

RESEARCH ARTICLE

FabricFolding: learning efficient fabric folding without expert demonstrations

Can He^{1,2} , Lingxiao Meng¹, Zhirui Sun^{1,2} , Jiankun Wang^{1,2}  and Max Q.-H. Meng¹

¹Shenzhen Key Laboratory of Robotics Perception and Intelligence, Department of Electronic and Electrical Engineering, Southern University of Science and Technology, Shenzhen, China and ²Jiaxing Research Institute, Southern University of Science and Technology Jiaxing, China

Corresponding authors: Jiankun Wang; Email: wangjk@sustech.edu.cn, Max Q.-H. Meng; Email: max.meng@ieee.org

Received: 11 September 2023; **Revised:** 10 January 2024; **Accepted:** 30 January 2024

Keywords: Fabric unfold; heuristic folding; keypoint detection; self-supervised learning

Abstract

Autonomous fabric manipulation is a challenging task due to complex dynamics and potential self-occlusion during fabric handling. An intuitive method of fabric-folding manipulation first involves obtaining a smooth and unfolded fabric configuration before the folding process begins. However, the combination of quasi-static actions like pick & place and dynamic action like fling proves inadequate in effectively unfolding long-sleeved T-shirts with sleeves mostly tucked inside the garment. To address this limitation, this paper introduces an enhanced quasi-static action called pick & drag, specifically designed to handle this type of fabric configuration. Additionally, an efficient dual-arm manipulation system is designed in this paper, which combines quasi-static (including pick & place and pick & drag) and dynamic fling actions to flexibly manipulate fabrics into unfolded and smooth configurations. Subsequently, once it is confirmed that the fabric is sufficiently unfolded and all fabric keypoints are detected, the keypoint-based heuristic folding algorithm is employed for the fabric-folding process. To address the scarcity of publicly available keypoint detection datasets for real fabric, we gathered images of various fabric configurations and types in real scenes to create a comprehensive keypoint dataset for fabric folding. This dataset aims to enhance the success rate of keypoint detection. Moreover, we evaluate the effectiveness of our proposed system in real-world settings, where it consistently and reliably unfolds and folds various types of fabrics, including challenging situations such as long-sleeved T-shirts with most parts of sleeves tucked inside the garment. Specifically, our method achieves a coverage rate of 0.822 and a success rate of 0.88 for long-sleeved T-shirts folding. Supplemental materials and dataset are available on our project webpage at <https://sites.google.com/view/fabricfolding>.

1. Introduction

In recent years, significant progress has been made in the field of robotic manipulation, especially in the handling of rigid objects, and breakthroughs have been made in multiple aspects, such as the re-grasping of complex objects [1, 2] and manipulation in cluttered environments [3, 4]. However, the autonomous manipulation of fabrics still faces significant challenges compared with rigid objects. This is mainly attributed to two key factors: the complex dynamic model of the fabric and the persistent self-occlusion problem during fabric manipulation. Therefore, further research is necessary to overcome these challenges and unlock the full potential of fabric manipulation for robotic applications.

Early fabric manipulation research aims to develop heuristic methods for quasi-static manipulations with a single robotic arm to accomplish tasks such as fabric unfolding [5], smoothing [6], and folding [7]. However, these methods have some inherent limitations, including strong assumptions about the initial state and fabric type. Recently, there has been a surge in the development of deep learning techniques for fabric manipulation, where researchers introduce self-supervised learning to replace the need for expert demonstrations [8]. By learning goal-conditioned [9] strategies, it is now possible to effectively fold a

single square fabric. However, this quasi-static method requires numerous iterations to obtain relatively smooth fabric. Ha *et al.* [10] propose a dynamic fling method to achieve fabric unfolding, but it cannot be generalized to other tasks. Some researchers have attempted to combine supervised learning from expert demonstrations to realize fabric folding and unfolding [11], but these methods require extensive human annotations, which are time-consuming and error-prone. Canberk *et al.* [12] proposed a method that utilizes canonicalized alignment to unfold fabric, complemented by heuristic keypoint detection for folding the unfolded fabric. This approach solely relies on pick & place and fling actions for unfolding the fabric.

However, the proposed method encounters challenges in achieving effective unfolding when dealing with complex fabrics like long-sleeved T-shirts, particularly when certain sections of the sleeves are concealed or tucked beneath other fabric layers. Consequently, this limitation adversely impacts the fabric-folding task. Moreover, due to the absence of publicly available keypoint datasets specifically designed for fabric folding, Canberk *et al.* only collected 200 simulated cloth data samples for each fabric type to train the keypoint model. Given the sim2real gap, employing simulation data for training purposes can result in inadequate detection accuracy and a lack of robustness in the detection model when applied to real-world scenarios. This limitation can significantly impact the success rate of the fabric-folding processes. To address this issue, we curate a dataset of fabric keypoints through the collection of real fabric images.

This paper presents the introduction of a novel quasi-static motion primitive called pick & drag. This primitive action is designed to address situations where sections of long-sleeved T-shirt sleeves are concealed or tucked beneath other fabrics. Additionally, we propose FabricFolding, a system that leverages the fabric's current coverage and corresponding keypoint information to intelligently select suitable primitive actions for unfolding the fabric from arbitrary initial configurations and subsequently folding the unfolded fabric. The system achieves the complete operational process of unfolding and folding for fabric with any initial configuration. Importantly, it no longer segregates fabric unfolding and folding into two distinct and independent research tasks. The system comprises two components: fabric unfolding and folding. The first component utilizes a self-supervised network to acquire knowledge of specific grasping points by analyzing RGBD input images. This knowledge is then applied to smoothen and unfold the fabric, which initially exists in a wrinkled state. The second component combines fabric keypoint detection with heuristic folding methods to enable efficient and precise folding of the fabric subsequent to its smoothening and unfolding. Moreover, the design of our fabric keypoints network eliminates the need for expert demonstrations and provides critical assistance during the fabric's unfolding stage. Figure 1 shows the FabricFolding work example. In instances of low fabric coverage, the system employs the fling action for efficient fabric unfolding. When the system detects some sleeves concealed beneath other fabric layers, the pick & drag action is chosen to address this issue. Subsequently, once the fabric is sufficiently unfolded and keypoints are detected, our method employs the pick & place action to achieve heuristic folding.

The contributions of this paper are summarized as:

- We enhance a quasi-static primitive action, namely pick & drag, to facilitate fabric unfolding, particularly when dealing with intricate fabric configurations like when sections of long-sleeved T-shirts are concealed or tucked beneath other layers of fabric.
- We propose a self-supervised learning unfolding strategy with a multi-primitive policy that can choose between dynamic fling and quasi-static (including pick & place and pick & drag) actions to efficiently unfold the fabric.
- To bridge the sim2real gap and enhance the accuracy of fabric keypoint detection, we conduct an extensive data collection process involving real images of various fabric types. Subsequently, we curate a dedicated dataset specifically designed for fabric-folding keypoint detection.

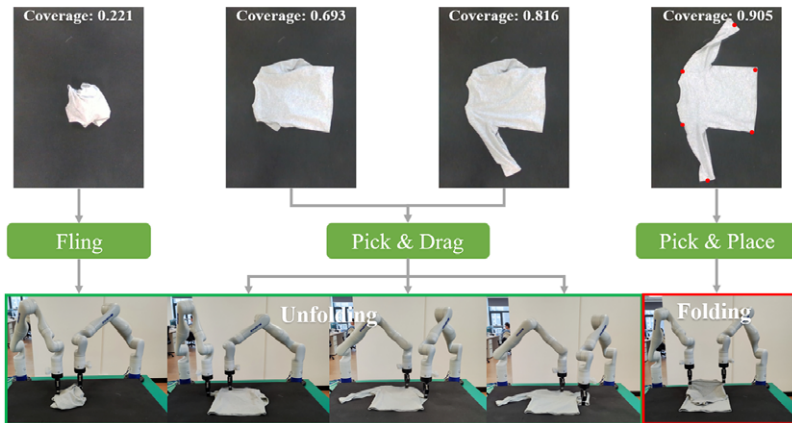


Figure 1. *FabricFold* divides the task of fabric folding into two stages for any given initial configuration. The first stage aims to achieve a relatively smooth fabric by dynamically selecting and executing actions (dynamic or quasi-static) based on the current state of the fabric. The second stage commences once the fabric is sufficiently unfolded and involves performing the folding task by detecting the keypoints of the fabric.

- Our method has been thoroughly evaluated using real robotic arms on a diverse range of fabrics. Specifically, it achieves a coverage rate of 0.822 and a fold success rate of 0.88 for long-sleeved T-shirts. Additionally, our method demonstrates a coverage rate of 0.958 and a fold success rate of 0.92 for towels.

This paper is organized as follows. Section 2 reviews some related works. Section 3 presents the *FabricFolding* method. The corresponding experiments are reported and analyzed in Section 4. Finally, Section 5 concludes this work and discusses some future work.

2. Related work

2.1. Fabric unfolding

Fabric unfolding mainly changes the fabric from an arbitrary crumpled configuration to a fully unfolded configuration. Prior work on fabric unfolding is based on heuristics and the extraction of some geometric features of the fabric, such as the edges [5], wrinkles [6, 13], and corners [14, 15] of the fabric. Then, these features are utilized to determine the subsequent manipulation to make the fabric as smooth as possible. Recently, reinforcement learning combines hard-coded heuristics [16] or expert demonstrations [17] to unfold fabric. Wu *et al.* [8] introduce self-supervised learning into fabric unfolding to replace the role of expert demonstrations or heuristics.

The algorithms previously discussed employ quasi-static primitive actions for fabric manipulation, including pick & place [18], and drag [17]. However, fabric unfolding based on quasi-static manipulation may require numerous iterations and interactions before achieving a relatively smooth fabric. This method's effectiveness can be hampered by the limited reach of a robotic arm and a single gripper's operational constraints, rendering the task impractical, particularly for intricate fabrics like T-shirts.

Compared with quasi-static manipulation, dynamic manipulation of robotic arms involves high-speed motion, which imparts velocity to the grasped deformable object. When the robotic arms come to a stop, the ungrasped portion of the deformable object continues moving due to inertia. Dynamic manipulation effectively expands the reachable fabric area and reduces the number of operations necessary to accomplish the desired task. Dynamic manipulation is initially applied to linearly deformable objects, such as cables [19]. Wang *et al.* [20] employ tactile sensors to capture information about objects and

integrate this data with recurrent neural networks to accomplish precise swinging movements. Jangir *et al.* [21] utilize reinforcement learning for dynamic operations on towels. However, this research was limited to simulations and did not include physical experiments. Ha *et al.* [10] propose a system based on self-supervised learning that dramatically increases the effectiveness of fabric unfolding by using high-speed fling action to smooth wrinkled fabric. Based on the research conducted by Ha *et al.*, Xu *et al.* [22] employ a commodity centrifugal air pump to smooth the fabric more efficiently. Dynamic flings alone, however, cannot fully unfold complicated textiles, such as long-sleeved T-shirts.

Employing multiple primitive actions is more effective than a singular primitive action for fabric-unfolding tasks. Avigal *et al.* [11] introduce the BiMaMa-Net architecture, which enables the selection of multiple primitive actions for fabric unfolding. Canberk *et al.* [12] present a clothing alignment algorithm designed for cloth unfolding. This algorithm utilizes self-supervised learning to determine suitable primitive actions for cloth unfolding by encoding both the current fabric state and the aligned fabric state. However, these algorithms generally do not account for intricate fabric configurations, such as when a portion of the long-sleeved T-shirt's sleeves is concealed beneath other layers of clothing. In contrast, Our algorithm demonstrates the capability to manage complex fabric configurations and is readily applicable in real-world scenarios.

2.2. Fabric folding

Fabric folding initially relied on heuristic algorithms that impose strict predefined constraints on the fabric's initial configuration [7, 23, 24]. Contemporary approaches to fabric manipulation involve training goal-conditioned policies using reinforcement learning [25–27], self-supervised learning [9, 28, 29], and imitation learning [16] in either simulated [30–32] or real robotic arms [9]. However, the strategy of employing simulation data for training essentially cannot achieve the desired effect of the simulation environment due to the sim2real gap on the robotic manipulator [26]. Moreover, a gap remains in the ability to generalize the approach to various types of fabrics [32]. To generalize fabric folding across different types of fabrics, some researchers [12, 33] proposed an approach that combines the detection of fabric keypoints with a heuristic for the folding process. However, owing to the absence of a dataset containing fabric keypoints derived from real fabric images, their study relies solely on simulated image data to generate diverse fabric keypoints datasets. This limitation affects the accuracy of fabric keypoints detection when applied in real-world scenarios. Instead, our fabric-folding keypoint dataset, created through the collection of real fabric images, is utilized for training. This approach effectively mitigates the issue of inadequate prediction precision attributable to the sim2real gap and enhances the success rate of fabric folding.

3. Method

FabricFolding is a novel approach for folding fabric, which involves breaking down the process into two distinct steps. Firstly, a multi-primitive policy unfolding algorithm is applied to ensure that the fabric is unfolded and smoothed out as much as possible. Subsequently, an adaptive folding manipulation technique is employed, which leverages keypoint detection to facilitate the folding of various types of fabric. FabricFolding can achieve effective and efficient fabric folding by utilizing this two-step approach, and the pipeline of the system is shown in Fig. 2.

3.1. Multi-primitive policy fabric unfolding

While dynamic actions can efficiently unfold fabrics but cannot entirely unfold complex garments like long-sleeved T-shirt, quasi-static actions are preferable for delicately handling fabrics but are inefficient. To efficiently manipulate the fabric into a desired configuration, we utilize a combination of quasi-static and dynamic primitives that are capable of delicate handling and efficient unfolding, respectively.

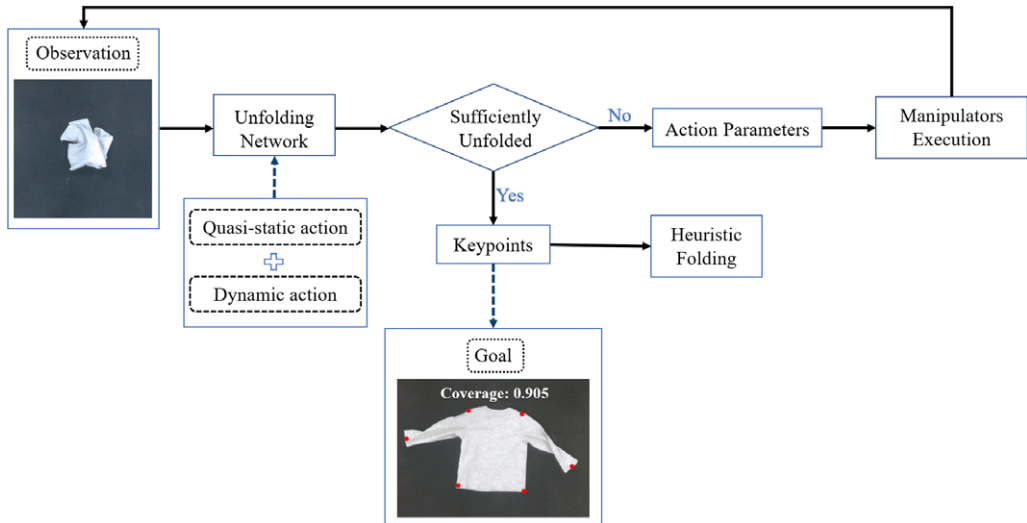


Figure 2. FabricFolding pipeline: The RGB image and depth image obtained from an overhead camera serve as inputs to the unfolding network, which generates a set of keypoints and action parameters. If the fabric is sufficiently unfolded and corresponding keypoints are detected, the system will proceed to fold the fabric using heuristics based on the keypoints. Otherwise, the dual-arm manipulation system will execute relevant primitive actions to unfold the fabric.



Figure 3. Fabric unfolding: The system adopts a dynamic fling action to unfold the fabric when it is in a low-coverage stacking configuration. However, when the fabric coverage exceeds S_1 , quasi-static actions such as pick & place and pick & drag are mainly used to fine adjustments. If the coverage of the fabric is greater than S_2 and the number of detected keypoints meets the requirements, the fabric is considered to be fully unfolded.

FabricFolding utilizes the RGBD image output from an overhead RealSense 435i camera to calculate the current fabric coverage. The fabric coverage \mathcal{C} is calculated as shown in Eq. (1).

$$\mathcal{C} = \frac{\sum_{i=1}^C \mathcal{P}_i}{\sum_{j=1}^T \mathcal{P}_j} \tag{1}$$

where \mathcal{P}_i represents the pixel occupied by the current fabric configuration in the image, while \mathcal{P}_j represents the pixel occupied in the image when the fabric is fully unfolded.

To determine whether the fabric is ready for the downstream folding task, we have two indicators: the fabric coverage \mathcal{C} and the number of detected keypoints \mathcal{K} . The fabric is considered smooth enough when the current coverage exceeds threshold S_2 and the number of detected keypoints meets the requirements. Figure 3 illustrates the fabric-unfolding pipeline with a multi-primitive policy. In our experiments, $S_1 = 0.65$ and $S_2 = 0.8$ perform best.

3.1.1. Primitive policy

To determine suitable primitive actions for efficient fabric unfolding, we have devised a primitive action weighting strategy based on heuristics, and the precise formula is detailed in Eq. (2). When the coverage is below a certain threshold (S_1), the robotic arm will use the dynamic fling action with a high probability to effectively unfold the fabric. As the coverage of fabric increases and exceeds S_1 , the system prefers to use quasi-static actions such as pick & place and pick & drag to operate the fabric. Finally, when the coverage is above S_2 , the robotic arms mainly rely on quasi-static actions to fine-tune the fabric. The action policy selection is shown below:

$$\mathcal{A}(\mathcal{C}) = \begin{cases} 0.99a_d + 0.01a_{qs} & , \mathcal{C} \leq S_1 \\ (S_2 - \mathcal{C}) a_d + [1 - (S_2 - \mathcal{C})] a_{qs} & , S_1 < \mathcal{C} < S_2 \\ 0.01a_d + 0.99a_{qs} & , S_2 \leq \mathcal{C} \end{cases} \quad (2)$$

where \mathcal{A} represents the chosen action, a_{qs} means the quasi-static including two actions of pick & place and pick & drag, and a_d means the dynamic primitive, fling.

Furthermore, when choosing between the two quasi-static primitive actions, the system relies on the detection of keypoints on the fabric sleeves. Specifically, when the distance between the keypoint on the sleeve and the keypoint on the shoulder on the same side significantly deviates from the standard value, the pick & drag primitive is chosen. Otherwise, the pick & place primitive is selected. The precise formula is presented in Eq. (3). For example, for long-sleeved T-shirts where part of the sleeve is concealed or tucked under other fabric, the robotic arms perform the pick & drag action to extract sleeves concealed beneath other fabrics, thereby increasing fabric coverage. Moreover, the pick & place action is also a quasi-static manipulation used to sufficiently smooth the fabric.

$$a_{qs} = \begin{cases} a_{pd} & , \mathcal{K}_{ss} \& (d_{ss} < d_0) \\ a_{pp} & , \text{others} \end{cases} \quad (3)$$

where a_{pd} means the pick & drag action, a_{pp} represents the pick & place action, \mathcal{K}_{ss} indicates whether the keypoints of the same side of the fabric are detected, such as the keypoints of the same side shoulder and sleeve of a long-sleeved T-shirt. Similarly, d_{ss} represents the distance between two detected keypoints on the same side, and d_0 represents the distance between the corresponding two keypoints when the fabric is fully unfolded.

All primitive actions are shown in Fig. 4, and the following are several primitive actions that we have defined:

- **Pick & Place:** The pick & place action is a quasi-static primitive. With a given pick pose and place pose, a single robotic arm grasps the fabric at the pick point, lifts it, moves it over the place point, and releases it. This primitive policy effectively handles situations where the hem or sleeves of the cloth are stacked on top of each other.
- **Pick & Drag:** It is a quasi-static primitive policy. Two pick poses are given, and the two robotic arms grasp the two pick points of the fabric, respectively. Then, one robotic arm remains stationary (close to the center of the fabric mask), while the other robotic arm drags the fabric away from the center point of the fabric mask for a certain distance. This primitive policy is effective in dealing with situations where most of the sleeves are concealed or tucked beneath other layers of cloth.
- **Fling:** This is a dynamic primitive policy designed for fabric manipulation. Given two pick poses. After the robotic arms grasp the two pick points of the fabric, the fabric is lifted to a certain height and stretched, while the camera in front of the robotic arms is used to estimate whether the fabric has been fully stretched. Then the two robotic arms simultaneously fling the fabric forward for a certain distance, then retreat for a certain distance while gradually reducing the height, and then

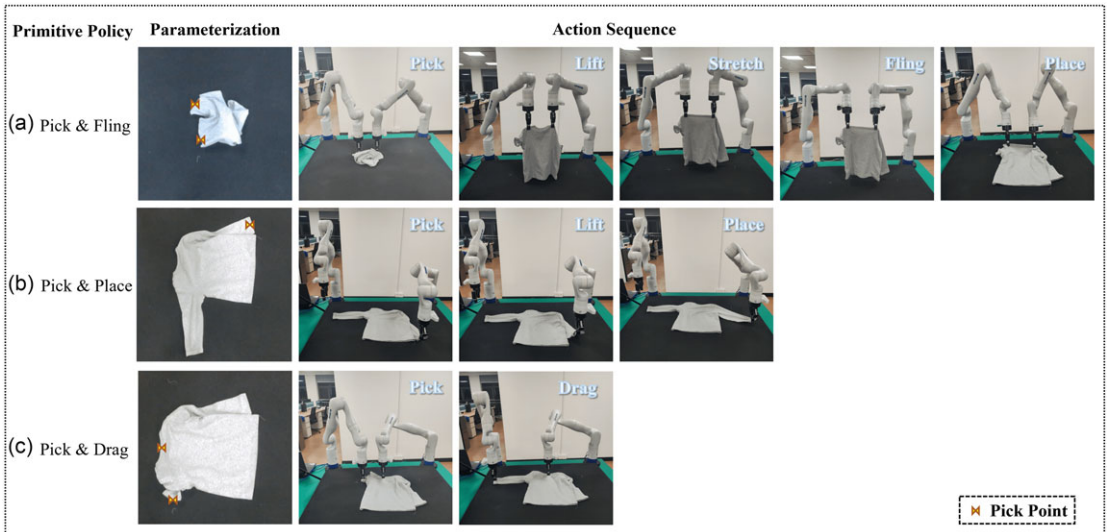


Figure 4. Primitive actions: Based on the input received from the overhead RGBD camera, the grasping network can predict a series of pick poses for the upcoming primitive actions.

release the fabric. This policy efficiently spreads the fabric and increases coverage, but it may not be effective in dealing with smaller wrinkles in the cloth.

- **Fold:** Both robotic arms execute the pick & place primitive action simultaneously. The two pick poses and their corresponding place poses are obtained through keypoint detection of the fabric.

3.1.2. Grasping action parameterization

To enhance the effectiveness of fabric unfolding and ensure that the fabric becomes smoother, we have made enhancements to the grasping framework of DextAIRity [22]. These improvements include modifying the grasping action parameters and incorporating a primitive action selection block. These modifications allow for more accurate and efficient predictions of the grasp poses required for subsequent primitive actions.

DextAIRity [22] made some adjustments to Flingbot’s [10] action parameterization, extending the two grasping positions L and R to the edge of the fabric mask, thereby reducing the likelihood of the gripper grasping multiple layers of fabric. Based on the action parameterization form of DextAIRity (C, θ, ω), we have made some minor adjustments, and our action parameters are shown in Eq. (4).

$$\mathcal{P}_{epo} = (C, \theta, \omega, \phi) \tag{4}$$

where the meaning of (C, θ, ω) corresponds to its definition in DextAIRity, ϕ represents the angle of rotation of the manipulator’s end effector relative to the y-axis of the base frame. There are left and right grab points $L(x_l, y_l)$ and $R(x_r, y_r)$ for fabric manipulation. If the four parameters are obtained by direct training, it is relatively difficult. To simplify the training difficulty, C is used to represent the midpoint of L-R, θ is used to represent the rotation angle of line L-R in the image plane, and ω is used to represent the length of line L-R in the pixel coordinate system. For a visual representation of the robot arm end effector’s coordinate system, please refer to Fig. 5. This parameter primarily serves the purpose of adjusting the clamping claw’s angle to facilitate the robot arm in grasping the fabric more effectively.

3.1.3. Unfolding network

The feature extraction block employs the Unet [34] architecture. In the encoder section, a feature map is derived from the $H \times W \times 3$ input. The feature map is then processed by the decoder, where it is resized

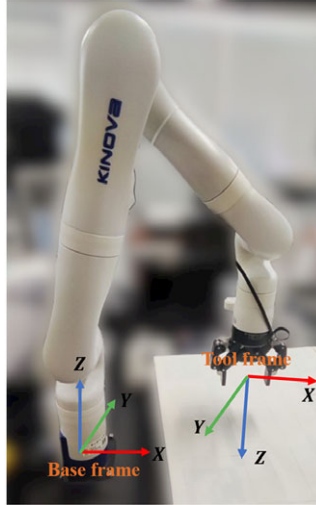


Figure 5. The left arm coordinate system in the dual-arm manipulation system.

to match the dimensions of the original image. During this process, high-level and low-level features are integrated using skip connections. The decoder at each layer is processed using in Eq. (5).

$$X_{De}^k = \begin{cases} X_{En}^k & , k = N \\ \mathcal{F}([C_{ov}(X_{En}^k), C_{ov}(\mathcal{U}(X_{De}^{k+1}))]) & , k = N - 1 \end{cases} \quad (5)$$

where k represents the down-sampling layer of the encoder, N denotes the total number of layers within the encoder, and $[\cdot]$ means skip connection. $\mathcal{F}(\cdot)$ indicates the process of feature fusion employing convolution, batch normalization, and ReLU activation function. $\mathcal{U}(\cdot)$ performs the up-sampling operation, and $C_{ov(\cdot)}$ implies convolution operation.

This grasping point prediction block comprises two integral parts: the primitive action selection block and the value map block. Specifically, to enhance the efficiency of fabric unfolding, the primitive action selection block assesses and selects appropriate primitive actions based on the current fabric coverage and the detected keypoints. Once the primitive action is determined, a set of action parameterizations denoted as \mathcal{P}_{epo} is acquired through the action value map module, where the parameters with the highest values serve as the grasping parameters. To achieve equivariance between the grasping action and the physical transformation of the fabric, we employ a spatial action map [10, 35]. The unfolding network structure is shown in Fig. 6.

The spatial action value map, employing a convolutional neural network and ResNet-inspired skip connections, predicts a set of grab values in the pixel space, characterized by constant scale and rotation. Specifically, the input is a top-down observation of $H \times W \times 3$. This input undergoes preprocessing to produce a series of rotated and scaled views, denoted as $K \times H \times W \times 3$. The preprocessing state includes 18 rotations, covering a full 360° range, and 5 scaling levels with values of $\{1, 1.5, 1.75, 2, 2.5\}$ ($K = 90$). Subsequently, a collection of value maps is predicted, wherein each pixel encompasses values set for action parameters (C, θ, ω) . The predictions of our value network are supervised by the delta coverage before and after action execution, utilizing mean squared error as the loss function.

3.1.4. Training setting

Before training the grasping network, the keypoint detection network is trained first through supervised learning. The sequence is necessary as the training of the fabric keypoint detection network is a prerequisite for providing the current keypoint information needed by the grasping points prediction network.

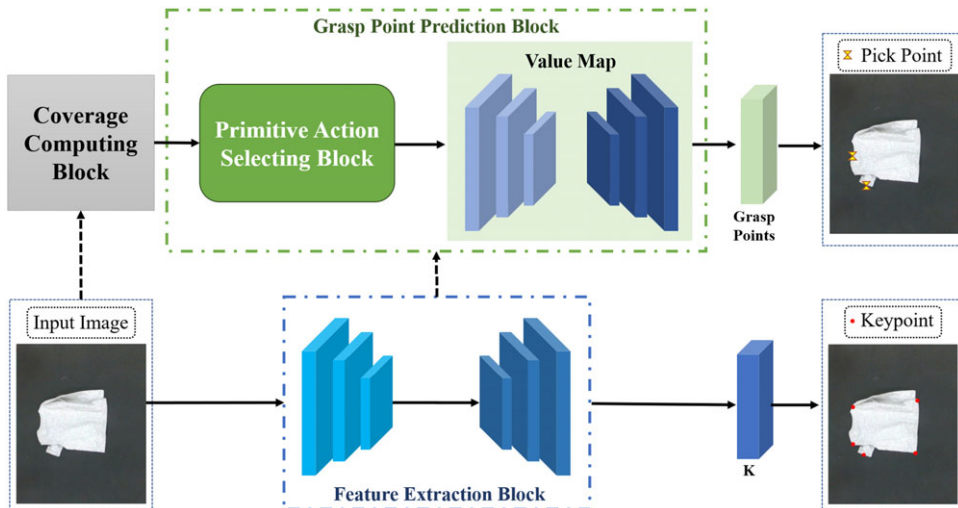


Figure 6. Fabric-folding structure: The RGB image captured by the overhead RealSense 435i camera serves as the input for our system.

We employed a simulation environment based on SoftGym [36] for the pre-training of the fabric-unfolding grasping network through self-supervised learning. This primarily involved training three parameters (C , θ , ω). Due to the simulation environment's inability to import the URDF models of the robotic arms, we utilized two grasp points to represent the end effectors of the two robotic arms. Certain constraints were imposed on these points to ensure their applicability to the physical robotic arms. Subsequently, the network parameters acquired during simulation training were further trained in real-world conditions. Parameter ϕ was introduced during this stage to facilitate the fabric gripping by the robot arm. To prevent potential damage to the cameras of the dual-arm manipulation system during manipulation, we imposed a constraint requiring the left and right robotic arms to rotate 90° and -90° , respectively, relative to the base frame's Z-axis. This ensured that the two cameras faced in opposite directions.

3.2. Heuristic fabric folding

3.2.1. Dataset for keypoint detection

Due to the sim2real gap, the keypoint detection performance of the fabric is not optimal when using the fabric keypoint dataset generated in the simulation, which includes various configurations of fabric. For example, certain configurations of fabric cannot detect all the expected keypoints. Additionally, there is no existing public dataset that is suitable for keypoint detection in fabric-folding tasks. In order to address this issue, we created a fabric keypoint dataset consisting of 1809 images that include four types of long-sleeved T-shirts and ten types of towels. Among them, all fabrics are configured to have a coverage of more than 65% since the information on the keypoints of the fabric mainly is used in the case of high fabric coverage. The long-sleeved T-shirt is sampled by rotating it 360° at intervals of 10° , involving eight distinct fabric coverages per angle. Similarly, in the case of the towel, a 180° rotation at 18° intervals was performed, with six different fabric coverages at each angle. The dataset images are acquired using a Realsense 435i camera positioned 1.2 m above the fabric with an image resolution of 640×480 pixels. Additionally, we considered the impact of real-world lighting conditions, capturing images of the fabric under various lighting scenarios.

For towels, we define four keypoints, starting with the upper left corner of the image as corner 1, and continuing in clockwise order as corner 2, corner 3, and corner 4. Long-sleeved T-shirts have five keypoints, including right_shoulder, left_shoulder, right_sleeve, right_waist, left_waist, and left_sleeve,

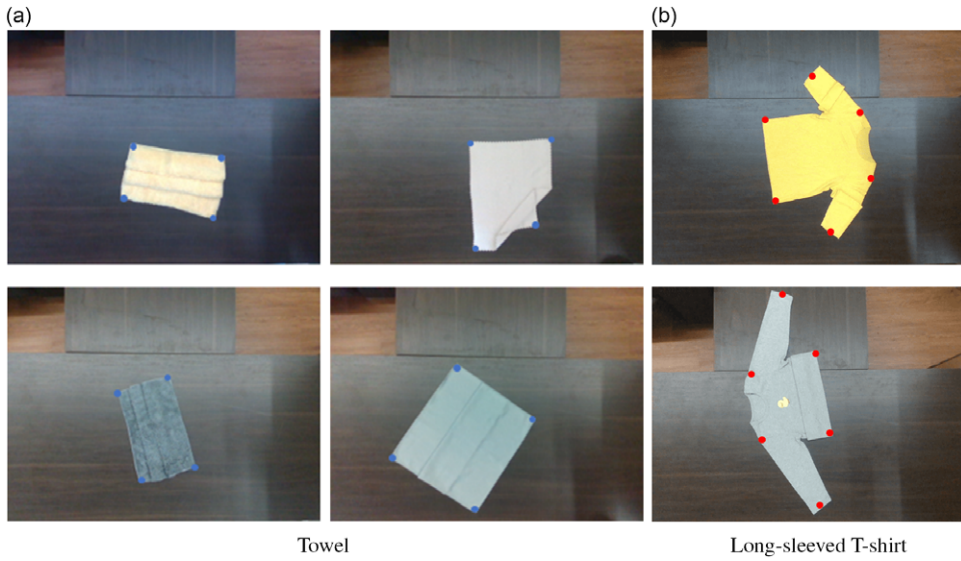


Figure 7. Sample keypoint detection dataset for fabric folding.

which is marked according to the outward direction of the vertical image. An example of the fabric keypoint dataset can be seen in Fig. 7.

3.2.2. Keypoint detection network

To identify keypoints in fc , the heatmap [37] regression method is utilized for pinpointing their locations. For a d -dimensional heatmap $g_i(x) : \mathbb{R}^d \rightarrow \mathbb{R}$ encompassing M keypoints, the coordinate $\mathbf{x}_i \in \mathbb{R}^d$ of a specific target keypoint $L_i, i = \{1, \dots, M\}$ is represented using a Gaussian function.

$$g_i(\mathbf{x}; \sigma_i) = \frac{\gamma}{(2\pi)^{d/2} \sigma_i^d} \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_i^*\|_2^2}{2\sigma_i^2}\right) \tag{6}$$

Eq. (6) indicates that in the heatmap image, pixel values near the target coordinate \mathbf{x}_i^* are higher, diminishing rapidly with increasing distance from \mathbf{x}_i^* . To circumvent numerical instability during training, arising from the Gaussian function’s generation of tiny values, a scaling factor γ is introduced. For each dimension, the standard deviation σ_i determines the Gaussian function’s peak width in the heatmap of keypoint L_i . Consequently, σ_i represents a variable parameter of the Gaussian function $g_i(x)$, learned concurrently with the network’s weights and biases. It ensures the learning of an optimal peak width for the heatmap of each keypoint.

The network is trained to generate M heatmaps by minimizing the discrepancy between the predicted heatmaps $h_i(\mathbf{x}; \mathbf{w}, \mathbf{b})$ for all keypoints L_i and their corresponding ground truth heatmaps $g_i(\mathbf{x}; \sigma_i)$. This process is described in Eq. (7). When the network confidently predicts keypoints, it generates heatmaps with narrower peak widths; conversely, for keypoints with lower confidence, it produces heatmaps with wider peaks. The parameter ϵ serves as the penalty factor that determines the peak width of the heatmaps, while η modulates the impact of the network’s weight L_2 norm to mitigate the risk of overfitting.

$$\min_{\mathbf{w}, \mathbf{b}, \sigma} \sum_{i=1}^M \sum_{\mathbf{x}} \|h_i(\mathbf{x}; \mathbf{w}, \mathbf{b}) - g_i(\mathbf{x}; \sigma_i)\|_2^2 + \epsilon \|\sigma\|_2^2 + \eta \|\mathbf{w}\|_2^2 \tag{7}$$

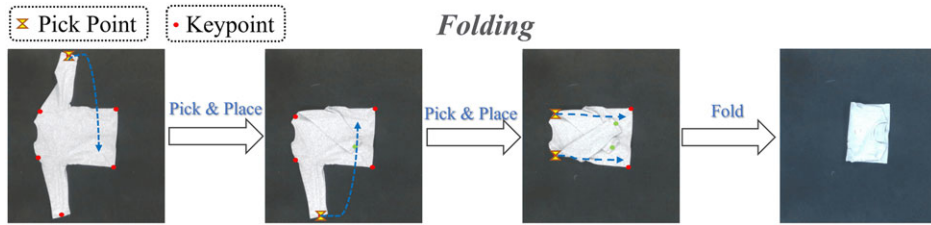


Figure 8. Fabric folding: Based on the input received from the overhead RGBD camera, the grasp network can predict a series of pick poses for the upcoming primitive actions.

Finally, the predicted coordinates $\hat{\mathbf{x}}_i \in \mathbb{R}^d$ of each keypoint L_i are determined by identifying the locations of the maximum values on the heatmaps, as shown in Eq. (8).

$$\hat{\mathbf{x}}_i = \arg \max_{\mathbf{x}} h_i(\mathbf{x}; \mathbf{w}, \mathbf{b}) \quad (8)$$

We divide the data into training and validation sets in an 8:2 ratio. After training for 4 hours on an NVIDIA RTX3080Ti, the average pixel error of the detected keypoints on a 640x480 validation set image can be guaranteed to be within 3 pixels.

3.2.3. Heuristic folding

Taking inspiration from the Cloth Funnels [12] heuristic folding method, as depicted in Fig. 8, we use a similar approach for folding a long-sleeved T-shirt. Initially, we utilize the pick & place primitive action to fold the two sleeves of the garment onto the main part of the garment. Following this, we pick the keypoints of the shoulders and place them at the keypoints of the waist.

4. Experiments

We conducted an experiment to identify the optimal threshold S_1 . Additionally, we carried out ablation studies on primitive actions and multi-primitive policy. In the real world, we evaluated our method's effectiveness in unfolding fabric and its success rate in folding, using a range of long-sleeved t-shirts and towels. This was further complemented by a comparative analysis with data from several contemporary advanced algorithms.

4.1. Experimental setup

Our experimental setup comprises two Kinova Gen3 robotic arms, each outfitted with a Robotiq 2F-85 gripper. For visual data acquisition, we utilize a top-down Intel RealSense D435i camera for capturing RGBD information during fabric manipulation and another Intel RealSense D435i, positioned to monitor the robotic arm, to assess the degree of fabric stretch. Additionally, a Dell G15 5520 laptop and a sponge sheet are included for experimental deployment, as detailed in Fig. 9.

4.2. Metrics

We evaluate FabricFolding on both the fabric-unfolding and fabric-folding tasks. Performance in the fabric-unfolding task is evaluated based on the coverage achieved at the end of each episode. Furthermore, we evaluate the algorithm's success rate for the folding task, as well as its generalization ability to handle unseen fabrics on real robotic arms. To reduce experimental randomness, each experiment is repeated 25 times, and the weighted average of all results is calculated. If the folding result is deemed a failure by the majority of the 5 judges or if the folding cannot be completed after 20 action sequences, the experiment is considered a failure.

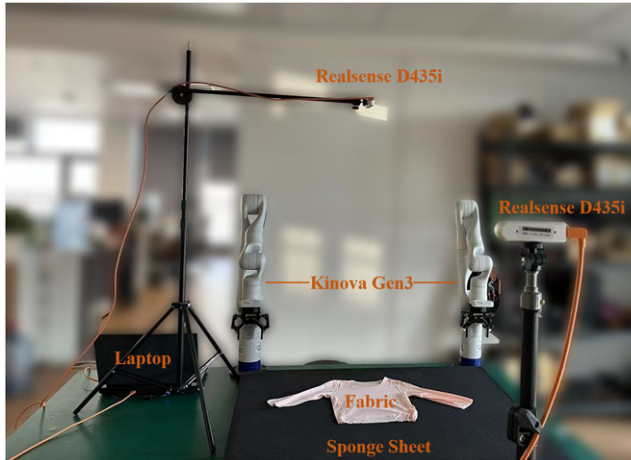


Figure 9. The experiment setup for the dual-arm manipulation system.

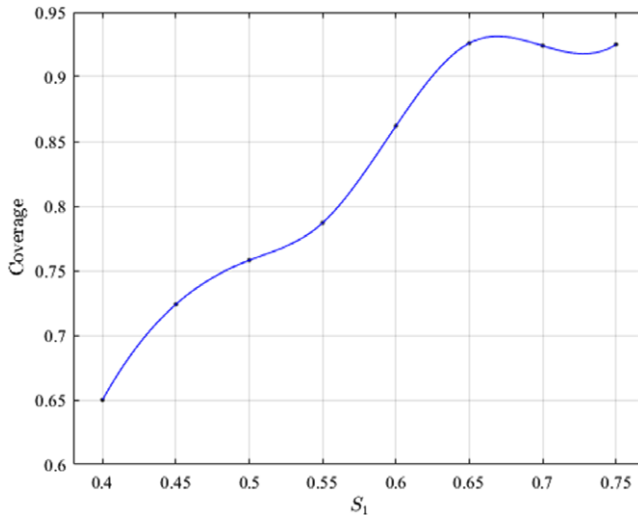


Figure 10. The normalized coverage of a long-sleeved T-shirt (Initial coverage is 0.5) is evaluated after five primitive actions under various threshold parameters S_1 .

4.3. Parameter optimization

To optimize the grasping network, we use S_1 as a threshold to determine the weight between dynamic and quasi-static primitive action. To determine the optimal value for S_1 , we compare the coverage achieved after 5 primitive actions for different values of S_1 .

When S_1 is below 0.5, and since the initial fabric coverage is 0.5, our primitive action selection strategy favors quasi-static primitive actions over dynamic ones. As shown in Fig. 10, the fabric's coverage after five operations does not exhibit a significant increase, suggesting that quasi-static primitive actions are less efficient in unfolding the fabric. For $0.5 < S_1 \leq 0.65$, our multi-primitive action selection strategy prompts the algorithm to assign greater weight to the dynamic primitive action fling. Figure 10 demonstrates that this action results in more efficient fabric unfolding. Besides, for $0.65 < S_1 \leq 0.75$, despite the algorithm continuing to heavily weigh the dynamic primitive action fling, the fabric's coverage after five operations is fluctuating. It indicates that while fling is efficient in unfolding the fabric, it is less capable of executing finer operations. Therefore, the best folding efficiency is achieved when $S_1 = 0.65$.

Table I. *Validity of primitive actions: the fabric is a long-sleeved T-shirt.*

Fabric Status	Primitive Actions		Cov. ↑
Ini_cov \geq 0.7	Quasi-static primitive action	Pick & Place (P&P)	0.872
		Pick & Drag (P&D)	0.876
		P&P + P&D	0.891
	Dynamic primitive action	Fling	0.818
	Muti-primitive actions	P&P + P&D + Fling	0.902
Ini_cov \leq 0.3	Quasi-static primitive action	Pick & Place (P&P)	0.624
		Pick & Drag (P&D)	0.636
		P&P + P&D	0.694
	Dynamic primitive action	Fling	0.742
	Multi-primitive actions	P&P + P&D + Fling	0.822

Table II. *Validity of Pick & Drag: the fabric is long-sleeved T-shirts with sleeves partially concealed beneath other layers of cloth.*

Fabric Status	Primitive Actions		Cov. ↑
Ini_cov \geq 0.75	Quasi-static primitive action	Pick & Place (P&P)	0.816
		Pick & Drag (P&D)	0.879
		P&P + P&D	0.895
	Dynamic primitive action	Fling	0.811
	Muti-primitive actions	P&P+P&D+Fling	0.899

4.4. Primitive actions ablation

Dynamic action can efficiently unfold the fabric, and quasi-static actions can make some fine adjustments to the fabric. To verify the effectiveness of our multi-primitive policy mechanism, we conduct experiments on long-sleeved T-shirts with different coverage. Table I shows that when the fabric has high initial coverage and mild self-occlusion, the quasi-static actions have better coverage compared to the dynamic action. This substantiates the capability of quasi-static actions to execute finer tasks in comparison to dynamic action. On the other hand, when the fabric has low initial coverage and severe self-occlusion, the dynamic action effectively improves coverage, which is in line with the findings of Flingbot [10]. It also demonstrates that our multi-primitive policy outperforms a single-primitive action in achieving higher coverage, regardless of the initial coverage of the fabric. This highlights the effectiveness of the multi-primitive policy in enhancing fabric coverage.

To assess the validity of the enhanced Pick & Drag primitive action in unfolding severely self-occluding fabrics, we conducted tests using a long-sleeved T-shirt, with certain sleeves concealed or tucked beneath other layers of cloth. Table II demonstrates that the enhanced pick & drag primitive action significantly outperforms other primitive actions in managing fabrics with severe self-coupling of sleeves. Furthermore, it highlights the superiority of our multi-primitive policy over a single-primitive approach.

4.5. Effectiveness on fabric category

Table III presents the coverage attained by various algorithms on diverse real-world fabrics with initial coverage less than 30%. Unfortunately, despite the availability of open-source code for SpeedFold [11] and Canberk [12], substantial challenges exist when it comes to deploying these two tasks in a dual-arm manipulation system using Kinova Gen3 due to inconsistencies in the robotic arms' utilization. Therefore, the data presented in this table are the original results reported in their respective papers.

Table III. *Real-world coverage: * indicates the original data in paper.*

Approach	Fabric	Cov. \uparrow
Flingbot [10]	Towels	0.905
	Long-sleeved T-shirts	0.742
SpeedFold [11]	Towels	0.92*
	T-shirts	0.8*
	Long-sleeved T-shirts	\
Canberk [12]	Towels	\
	Long-sleeved T-shirts	0.806*
Our	Towels	0.958
	Long-sleeved T-shirts	0.822

Table IV. *Folding's success: * indicates the original data in paper.*

Approach	Fabric	Success \uparrow
Doumanoglou [7]	Towels	0.78*
	T-shirts	0.66*
SpeedFold [11]	T-shirt	0.93*
	Long-sleeved T-shirts	\
Canberk [12]	Towels	\
	Long-sleeved T-shirts	0.878*
Our	Towels	0.92
	Long-sleeved T-shirts	0.88

To ensure a fair comparison, the fabrics tested in our algorithm are chosen to be as similar as possible to those used in the previous works.

In our experimental setup, we independently deployed and evaluated FlingBot [10] and our algorithm. As indicated in Table III, while FlingBot demonstrates proficiency with simpler fabrics like towels, it encounters challenges with more complex items such as long-sleeved T-shirts. In contrast, our algorithm not only effectively unfolds towels but also outperforms comparative algorithms in unfolding long-sleeved T-shirts, showcasing superior performance. In addition, it exhibits the efficacy of our advanced pick & drag primitive action and multi-primitive policy in fabric unfolding.

The success rate of fabric folding is a crucial performance indicator. As shown in Table IV, the complexity of the fabric has a direct effect on the success rate of folding, which decreases continuously as the fabric complexity increases. In particular, when dealing with fabrics that are not whole pieces, such as long-sleeved T-shirts with two sleeves, it is common for the fabric to self-occlude during the folding process to reduce the success rate. Among the four algorithms compared, our algorithm outperformed the others in terms of the success rate achieved in folding towels and long-sleeved T-shirts. The result of Table IV not only illustrates the significance of our enhanced pick & drag primitive action and multi-primitive policy in simplifying tasks like fabric folding but also confirms the pivotal role of our realistic image dataset in keypoint-based heuristic fabric folding.

5. Conclusions

In this paper, we enhance a quasi-static primitive action, pick & drag, enabling it to address severely self-occluding fabric scenarios, including instances such as long-sleeved T-shirts with sleeves concealed or tucked beneath other layers of cloth. Simultaneously, we have developed the FabricFolding system,

which dynamically selects multiple primitive actions to efficiently unfold fabric in any initial configuration. Additionally, we have developed a keypoint detection dataset for fabric folding to enhance the precision of fabric keypoint detection, consisting of approximately 2,000 images. Our algorithm achieves a coverage rate of 0.822 and a folding success rate of 0.88 for long-sleeved T-shirts. During our experiments, we found that when the two pick points of fling are at diagonally opposite corners of the fabric, it can be challenging to fully unfold the fabric even after multiple interactions. In the future, we plan to investigate this issue and develop solutions to address it.

Author contributions. Can He conceived and designed this work, including setting up the simulation and physical hardware platforms, conducting relevant experiments, production of the dataset, and manuscript writing. Lingxiao Meng was responsible for debugging the dual-arm coordination communication, while Zhirui Sun contributed to dataset production and hardware platform construction. Jiankun Wang and Max Q.-H. Meng provided guidance and supervision.

Financial support. This work is supported by Shenzhen Science and Technology Program under Grant RCBS20221008093305007, National Natural Science Foundation of China under Grant 62103181, Shenzhen Science and Technology Program under Grant 20231115141459001, Young Elite Scientists Sponsorship Program by CAST under Grant 2023QNRC001.

Competing interests. The authors declare no competing interests exist.

Ethical approval. Not applicable.

References

- [1] T. Fukuda, H. Dobashi, H. Nagano, Y. Tazaki, R. Katayama and Y. Yokokohji, “Jigless assembly of an industrial product by a universal robotic hand mounted on an industrial robot,” *Robotica* **41**(8), 2464–2488 (2023).
- [2] Y. Wu, Y. Fu and S. Wang, “Global motion planning and redundancy resolution for large objects manipulation by dual redundant robots with closed kinematics,” *Robotica* **40**(4), 1125–1150 (2022).
- [3] W. Bejjani, M. Leonetti and M. R. Dogar, “Learning image-based receding horizon planning for manipulation in clutter,” *Robot. Auton. Syst.* **138**, 103730 (2021).
- [4] K. Ren, L. E. Kavraki and K. Hang, “Rearrangement-based Manipulation via Kinodynamic Planning and Dynamic Planning Horizons,” **In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)**, (2022) pp. 1145–1152.
- [5] D. Triantafyllou, I. Mariolis, A. Kargakos, S. Malassiotis and N. Aspragathos, “A geometric approach to robotic unfolding of garments,” *Robot. Auton. Syst.* **75**, 233–243 (2016).
- [6] L. Sun, G. Aragon-Camarasa, P. Cockshott, S. Rogers and J. P. Siebert, “A Heuristic-based Approach for Flattening Wrinkled Clothes,” *Towards Autonomous Robotic Systems: 14th Annual Conference, TAROS 2013, Oxford, UK, August 28-30, 2013, Revised Selected Papers 14* (2014) pp. 148–160.
- [7] A. Doumanoglou, J. Stria, G. Peleka, I. Mariolis, V. Petrik, A. Kargakos, L. Wagner, V. Hlaváč, T.-K. Kim and S. Malassiotis, “Folding clothes autonomously: A complete pipeline,” *IEEE Trans. Robot.* **32**(6), 1461–1478 (2016).
- [8] Y. Wu, W. Yan, T. Kurutach, L. Pinto and P. Abbeel, “Learning to Manipulate Deformable Objects without Demonstrations,” **In: Proceedings of the 2020 Robotics and Systems (RSS)**, (2020).
- [9] R. Lee, D. Ward, V. Dasagi, A. Cosgun, J. Leitner and P. Corke, “Learning Arbitrary-goal Fabric Folding with One Hour of Real Robot Experience,” **In: Proceedings of the 2020 Conference on Robot Learning**, PMLR **155**, pp. 2317–2327 (2021).
- [10] H. Ha and S. Song, “Flingbot: The Unreasonable Effectiveness of Dynamic Manipulation for Cloth Unfolding,” **In: Proceedings of the 2022 Conference on Robot Learning**, PMLR **164**, pp. 24–33 (2022).
- [11] Y. Avigal, L. Berscheid, T. Asfour, T. Kröger and K. Goldberg, “Speedfolding: Learning Efficient Bimanual Folding of Garments,” **In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)**, IEEE (2022) pp. 1–8.
- [12] A. Canberk, C. Chi, H. Ha, B. Burchfiel, E. Cousineau, S. Feng and S. Song, “Cloth Funnels: Canonicalized-Alignment for Multi-purpose Garment Manipulation,” **In: 2023 IEEE International Conference of Robotics and Automation (ICRA)**, (2023) pp. 5872–5879.
- [13] H. Yuba, S. Arnold and K. Yamazaki, “Unfolding of a rectangular cloth from unarranged starting shapes by a dual-armed robot with a mechanism for managing recognition error and uncertainty,” *Adv. Robot.* **31**(10), 544–556 (2017).
- [14] J. Maitin-Shepard, M. Cusumano-Towner, J. Lei and P. Abbeel, “Cloth Grasp Point Detection based on Multiple-view Geometric Cues with Application to Robotic Towel Folding,” **In: 2010 IEEE International Conference on Robotics and Automation (ICRA)**, (2010) pp. 2308–2315.
- [15] D. Seita, N. Jamali, M. Laskey, R. Berenstein, A. K. Tanwani, P. Baskaran, S. Iba, J. Canny and K. Goldberg, “Deep transfer learning of pick points on fabric for robot bed-making,” **In: 2019 The International Symposium of Robotics Research (ISRR)**, Springer (2019) pp. 275–290.

- [16] 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)*, (2020) pp. 9651–9658.
- [17] A. Ganapathi, P. Sundaresan, B. Thananjeyan, A. Balakrishna, D. Seita, J. Grannen, M. Hwang, R. Hoque, J. E. Gonzalez, N. Jamali, et al., “Learning Dense Visual Correspondences in Simulation to Smooth and Fold Real Fabrics,” *In: 2021 IEEE International Conference on Robotics and Automation (ICRA)*, (2021) pp. 11515–11522.
- [18] D. Morrison, A. W. Tow, M. Mctaggart, R. Smith, N. Kelly-Boxall, S. Wade-McCue, J. Erskine, R. Grinover, A. Gurman, T. Hunn, et al., “Cartman: The Low-cost Cartesian Manipulator that Won the Amazon Robotics Challenge,” *In: 2018 IEEE International Conference on Robotics and Automation (ICRA)*, (2018) pp. 7757–7764.
- [19] V. Lim, H. Huang, L. Y. Chen, J. Wang, J. Ichnowski, D. Seita, M. Laskey and K. Goldberg, “Real2sim2real: Self-supervised Learning of Physical Single-step Dynamic Actions for Planar Robot Casting,” *In: 2022 IEEE International Conference on Robotics and Automation (ICRA)*, (2022) pp. 8282–8289.
- [20] C. Wang, S. Wang, B. Romero, F. Veiga and E. Adelson, “Swingbot: Learning Physical Features from In-hand Tactile Exploration for Dynamic Swing-up Manipulation,” *In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (2020) pp. 5633–5640.
- [21] R. Jangir, G. Alenya and C. Torras, “Dynamic Cloth Manipulation with Deep Reinforcement Learning,” *In: 2020 IEEE International Conference on Robotics and Automation (ICRA)*, (2020) pp. 4630–4636.
- [22] Z. Xu, C. Chi, B. Burchfiel, E. Cousineau, S. Feng and S. Song, “Dexterity: Deformable Manipulation can be a Breeze,” *In: Proceedings of the 2022 Robotics: Science and Systems (RSS)*, (2022).
- [23] M. Cusumano-Towner, A. Singh, S. Miller, J. F. O’Brien and P. Abbeel, “Bringing Clothing into Desired Configurations with Limited Perception,” *In: 2011 IEEE International Conference on Robotics and Automation*, (2011) pp. 3893–3900.
- [24] I. G. Camacho, J. B.às Sol and G. A.à Ribas, “Knowledge Representation to Enable High-Level Planning in Cloth Manipulation Tasks,” *In: Proceeding of the 2022 ICAPS Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*, (2022) p. 9.
- [25] J. Hietala, D. Blanco-Mulero, G. Alcan and V. Kyrki, “Learning Visual Feedback Control for Dynamic Cloth Folding,” *In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (2022) pp. 1455–1462.
- [26] J. Matas, S. James and A. J. Davison, “Sim-to-real Reinforcement Learning for Deformable Object Manipulation,” *In: Proceedings of the 2018 Conference on Robot Learning*, PMLR **87**, pp. 734–743 (2018).
- [27] P.-C. Yang, K. Sasaki, K. Suzuki, K. Kase, S. Sugano and T. Ogata, “Repeatable folding task by humanoid robot worker using deep learning,” *IEEE Robot. Autom. Lett.* **2**(2), 397–403 (2016).
- [28] R. Hoque, D. Seita, A. Balakrishna, A. Ganapathi, A. K. Tanwani, N. Jamali, K. Yamane, S. Iba and K. Goldberg, “Visuospatial foresight for physical sequential fabric manipulation,” *Auton. Robot.*, 1–25 (2022).
- [29] Z. Huang, X. Lin and D. Held, “Mesh-based Dynamics with Occlusion Reasoning for Cloth Manipulation,” *In: Proceedings of the 2022 Robotics: Science and Systems (RSS)*, (2022).
- [30] K. Mo, C. Xia, X. Wang, Y. Deng, X. Gao and B. Liang, “Foldsformer: Learning sequential multi-step cloth manipulation with space-time attention,” *IEEE Robot. Autom. Lett.* **8**(2), 760–767 (2022).
- [31] B. Thach, B. Y. Cho, A. Kuntz and T. Hermans, “Learning Visual Shape Control of Novel 3D Deformable Objects from Partial-view Point Clouds,” *In: 2022 IEEE International Conference on Robotics and Automation (ICRA)*, (2022) pp. 8274–8281.
- [32] T. Weng, S. M. Bajracharya, Y. Wang, K. Agrawal and D. Held, “Fabricflownet: Bimanual Cloth Manipulation with a Flow-based Policy,” *In: Proceedings of the 2022 Conference on Robot Learning*, PMLR **164**, pp.192–202 (2022).
- [33] T. Lips, V.-L. De Gussemme, et al., “Learning keypoints from synthetic data for robotic cloth folding,” *In: 2nd Workshop on Representing and Manipulating Deformable Objects - ICRA* (2022).
- [34] O. Ronneberger, P. Fischer and T. Brox, “U-net: Convolutional Networks for Biomedical Image Segmentation,” *In: Proceedings of the 2015 Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, Springer (2015) pp. 234–241.
- [35] J. Wu, X. Sun, A. Zeng, S. Song, J. Lee, S. Rusinkiewicz and T. Funkhouser, “Spatial Action Maps for Mobile Manipulation,” *In: Proceedings of the 2020 Robotics: Science and Systems (RSS)*, (2020).
- [36] X. Lin, Y. Wang, J. Olkin and D. Held, “Softgym: Benchmarking Deep Reinforcement Learning for Deformable Object Manipulation,” *In: Proceedings of the 2021 Conference on Robot Learning*, PMLR **155**, pp. 432–448 (2021).
- [37] C. Payer, D. Štern, H. Bischof and M. Urschler, “Integrating spatial configuration into heatmap regression based cnns for landmark localization,” *Med. Image Anal.* **54**, 207–219 (2019).