Contents

A Additional results **2**
- A.1 Simulation Experiments . . . 2
  - A.1.1 Normalized improvements for cloth flattening and canonicalization . . . 2
  - A.1.2 Example trajectories in simulation . . . 2
- A.2 Physical Experiments . . . 2

B Implementation details of MEDOR **3**
- B.1 GarmentNets-style Mesh Reconstruction Model . . . 3
  - B.1.1 Model Architecture . . . 3
  - B.1.2 Training Details . . . 4
- B.2 Dynamics Model . . . 5
- B.3 Planning . . . 5

C Implementation Details of Baselines **6**
- C.1 VisuoSpatial Forsight (VSF) . . . 6
- C.2 Visible Connectivity Dynamics (VCD) . . . 7

D Experiments Setup **8**
- D.1 Simulation Setup . . . 8

A Additional results

A.1 Simulation Experiments

A.1.1 Normalized improvements for cloth flattening and canonicalization

	Task	Flattening			Canonicalization		
	Number of Pick-and-Place	1	2	3	1	2	3
Trousers	VSF [1]	0.09 ± 0.13	0.15 ± 0.07	0.14 ± 0.13	0.02 ± 0.02	0.03 ± 0.04	0.05 ± 0.05
	VCD [2]	0.34 ± 0.15	0.41 ± 0.14	0.53 ± 0.24	0.22 ± 0.14	0.26 ± 0.17	0.46 ± 0.16
	GarmentNets [3]	0.27 ± 0.15	0.37 ± 0.14	0.53 ± 0.16	0.14 ± 0.06	0.21 ± 0.09	0.33 ± 0.14
	MEDOR (no fine-tuning)	0.41 ± 0.14	0.57 ± 0.16	0.72 ± 0.08	0.48 ± 0.16	0.64 ± 0.10	0.77 ± 0.10
	MEDOR	$\mathbf{0.52 \pm 0.12}$	$\mathbf{0.69 \pm 0.12}$	$\mathbf{0.79 \pm 0.10}$	$\mathbf{0.59 \pm 0.13}$	$\mathbf{0.73 \pm 0.10}$	$\mathbf{0.77 \pm 0.07}$
Shirt	VSF [1]	0.12 ± 0.07	0.15 ± 0.08	0.20 ± 0.09	0.01 ± 0.02	0.01 ± 0.03	0.03 ± 0.04
	VCD [2]	0.39 ± 0.14	0.51 ± 0.23	0.70 ± 0.20	0.10 ± 0.20	0.14 ± 0.23	0.22 ± 0.21
	GarmentNets [3]	0.34 ± 0.21	0.47 ± 0.17	0.55 ± 0.22	0.08 ± 0.12	0.10 ± 0.14	0.11 ± 0.16
	MEDOR (no fine-tuning)	$\mathbf{0.59 \pm 0.17}$	$\mathbf{0.78 \pm 0.26}$	0.94 ± 0.22	0.46 ± 0.26	$\mathbf{0.60 \pm 0.12}$	$\mathbf{0.62 \pm 0.24}$
	MEDOR	0.59 ± 0.22	0.77 ± 0.19	$\mathbf{0.96 \pm 0.13}$	$\mathbf{0.53 \pm 0.26}$	0.56 ± 0.12	0.61 ± 0.09
Dress	VSF [1]	0.12 ± 0.07	0.15 ± 0.08	0.20 ± 0.09	0.02 ± 0.03	0.03 ± 0.03	0.04 ± 0.04
	VCD [2]	0.39 ± 0.14	0.51 ± 0.23	0.70 ± 0.20	0.18 ± 0.17	0.25 ± 0.16	0.34 ± 0.19
	GarmentNets [3]	0.38 ± 0.16	0.51 ± 0.22	0.57 ± 0.16	0.07 ± 0.10	0.13 ± 0.13	0.21 ± 0.19
	MEDOR (no fine-tuning)	0.43 ± 0.15	0.63 ± 0.19	0.69 ± 0.18	0.36 ± 0.17	0.53 ± 0.22	0.65 ± 0.15
	MEDOR	$\mathbf{0.50 \pm 0.14}$	$\mathbf{0.65 \pm 0.11}$	$\mathbf{0.80 \pm 0.17}$	$\mathbf{0.51 \pm 0.14}$	$\mathbf{0.60 \pm 0.08}$	$\mathbf{0.72 \pm 0.09}$
Skirt	VSF [1]	0.25 ± 0.13	0.32 ± 0.19	0.27 ± 0.24	0.06 ± 0.04	0.07 ± 0.05	0.10 ± 0.07
	VCD [2]	0.56 ± 0.13	0.67 ± 0.15	0.87 ± 0.12	0.19 ± 0.14	0.20 ± 0.14	0.21 ± 0.19
	GarmentNets [3]	0.46 ± 0.16	0.59 ± 0.13	0.70 ± 0.13	0.12 ± 0.10	0.20 ± 0.12	0.25 ± 0.16
	MEDOR (no fine-tuning)	0.56 ± 0.16	0.63 ± 0.19	0.73 ± 0.16	0.39 ± 0.14	0.41 ± 0.17	0.46 ± 0.21
	MEDOR	$\mathbf{0.58 \pm 0.12}$	$\mathbf{0.78 \pm 0.14}$	$\mathbf{0.91 \pm 0.13}$	$\mathbf{0.42 \pm 0.14}$	$\mathbf{0.47 \pm 0.18}$	$\mathbf{0.56 \pm 0.21}$
Jumpsuit	VSF [1]	0.12 ± 0.06	0.15 ± 0.07	0.19 ± 0.10	0.02 ± 0.03	0.03 ± 0.03	0.04 ± 0.05
	VCD [2]	0.36 ± 0.10	0.45 ± 0.15	0.64 ± 0.19	0.23 ± 0.09	0.19 ± 0.11	0.45 ± 0.19
	GarmentNets [3]	0.33 ± 0.12	0.41 ± 0.12	0.55 ± 0.14	0.13 ± 0.15	0.18 ± 0.14	0.33 ± 0.21
	MEDOR (no fine-tuning)	0.45 ± 0.17	0.65 ± 0.14	0.78 ± 0.19	$\mathbf{0.59 \pm 0.14}$	0.67 ± 0.09	0.76 ± 0.08
	MEDOR	$\mathbf{0.53 \pm 0.17}$	$\mathbf{0.73 \pm 0.17}$	$\mathbf{0.82 \pm 0.10}$	0.57 ± 0.14	$\mathbf{0.74 \pm 0.09}$	$\mathbf{0.81 \pm 0.05}$

Table 1: Normalized Improvement (NI) of cloth flattening and cloth canonicalization, for varying numbers of allowed pick and place actions.

A.1.2 Example trajectories in simulation

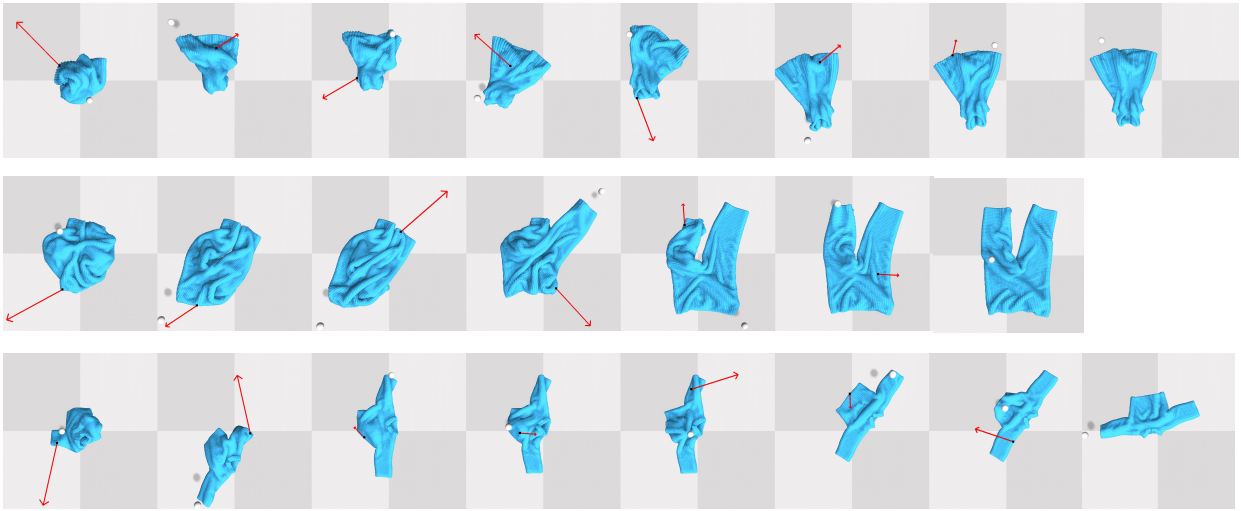

Figure 1: Examplar trajectories of canonicalization task in simulation. As we can see, our method is able to quickly unfold the cloths from extremely crumpled configurations in a few steps.

A.2 Physical Experiments

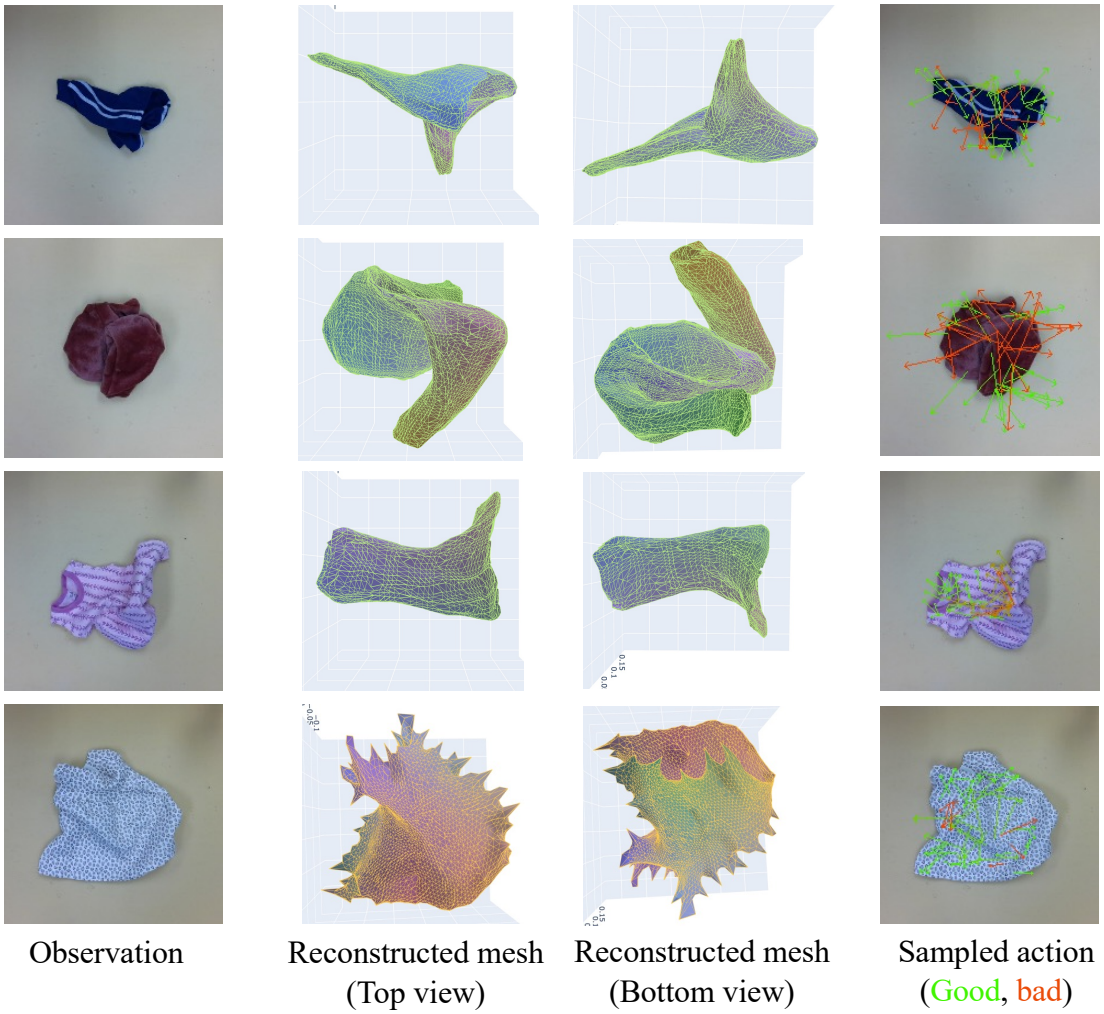

Figure 2: Reconstruction results in real world.

B Implementation details of MEDOR

B.1 GarmentNets-style Mesh Reconstruction Model

As described in the main paper, we employ a GarmentNets-style model to reconstruct the mesh from depth image. GarmentNets formulates the pose estimation problem of clothes as a shape completion task in the canonical space. By doing so, the model learns meaningful correspondence between There are several advantages about GarmentNets: 1) it reconstructs the occluded part of the clothes due to self-occlusion; 2) it estimates the correspondence between clothes in canonical space and observation space; 3) it estimates the pose of the clothes (per-vertex location).

B.1.1 Model Architecture

We build our mesh reconstruction model based on GarmentNets [3] with some decisive modifications to make it fit in our setup. Here, we give a brief description of the reconstruction pipeline and the modifications we made. For details, please refer to GarmentNets [3].

Canonicalization Given the observation at current state, we first use **canonicalization model** to map it to canonical space by conducting pixel-wise canonical coordinate prediction. We don't pick up the clothes for state estimation, because it may disrupt the configuration of partially folded or almost smooth clothes. We use depth images captured

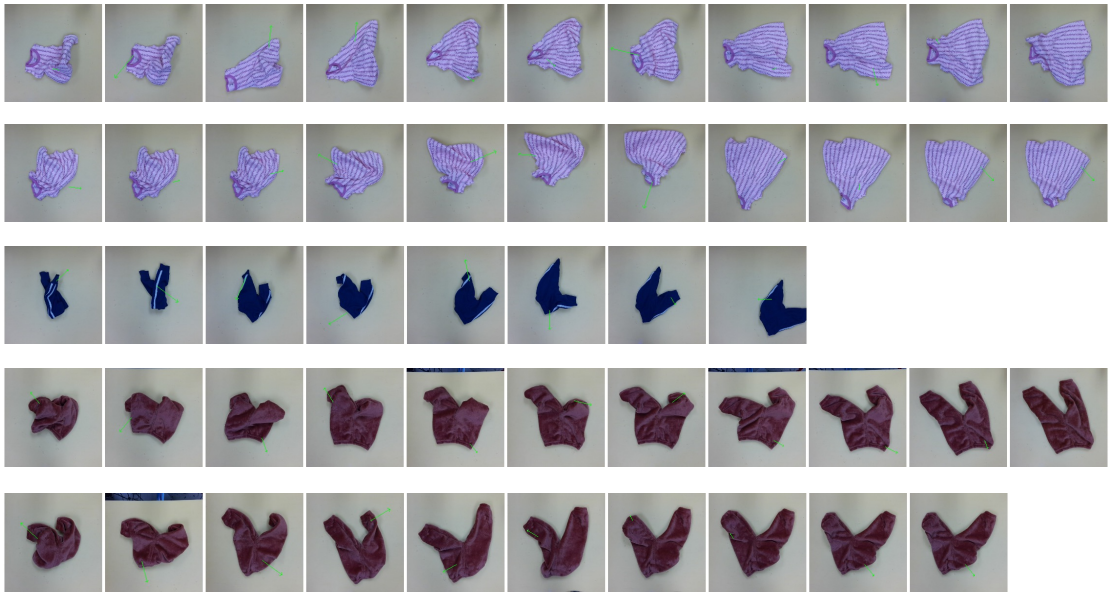

Figure 3: Rollout of cloth flattening in real world.

by a top-down camera as the observation and High-Resolution Network (HRNet-32) [4] as the backbone of canonicalization model. HRNet is a convolutional architecture that specializes in producing high-resolution and spatially precise representations. It uses smaller convolutional kernels and avoids overly downsampling the feature map. This is critical for our task because our model needs to infer the structure of crumpled clothes by subtle changes at the contour of different layers. The architecture change improves the performance on cloth smoothing and canonicalization tasks for 75%. Following GarmentNets [3], we formulate the prediction problem as a classification task by dividing each axis into 64 bins and use Cross-entropy loss for training.

Feature Scattering Given the predicted canonical coordinate, we scattered the features of each pixel to corresponding locations in the canonical space. The aggregated feature volume is further transformed by a 3D UNet [5], which is trained by shape completion and flow prediction.

Shape Completion Before estimating the pose of occluded surface, we first perform volumetric shape completion in the canonical space, which helps capture the shared structure within the same category. Since the structure of clothes are thin and non-watertight, GarmentNets [3] proposes to use Winding Number Field [6] as the shape representation. The shape completion network is instantiated as an implicit network. It takes the dense feature produced by 3D UNet and a canonical coordinate and output the winding number field in that coordinate.

Predicting pose in the observation space After we complete the shape of clothes in the canonical space, we estimate the pose of the clothes in observation space. We cast it as a 3D flow prediction problem by predicting per-vertex flow that transforms the clothes from canonical space to observation space.

$$\tilde{x}_i^o = \tilde{x}_i^c + \tilde{f}_i \tag{1}$$

B.1.2 Training Details

Dataset collection Our model is category-specific, so we collect a dataset for each category separately in Softgym [7]. We obtained 3D clothes models from CLOTH3D dataset [8].

Each dataset contains 4,000 trajectories of length 5 for training and 400 for testing. The At the beginning of each trajectory, we randomly sample a clothes mesh and initialize it by random drop or flattened pose with equal probability. Then we disrupt the clothes with random pick-and-place actions. Random actions are biased towards picking corners (obtained by Harris corner detection[9]) 90% of the time, otherwise it is sampled uniformly on the clothes. The distance between the pick point and the place point is uniformly sampled between [25, 150] pixels.

Training details The model is trained in two-stage. In the first stage, we train the canonicalization network with Cross-entropy loss till convergence. In the second stage, we freeze the canonicalization network and train the rest of the models for shape completion and flow prediction by Mean-square error.

Inference details At test-time, given a depth image, we first use canonicalization network and 3D UNet to obtain dense feature volume in the canonical space. Then we discretize the canonical space into 128x128x128 grid and evaluate shape completion network at every cell. To retrieve the mesh, we compute the Gaussian derivatives for the predicted winding number field and run Marching Cube algorithm [10].

Model parameter	Value
Canonicalization Network	
Backbone	HRNet-32
Dimension of output feature	489
3D CNN	
Backbone	3D Unet
Level	4
Feature maps	32
Dimension of output feature	128
Implicit Shape Completion Network	
Backbone	MLP
Number of hidden layers	3
Size of hidden layers	512
Implicit Shape Completion Network	
Backbone	MLP
Number of hidden layers	3
Size of hidden layers	512
Training parameters	**Value**
Optimizer	Adam
Learning rate	0.0001
Gaussian noise std	0.005
Random rotation	[-180, 180]

Table 2: Hyper-parameters of mesh reconstruction model

B.2 Dynamics Model

Similar to VCD [2], we use a learned GNN-based dynamics model proposed in GNS [11]. The difference between VCD and ours is that we don't have a GNN edge model because edges are estimated by our mesh reconstruction model. The original mesh models in CLOTH3D [8] (see Table. 4) are too dense that the rollout becomes computationally infeasible. Therefore, we downsample the mesh by using Vertex Clustering [12] with a voxel size of 0.025m. For the complete list of hyperparameters of the GNN dynamics model, please refer to Table 3.

B.3 Planning

The planning algorithm is outlined in Algorithm 1. We plan in the space of pick-and-place primitive with horizon equal to 1. To simulate the effect of each pick-and-place, we divide them into low-level actions and roll out by the dynamics model in parallel. Following [2], the action is encoded into the input mesh by directly modifying the position and picked point, and the displacement will be propagated to the rest of mesh during message passing.

Action sampling during planning For both cloth flattening and canonicalization, we bias the actions sampling toward the contour of the cloth. More specifically, we first obtain the bounding box of the cloth in current observation and expand it by 30 pixels in each direction. Then we randomly sample picked points within the the bounding box region. For points that are not on the cloth, we map them to the nearest point on the cloth. The place direction is

Model parameter	Value
Encoder(same for both node encoder and edge encoder)	
Number of hidden layers	3
Size of hidden layers	128
Processor	
Number of message passing steps	10
Number of hidden layers in each edge/node update MLP	3
Size of hidden layers	128
Decoder	
Number of hidden layers	3
Size of hidden layers	128
Training parameters	**Value**
Number of trajectories	5000
Learning rate	0.0001
Batch size	16
Training epoch	120
Optimizer	Adam
Beta1	0.9
Beta2	0.999
Weight decay	0
Others	**Value**
dt	0.05 second
Particle radius	0.005 m
Vertex clustering voxel size	0.025 m
Neighbor radius R	0.036 m

Table 3: Hyper-parameters of GNN dynamics model.

uniformly sampled in all directions and place distance is sampled uniformly from [0.05, 0.2]. A dummy action which corresponds to "no action" is added to the list of candidate actions. We sample 500 pick-n-place actions at each timestep.

Reward computation For flattening, we treat each mesh vertex as a sphere of radius 0.01, and the total reward is the covered area of the projection of all vertices to the ground plane. For canonicalization, we rotate the predicted canonical pose according to the predefined rotation symmetry (see Table. 6). For each valid canonical pose, we input it into the simulator to flatten by gravity, which constitutes a predicted goal set. The reward is computed as negative of smallest distance to goals in goal set. We use pairwise l2 distance.

C Implementation Details of Baselines

C.1 VisuoSpatial Forsight (VSF)

We use the official codebase of VSF[1]. Image: we train the model with RGB-D images of size 56 x56 pixels, according to the original paper. To reproduce the performance of the original paper, we collected 7115 trajectories with 15 pick-and-place actions for each category. In total, it amounts to 100,000 environment steps, which is 5 times the data compared to our methods.

Action sampling We use a similar action sampling strategy as MEDOR for VSF, that is, bounding box sampling B.3. We use a smaller padding size (6 pixels) because of the smaller image size.

[1] https://github.com/ryanhoque/fabric-vsf

Algorithm 1: Planning pipeline of MEDOR

input : Depth Image D, partial point cloud P, mesh reconstruction Model ϕ, dynamics GNN G_{dyn}, number of sampled actions K

output: pick-and-place action $a = \{x_{pick}, x_{place}\}$

Estimate the full mesh of clothes by mesh reconstruction model: $\tilde{M}_{init} = \phi(D)$

Perform test-time fine-tuning: $\tilde{M}^0_{tuned} = finetune(\tilde{M}_{init}) = (\tilde{V}^0, \tilde{E}_M)$.

for $i \leftarrow 1$ ***to*** K **do**

  Sample a pick-and-place action x_{pick}, x_{place}

  Compute low-level actions $\Delta x_1, ..., \Delta x_H$

  Get picked point v_{picked} from x_{pick}

  Pad historic velocities with 0: $\mathbf{x}_0 \leftarrow \tilde{V}^0, \dot{\mathbf{x}}_{-m...0} \leftarrow \mathbf{0}$

  for $t \leftarrow 0$ ***to*** H **do**

    Build collision edges E^t_C with $\mathbf{x}_t$

    Move picked point according to gripper movement by :

    $x_{u,t} \leftarrow x_{u,t} + \Delta x_t,\ \dot{x}_{u,t} \leftarrow \Delta x_t / \Delta t$

    Predict accelerations using G_{dyn}: $\ddot{\mathbf{x}}_t \leftarrow G_{dyn}(\mathbf{x}_t, \dot{\mathbf{x}}_{t-m...t}, E_M, E^t_C)$

    Update point cloud predicted positions & velocities:

    $\dot{\mathbf{x}}_{t+1} = \dot{\mathbf{x}}_t + \ddot{\mathbf{x}}_t \Delta t,\ \mathbf{x}_{t+1} = \mathbf{x}_t + \dot{\mathbf{x}}_{t+1} \Delta t$

    Readjust picked point according to gripper movement by

    $x_{u,t} \leftarrow x_{u,t} + \Delta x_t,\ \dot{x}_{u,t} \leftarrow \Delta x_t / \Delta t$

  end

  Compute reward r based on final mesh nodes position $\mathbf{x}_H$

end

return *pick and place action with maximal reward*

Reward computation For flattening, we use color thresholding to compute the coverage of cloth. For canonicalization, we project the predicted and goal RGB-D image into 3D rgb point cloud. The RGB values are scaled to be similar to the coordinate values. Then we run ICP [13] for 5 iterations to align the predicted point cloud with the goal point cloud.

C.2 Visible Connectivity Dynamics (VCD)

We use the official code of VCD, with modifications so that it works well on our dataset.

Given point cloud and mesh, the original VCD conduct bipartite matching to map point cloud to mesh nodes. If the corresponding mesh nodes are connected by mesh edges, we also construct mesh edges for the point cloud points. We found that this approach is highly sensitive to density of point cloud and mesh. Imagine the mesh is denser than the point cloud, there might be many mesh vertices between two adjacent points on point cloud. Thus they are not connected although they should.

To solve this issue, we design a more robust approach for mesh edges construction that is agnostic to the density of mesh. First, for each point on the point cloud, we find the nearest mesh vertex. Then we compute the distance between neighboring points on point cloud by the pairwise geodesic distance of corresponding mesh vertices. A mesh edge is constructed if the geodesic distance is below a threshold.

Training To makes a fair comparison, we train the edge GNN dataset of different categories separately. Each dataset contains 20,000 envrironment steps, which is same as the dataset used for training the mesh reconsctruction model of MEDOR. For dynamics model, similar to MEDOR, we train a single model on Trousers dataset but use it for all categories at test-time.

Planning Same as MEDOR, except that we run ICP [13] for 5 iterations to align the predicted point cloud with the goal point cloud before computing the cost function for canonicalization task.

D Experiments Setup

D.1 Simulation Setup

	Trousers	Shirt	Dress	Skirt	Jumpsuit
No. of Meshes	1691	1111	2037	468	2279
Avg. No. of Vertices	7050±1954	5308±996	8030±2159	4643±1225	10686±2572
Avg. No. of Vertices (downsampled)	278±87	214±59	291±99	190±80	215±58
Rescaling Factor	0.42	0.36	0.29	0.28	0.28
Avg. X (m)	0.29±0.03	0.32±0.10	0.24±0.07	0.20±0.05	0.19±0.08
Avg. Y (m)	0.12±0.02	0.11±0.02	0.15±0.05	0.15±0.05	0.09±0.02
Avg. Z (m)	0.27±0.08	0.19±0.04	0.31±0.06	0.18±0.05	0.31±0.06

Table 4: The statistics of the CLOTH3D dataset [8] after pre-processing.

Simulation parameters	Value
Camera view	Top-down
Camera position	[0, 0.65 m, 0]
Field of view	90
Picker radius	0.01 m
Picker threshold	0.00625 m
dt	0.01 second
Damping	1
Dynamic friction	1.2
Particle friction	1
Stiffness (strech, bend, shear)	[1.2, 0.6, 1]
Mass	0.0003
Particle radius	0.005
Gravity	-9.8

Table 5: Hyper-parameters of softgym.

We conduct all simulation experiments in Softgym [7], a simulation environment for deformable objects built on the particle-based simulator, Nvidia Flex. We model a flying gripper as a spherical picker that can move freely in the 3D space. When a cloth particle is "picked", it will move rigidly with the gripper. We obtained the 3D cloth models from CLOTH3D dataset [8]. Considering the physical experiments, we rescale the cloth models so that they could fit in the workspace of our real robot. Also, to make the GNN dynamics computationally feasible, we create a downsampled version of the cloth models. During data collection, we still use the original dense mesh in softgym, but register the downsampled mesh onto the dense mesh by finding the nearest neighbor of each vertex. Therefore, we obtain the trajectory of downsampled mesh and use that for dynamics learning. The detailed specifications of the preprocessed CLOTH3D dataset can be found in Table. 4. We use a top-down camera and it's placed at a fixed height of 0.65 m. The valid workspace is 0.53 m $\times$ 0.53 m.

Flattening For flattening task, the goal is to maximize the coverage of the cloth in the current configuration. To compute the coverage, we treat each node on the graph as a sphere with a radius of 0.005 and compute the covered area when projected to the ground plane.

Canonicalization In CLOTH3D, the canonical pose of the cloth is defined to be the pose of cloth when wore by a T-pose human. However, since we are considering cloth manipulation on a planar surface, it is not achievable and cannot be used as the goal pose directly. Therefore, we use gravity (10 times the gravity on Earth) to obtain a flattened version of the canonical pose. Another thing that needs to be handled is the ambiguity caused by reflection symmetry and rotation symmetry. For example, trousers are approximately 180° rotation symmetry, which means that the shapes before and after rotating around the center axis for 180° are the same. Therefore, during the evaluation of cloth canonicalization, we define a canonical goal set $\mathcal{G} = \{G_i^{N\times 3}\}_{i=1\dots A}$, for each type of cloth. A is the number of valid

canonical poses. The cost is computed as the minimum of average pairwise distance to each of the possible canonical poses. Suppose that the current configuration of the cloth is $V^{N\times 3}$, where N is the number of vertices, g_j and v_j is the j-th vertex of the mesh, the cost is computed as

$$Cost_{canon} = \min_{G_i \in \mathcal{G}} \sum_{j=n}^{N} \frac{(g_j - v_j)^2}{N} \tag{2}$$

The rotation symmetry of each category is described in Table. 6. It should be noted that since we maintain a discrete set of plausible goal pose, although skirt has infinite order of rotation symmetry, we cannot iterate over all possible cases. Instead, we make an approximation that the order of rotation symmetry of skirt is 12. The order of rotation symmetry is the times the shape fits onto itself when rotating for 360 degrees. An order of 2 means that the shape remains unchanged when rotating for 180 or 360 degrees.

Trousers	Shirt	Dress	Skirt	Jumpsuit
2	1	2	12	2

Table 6: Order of rotation symmetry of different types of cloth. The order of rotation symmetry is the times the shape fits onto itself when rotating for 360 degrees. An order of 2 means that the shape remains unchanged when rotating for 180 or 360 degrees. We use it to construct goal sets for cloth canonicalization task.

References

[1] Ryan Hoque, Daniel Seita, Ashwin Balakrishna, Aditya Ganapathi, Ajay Tanwani, Nawid Jamali, Katsu Yamane, Soshi Iba, and Ken Goldberg. VisuoSpatial Foresight for Multi-Step, Multi-Task Fabric Manipulation. In *Robotics: Science and Systems (RSS)*, 2020. 2

[2] Xingyu Lin, Yufei Wang, Zixuan Huang, and David Held. Learning visible connectivity dynamics for cloth smoothing. In *Conference on Robot Learning*, pages 256–266. PMLR, 2022. 2, 5

[3] Cheng Chi and Shuran Song. Garmentnets: Category-level pose estimation for garments via canonical space shape completion. *ICCV*, 2021. 2, 3, 4

[4] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5693–5703, 2019. 4

[5] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *International conference on medical image computing and computer-assisted intervention*, pages 424–432. Springer, 2016. 4

[6] Alec Jacobson, Ladislav Kavan, and Olga Sorkine-Hornung. Robust inside-outside segmentation using generalized winding numbers. *ACM Transactions on Graphics (TOG)*, 32(4):1–12, 2013. 4

[7] Xingyu Lin, Yufei Wang, Jake Olkin, and David Held. SoftGym: Benchmarking deep reinforcement learning for deformable object manipulation. In *Conference on Robot Learning*, 2020. 4, 8

[8] Hugo Bertiche, Meysam Madadi, and Sergio Escalera. Cloth3d: Clothed 3d humans. In *European Conference on Computer Vision*, pages 344–359. Springer, 2020. 4, 5, 8

[9] C. G. Harris and M. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, 1988. 4

[10] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987. 5

[11] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *International Conference on Machine Learning*, pages 8459–8468. PMLR, 2020. 5

[12] Kok-Lim Low and Tiow-Seng Tan. Model simplification using vertex-clustering. In *Proceedings of the 1997 symposium on Interactive 3D graphics*, pages 75–ff, 1997. 5

[13] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International journal of computer vision*, 13(2):119–152, 1994. 7