the network FCN-R. Compared with LR and CNN autoencoders, our proposed FCN-F performs better both in L1 loss and IoU. The main reason is that autoencoders aim to reconstruct the entire information of the input (even the details) instead of paying attention to the fundamental features. Although [28] achieves well in L1 loss, its performance concerning IoU is poor. This is because it is only able to reconstruct the original data with a fixed size (due

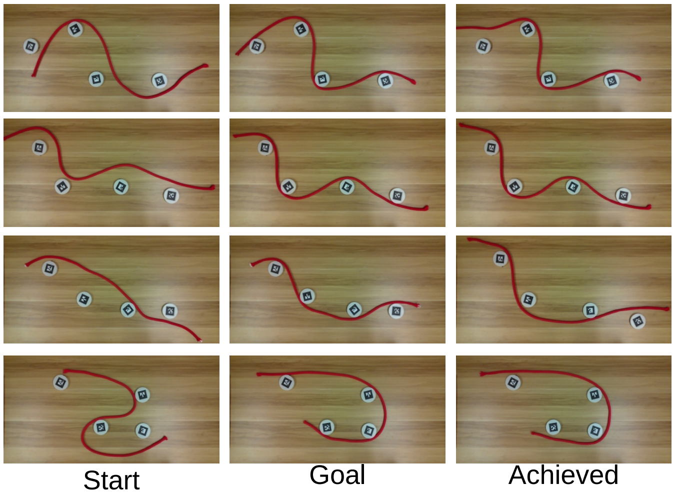

Fig. 10. Our designed DLO manipulation with environmental contacts scenarios. From left to right: the start state, the goal state and achieved state with our framework. All the images are taken by our top-down Realsense L515 depth camera.

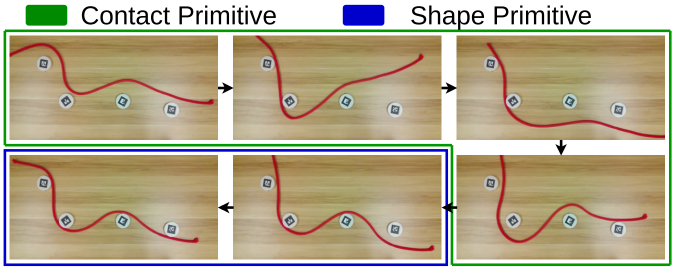

Fig. 11. The detailed procedure of a typical coarse-to-fine manipulation example.

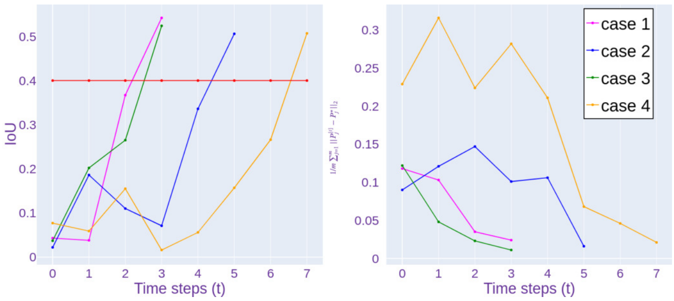

Fig. 12. Shape error minimization process. (a) IoU between the goal state S^* and the state $S^{[t]}$ in each time step t. The red line represents the successful baseline and also the termination conditions. (b) The keypoints error between P^* and $P^{[t]}$.

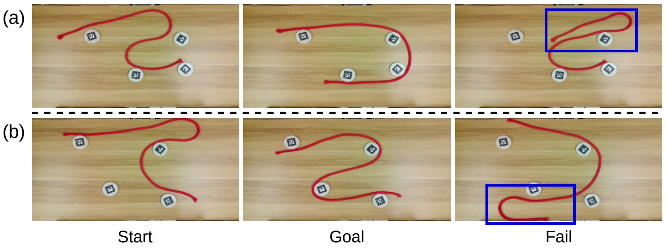

Fig. 13. Graphical representation about the failure cases. (a) Rolling. (b) Open-loop control.

to the identical input dimension); thus loses some information inevitably.

C. Manipulation

To validate our hierarchical action framework, we evaluate the performance with multiple experiments using various fixtures configurations and goals. Fig. 10 shows four designed tasks in our experiment. Note that the configuration of the DLO at the beginning $S^{[0]}$ is placed randomly on the table and the desired goal is provided artificially. For each experiment, we assume that the goal shape S^* keeps stable with the support of the fixtures and the table. The third column in Fig. 10 illustrates our achieved results. Since our hierarchical action framework is iterative, the robot continuously manipulates the DLO until the shape similarity between the goal S^* and the achieved one $S^{[t]}$ reaches a given threshold. In this experimental study, the shaping tasks are conducted with multistep actions depending on the feature extraction of DLOs without learning their physical dynamics.

As a multistep decision-making process, we provide a typical example of the manipulation, as shown in Fig. 11. At the beginning, our algorithm computes the prior knowledge for the hierarchical action framework based on the goal image I^*: 1) segment the DLO S^* with the color filter and detect the corresponding sequential keypoints P^* through our perception network and 2) localize the fixtures $C = \{c_1, \cdots, c_k, \cdots, c_K\}$ and compute the contact-based benchmarks $(\mathcal{B}, \mathcal{B}', \mathcal{J})$. Then, our algorithm enters into the action loop. At each time-step t, we sense the DLO $S^{[t]}$ and detect its keypoints $P^{[t]}$ via our perception network. With this, we check the contact completion based on our search benchmarks $\mathcal{B}$. If it is incomplete, we utilize the contact primitive to make contacts with the corresponding fixture c_k. Once the action sequences $(\mathcal{T}_L, \mathcal{T}_R)$ is accomplished, we update the state of the DLO $S^{[t+1]}$ and check the contact completion again. If the contact constraints are complete, we move on to the shape primitive for finetuning. The entire algorithm iterates until reaching the goal state S^*, which the criteria is defined as the binary IoU between $S^{[t]}$ and S^* according to Eq. 17 should be larger than 40%. Note that we choose IoU as the evaluation metric since it is intuitive to measure their similarity ratio and the comparison objects are both binary. We also provide supplementary material for robotic bimanual manipulation videos.

Based on the goal shape in Fig. 10, we implement four trials under various initial configurations. Fig. 12 depicts the quantitative measurements of the scenarios in Fig. 10. Specifically, the minimization of the magnitude error ΔP is shown in Fig. 12(b). These results corroborate that the detected sequential keypoints can be used to manipulate the DLO into the desired specification. Fig. 12(a) demonstrates the similarity level of the state at each time step with the goal shape S^*, which IoU$= 40\%$ serves as a baseline. Note that the IoU value decreases compared to the previous time step in some cases since the contact-based manipulation task is not continuous. Hence, a coarse-to-fine manner is necessary for this challenging task, otherwise, we probably get stuck in a local optimum. These results also reveal that our algorithm is superior in feature description and action planning versus this kind of challenging task.

Although our action framework is capable of dealing with the majority of these challenging tasks, there are some cases that the system fails. Fig. 13 presents two typical failure examples. Although our perception network plays well in most cases, its performance is severely affected by rolling (a region of high curvature to form a closed loop). That is because the convolution is not good at dealing with the details of the pixels and the finetuning regresses the keypoints to the wrong section of the DLO, resulting in disordered keypoints. Another case is caused by the lack of physical dynamics. Without any forecasting and feedback, our framework replans the action in an open-loop form. Thus, the system probably traps in a local area around a fixture.

VI. Conclusions

In this paper, we demonstrate a keypoint-based bimanual manipulation for DLOs under environmental constraints. Training on a synthetic image dataset, our perception model describes a DLO with sequential keypoints. The hierarchical action framework performs the task with two defined primitives in a coarse-to-fine manner. The whole algorithm is explicit without requiring any manual data collection and annotations. However, our methods exhibit some limitations. The perception network has poor performance in the knotted case. As an open-loop method, the stability of the planner is not guaranteed. For future directions, we are interested to include the prior spatial-temporal knowledge about the DLO into the perception and the effect of the action as feedback to form a closed-loop control.

References

[1] J. Zhu, B. Navarro, P. Fraisse, A. Crosnier, and A. Cherubini, “Dual-arm robotic manipulation of flexible cables,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 479–484, IEEE, 2018.

[2] I. Garcia-Camacho, M. Lippi, M. C. Welle, H. Yin, R. Antonova, A. Varava, J. Borras, C. Torras, A. Marino, G. Alenya, *et al.*, “Benchmarking bimanual cloth manipulation,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1111–1118, 2020.

[3] D. Navarro-Alarcon, H. M. Yip, Z. Wang, Y.-H. Liu, F. Zhong, T. Zhang, and P. Li, “Automatic 3-d manipulation of soft objects by robotic arms with an adaptive deformation model,” *IEEE Transactions on Robotics*, vol. 32, no. 2, pp. 429–441, 2016.

[4] 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,” *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 688–716, 2018.

[5] D. Navarro-Alarcon, Y.-H. Liu, J. G. Romero, and P. Li, “Model-free visually servoed deformation control of elastic objects by robot manipulators,” *IEEE Transactions on Robotics*, vol. 29, no. 6, pp. 1457–1468, 2013.

[6] J. Zhu, D. Navarro-Alarcon, R. Passama, and A. Cherubini, “Vision-based manipulation of deformable and rigid objects using subspace projections of 2d contours,” *Robotics and Autonomous Systems*, vol. 142, p. 103798, 2021.

[7] 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.

[8] J. Zhu, A. Cherubini, C. Dune, D. Navarro-Alarcon, F. Alambeigi, D. Berenson, F. Ficuciello, K. Harada, X. Li, J. Pan, *et al.*, “Challenges and outlook in robotic manipulation of deformable objects,” *arXiv preprint arXiv:2105.01767*, 2021.

[9] P. Zhou, J. Zhu, S. Huo, and D. Navarro-Alarcon, “Lasesom: A latent and semantic representation framework for soft object manipulation,” *IEEE Robotics and Automation Letters*, 2021.

[10] D. Tanaka, S. Arnold, and K. Yamazaki, “Emd net: An encode–manipulate–decode network for cloth manipulation,” *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1771–1778, 2018.

[11] E. Johns, S. Leutenegger, and A. J. Davison, “Deep learning a grasp function for grasping under gripper pose uncertainty,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4461–4468, IEEE, 2016.

[12] D. Seita, A. Ganapathi, R. Hoque, M. Hwang, E. Cen, A. K. Tanwani, A. Balakrishna, B. Thananjeyan, J. Ichnowski, N. Jamali, *et al.*, “Deep imitation learning of sequential fabric smoothing from an algorithmic supervisor,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9651–9658, IEEE, 2020.

[13] 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)*, pp. 9411–9418, IEEE, 2020.

[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] T. Bretl and Z. McCarthy, “Quasi-static manipulation of a kirchhoff elastic rod based on a geometric analysis of equilibrium configurations,” *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 48–68, 2014.

[16] 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)*, pp. 6586–6593, IEEE, 2019.

[17] T. Tang and M. Tomizuka, “Track deformable objects from point clouds with structure preserved registration,” *The International Journal of Robotics Research*, p. 0278364919841431, 2018.

[18] D. McConachie, A. Dobson, M. Ruan, and D. Berenson, “Manipulating deformable objects by interleaving prediction, planning, and control,” *The International Journal of Robotics Research*, vol. 39, no. 8, pp. 957–982, 2020.

[19] J. Zhu, B. Navarro, R. Passama, P. Fraisse, A. Crosnier, and A. Cherubini, “Robotic manipulation planning for shaping deformable linear objects withenvironmental contacts,” *IEEE Robotics and Automation Letters*, vol. 5, no. 1, pp. 16–23, 2019.

[20] O. Kroemer, S. Niekum, and G. Konidaris, “A review of robot learning for manipulation: Challenges, representations, and algorithms.,” *J. Mach. Learn. Res.*, vol. 22, pp. 30–1, 2021.

[21] 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)*, pp. 9545–9552, IEEE, 2020.

[22] D. Seita, P. Florence, J. Tompson, E. Coumans, V. Sindhwani, K. Goldberg, and A. Zeng, “Learning to Rearrange Deformable Cables, Fabrics, and Bags with Goal-Conditioned Transporter Networks,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[23] P. Song and V. Kumar, “A potential field based approach to multi-robot manipulation,” in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 2, pp. 1217–1222, IEEE, 2002.

[24] G. Bradski, “The opencv library,” *Dr Dobb’s J. Software Tools*, vol. 25, pp. 120–125, 2000.

[25] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, 2015.

[26] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[27] T. Y. Zhang and C. Y. Suen, “A fast parallel algorithm for thinning digital patterns,” *Communications of the ACM*, vol. 27, no. 3, pp. 236–239, 1984.

[28] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, “Learning representations and generative models for 3d point clouds,” in *International conference on machine learning*, pp. 40–49, PMLR, 2018.