2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

TRTM: Template-based Reconstruction and Target-oriented Manipulation of Crumpled Cloths

Wenbo Wang, Gen Li, Miguel Zamora, and Stelian Coros

***Abstract*— Precise reconstruction and manipulation of the crumpled cloths is challenging due to the high dimensionality of cloth models, as well as the limited observation at self-occluded regions. We leverage the recent progress in the field of single-view reconstruction to template-based reconstruct the crumpled cloths from their top-view depth observations only, with our proposed sim-real registration protocols. In contrast to previous implicit cloth representations, our reconstruction mesh explicitly describes the positions and visibilities of the entire cloth mesh vertices, enabling more efficient dual-arm and single-arm target-oriented manipulations. Experiments demonstrate that our TRTM system can be applied to daily cloths that have similar topologies as our template mesh, but with different shapes, sizes, patterns, and physical properties. Videos, datasets, pre-trained models, and code can be downloaded from our project website: `https://wenbwa.github.io/TRTM/`.**

I. Introduction

Cloth products have been a crucial part of our daily life, where repeated human resources are spent on cloth arranging tasks. For this reason, several studies have been performed to identify [1], perceive [2], and organize [3], [4] different cloth items using both computer vision and robotic approaches.

However, manipulating while perceiving the entire state of one crumpled cloth is challenging due to the complex cloth model and the limited observation at self-occluded regions. In previous research, crumpled cloths are mostly represented as either visible pixel values [5], [6], [7], sampled surface points [8], sparse feature groups [9], [10], or encoded latent vectors [11], as shown in Fig. 1. They train or optimize their implicit manipulation policies with those implicit and simplified cloth representations by either reinforcement learning or dynamics learning mostly within the simulation environment. Few of the previous work fully and precisely understand the entire crumpled cloth configuration, not to mention explicitly locating and manipulating the visible cloth mesh vertices.

Different from the previous studies, we employ the template-based graph neural networks (GNNs) to explicitly reconstruct the entire meshes of crumpled cloths, using their top-view depth observations only. We demonstrate that, with our sim-real registration protocols, the distribution of cloth configurations in the real world can be properly described using adequate simulated cloth meshes, among which the ground truth mesh of each depth observation is known. Benefiting from our explicit mesh representation, the task of manipulating crumpled cloths to some target configurations can be more efficiently performed with fewer operation episodes. Our work contributes to the following:

The authors are with ETH Zurich, Switzerland. {wenbwang}@student.ethz.ch, {gen.li, mimora, scoros}@inf.ethz.ch

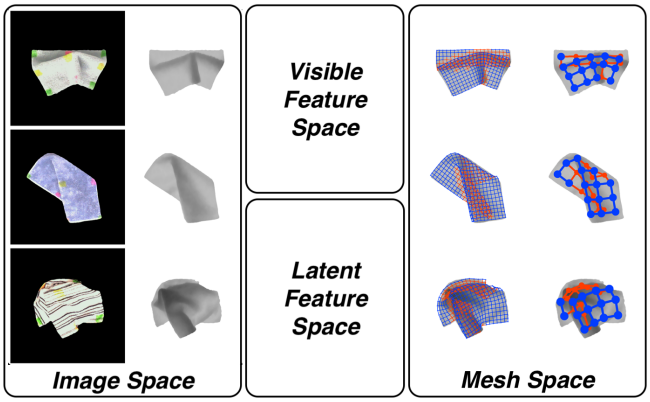

Fig. 1. Different state representations of crumpled cloths. From left to right: top-view color images; top-view depth images or point clouds; sparse visible features or encoded latent vectors; our template-based reconstruction mesh and clustered mesh group for robot manipulation. From top to bottom: configurations of the randomly one-time dragged rectangle cloth, two-times folded template square cloth, and one-time dropped larger square cloth.

1) a novel template-based reconstruction method that can explicitly predict the positions and visibilities of the entire cloth mesh vertices from its top-view depth observation only.

2) a synthetic dataset with 120k+ simulated cloth meshes and rendered top-view RGBD images, together with one real-world dataset consisting of 3k+ collected cloth configurations and keypoint-labeled top-view RGBD images.

3) a robot system that can manipulate crumpled cloths to some target or near-target configurations by querying and selecting corresponding visible mesh vertices.

II. Related Work

Research on the cloth perception and manipulation is extensive. Earlier approaches typically employ handcrafted or learning-based methods to identify specific cloth features, such as corners, edges, and wrinkles, within the top-view color or depth images [9], [10], [12]. Their manipulation policies are mostly generated from those independently detected image features [13], [14], which may be noisy, sparse, sensitive, and ambiguous. As shown in Fig. 1, it is intrinsically difficult to pixel-wise distinguish all those visible cloth corners, edges, and wrinkles.

Other studies try to simplify the infinite configuration space by firstly hanging the crumpled cloth in midair, from which some feature detection [15], mesh matching [16], [17], [18], and dual-arm manipulation [19] can be more easily performed. In contrast to these approaches, we focus on directly reconstructing and manipulating the initially crumpled cloths, using their top-view observations only.

979-8-3503-8457-4/24/$31.00 ©2024 IEEE

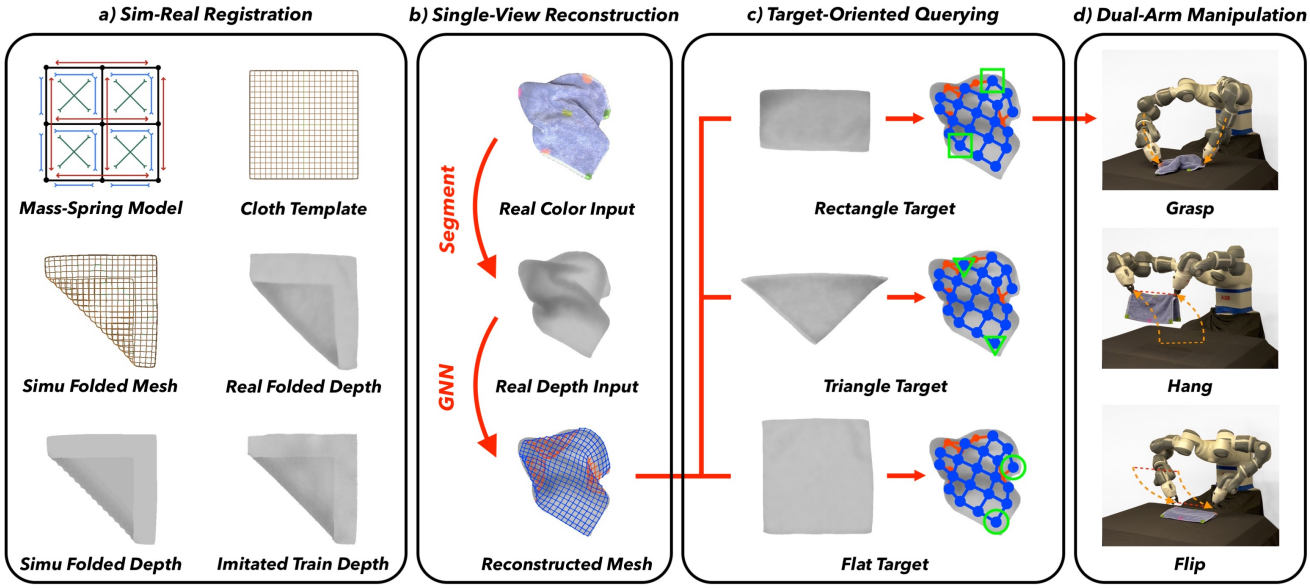

Fig. 2. **System Overview.** a) Sim-real registration of one real-world cloth to a synthetic mass-spring cloth mesh with imitated top-view depth observations. b) Single-view template-based reconstruction of a crumpled cloth from its top-view depth observation only, using our template-based cloth GNN. c) Querying the best visible vertex pairs within the reconstructed and clustered mesh group, according to different target configurations: flat, triangle, and rectangle. d) Dual-arm manipulation using one ABB YuMi Robot at the selected cloth vertex pair with optimized grasp-hang-and-flip trajectories.

Recently, data-driven methods have been proposed to achieve more sophisticated perception and manipulation of crumpled cloths. In these studies, some parameterized single-arm or dual-arm actions, together with some task-specific folding and unfolding policies, are trained in the ways of reinforcement learning with image-based rewards [20], [21], deep imitation learning from predefined policies [22], [23], or value function learning at pixel observations [7], [24], during which the cloth deformations are not explicitly perceived. In addition to those model-free learning methods, some other studies directly optimize the random-shooting actions by dynamics learning within the simulation environment [8], [25], [26], where the synthetic cloth may look and perform differently from the real-world cloths that have various textures and physical properties. In summary, few of the above work fully understands the crumpled cloth configuration, not to mention explicitly locating the visible cloth vertices and manipulating them with some target configurations.

Inspired by the above work and the recent progress in the learning-based reconstruction [27], we aim to achieve precise mesh reconstruction of the randomly dragged, folded, and dropped cloths from their top-view depth observations only. Compared with the previous implicit and simplified cloth representations, our reconstruction mesh explicitly indicates the entire cloth mesh vertices' positions and visibilities, as shown in Fig. 1. Experiments demonstrate that our explicit mesh representation promotes more explicit dual-arm and single-arm target-oriented manipulations, which are more efficient and more similar to the human-wise decision, i.e., in front of a real-world cloth, we mostly generate actions from its 3D mesh embedding, instead of those 2D pixel observations, latent feature vectors, or random shooting optimizations, that are used by the previous work.

III. Methodology

A. Sim-Real Registration

Different from the previous studies that mostly train their policies within simulation and employ the sim2real transfer either directly [8], [28] or rely on fine-tuning [24], [25], we perform several sim-real registrations to directly shrink the gap between the simulation and the real world.

Cloth model registration. Concretely, we register one $0.3m \times 0.3m$ real-world cloth used in VCD [8] to one mass-spring cloth mesh within Blender [29], a physical engine that provides powerful simulation tools. The registered synthetic cloth mesh has 21×21 vertices, and its simulation parameters, like bending stiffness and thickness, are manually tuned from the real-world depth observations of the folded wrinkle size and thick difference, as shown in Fig. 2.

Depth observation registration. Both in simulation and the real world, we centralize each cloth mesh around its image center and normalize the depth observation with a constant scale. Without loss of generality, the cloth image region is one-time scaled according to the longest canonical edge length with a constant ratio of $l_{cloth} : l_{image} = 2 : 3$, which generalizes our reconstruction model to other cloths with different shapes and sizes, as shown in the Ablation Study. Finally, we introduce Gaussian noises to imitate camera noise and non-smooth cloth surfaces observed in the real world. These sim-real registrations only need to be performed once when generating the synthetic dataset and training the GNN.

B. Single-view Reconstruction

We employ the template-based GNN from the single-view human body reconstruction work [27] to our crumpled cloth setting. We design our mass-spring cloth GNN as the encoder, updater, and decoder, as described below.

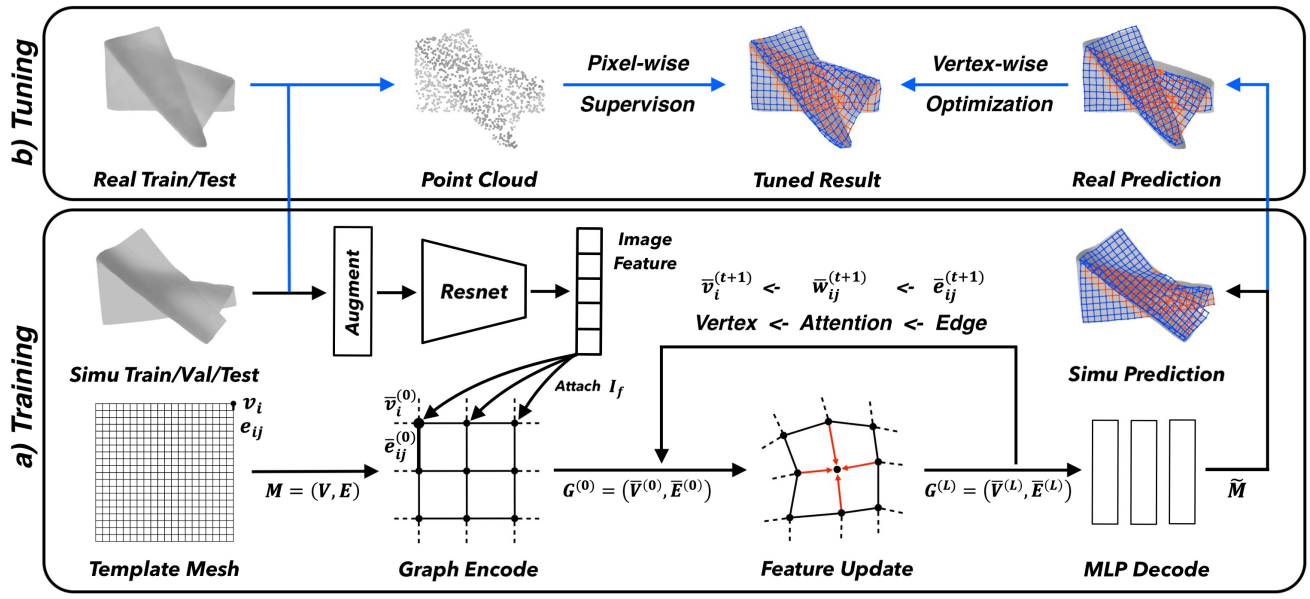

Fig. 3. **Template-based GNN.** a) Synthetic training with simulated cloth meshes and depth images. From left to right: synthetic cloth dataset and template mesh, image feature encoding and template graph encoding, graph feature updating with attention message flow, mesh decoding and supervising. b) Real-world tuning with collected cloth configurations and depth images. From left to right: real-world cloth dataset, point cloud observation, pixel-wise tuned result from the GNN prediction. In our work, we observe small improvements during the tuning process, as discussed in the Ablation Study.

Mass-spring Cloth Model. As shown in Fig. 3, one cloth can be represented as a mass-spring mesh $M = (V, E)$ [30] which contains a vertex group $V = \{v_i\}$ and a bidirectional edge group $E = \{e_{ij}\}$, where $i, j = 1 : N^v$. Each mesh vertex v_i here contains a 3D position vector p_i and a visible flag f_i inferred from its surrounding vertices. Each mesh edge e_{ij} here connects two neighboring vertices and consists of their relative position $p_j - p_i$ and length $||e_{ij}||_2 = ||p_j - p_i||_2$.

Template Graph Encoding. Our GNN encoder includes the image feature encoding and the template graph encoding. One ResNet is used here as a backbone feature extractor to encode the depth observation I into one image feature I_f. This image feature, together with the vertices v_i and edges e_i of our template mesh M, are encoded into a template graph $G = (\bar{V}, \bar{E})$ which contains one vertex feature group $\bar{V} = \{\bar{v}_i\}_{i=1:N^v}$ and one edge feature group $\bar{E} = \{\bar{e}_{ij}\}_{i,j=1:N^v}$:

$$\bar{v}_i = MLP_V([p_i, I_f]),\ I_f = ResNet(I) \tag{1}$$

$$\bar{e}_{ij} = MLP_E([p_j - p_i,\ \|p_j - p_i\|_2]) \tag{2}$$

The encoded template graph $G^{(0)} = (\bar{V}^{(0)}, \bar{E}^{(0)})$ will be iteratively updated through the attention message flow and finally decoded into the predicted cloth mesh $\tilde{M} = (\tilde{V}, \tilde{E})$.

Graph Attention Updating. In the vanilla GNN, edge features $\bar{e}_{ij}^{(t)}$ and vertex features $\bar{v}_i^{(t)}$ are updated iteratively by averaging their neighboring features. To further improve this message updating efficiency, we introduce the attention mechanism to update vertex features by pooling neighboring edge features with learnable attention weights, following the GAT work [31]. During the multi-step feature regression, the neural network will learn on its own which edge connection is more informative, with the weight $\bar{w}_{ij}^{(t+1)}$ shown below:

$$\bar{e}_{ij}^{(t+1)} = \phi_E^{(t)}([\bar{e}_{ij}^{(t)}, \bar{v}_i^{(t)}, \bar{v}_j^{(t)}]) \tag{3}$$

$$\bar{w}_{ij}^{(t+1)} = \frac{exp(\phi_A^{(t)}([\bar{e}_{ij}^{(t+1)}]))}{\sum exp(\phi_A^{(t)}([\bar{e}_{ik}^{(t+1)}]))},\ k \in Neighbor(i) \tag{4}$$

$$\bar{v}_i^{(t+1)} = \phi_V^{(t)}([\bar{v}_i^{(t)}, \sum \bar{w}_{ik}^{(t+1)}\, \bar{e}_{ik}^{(t+1)}]),\ k \in Neighbor(i) \tag{5}$$

In our work, the edge $\phi_E^{(t)}$, attention $\phi_A^{(t)}$, and vertex $\phi_V^{(t)}$ updaters at each iteration are parameterized using MLPs. The contribution of our learnable attention weights is around 11% vertex-wise loss decay, as shown in the Ablation Study.

Cloth Mesh Decoding. After $L = 15$ times of the above graph updating, we decode each vertex feature $\bar{v}_i^{(L)}$ into the predicted vertex position $\tilde{p}_i$, which can be supervised within our synthetic dataset. The visible flag $\tilde{f}_i$ at each vertex is determined by checking whether it is on the top layer of the cloth mesh within a vertical cylinder voxel: $Voxel(\tilde{p}_i)$.

$$\tilde{p}_i = MLP_D([\bar{v}_i^{(L)}]) \tag{6}$$

$$\tilde{f}_i = TOP_L(\tilde{p}_i, \tilde{p}_k),\ \tilde{p}_k \in Voxel(\tilde{p}_i) \tag{7}$$

In our work, we visualize visible vertices with blue and hidden vertices with red, while targeting to explicitly locate and manipulate different visible mesh vertices.

Implementation Details. The above template-based cloth GNN is supervised by the randomly dragged, folded, and dropped cloth meshes simulated within Blender, together with their rendered top-view depth images. The synthetic training loss consists of five terms, as described below:

$$L_{train} = L_{vtx,p} + \lambda_k L_{key,p} + \lambda_s L_{sil} + \lambda_c L_{cham} + \lambda_r L_{regu} \quad (8)$$

The first loss term $L_{vtx,p}$ is the L1 distance between the predicted $\tilde{p}_i$ and the ground truth vertex positions $\hat{p}_i$. The second loss term $L_{key,p}$ is one additional L1 loss at nine cloth keypoints: four corners, four middle points of edges, and one center. These two vertex-wise losses work together to assign the vertex corresponding between the predicted $\tilde{M}$ and the ground truth cloth mesh $\hat{M}$, as shown below:

$$L_{vtx,p} = \frac{1}{N^v} \sum ||\tilde{p}_i - \hat{p}_i||_1,\ i = 1 : N^v \quad (9)$$

$$L_{key,p} = \frac{1}{9} \sum ||\tilde{p}_i - \hat{p}_i||_1,\ i \in Keypoints \quad (10)$$

The third loss term L_{sil} is the image difference between the ground truth $\hat{S}$ and the predicted silhouette $\tilde{S}$ rendered from a differentiable renderer Pytorch3D [32]. The fourth term L_{cham} is the unidirectional chamfer loss from the observed depth point cloud $\hat{D}$ to the predicted vertex positions $\tilde{P}$ [25]. These two self-supervised pixel-wise losses work together to refine the boundary and surface similarities between the predicted $\tilde{M}$ and the ground truth mesh $\hat{M}$. The last term L_{regu} is a regularization loss for edge lengths.

$$L_{sil} = \frac{1}{||\hat{S}||_2^2} \sum_{p \in \hat{S}} ||\tilde{S}_p - \hat{S}_p||_2^2 \quad (11)$$

$$L_{cham} = \frac{1}{|\hat{D}|} \sum_{\hat{d}_i \in \hat{D}} \min_{\tilde{p}_k \in \tilde{P}} ||\tilde{p}_k - \hat{d}_i||_2^2 \quad (12)$$

$$L_{regu} = \frac{1}{N^e} \sum |||\tilde{e}_{ij}||_2 - ||\hat{e}_{ij}||_2|,\ i, j = 1 : N^v \quad (13)$$

We set the above loss ratios as $\lambda_k = 1$, $\lambda_s = 0.5$, $\lambda_c = 0.5$, and $\lambda_r = 1$ respectively. During the synthetic training, we augment the dataset by introducing Gaussian noises and rotating with random angles. Inspired by the previous study [24], we augment each test observation by rotating eight times while predicting their cloth meshes, among which the best mesh prediction is selected using the self-supervised pixel-wise losses. The contributions of these augmentation steps are demonstrated in the Ablation Study.

The synthetic training process takes 2 days on one 2080Ti GPU, while the inference time is around 30ms per image.

C. Target-oriented Querying Policy

Unlike the previous implicit manipulation work [7], [24], we use our template-based reconstruction mesh to generate our dual-arm grasp-hang-and-flip actions as shown in Fig. 2. In principle, each visible mesh vertex can be queried and selected for real-world robot manipulation. However, due to the non-negligible gripper size in practice and the prediction uncertainty at individual vertices, we cluster the reconstruction mesh $\tilde{M} = (\tilde{V}, \tilde{E})$ into a lower-dimensional mesh group $\tilde{M}^g = (\tilde{V}^g, \tilde{E}^g)$ to make our querying policy more efficient and more robust, as shown in Fig. 1 and 4.

TABLE I

QUANTITATIVE EVALUATION OF TEMPLATE-BASED RECONSTRUCTION.

	Averaged over Dragged, Folded, Dropped Cloths					
	T_{simu}	T_{real}	Sq_s	Sq_l	*Rect*	*Shirt*
$L_{vtx,f}(\%)$	10.5	None				
$L_{vtx,p}(mm)$	**11.8**					
$L_{key,f}(\%)$	11.6	14.5	15.0	16.4	18.9	22.3
$L_{key,p}(mm)$	**12.6**	**17.3**	15.8	21.5	20.9	26.7
$L_{sil}(\%)$	5.8	8.1	7.6	8.7	9.2	11.3
$L_{cham}(mm)$	7.3	9.6	7.8	11.5	10.4	14.9
$\mathbf{T_{loss}}$	**19.2**	**26.1**	**23.5**	**31.6**	**30.7**	**39.8**

The clustered mesh group explicitly indicates the positions and visibilities of different cloth regions, from which we can explicitly dual-arm flip a crumpled cloth to some target or near-target configurations. To do so, we dual-arm flip the real-world cloth through each pair of the group vertices. For each target configuration, such as flat, triangle, and rectangle, we score the group vertex pairs by evaluating the silhouette difference between the flipped and the targeted cloth configurations, from which a hierarchical querying list can be generated. At each operation time, the manipulation agent will search through the above querying list within the clustered mesh group and report the best visible group vertex pair $(\tilde{p}_L^{g_{real}}, \tilde{p}_R^{g_{real}})$ for the real-world robot manipulation.

D. Dual-arm Robotic Manipulation

We formulate our dual-arm grasp-hang-and-flip actions according to the FlingBot work [24], as shown in Fig. 2. One ABB YuMi robot is used here to achieve the manipulation task, during which the joint trajectories are optimized from the gripping trajectories using Newton's method [33]. We invite interested readers to check the supplementary video and our project website for manipulation demos.

IV. EXPERIMENTS

We design and execute a series of synthetic and real-world experiments to both qualitatively and quantitatively evaluate our TRTM system with different cloths and tasks.

A. Single-view Cloth Reconstruction Results

Three tiers of cloth configurations are generated within Blender by one-time dragging, two-times folding, and one-time dropping the synthetic template mesh T_{simu}, which provides $120k$ cloth meshes and depth images for training.

To evaluate the real-world reconstruction, we randomly generate the above three cloth tiers using our marked template cloth T_{real}, with a total of 600 cloth configurations. Since it is impossible to label the ground-truth mesh, we only label the marked keypoints, i.e., positions and visibilities of four corners, four middle-edges, and one center. We evaluate our reconstruction results using the supervised vertex-wise losses and pixel-wise losses, as well as the error of visible flags $L_{vtx,f}/L_{key,f}$. The total loss T_{loss} is calculated using Equation (8) without $L_{vtx,p}$ and L_{regu}, as shown in Table I.

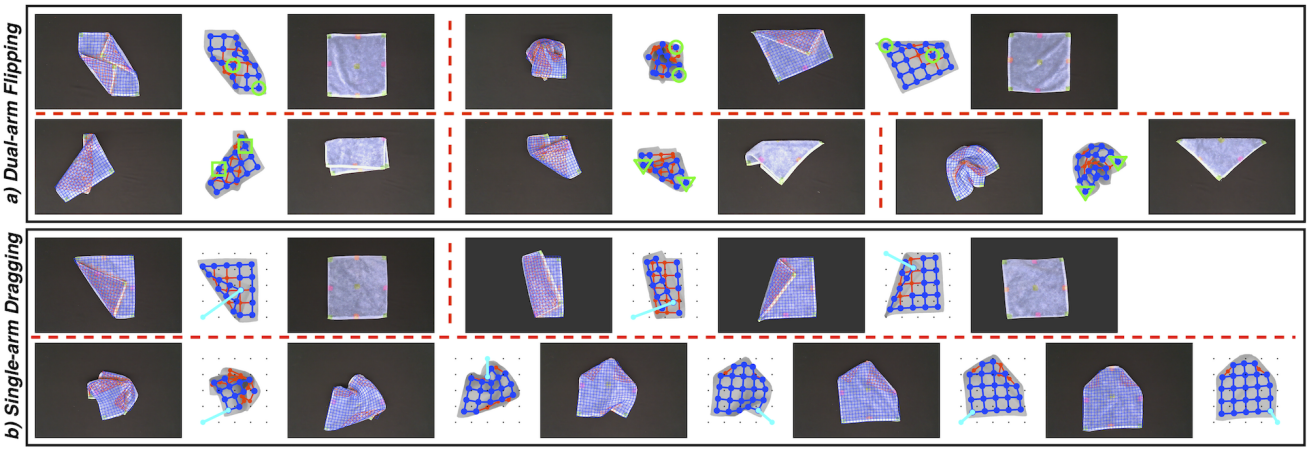

Fig. 4. **Qualitative Evaluation of our Target-oriented Manipulation.** a) Dual-arm flipping by querying visible group vertex pairs according to different target configurations: flat, triangle, and rectangle. b) Single-arm flattening by sequentially dragging the visible group vertex to its canonical target position.

The above reconstruction experiments demonstrate that our synthetic-trained GNN can explicitly and precisely reconstruct our template cloth both in simulation and the real world, with on average vertex-wise losses of 1.22 cm and 1.73 cm respectively. In addition, the direct reconstruction results of four other real-world cloths: one smaller square cloth Sq_s, one larger square cloth Sq_l, one rectangle cloth *Rect*, and one shirt *shirt*, are also demonstrated in Table I.

B. Dual-Arm Cloth Manipulation Results

We dual-arm grasp-hang-and-flip the randomly dragged, folded, and dropped template cloths both in the simulation $(3\times500\times3)$ and the real world $(3\times50\times3)$, with three target configurations: flat, triangle, and rectangle, as shown in Fig. 4 (a) and Fig. 5 (a). In our work, the dual-arm manipulation episode is set as two for the flat target while only one for the triangle and rectangle targets. We evaluate the dual-arm flattened configurations using their top-view coverage values. For the triangle and rectangle targets, we evaluate through the top-view silhouette similarity between the flipped and the targeted configurations: $1-L_{sil}(S_{flipped}, S_{targeted})$.

We compare our real-world dual-arm flattening results with the FlingBot [24], where the flipping points are selected from the top-view color images using a task-specific value network trained within simulation with coverage rewards. For a fair comparison, we employ the FlingBot value network to select flipping points for 3×50 randomly dragged, folded, and dropped template cloths, while keeping the rest setting the same, the results are shown in Fig. 5 (a), in brown.

Experimentally, using our explicit mesh representation, our dual-arm flipping agent can flip most of the randomly dragged, folded, and dropped cloths to flat (97.6% coverage) within two operation episodes, outperforming the implicit FlingBot agent (78.3% coverage). For triangle and rectangle targets, our dual-arm agent can achieve on average 84.5% (Real-RE) / 90.8% (Simu-GT) and 82.3% (Real-RE) / 89.3% (Simu-GT) top-view similarities within only one operation episode. Among three cloth tiers, the two-times folded cloths are mostly difficult to be reconstructed and manipulated.

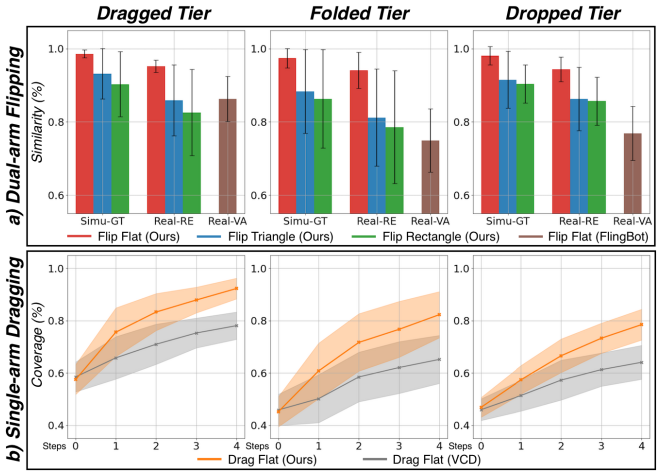

Fig. 5. **Quantitative Evaluation of our Target-oriented Manipulation.** a) Dual-arm flipping experiments with flat (red), triangle (blue), and rectangle (green) targets. We demonstrate the flipping results with the simulated ground truth meshes (Simu-GT), with the real-world reconstruction meshes (Real-GT), and value networks (Real-VA, flatten only). b) Real-world single-arm dragging for flattening experiments: ours (orange) and VCD (gray).

C. Single-arm Cloth Manipulation Results

We modify one single-arm flattening strategy [22] to our mesh group setting, where a canonical group target is assigned around the cloth image center, as shown in Fig. 4 (b). At each operation episode, our single-arm agent will drag the visible group vertex to its target position that has the longest distance. Within the real world, we single-arm flatten 3×50 randomly dragged, folded, and dropped cloth configurations four times each, during which we report the top-view coverage values, as shown in Fig. 5 (b), in orange.

We compare our single-arm flattening results with the VCD work [8], where some random-shooting actions are optimized by a dynamic model trained within the simulation environment. For a fair comparison, we let the VCD agent generate actions to flatten 3×50 randomly dragged, folded, and dropped template cloths, while keeping the rest setting the same, their results are shown in Fig. 5 (b), in gray.

TABLE II

ABLATION STUDY OF TEMPLATE-BASED RECONSTRUCTION.

Methods	T_{real}			
	$L_{key,f}$	$L_{key,p}$	L_{sil}	L_{cham}
No attention weights	16.6%	19.4mm	10.4%	11.6mm
No train augmentation	22.1%	21.8mm	11.6%	12.5mm
No test augmentation	17.4%	19.7mm	10.7%	11.3mm
Synthetic-trained GNN	**14.5%**	**17.3mm**	**8.1%**	**9.6mm**
Tune by training	14.3%	17.0mm	7.8%	9.7mm
Tune by optimizing	14.0%	16.6mm	7.7%	9.3mm

Experimentally, for those boundary-dragged cloth configurations, our single-arm flattening agent can directly recover the dragging action from the mesh distribution, and thus can unfold the cloth within two operation episodes. However, compared with the above dual-arm flipping agent, the single-arm dragging agent requires more operations to flatten a crumpled cloth, and usually sticks around some configurations where corners and edges are folded inside, like the last row in Fig. 4 (b). To flatten these states, some explicit reveal-and-drag actions can be introduced in future work.

D. Ablation Studies

Templated-based GNN. In this section, we first examine some training and testing designs of our reconstruction model in front of the real-world template cloth T_{real}, as shown in Table II. The loss numbers represent the average reconstruction losses in front of the entire dragged, folded, and dropped real-world template cloth configurations.

We also employed some tuning strategies [24], [25] to our cloth reconstruction setting. Specifically, we additionally tune the synthetic GNN with part of the real-world depth observations (400) for another 50 epochs, supervised with the pixel-wise losses only. After this, we further optimize the entire reconstruction mesh with the pixel-wise losses only. The small improvements demonstrate that our synthetic trained model doesn't benefit much from the small-sized real-world data with only pixel-wise supervision.

Sim-real Registration. In this section, we demonstrate that our square-template GNN can be directly and reasonably applied to other daily cloths with a similar topology but different shapes, sizes, textures, and physical properties. To do so, we find another four real-world cloths $(Sq_s, Sq_l, Rect, Shirt)$ and randomly generate their dragged, folded, and dropped cloth configurations with the size of 300 per cloth. We both quantitatively and qualitatively evaluate the direct reconstruction performance of our square-template GNN in front of the above real-world cloths, as shown in Table I and Fig. 6. More synthetic and real-world cloth data can be downloaded from our project website.

These experiments demonstrate that our synthetic-trained square-template GNN can be directly applied to daily square and rectangle cloths with different sizes, textures, and physical properties. It achieves nearly the same pixel-wise reconstruction results and is slightly vertex-wise less accurate. However, in front of those two-layered human garments that have totally different topologies from our template mesh, our square-template GNN can still fit a reasonable crumpled square mesh to their top-view depth observations, but reaching nearly doubled vertex-wise losses, especially when two garment layers are randomly detached and expanded. In future work, a synthetic shirt dataset can be simulated, from which a shirt GNN can be trained to achieve better reconstruction of crumpled shirts.

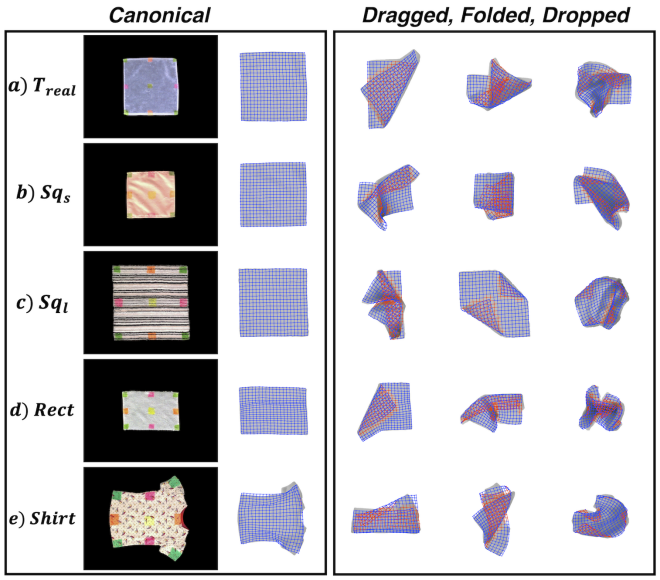

Fig. 6. **Qualitative Evaluation of our Template-based Reconstruction.** Direct reconstruction results of our synthetic-trained square-template GNN in front of different real-world cloths. From top to bottom: a) our sim-real registered $0.3m \times 0.3m$ template cloth T_{real}; b) another $0.25m \times 0.25m$ stiffer square cloth Sq_s; c) another $0.4m \times 0.4m$ softer square cloth Sq_l; d) another $0.2m \times 0.3m$ rectangle cloth *Rect*; e) another $0.3m \times 0.4m$ softer *Shirt* with two layers of rectangle bodies and sleeves. From left to right: canonical configurations; randomly dragged, folded, and dropped cloth configurations.

V. CONCLUSION AND FUTURE WORK

In this paper, we propose a TRTM system that can precisely reconstruct and explicitly manipulate the randomly dragged, folded, and dropped cloths from their top-view observations only. Compared with the previous implicit and simplified cloth representations, our template-based reconstruction mesh explicitly indicates the positions and visibilities of the entire cloth mesh vertices. Experiments demonstrate that our explicit mesh representation promotes more explicit dual-arm and single-arm target-oriented manipulations, which can significantly outperform the previous implicit and task-specific cloth manipulation agents.

Regarding the future work, instead of fitting a TRTM system for each daily cloth with various template embeddings, we believe the fixed template used in our work can be further parameterized like the human body model [34]. From that stage, auto-template-registration [35], [36] can be introduced to improve the reconstruction robustness in front of different cloths with various canonical properties.

REFERENCES

[1] B. Willimon, S. Birchfield, and I. Walker, "Classification of clothing using interactive perception," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 1862–1868.

[2] A. Ganapathi, P. Sundaresan, B. Thananjeyan, A. Balakrishna, D. Seita, J. Grannen, M. Hwang, R. Hoque, J. E. Gonzalez, N. Jamali, K. Yamane, S. Iba, and K. Goldberg, "Learning to smooth and fold real fabric using dense object descriptors trained on synthetic color images," *CoRR*, vol. abs/2003.12698, 2020.

[3] 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*, 2010, pp. 2308–2315.

[4] Y. Li, Y. Yue, D. Xu, E. Grinspun, and P. K. Allen, "Folding deformable objects using predictive simulation and trajectory optimization," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 6000–6006.

[5] 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," *CoRR*, vol. abs/2012.03385, 2020.

[6] J. Qian, T. Weng, L. Zhang, B. Okorn, and D. Held, "Cloth region segmentation for robust grasp selection," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 9553–9560.

[7] Y. Avigal, L. Berscheid, T. Asfour, T. Kroger, and K. Goldberg, "Speedfolding: Learning efficient bimanual folding of garments," *ArXiv*, vol. abs/2208.10552, 2022.

[8] X. Lin, Y. Wang, and D. Held, "Learning visible connectivity dynamics for cloth smoothing," in *CoRL*, 2021.

[9] H. Yuba, S. Arnold, and K. Yamazaki, "Unfolding of a rectangular cloth based on action selection depending on recognition uncertainty," in *2015 IEEE/SICE International Symposium on System Integration (SII)*, 2015, pp. 623–628.

[10] K. Yamazaki, "Gripping positions selection for unfolding a rectangular cloth product," in *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, 2018, pp. 606–611.

[11] W. Yan, A. Vangipuram, P. Abbeel, and L. Pinto, "Learning predictive representations for deformable objects using contrastive estimation," *CoRR*, vol. abs/2003.05436, 2020.

[12] K. Sun, G. Aragon-Camarasa, P. Cockshott, S. Rogers, and J. Siebert, "A heuristic-based approach for flattening wrinkled clothes," vol. 8069, 08 2013.

[13] T. Oshima, T. Yoshimi, M. Mizukawa, and Y. Ando, "A study of towel folding by a robot arm - spreading and vertex detection using image processing," in *2014 14th International Conference on Control, Automation and Systems (ICCAS 2014)*, 2014, pp. 627–631.

[14] J. Stria, D. Průša, V. Hlaváč, L. Wagner, V. Petrík, P. Krsek, and V. Smutný, "Garment perception and its folding using a dual-arm robot," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 61–67.

[15] I. Mariolis, G. Peleka, A. Kargakos, and S. Malassiotis, "Pose and category recognition of highly deformable objects using deep learning," in *2015 International Conference on Advanced Robotics (ICAR)*, 2015, pp. 655–662.

[16] Y. Kita and N. Kita, "A model-driven method of estimating the state of clothes for manipulating it," in *Sixth IEEE Workshop on Applications of Computer Vision, 2002. (WACV 2002). Proceedings.*, 2002, pp. 63–69.

[17] Y. Kita, T. Ueshiba, E. S. Neo, and N. Kita, "Clothes state recognition using 3d observed data," in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 1220–1225.

[18] C. Chi and S. Song, "Garmentnets: Category-level pose estimation for garments via canonical space shape completion," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 3304–3313.

[19] A. Doumanoglou, A. Kargakos, T.-K. Kim, and S. Malassiotis, "Autonomous active recognition and unfolding of clothes using random decision forests and probabilistic planning," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 987–993.

[20] Y. Wu, W. Yan, T. Kurutach, L. Pinto, and P. Abbeel, "Learning to manipulate deformable objects without demonstrations," *CoRR*, vol. abs/1910.13439, 2019.

[21] R. Lee, D. Ward, A. Cosgun, V. Dasagi, P. Corke, and J. Leitner, "Learning arbitrary-goal fabric folding with one hour of real robot experience," *CoRR*, vol. abs/2010.03209, 2020.

[22] D. Seita, A. Ganapathi, R. Hoque, M. Hwang, E. Cen, A. K. Tanwani, A. Balakrishna, B. Thananjeyan, J. Ichnowski, N. Jamali, K. Yamane, S. Iba, J. F. Canny, and K. Goldberg, "Deep imitation learning of sequential fabric smoothing policies," *CoRR*, vol. abs/1910.04854, 2019.

[23] M. Laskey, C. Powers, R. Joshi, A. Poursohi, and K. Goldberg, "Learning robust bed making using deep imitation learning with dart," 11 2017.

[24] H. Ha and S. Song, "Flingbot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding," *CoRR*, vol. abs/2105.03655, 2021.

[25] Z. Huang, X. Lin, and D. Held, "Mesh-based dynamics with occlusion reasoning for cloth manipulation," 06 2022.

[26] R. Hoque, D. Seita, A. Balakrishna, A. Ganapathi, A. K. Tanwani, N. Jamali, K. Yamane, S. Iba, and K. Goldberg, "Visuospatial foresight for multi-step, multi-task fabric manipulation," *CoRR*, vol. abs/2003.09044, 2020.

[27] N. Kolotouros, G. Pavlakos, and K. Daniilidis, "Convolutional mesh regression for single-image human shape reconstruction," *CoRR*, vol. abs/1905.03244, 2019.

[28] S. Sharma, E. Novoseller, V. Viswanath, Z. Javed, R. Parikh, R. Hoque, A. Balakrishna, D. S. Brown, and K. Goldberg, "Learning switching criteria for sim2real transfer of robotic fabric manipulation policies," in *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*, 2022, pp. 1116–1123.

[29] B. O. Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. [Online]. Available: http://www.blender.org

[30] X. Provot, "Deformation constraints in a mass-spring model to describe rigid cloth behavior," *Graphics Interface*, vol. 23(19), 09 2001.

[31] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rJXMpikCZ

[32] N. Ravi, J. Reizenstein, D. Novotný, T. Gordon, W. Lo, J. Johnson, and G. Gkioxari, "Accelerating 3d deep learning with pytorch3d," *CoRR*, vol. abs/2007.08501, 2020.

[33] S. Zimmermann, G. Hakimifard, M. Zamora, R. Poranne, and S. Coros, "A multi-level optimization framework for simultaneous grasping and motion planning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2966–2972, 2020.

[34] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. Black, "Smpl: a skinned multi-person linear model," vol. 34, 11 2015.

[35] F. Hong, L. Pan, Z. Cai, and Z. Liu, "Garment4d: Garment reconstruction from point cloud sequences," *CoRR*, vol. abs/2112.04159, 2021.

[36] C. Guo, X. Chen, J. Song, and O. Hilliges, "Human performance capture from monocular video in the wild," in *2021 International Conference on 3D Vision (3DV)*, 2021, pp. 889–898.