

Non-rigid Relative Placement through 3D Dense Diffusion

Eric Cai, Octavian Donca, Ben Eisner, David Held

Robotics Institute, School of Computer Science

Carnegie Mellon University, United States

{eycai, odonca, baeisner, dheld}@andrew.cmu.edu

Abstract: The task of “relative placement” is to predict the placement of one object in relation to another, e.g. placing a mug onto a mug rack. Through explicit object-centric geometric reasoning, recent methods for relative placement have made tremendous progress towards data-efficient learning for robot manipulation while generalizing to unseen task variations. However, they have yet to represent deformable transformations, despite the ubiquity of non-rigid bodies in real world settings. As a first step towards bridging this gap, we propose “cross-displacement” - an extension of the principles of relative placement to geometric relationships between deformable objects - and present a novel vision-based method to learn cross-displacement through dense diffusion. To this end, we demonstrate our method’s ability to generalize to unseen object instances, out-of-distribution scene configurations, and multimodal goals on multiple highly deformable tasks (both in simulation and in the real world) beyond the scope of prior works. Supplementary information and videos can be found at our [website](#).

Keywords: Deformable, Non-rigid, Manipulation, Relative Placement

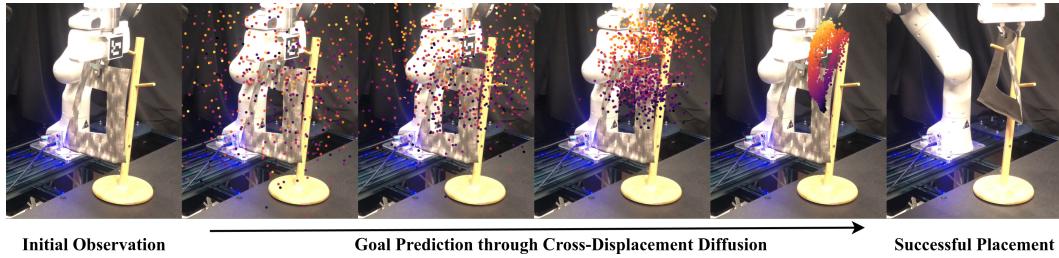


Figure 1: Our method (TAX3D) uses dense displacement diffusion to determine how to perform a deformable cloth-hanging task for an unseen scene configuration.

1 Introduction

Learning from demonstrations has emerged as a powerful technique to imbue robots with complex manipulation capabilities. Recent approaches have achieved remarkable success on a wide variety of tasks, including mug-hanging [1, 2, 3], book-shelving [4], cloth-folding [5], and sauce-pouring [6]. One common approach for imitation learning is to learn end-to-end visuomotor policies that map directly from observations to low-level robot actions; however, such approaches often struggle to generalize to novel variations of the scene (e.g. unseen object instances or object configurations). In this work, we aim to train robots to be robust to these variations for *multimodal, deformable* object placement tasks. Given demonstrations of a cloth-hanging task, for example, a robot should be able to successfully hang the cloth for unseen cloth instances or hanger positions. Similarly, if the task has multiple possible goal configurations (e.g. multiple holes on the cloth through which to orient the hanger), the robot should be able to output multimodal predictions to complete the task.

A simple solution is to collect an increasingly large dataset with all of the variations that one wishes to be robust to. However, collecting such a dataset, particularly in the real world, may be prohibitively infeasible. Other recent works have taken an alternative approach by reasoning more explicitly about object geometry and casting the manipulation task as a “relative placement” problem [1, 2, 3, 4]. These approaches directly model the pose relationship between the object being manipulated (the “action object”) and the rest of the scene (the “anchor”) in an object-centric fashion by separately featurizing action and anchor geometries; these approaches have shown the ability to learn precise manipulation tasks with impressive generalization capabilities from a small number (e.g. 10) of demonstrations [1]. However, these approaches also make strong assumptions about object rigidity, predicting an $SE(3)$ transformation that does not apply to deformable objects.

In this paper, we present **TAX3D** (**T**ask-Specific **C**ross-**D**isplacement through **D**ense **D**iffusion), a deep 3D-vision framework to handle multimodal goal prediction for *deformable* relative placement tasks (e.g. hanging a cloth on a hanger). Namely, our contributions include:

- (1) A formal definition of “cross-displacement,” in which we extend the idea of relative placement beyond rigid poses to arbitrary deformable settings through a dense representation.
- (2) A novel method to predict cross-displacement through object-centric point-wise diffusion.
- (3) A novel task benchmark for multimodal relative placement for deformable object manipulation from demonstrations, building on DEDO [7].

We design experiments to rigorously evaluate performance under goal multimodality, out-of-distribution scene configurations, and variations in object geometry. Contrary to prior relative placement methods, we demonstrate that our method is robust to all of these variations for simulated and real-world cloth-hanging tasks, paving the way for generalizable non-rigid relative placement.

2 Related Work

Relative Placement for Object Manipulation: As previously described, there exists a corpus of work that deconstructs a placement task as a relative pose prediction problem [8, 9, 10, 11, 12]. Dense Object Nets (DON) [3] and Neural Descriptor Fields (NDF) [2] accomplish this by learning dense embeddings and matching observation embeddings to demonstration embeddings. While these approaches work in constrained placement settings, they assume that the target object is moved relative to a static reference object. TAX-Pose [1] addresses this issue by directly modeling the relative pose of objects in the scene and regressing a task-specific “cross-pose” from learned dense embeddings. This inference pipeline, however, renders TAX-Pose unsuitable for multimodal goal prediction. Recent work (TAX-PoseD [13]) extends TAX-Pose to multimodal tasks using a conditional Variational AutoEncoder (cVAE) with a dense spatially-grounded latent space. Relational Pose Diffusion (RPDiff) [4] leverages iterative pose de-noising to learn object-scene relationships that are multimodal, but diffuses directly in the space of $SE(3)$ transformations. This fundamentally restricts RPDiff to rigid placement tasks. In contrast to these approaches, our method is able to handle both multi-modal placements and deformable objects for relative placement tasks.

Diffusion Models for Point Cloud Generation: Diffusion models [14, 15, 16, 17, 18, 19, 20] have emerged as the state-of-the-art in 2D image generation; their variational loss allows for more stable training and mode capture (in contrast to GANs), and the lack of aggressive regularization on their latent distribution mitigates the coverage-fidelity tradeoff inherent to VAEs. Following this success, there have been many efforts to transfer these advantages to 3D point cloud generation, either by applying 2D diffusion techniques to volumetric representations [21, 22] or by denoising individual points independently [23, 24]. Our methodology aligns with the latter, as we adapt the Diffusion Transformer [25] architecture to de-noise pointwise displacements.

Deformable Object Manipulation: Despite its prevalence in the real world, deformable manipulation remains a challenge in robotics due to the complexity of its underlying dynamics and state spaces. Early works have demonstrated success on deformable tasks such as cloth and bedding

manipulation [26, 27, 28, 29, 30, 31, 32], rope manipulation [33], dough rolling [34, 35], and bag opening [36, 37]. These methods, however, often rely on pre-defined features (e.g. corner or wrinkle detection) [38, 32, 30], manually designed action primitives (e.g. pick-and-place, fold, or fling) [27, 28, 26, 38, 33, 31, 29, 34], or simplifying assumptions about the scene (e.g. a 2D tabletop environment) [31, 37, 36, 32, 35], limiting their generalization to a broad range of deformable tasks. More recently, end-to-end visuomotor policies [6, 39, 5, 40] have shown some capacity to learn non-rigid manipulation tasks. However, our experiments show that one such method (DP3 [39]) fails to generalize to novel objects or unseen object configurations, whereas our approach of non-rigid relative placement shows significantly improved generalization abilities.

3 Problem Statement

Relative Placement for Rigid Objects: In this paper, we study “relative placement” as a general framework that encapsulates many robotics tasks (e.g. placing a mug on a rack, or hanging a cloth on a hanger). Given two objects \mathcal{A} and \mathcal{B} , the goal of a relative placement task is to manipulate object \mathcal{A} (the “action” object) into some position relative to object \mathcal{B} (the “anchor” object).

We will briefly review relative placement tasks for rigid objects as defined in prior work [1]: for objects \mathcal{A} and \mathcal{B} , let point clouds $\mathbf{P}_\mathcal{A}^*$ and $\mathbf{P}_\mathcal{B}^*$ denote their respective object geometries in a desired goal configuration. Suppose, for example, that object \mathcal{A} is a mug, object \mathcal{B} is a mug rack, and $(\mathbf{P}_\mathcal{A}^*, \mathbf{P}_\mathcal{B}^*)$ are the point clouds of these objects when the mug is hanging on the rack. For a relative placement task, if both objects are transformed by the same $SE(3)$ -transformation \mathbf{T} , then the resulting configuration should also be considered a successful task completion; for example, if the mug and rack are transformed together, then the mug is still hanging on the rack. Formally, if $(\mathbf{P}_\mathcal{A}, \mathbf{P}_\mathcal{B})$ denote the point clouds of objects \mathcal{A} and \mathcal{B} in some arbitrary configuration, we can define whether this configuration represents a successful relative placement as:

$$\text{RelPlace}(\mathbf{P}_\mathcal{A}, \mathbf{P}_\mathcal{B}) = \text{SUCCESS} \iff \exists \mathbf{T} \in SE(3) \text{ s.t. } \mathbf{P}_\mathcal{A} = \mathbf{T} \cdot \mathbf{P}_\mathcal{A}^* \text{ and } \mathbf{P}_\mathcal{B} = \mathbf{T} \cdot \mathbf{P}_\mathcal{B}^*. \quad (1)$$

Then, given observed object point clouds $\mathbf{P}_\mathcal{A}$ and $\mathbf{P}_\mathcal{B}$, the goal of a rigid relative placement task [1] is to learn a function $f(\mathbf{P}_\mathcal{A}, \mathbf{P}_\mathcal{B}) = \mathbf{T}_{AB}$ that predicts a rigid transformation of object \mathcal{A} such that $\text{RelPlace}(\mathbf{T}_{AB} \cdot \mathbf{P}_\mathcal{A}, \mathbf{P}_\mathcal{B}) = \text{SUCCESS}$. The transform \mathbf{T}_{AB} is referred to as the “cross-pose” [1] since it captures the pose *relationship* between objects \mathcal{A} and \mathcal{B} . Using motion planning, the robot can then move object \mathcal{A} into a new pose determined by \mathbf{T}_{AB} to complete the relative placement task, e.g. to place the mug onto the rack [1].

Cross-Displacement for Deformables: In the deformable case, however, this formulation is ill-defined. For a task with deformable object \mathcal{A} , there is no guarantee that there exists a rigid transformation \mathbf{T}_{AB} that can transform an observed point cloud $\mathbf{P}_\mathcal{A}$ into a goal configuration $\mathbf{P}_\mathcal{A}^*$ such that $\text{RelPlace}(\mathbf{T}_{AB} \cdot \mathbf{P}_\mathcal{A}, \mathbf{P}_\mathcal{B}) = \text{SUCCESS}$, as object \mathcal{A} may need to undergo a non-rigid transformation to achieve its goal configuration. Consider, for example, the task of hanging a towel on a towel rack - the deformations undergone by the towel as it folds and drapes over the hanger cannot be captured by a rigid transformation.

Instead, to model object deformations, we learn a dense transformation function that predicts where each *point* in object \mathcal{A} must move. Namely, given an initial observation of object \mathcal{A} as point cloud $\mathbf{P}_\mathcal{A} = \{x_0, \dots, x_N\}$ with $x_i \in \mathbb{R}^3$, we aim to learn a function $g(\mathbf{P}_\mathcal{A}, \mathbf{P}_\mathcal{B}) = \Delta X$, where $\Delta X = \{\Delta x_0, \dots, \Delta x_N\}$ with $\Delta x_i \in \mathbb{R}^3$, such that $\text{RelPlace}(\mathbf{P}_\mathcal{A} + \Delta X, \mathbf{P}_\mathcal{B}) = \text{SUCCESS}$. Intuitively, ΔX is a set of dense displacements such that $x_i + \Delta x_i$ brings each point x_i to the goal configuration relative to $\mathbf{P}_\mathcal{B}$. As such, we refer to ΔX as the “**cross-displacement**” of objects \mathcal{A} and \mathcal{B} , analogous to the “cross-pose” defined previously.

Goal Multimodality: For some tasks, there may be more than one way to achieve a goal configuration. For example, for the task of hanging a towel on a rack, if object \mathcal{B} consists of multiple towel racks, then hanging the towel on any of the racks should yield a valid goal configuration. To this end, we can define a *distributional* relative placement task: for a given point cloud $\mathbf{P}_\mathcal{B}^*$ of object \mathcal{B} ,

let $m(\mathbf{P}_B^*)$ be a set of point clouds for object \mathcal{A} such that for all $\mathbf{P}_A^* \in m(\mathbf{P}_B^*)$, the configuration pair $(\mathbf{P}_A^*, \mathbf{P}_B^*)$ completes the task. We can then explicitly define success for a distributional relative placement task as:

$$\text{RelPlace}_D(\mathbf{P}_A, \mathbf{P}_B) = \text{SUCCESS} \iff \exists \mathbf{T} \in SE(3), \mathbf{P}_A^* \in m(\mathbf{P}_B^*) \\ \text{s.t. } \mathbf{P}_A = \mathbf{T} \cdot \mathbf{P}_A^* \text{ and } \mathbf{P}_B = \mathbf{T} \cdot \mathbf{P}_B^*. \quad (2)$$

This definition extends the idea of distributional relative placement defined in prior work [13] to a continuous rather than discrete set of possible goal configurations. To achieve a multi-modal deformable relative placement task, we aim to learn a distribution over cross-displacements $g(\mathbf{P}_A, \mathbf{P}_B) = p(\Delta \mathbf{X})$ from which we can sample a candidate cross-displacement $\Delta \mathbf{X} \sim p(\Delta \mathbf{X})$.

Assumptions: For our method, we assume access to segmentations for action object \mathcal{A} and anchor object \mathcal{B} . During training, we assume access to a set of M demonstration configurations that complete the task $\{(\mathbf{P}_{A,1}, \mathbf{P}_{A,1}^*, \mathbf{P}_{B,1}^*), \dots, (\mathbf{P}_{A,M}, \mathbf{P}_{A,M}^*, \mathbf{P}_{B,M}^*)\}$, where $(\mathbf{P}_{A,i}, \mathbf{P}_{A,i}^*, \mathbf{P}_{B,i}^*)$ respectively denote the initial action object point cloud, goal action object point cloud, and goal anchor object point cloud in demonstration i . We also assume access to correspondences between the initial $\mathbf{P}_{A,i}$ and the goal $\mathbf{P}_{A,i}^*$ point clouds; these correspondences are required to compute the ground-truth cross-displacement $\Delta \mathbf{X}^* = \mathbf{P}_A^* - \mathbf{P}_A$ used to supervise our model as explained below.

4 Method

Given point clouds $(\mathbf{P}_A, \mathbf{P}_B)$ of objects \mathcal{A} and \mathcal{B} in an arbitrary configuration, our goal is to learn a function that predicts a distribution over cross-displacements $g(\mathbf{P}_A, \mathbf{P}_B) = p(\Delta \mathbf{X})$. From this distribution, we can sample a cross-displacement $\Delta \mathbf{X} \sim p(\Delta \mathbf{X})$ to transform \mathbf{P}_A into a successful goal configuration $\hat{\mathbf{P}}_A = \mathbf{P}_A + \Delta \mathbf{X}$ relative to object \mathcal{B} .

To solve this distributional non-rigid relative placement problem, we leverage diffusion models [14, 15], a class of generative models that rely on iterative noising and de-noising. More specifically, our method builds on Improved Denoising Diffusion Probabilistic Models [41]. Given some Gaussian noise x_T (noised from the forward process [14]), we train our model to de-noise x_T by learning the reverse diffusion process [14]: $p_\theta(x_{t-1} | x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t), \Sigma_\theta(x_t))$. Following [41], we re-parameterize the mean $\mu_\theta(x_t)$ and the covariance $\Sigma_\theta(x_t)$ using a noise prediction $\epsilon_\theta(x_t)$ and interpolation vector $v_\theta(x_t)$, respectively, and supervise with a hybrid loss function that combines the standard noise prediction error with a variational lower bound loss; for further details, we defer to Nichol and Dhariwal [41].

4.1 Point Cloud Generation for Relative Placement

Training: Given a sample demonstration $(\mathbf{P}_A, \mathbf{P}_A^*, \mathbf{P}_B^*)$, where \mathbf{P}_A is the initial action object point cloud $\{x_{A,0}, \dots, x_{A,N}\}$, \mathbf{P}_A^* is the goal action object point cloud $\{x_{A,0}^*, \dots, x_{A,N}^*\}$, and \mathbf{P}_B^* is the goal anchor object point cloud, we train our model to de-noise a set of per-point displacements $\Delta \mathbf{X}$, with the ground-truth displacements defined as $\Delta \mathbf{X}^* = \mathbf{P}_A^* - \mathbf{P}_A = \{x_{A,0}^* - x_{A,0}, \dots, x_{A,N}^* - x_{A,N}\}$. Following Ho et al. [15], we sample partially noised displacements ΔX_t :

$$\Delta X_t = \sqrt{\bar{\alpha}_t} \Delta X_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon \quad (3)$$

where $\Delta X_0 = \Delta X^*$, $t \sim [0, T]$, $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and $\bar{\alpha}_t$ is a function of the noise schedule. We then supervise our model's noise $\epsilon_\theta(\Delta X_t, \mathbf{P}_A, \mathbf{P}_B^*, t)$ and interpolation vector $v_\theta(\Delta X_t, \mathbf{P}_A, \mathbf{P}_B^*, t)$ predictions using the hybrid loss function from Nichol and Dhariwal [41]. Note that the model is conditioned on the anchor \mathbf{P}_B^* , as well as the initial action point cloud \mathbf{P}_A , which we refer to below as the “action context.” For architecture and training specifics, see Appendix B.

Inference: At inference, given some object configuration $(\mathbf{P}_A, \mathbf{P}_B)$, we initialize a set of per-point displacements as Gaussian noise: $\Delta X_T \sim \mathcal{N}(0, \mathbf{I})$. These displacements are iteratively de-noised using our model's outputs: $\Delta X_{t-1} \sim \mathcal{N}(\Delta X_{t-1}; \mu_\theta, \Sigma_\theta)$, with μ_θ and Σ_θ re-parameterized as in Nichol and Dhariwal [41]. The final de-noised displacements ΔX_0 are then used to transform the points of \mathbf{P}_A into a predicted placement $\mathbf{P}_A + \Delta X_0$ relative to \mathbf{P}_B .

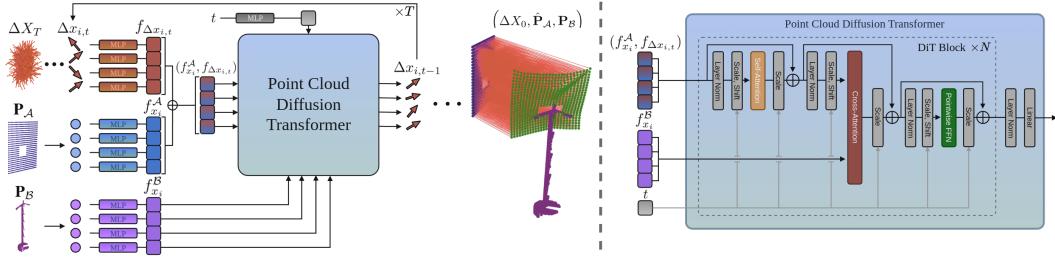


Figure 2: (Left). During inference, randomly sampled displacements $\Delta X_T \sim \mathcal{N}(0, \mathbf{I})$ are de-noised conditioned on action (\mathbf{P}_A) and anchor (\mathbf{P}_B) features; the final ΔX_0 is predicted to displace the action into a goal configuration. (Right). Our modified DiT [25] architecture combines self-attention and cross-attention for object-centric and scene-level reasoning.

Object-Specific Frames: To guarantee translation-invariance in our model, we process input point clouds ($\mathbf{P}_A, \mathbf{P}_B$) in object-specific frames. Namely, we center the action context \mathbf{P}_A in the action frame ($\mathbf{P}'_A = \mathbf{P}_A - \bar{\mathbf{P}}_A$) and the anchor \mathbf{P}_B in the anchor frame ($\mathbf{P}'_B = \mathbf{P}_B - \bar{\mathbf{P}}_B$) by subtracting their respective point cloud means $\bar{\mathbf{P}}_A$ and $\bar{\mathbf{P}}_B$. Accordingly, our model is conditioned on $(\mathbf{P}'_A, \mathbf{P}'_B)$ instead of $(\mathbf{P}_A, \mathbf{P}_B)$. During training, we further center the ground-truth goal action point cloud in the anchor frame ($\mathbf{P}^{*'}_A = \mathbf{P}_A^* - \bar{\mathbf{P}}_B$) and modify the ground-truth displacements to $\Delta X^{*'} = \mathbf{P}_A^* - \mathbf{P}'_A$. During inference, we can easily transform our predictions back into the world frame by inverting these translations. In our experiments, we show that this use of object-specific frames is crucial for robust out-of-distribution generalization.

4.2 Diffusion Transformer for Point Cloud Generation

We adapt the Diffusion Transformer (DiT) [25] architecture for our model, as shown in Figure 2. At each diffusion timestep, our model takes as input the noised displacements ΔX_t , the action context \mathbf{P}_A , the anchor \mathbf{P}_B , and the timestep t . Using MLP encoders, we first compute per-point displacement features $f_{\Delta x}$, per-point action context features f_x^A , and per-point anchor features f_x^B from ΔX_t , \mathbf{P}_A , and \mathbf{P}_B , respectively. MLP weights are shared within sets of features. The action context features and displacement features are then concatenated into $(f_x^A, f_{\Delta x})$ as input to our modified DiT model alongside anchor features f_x^B (see Figure 2, left).

Within a DiT block, self-attention is applied to the combined action features $(f_x^A, f_{\Delta x_i})$ to aggregate information across the entire action point cloud and facilitate coordinated displacement predictions. Cross-attention is then applied to these features with the anchor features f_x^B , allowing for scene-level global reasoning between the action and anchor objects. This is repeated for N blocks, before the network outputs a noise ϵ_θ and interpolation vector v_θ prediction, as described above.

We refer to the above approach as the “**Cross-Displacement (CD)**” variant of our TAX3D architecture, since we directly predict the cross-displacement ΔX . We also propose a “**Cross-Point (CP)**” variant in which we directly encode and diffuse over the positions of the predicted goal point cloud $\hat{\mathbf{P}}_A = \mathbf{P}_A + \Delta X$ instead; we find that both variants perform similarly in our experiments. Following Peebles and Xie [25], we incorporate timestep conditioning using adaptive layer normalization (adaLN) blocks. For permutation equivariance, we do not incorporate any positional encoding.

5 Experiments

Setup: To the best of our knowledge, quantitative benchmarks do not exist for the setting of deformable object relative placement. As such, to evaluate our approach, we design a novel experimental benchmark building on DEDO: Dynamic Environments with Deformable Objects [7], a suite of simulation environments for deformable manipulation. We experiment on two cloth-hanging tasks (Figure 3): HangProcCloth, in which a cloth must be hung through one of its holes, and HangBag,



Figure 3: TAX3D generalizes to diverse cloths and anchor positions (*top*); we also visualize the corresponding goal predictions (*middle*) and successful rollouts (*bottom*) after releasing the cloth. The two rightmost columns are HangBag configurations - all others are HangProcCloth configurations.

in which a bag must be hung on one of its handles. For environment and demonstration details, see Appendix A. We also evaluate our method in the real world, described further below.

Evaluation Metrics: We evaluate our approach using two metrics: point prediction error and downstream policy performance. To measure point prediction error, we use point-wise **root-mean-square-error (RMSE)** to compute the distance between a prediction $\hat{\mathbf{P}}_{\mathcal{A}}$ and a ground truth goal configuration $\mathbf{P}_{\mathcal{A}}^*$. To evaluate *distributional* placements for multimodal experiments, we also report **Coverage RMSE**, in which we compute the minimum distance to a set of sampled predictions $\{\hat{\mathbf{P}}_{\mathcal{A},j}\}$ for each ground truth goal $\mathbf{P}_{\mathcal{A},i}^*$, and **Precision RMSE**, in which we compute the minimum distance to the set of ground truth goals $\{\mathbf{P}_{\mathcal{A},i}^*\}$ for each sampled prediction $\hat{\mathbf{P}}_{\mathcal{A},j}$. To evaluate downstream policy performance, we use a primitive goal-conditioned policy that directly converts a predicted goal point cloud $\hat{\mathbf{P}}_{\mathcal{A}}$ into position control inputs and report the cloth hanging **Success Rate**. Further details regarding evaluation can be found in Appendix C.

Ablations and Baselines: We evaluate the two variants of our method described in Section 4.2 - “Cross-Displacement” (CD) and “Cross-Point” (CP) - against several architectural ablations:

- (1) **Scene Displacement/Point (SD/SP):** a variant of our CD/CP architectures *without object-centric reasoning*. The action and anchor point clouds are combined into a single scene point cloud \mathbf{P}_S , and the DiT blocks are implemented with self-attention only.
- (2) **Cross Displacement/Point - World Frame (CD-W/CP-W):** a variant of our CD/CP architectures *without object-specific frames*: $\mathbf{P}_{\mathcal{A}}$ and $\mathbf{P}_{\mathcal{B}}$ are not mean-centered.
- (3) **Cross Displacement/Point - No Action Context (CD-NAC/CP-NAC):** a variant of our CD/CP architectures *without action context features*. $\mathbf{P}_{\mathcal{A}}$ is not encoded, but cross-attention between displacement features $f_{\Delta x}$ and anchor features $f_x^{\mathcal{B}}$ is retained.

We also compare our method to **3D Diffusion Policy (DP3)** [39], a recent end-to-end visuomotor policy that has achieved state-of-the-art manipulation results using point cloud inputs.

Generalization to Unseen Configurations and Geometries: Experiments are conducted on two versions of the HangProcCloth task (`unimodal1`, in which the cloth has one hole, and `multimodal`, in which the cloth has two holes) as well as on the HangBag task. Important comparisons are shown in Tables 1, 3, and 4, respectively, with full results in Appendix D. Training demonstrations are generated under random anchor poses (64 demonstrations for HangProcCloth and 16 for HangBag), and models are evaluated on two sets of configurations (40 trials each): **Unseen**, which contains novel anchor poses from the training distribution, and **Unseen (Out-of-Distribution)**, which contains out-of-distribution anchor poses. For both HangProcCloth tasks, cloth and hole shape are

	RMSE (\downarrow)			Success Rate (\uparrow)		
	Train	Unseen	Unseen (OOD)	Train	Unseen	Unseen (OOD)
SD	0.127	0.805	4.246	0.27	0.20	0.00
SP	0.095	0.779	3.955	0.45	0.48	0.08
CD-W	0.099	0.464	12.464	0.95	0.88	0.00
CP-W	0.085	0.511	19.923	0.95	0.85	0.00
CD-NAC	2.233	2.243	2.326	0.16	0.28	0.10
CP-NAC	3.987	3.934	3.923	0.02	0.00	0.03
TAX3D-CD (<i>Ours</i>)	0.052	0.405	0.395	0.94	0.90	0.85
TAX3D-CP (<i>Ours</i>)	0.051	0.390	0.422	0.93	0.95	0.80
DP3	-	-	-	0.98	0.38	0.03

Table 1: HangProcCloth-unimodal: Random Cloth Geometry (1-Hole). We do not report RMSEs for DP3 since it is an end-to-end policy and does not predict point clouds.

randomized across demonstrations, with **Unseen** and **Unseen (OOD)** consisting only of unseen cloth geometries. For the HangBag task, only a single fixed cloth geometry is used.

	Success Rate (\uparrow)	
	Unseen	Unseen (OOD)
TAX3D-CD	1.00	0.98
TAX3D-CP	1.00	0.98
DP3	0.93	0.00

Table 2: HangProcCloth-simple: Fixed Cloth Geometry

All ablations (Table 1) fail on out-of-distribution anchor poses, demonstrating that our method’s combination of *object-level* reasoning in *object-specific* frames is crucial for generalization to novel scene configurations. As the point prediction errors indicate, NAC (No Action Context) ablations are unable to produce meaningful cloth geometries (Appendix D) due to the lack of action context - crucially, without combined action features ($f_x^A, f_{\Delta x}$), the network cannot maintain correspondences between displacements Δx_i and action points x_i .

We also find that DP3 fails to generalize both to novel scene configurations *and* to novel object instances, despite fitting well to training demonstrations. To understand why, we conduct an additional experiment with *fixed cloth geometry* (HangProcCloth-simple, Table 2), in which DP3 achieves decent performance on **Unseen** (but in-distribution) anchor poses. This indicates that the poor performance on **Unseen** in HangProcCloth-unimodal is a result of DP3’s inability to adapt to *variations in the cloth geometry*. TAX3D, on the other hand, generalizes well to both scene and object variations, maintaining significantly superior performance across all HangProcCloth and HangBag experiments (Tables 1, 3, 4) in comparison to DP3.

Multimodal Goal Prediction: Our diffusion-based architecture naturally accommodates multimodal settings (Figure 4). To explicitly measure this, we additionally compare against regression baselines (**Regression Displacement/Point (RD/RP)**) for HangProcCloth-multimodal. To do so, we remove the timestep conditioning from the architecture and train the model to directly predict $\hat{\mathbf{P}}_A$ via a regression loss (MSE). As Table 3 indicates, the regression models lead to poor performance on the multimodal task variant of a cloth with two holes. In contrast, our method continues to achieve good performance on this task; the low coverage and precision RMSEs indicate that it can predict both modes, while avoiding the mode-averaging tendencies of regressive inference.

Real World Experiments: We also qualitatively demonstrate our approach in a real-world cloth-hanging setup. Human demonstration videos are captured using a single Azure Kinect camera. Segmentations and correspondences are then obtained using point-prompted Segment Anything [42]

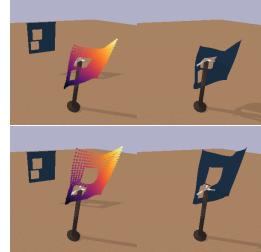


Figure 4: Multimodal TAX3D predictions (left), with successful rollouts (right).

	Coverage RMSE (\downarrow)		Precision RMSE (\downarrow)		Success Rate (\uparrow)	
	Unseen	Unseen (OOD)	Unseen	Unseen (OOD)	Unseen	Unseen (OOD)
RD	2.987	3.057	1.087	1.114	0.60	0.50
RP	1.417	1.432	1.192	1.154	0.65	0.48
TAX3D-CD (<i>Ours</i>)	0.457	0.543	0.403	0.558	0.98	0.73
TAX3D-CP (<i>Ours</i>)	0.453	0.540	0.495	0.631	0.85	0.70
DP3	-	-	-	-	0.45	0.00

Table 3: HangProcCloth-multimodal: Random Cloth Geometry (2-Hole)

	RMSE (\downarrow)			Success Rate (\uparrow)		
	Train	Unseen	Unseen (OOD)	Train	Unseen	Unseen (OOD)
TAX3D-CD (<i>Ours</i>)	0.183	0.204	0.245	0.94	0.83	0.78
TAX3D-CP (<i>Ours</i>)	0.173	0.192	0.233	0.94	0.93	0.78
DP3	-	-	-	0.98	0.38	0.03

Table 4: HangBag: Fixed Cloth Geometry



Figure 5: Real world results. TAX3D succeeds under varying anchor poses, varying peg placements (*left, middle-left*), and can model multimodal placements with multiple pegs (*middle-right, right*).

and SpatialTracker [43], respectively. We train our model on 10 demonstrations, containing multimodal placements and various anchor poses. To perform robot actions, we use position control to track the predicted goal point cloud. As shown in Figure 5, our method can complete the task under novel anchor poses and anchor geometries, while accommodating multimodal goal predictions.

6 Conclusion

In this paper, we present a novel framework to perform relative placement for non-rigid manipulation tasks. We formulate the task of “cross-displacement” prediction to handle relative placement for arbitrary object deformations, and we show that our dense diffusion architecture can learn cross-displacements on multiple cloth hanging tasks in simulation and in the real world. Our experiments demonstrate our approach’s ability to generalize to out-of-distribution scene configurations, unseen object instances, and multimodal placements for robust deformable manipulation.

Limitations: There are several limitations to our method, which we leave for future work:

1. **Segmentations.** Our method requires segmented action and anchor point clouds. Our current approach for obtaining such segmentations (point-prompted Segment Anything [42]) requires human involvement, limiting scalability. This can potentially be addressed using language-conditioned segmentation methods [44] to automate demonstration labeling.
2. **Open-loop control.** Our method currently relies on open-loop position control to the predicted goal for robot execution, which limits the ability to perform more complex placement tasks. This can potentially be addressed by incorporating TAX3D goal predictions into a goal-conditioned manipulation policy.

Acknowledgments

This material is based upon work supported by the Toyota Research Institute, the National Science Foundation under NSF CAREER Grant No. IIS-2046491, and NIST under Grant No. 70NANB23H178. We are grateful to Prof. Shubham Tulsiani for his valuable feedback and discussion at various stages of the project, and to Lifan Yu for her assistance with real-world experiments.

References

- [1] C. Pan, B. Okorn, H. Zhang, B. Eisner, and D. Held. Tax-pose: Task-specific cross-pose estimation for robot manipulation. In *Conference on Robot Learning*, pages 1783–1792. PMLR, 2023.
- [2] A. Simeonov, Y. Du, A. Tagliasacchi, J. B. Tenenbaum, A. Rodriguez, P. Agrawal, and V. Sitzmann. Neural descriptor fields: Se (3)-equivariant object representations for manipulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6394–6400. IEEE, 2022.
- [3] P. R. Florence, L. Manuelli, and R. Tedrake. Dense object nets: Learning dense visual object descriptors by and for robotic manipulation. *arXiv preprint arXiv:1806.08756*, 2018.
- [4] A. Simeonov, A. Goyal, L. Manuelli, Y.-C. Lin, A. Sarmiento, A. R. Garcia, P. Agrawal, and D. Fox. Shelving, stacking, hanging: Relational pose diffusion for multi-modal rearrangement. In *Conference on Robot Learning*, pages 2030–2069. PMLR, 2023.
- [5] J. Yang, C. Deng, J. Wu, R. Antonova, L. Guibas, and J. Bohg. Equivact: Sim (3)-equivariant visuomotor policies beyond rigid object manipulation. *arXiv preprint arXiv:2310.16050*, 2023.
- [6] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.
- [7] R. Antonova, P. Shi, H. Yin, Z. Weng, and D. K. Jensfelt. Dynamic environments with deformable objects. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.
- [8] B. Eisner, Y. Yang, T. Davchev, M. Vecerik, J. Scholz, and D. Held. Deep se(3)-equivariant geometric reasoning for precise placement tasks, 2024.
- [9] Y.-C. Lin, P. Florence, A. Zeng, J. T. Barron, Y. Du, W.-C. Ma, A. Simeonov, A. R. Garcia, and P. Isola. Mira: Mental imagery for robotic affordances. In K. Liu, D. Kulic, and J. Ichnowski, editors, *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 1916–1927. PMLR, 14–18 Dec 2023. URL <https://proceedings.mlr.press/v205/lin23c.html>.
- [10] J. Seo, N. P. S. Prakash, X. Zhang, C. Wang, J. Choi, M. Tomizuka, and R. Horowitz. Contact-rich se(3)-equivariant robot manipulation task learning via geometric impedance control. *IEEE Robotics and Automation Letters*, 9(2):1508–1515, Feb. 2024. ISSN 2377-3774. doi:10.1109/Lra.2023.3346748. URL <http://dx.doi.org/10.1109/LRA.2023.3346748>.
- [11] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, A. Wahid, V. Sindhwani, and J. Lee. Transporter networks: Rearranging the visual world for robotic manipulation, 2022.
- [12] A. Simeonov, Y. Du, L. Yen-Chen, A. Rodriguez, L. P. Kaelbling, T. Lozano-Perez, and P. Agrawal. Se(3)-equivariant relational rearrangement with neural descriptor fields, 2022.
- [13] J. Wang, O. Donca, and D. Held. Learning distributional demonstration spaces for task-specific cross-pose estimation. *IEEE International Conference on Robotics and Automation (ICRA)*, 2024, 2024.

- [14] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.
- [15] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.
- [16] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models, 2022.
- [17] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis, 2021.
- [18] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with clip latents, 2022.
- [19] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. Denton, S. K. S. Ghasemipour, B. K. Ayan, S. S. Mahdavi, R. G. Lopes, T. Salimans, J. Ho, D. J. Fleet, and M. Norouzi. Photorealistic text-to-image diffusion models with deep language understanding, 2022.
- [20] L. Zhang, A. Rao, and M. Agrawala. Adding conditional control to text-to-image diffusion models, 2023.
- [21] L. Zhou, Y. Du, and J. Wu. 3d shape generation and completion through point-voxel diffusion, 2021.
- [22] X. Zeng, A. Vahdat, F. Williams, Z. Gojcic, O. Litany, S. Fidler, and K. Kreis. Lion: Latent point diffusion models for 3d shape generation. *Conference on Neural Information Processing Systems (NeurIPS)*, 2022.
- [23] S. Luo and W. Hu. Diffusion probabilistic models for 3d point cloud generation. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
- [24] G. Nakayama, M. Uy, J. Huang, S. Hu, K. Li, and L. Guibas. Difffacto: Controllable part-based 3d point cloud generation with cross diffusion. *International Conference on Computer Vision (ICCV)*, 2023.
- [25] W. Peebles and S. Xie. Scalable diffusion models with transformers. *International Conference on Computer Vision (ICCV)*, 2023.
- [26] X. Lin, Y. Wang, Z. Huang, and D. Held. Learning visible connectivity dynamics for cloth smoothing, 2022. URL <https://arxiv.org/abs/2105.10389>.
- [27] H. Ha and S. Song. Flingbot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding, 2021. URL <https://arxiv.org/abs/2105.03655>.
- [28] Z. Xu, C. Chi, B. Burchfiel, E. Cousineau, S. Feng, and S. Song. Dextairity: Deformable manipulation can be a breeze, 2022. URL <https://arxiv.org/abs/2203.01197>.
- [29] Y. Wu, W. Yan, T. Kurutach, L. Pinto, and P. Abbeel. Learning to manipulate deformable objects without demonstrations, 2020. URL <https://arxiv.org/abs/1910.13439>.
- [30] 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*, pages 2308–2315, 2010. doi: 10.1109/ROBOT.2010.5509439.
- [31] T. Weng, S. Bajracharya, Y. Wang, K. Agrawal, and D. Held. Fabricflownet: Bimanual cloth manipulation with a flow-based policy, 2022. URL <https://arxiv.org/abs/2111.05623>.

- [32] K. Sun, G. Aragon-Camarasa, P. Cockshott, S. Rogers, and J. Siebert. A heuristic-based approach for flattening wrinkled clothes. volume 8069, 08 2013. ISBN 978-3-662-43644-8. doi:[10.1007/978-3-662-43645-5_16](https://doi.org/10.1007/978-3-662-43645-5_16).
- [33] 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, 2023. URL <https://arxiv.org/abs/2012.03385>.
- [34] J.-T. Kim, F. Ruggiero, V. Lippiello, and B. Siciliano. *Planning Framework for Robotic Pizza Dough Stretching with a Rolling Pin*, pages 229–253. 03 2022. ISBN 978-3-030-93289-3. doi:[10.1007/978-3-030-93290-9_9](https://doi.org/10.1007/978-3-030-93290-9_9).
- [35] C. Qi, X. Lin, and D. Held. Learning closed-loop dough manipulation using a differentiable reset module, 2022. URL <https://arxiv.org/abs/2207.04638>.
- [36] L. Y. Chen, B. Shi, D. Seita, R. Cheng, T. Kollar, D. Held, and K. Goldberg. Autobag: Learning to open plastic bags and insert objects, 2023. URL <https://arxiv.org/abs/2210.17217>.
- [37] A. Bahety, S. Jain, H. Ha, N. Hager, B. Burchfiel, E. Cousineau, S. Feng, and S. Song. Bag all you need: Learning a generalizable bagging strategy for heterogeneous objects, 2023. URL <https://arxiv.org/abs/2210.09997>.
- [38] D. Seita, N. Jamali, M. Laskey, A. K. Tanwani, R. Berenstein, P. Baskaran, S. Iba, J. Canny, and K. Goldberg. Deep transfer learning of pick points on fabric for robot bed-making, 2019. URL <https://arxiv.org/abs/1809.09810>.
- [39] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations, 2024. URL <https://arxiv.org/abs/2403.03954>.
- [40] J. Yang, Z. ang Cao, C. Deng, R. Antonova, S. Song, and J. Bohg. Equibot: Sim(3)-equivariant diffusion policy for generalizable and data efficient learning, 2024. URL <https://arxiv.org/abs/2407.01479>.
- [41] A. Q. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.
- [42] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick. Segment anything, 2023. URL <https://arxiv.org/abs/2304.02643>.
- [43] Y. Xiao, Q. Wang, S. Zhang, N. Xue, S. Peng, Y. Shen, and X. Zhou. Spatialtracker: Tracking any 2d pixels in 3d space, 2024. URL <https://arxiv.org/abs/2404.04319>.
- [44] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024.
- [45] E. Coumans and Y. Bai. Pybullet, a python module for physics simulation for games, robotics, and machine learning, 2016-2020. URL <http://pybullet.org>.

Appendix

Table of Contents

A DEDO Environment Details	13
A.1 HangProcCloth - Task Definition	13
A.2 HangProcCloth - Cloth Generation	13
A.3 HangBag - Task Definition	14
A.4 Cloth Control	15
A.5 Episode Rollout	15
A.6 Demonstration Generation	16
B Training Details	17
B.1 Model Architecture	17
B.2 Training Pre-Processing & Hyperparameters	18
C Evaluation Metrics	18
D Experiments	18
D.1 HangProcCloth-simple Results	19
D.2 HangProcCloth-unimodal Results	19
D.3 HangProcCloth-multimodal Results	19
D.4 HangBag Results	20
D.5 Additional Visualizations	20

A DEDO Environment Details

DEDO: Dynamic Environments with Deformable Object [7] is a suite of task-based simulation environments (hanging a bag, dressing a mannequin, etc.) involving highly deformable, topologically non-trivial objects. The environments are built on the PyBullet physics engine [45].

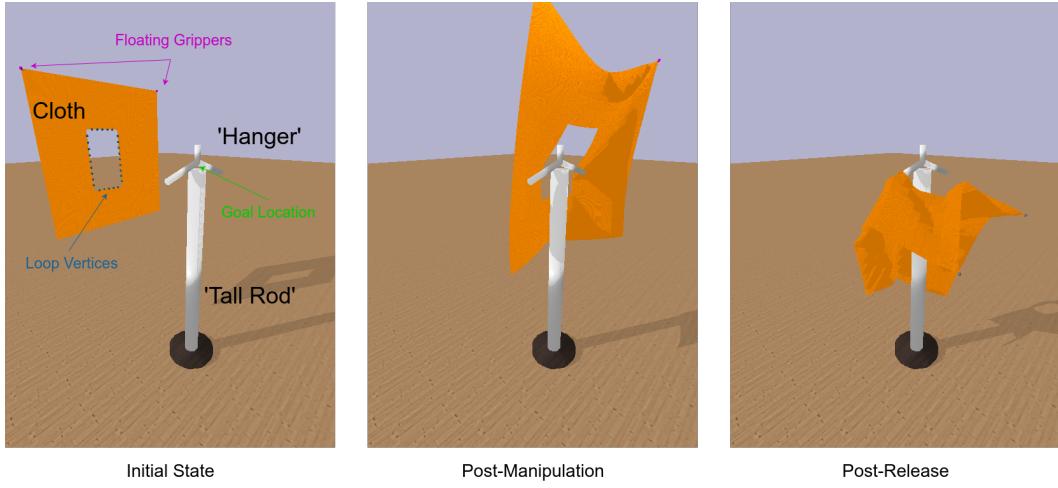


Figure 6: Sample demonstration of the HangProcCloth task.

A.1 HangProcCloth - Task Definition

For most of our experiments, we focus on the HangProcCloth task (Figure 6), in which a procedurally generated cloth must be placed on a hanger. More specifically, the cloth is generated to contain a hole in its topology - to successfully complete the task, the vertical part of the hanger should be aligned *through* the hole.

The hanger (anchor) is loaded into the PyBullet engine as a pre-defined rigid body, and contains two components: a ‘tall rod’, and the ‘hanger’ itself. While we randomize the anchor pose throughout our experiments, this geometry remains fixed. The **goal** of the task is explicitly formulated in the environment as the center of the ‘hanger’ component (Figure 6) - while this goal definition is not passed as input to our models, it is used later by our success metric for evaluation.

A.2 HangProcCloth - Cloth Generation

Following DEDO’s implementation, every cloth in our experiments is procedurally generated as a rectangular mesh, and can be represented using the following parameters:

<code>node_density</code>	25	The amount of vertices to initialize the cloth mesh with. Every cloth is initialized as an evenly spaced <code>node_density</code> \times <code>node_density</code> grid ($25 \times 25 = 625$ vertices for all of our cloths). Vertices are then removed during the hole generation process.
<code>width</code>	[0.8, 1.2]	The width of the cloth.
<code>height</code>	[0.8, 1.2]	The height of the cloth.
<code>num_holes</code>	(1..2)	The number of holes in the cloth.
<code>holes</code>	See A.2.1	See A.2.1

A.2.1 HangProcCloth - Hole Generation

Holes are created by removing mesh vertices. All generated holes are rectangular - as such, they can be represented topologically with respect to the procedurally generated cloth by their bottom-left and top-right corners. Accordingly, the `holes` parameter is a list, where each element corresponding to a specific hole in the cloth is a dictionary with elements:

x0	The x vertex coordinate of the bottom-left corner of the hole.
y0	The y vertex coordinate of the bottom-left corner of the hole.
x1	Similar to x_0 , for the top-right corner.
y1	Similar to y_0 , for the top-right corner.

For reference, the single-hole cloth used in our `HangProcCloth-simple` experiment is defined as:

```

1 {
2     "node_density": 25,
3     "width": 1.0,
4     "height": 1.0,
5     "num_holes": 1,
6     "holes": [
7         {"x0": 8, "y0": 9, "x1": 16, "y1": 13}
8     ]
9 }
```

In general, holes are randomly generated under the following constraints:

x_range	(2, node_density - 2)	The range of possible values for x_0 .
y_range	(2, node_density - 2)	The range of possible values for y_0 .
width_range	(5, 7)	The range of possible values for w_h , such that $x1 = x0 + w_h$.
height_range	(5, 7)	The range of possible values for h_h , such that $x1 = x0 + h_h$.

More precisely, when generating holes, x_0 and y_0 are first sampled based on `x_range` and `y_range`, respectively - x_1 and y_1 are then sampled based on `width_range` and `height_range`. To ensure that the resulting cloth geometry is valid topologically, DEDO generates cloths using a Monte Carlo method, only returning valid holes if they pass a boundary check (all vertices lie within the cloth boundary) and an overlap check (different holes do not overlap). For further implementation details, we refer to the DEDO codebase [7].

Since holes are generated by directly manipulating the cloth mesh, they can also be represented as deformable loops, defined by a set of “loop vertices” (Figure 6) - while information about these vertices is not passed as input to our models, it is used later by our success metric for evaluation.

A.3 HangBag - Task Definition

We also perform an additional experiment on the `HangBag` task (Figure 7), in which a deformable bag with two handles must be placed on a hanger. To successfully complete the task, at least one of the handles must lie on the handle.

Similar to `HangProcCloth`, the hanger (anchor) is a pre-defined rigid body - for our experiments, we randomize the anchor pose, but keep the geometry fixed. Unlike `HangProcCloth`, however, the cloth is loaded from a pre-defined mesh rather than procedurally generated. As such, we do not randomize the cloth geometry for this task, leaving such experiments for future work (notably, the DEDO environment does provide multiple meshes for the bag).

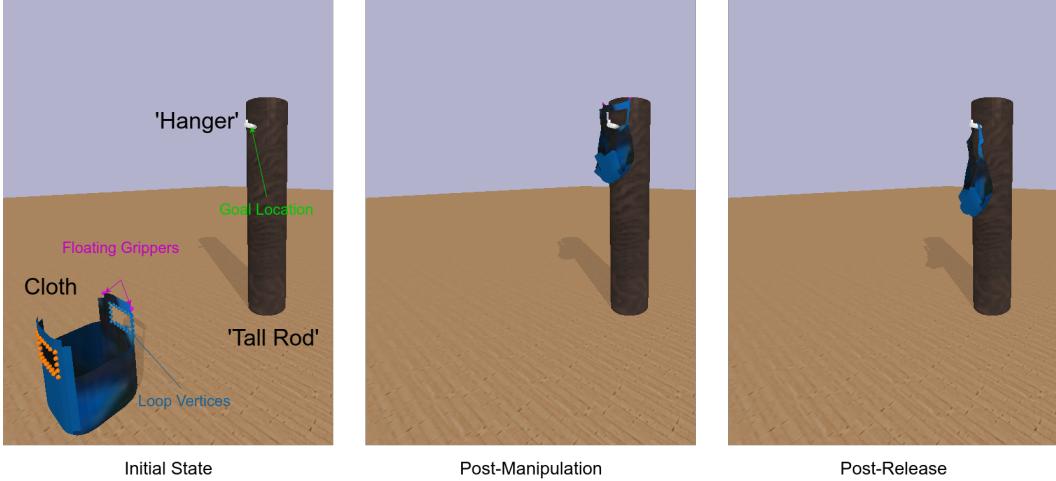


Figure 7: Sample demonstration of the HangBag task.

A.4 Cloth Control

The HangProcCloth and HangBag environments do not model a robot grasp - instead, the cloth is manipulated by applying force controls to floating “grippers”¹ attached to pre-defined points on the cloth (Figure 6,7). The grippers themselves are zero-mass and collision free.

Pseudo-expert Policy: To generate demonstrations, we hard-code a pseudo-expert policy with access to privileged environment information. At the initial state of the scene, the policy computes a target position for each gripper using the distance from the centroid of the loop vertices to the goal location in the hanger. If the cloth has multiple holes, a single hole is selected. When computing this target, we apply the same transformation that we apply to the anchor to randomize its pose - this ensures that the cloth is rotated appropriately to align with the hanger. At each time step, we also add a small correction to the target position based on the distance between the current centroid of the loop vertices and the goal position - this allows the pseudo-expert policy to adapt to small deformations in the cloth geometry, which we find meaningfully improves the success rate. For control, we pass the target positions for each gripper as inputs to a custom proportional-derivative controller (in addition to a target velocity of 0) with a velocity and position gains of 50 and a maximum force² of 5.

Evaluation Policy: To evaluate our models, we implement a separate evaluation policy without access to privileged environment information (e.g. goal location, deformable loop vertices). At the initial state of the scene, we run TAX3D on the full point cloud of the cloth³, and obtain the predicted position in the world frame of the two grippers (which we assume are attached to known points of the cloth). These target positions are then passed as inputs (in addition to a target velocity of zero) to a custom proportional-derivative controller, with the same gains as the pseudo-expert policy.

A.5 Episode Rollout

A.5.1 Rollout Phases

Each episode rollout consists of two phases (Figure 6): a **manipulation phase**, in which the grippers receive force control inputs at each time step to manipulate the cloth, and a **release phase**, in which

¹DEDO refers to these grippers as “anchors.” We refrain from this terminology since “anchor” denotes an entirely different object for our purposes.

²Following the DEDO implementation, this is not an overall maximum force magnitude - it is the maximum magnitude of the force along the x -, y -, and z -axes.

³Note that this is different from our training procedure (B.2), where we downsample cloth point clouds to 512 points.

the grippers “release” the cloth and allow it to fall. The release phase is fixed at 500 simulation steps, whereas the manipulation phase has a variable episode length depending on the setting (with each environment step corresponding to 8 simulation steps).

If the task is completed successfully, the cloth should be supported by the rigid anchor *after* the release phase. However, because we are learning a goal-prediction module to condition a policy’s control outputs, we use the post-manipulation, pre-release state of the cloth to label ground truth demonstrations.

A.5.2 Success Metric

To robustly determine whether or not the task has been successfully completed, we implement our own success metric consisting of two components:

1. **Centroid Check:** a binary metric that checks if the centroid of the deformable loop vertices is within a threshold distance of the goal location.
2. **Polygon Check:** a binary metric that projects the loop vertices and goal location onto the xy -plane, and then checks if the projected goal point lies on the interior of the polygon defined by the projected loop vertices. This is an intuitive heuristic that checks whether or not the hole “wraps” around the vertical rod of the hanger.

If the cloth has multiple holes, these metrics are computed individually for each hole - the task is considered successful if both are true for at least one hole.

For the HangProcCloth task, the success metric consists of a centroid check (with threshold 1.3) pre-release and a polygon check post-release. For the HangBag task, the success metric consists of two centroid checks, both with threshold 1.4, pre- and post-release - we do not use the polygon check for the HangBag task, as we found that it produces a larger number of false negatives.

A.6 Demonstration Generation

A.6.1 Randomizing Scene Configuration

For all HangProcCloth experiments, the objects in the scene are initialized to the following pose, shown in Figure 6:

	position (xyz)	orientation (Euler)
cloth	(0, 5, 8)	$(-\frac{\pi}{2}, 0, \frac{3\pi}{2})$
hanger	(0, 0, 8)	(0, 0, 0)
tall rod	(0, 0, 0)	(0, 0, 0)

Table 5: Initial HangProcCloth configuration

For all HangBag experiments, the objects in the scene are initialized to the following pose instead (shown in Figure 7):

	position (xyz)	orientation (Euler)
cloth	(0, 8, 2)	$(\frac{\pi}{2}, 0, 0)$
hanger	(0, 1.28, 9)	$(0, 0, \frac{\pi}{2})$
tall rod	(0, 0, 5)	$(\frac{\pi}{2}, 0, 0)$

Table 6: Initial HangBag configuration

To randomize scene configuration, the anchor is then transformed with a randomly sampled translation, and a randomly sampled rotation about the z -axis.

	Unseen	Unseen (OOD)
x -translation	$(-5, 5)$	$(-10, -5) \cup (5, 10)$
y -translation	$(0, -10)$	$(0, -10)$
z -translation	0	$(1, 5)$
z -rotation	$(-\frac{\pi}{3}, \frac{\pi}{3})$	$(-\frac{\pi}{3}, \frac{\pi}{3})$

All transformations are sampled uniformly at random from their respective ranges, with one small caveat: x -translations are chosen such that their signs match the sign of the sampled rotation. That is, if the z -rotation is sampled to be non-negative, then the x -translations are only sampled from the non-negative subset of the corresponding range. This ensures that the anchor always “faces” the cloth, such that the cloth need not undergo significant rotations for a successful placement.

As for the point clouds themselves, we obtain \mathbf{P}_B^* as a partial point cloud from an RGB-D render of the initial state of the environment (since the anchor is static). To guarantee correspondences, \mathbf{P}_A and \mathbf{P}_A^* are directly extracted from the mesh vertices of the cloth at its initial and post-manipulation states, respectively.

A.6.2 Experiment Datasets

As a reminder, cloth geometry (including holes) is randomized by following the parameter ranges and procedures described in A.2.1. The datasets for each experiment are generated as follows, where each tuple entry corresponds to the (**Train**, **Unseen**, **Unseen (OOD)**) settings, respectively:

	# cloths	# holes per cloth	# total demonstrations
HangProcCloth-simple	(1, 1, 1)	(1, 1, 1)	(16, 40, 40)
HangProcCloth-unimodal	(64, 40, 40)	(1, 1, 1)	(64, 40, 40)
HangProcCloth-multimodal	(32, 20, 20)	(2, 2, 2)	(64, 40, 40)
HangBag	(1, 1, 1)	(1, 1, 1)	(16, 40, 40)

For HangProcCloth-multimodal, we generate a successful demonstration for each hole per cloth - as such, there are half as many unique cloths as there are total demonstrations. For all demonstrations across all experiments, the anchor pose is randomized as described in A.6.1.

B Training Details

B.1 Model Architecture

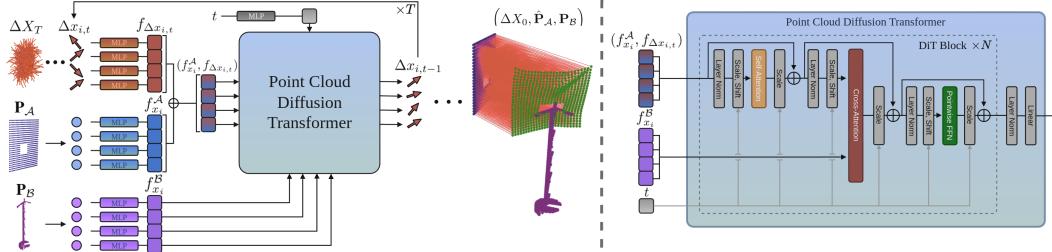


Figure 8: TAX3D model architecture. (Left). During inference, randomly sampled displacements $\Delta X_T \sim \mathcal{N}(0, \mathbf{I})$ are de-noised conditioned on action (\mathbf{P}_A) and anchor (\mathbf{P}_B) features; the final ΔX_0 is predicted to displace the action into a goal configuration. (Right). Our modified DiT [25] architecture combines self-attention and cross-attention for object-centric and scene-level reasoning.

As discussed in 4.2, we modify the standard DiT block [25] to include an additional cross-attention head 8. For all of our experiments, we train the same architecture:

depth	5	# of DiT blocks
num_heads	4	# heads per block
hidden_size	128	hidden size per block

These settings (namely, `num_heads` and `hidden_size`) are applied identically to the self-attention and cross-attention layers. During training and inference, our model always uses 100 diffusion steps, with a linear noise schedule.

B.2 Training Pre-Processing & Hyperparameters

For training, both the action and anchor point clouds are downsampled to 512 points using furthest point sampling⁴. The anchor point cloud is additionally augmented with z -axis rotations sampled uniformly at random from $[0, 2\pi]$.

All models are trained under the same hyperparameters with AdamW optimization and cosine scheduling with warmup:

learning_rate	1×10^{-4}
learning_rate_warmup_steps	100
weight_decay	1×10^{-5}
epochs	20,000
batch_size	16

C Evaluation Metrics

As discussed in Section 5, our method’s modeling of point-wise displacements allows us to directly use root-mean-squared-error (RMSE) as a distance metric between predicted and ground truth configurations of the cloth. To appropriately evaluate distributional predictions in our setting, we define two evaluation metrics⁵:

1. **Coverage RMSE:** For each demonstration with ground truth $\mathbf{P}_{\mathcal{A},i}^*$, we sample 20 predictions $\{\hat{\mathbf{P}}_{\mathcal{A},j}\}$, and keep the minimum RMSE. This is aggregated across all demonstrations in the dataset. Intuitively, this metric captures how well a model can produce all of the modes in a given dataset - that is, how well it *covers* a distribution.
2. **Precision RMSE:** We first collect demonstrations corresponding to a specific cloth geometry (this is either one demonstration for the unimodal case, or two demonstrations for the multimodal case) - for some cloth \mathcal{C} , this serves as a cloth-specific reference set $\{\mathbf{P}_{\mathcal{A},i}^*\}_{\mathcal{C}}$. We then sample 20 predictions conditioned on cloth \mathcal{C} , and compute for each prediction $\hat{\mathbf{P}}_{\mathcal{A},j}$ the minimum RMSE to ground truth point clouds in the reference set $\{\mathbf{P}_{\mathcal{A},i}^*\}_{\mathcal{C}}$ ⁶. This is aggregated across all 80 predictions, and then across all cloths. Intuitively, this metric captures how well a model can consistently produce predictions that are close to the dataset configurations - that is, how *precisely* it models a distribution.

D Experiments

The following sections contain all of the ablation and baseline comparisons for all experiments. Note that we do not report any RMSE metrics for DP3, since it is an end-to-end policy, and does not predict a point cloud.

⁴During policy evaluation, only the anchor point cloud is downsampled, as the full action point cloud is need to obtain target positions for the grippers.

⁵Both metrics bear strong similarity to the MMD metric, but are essentially modified to aggregate across different reference sets.

⁶In the multimodal case, all demonstrations for a single cloth are recorded under the same anchor configuration - as such, the world frame RMSEs are consistent across any cloth-specific reference set $\{\mathbf{P}_{\mathcal{A},i}^*\}_{\mathcal{C}}$.

D.1 HangProcCloth-simple Results

	RMSE (\downarrow)			Success Rate (\uparrow)		
	Train	Unseen	Unseen (OOD)	Train	Unseen	Unseen (OOD)
SD	0.054	0.741	2.898	0.94	0.98	0.00
SP	0.051	0.172	2.969	0.50	0.58	0.05
CD-W	0.052	0.224	3.680	1.00	1.00	0.00
CP-W	0.044	0.231	3.452	1.00	1.00	0.00
CD-NAC	1.586	1.498	1.741	0.56	0.60	0.53
CP-NAC	3.563	3.573	3.534	0.00	0.00	0.00
TAX3D-CD (Ours)	0.270	0.255	0.498	1.00	1.00	0.98
TAX3D-CP (Ours)	0.298	0.277	0.518	1.00	1.00	0.98
DP3	-	-	-	1.00	0.93	0.00

Table 7: HangProcCloth-simple: Fixed Cloth Geometry

D.2 HangProcCloth-unimodal Results

	RMSE (\downarrow)			Success Rate (\uparrow)		
	Train	Unseen	Unseen (OOD)	Train	Unseen	Unseen (OOD)
SD	0.127	0.805	4.246	0.27	0.20	0.00
SP	0.095	0.779	3.955	0.45	0.48	0.08
CD-W	0.099	0.464	12.464	0.95	0.88	0.00
CP-W	0.085	0.511	19.923	0.95	0.85	0.00
CD-NAC	2.233	2.243	2.326	0.16	0.28	0.10
CP-NAC	3.987	3.934	3.923	0.02	0.00	0.03
TAX3D-CD (Ours)	0.052	0.405	0.395	0.94	0.90	0.85
TAX3D-CP (Ours)	0.051	0.390	0.422	0.93	0.95	0.80
DP3	-	-	-	0.98	0.38	0.03

Table 8: HangProcCloth-unimodal: Random Cloth Geometry (1-Hole)

D.3 HangProcCloth-multimodal Results

	RMSE (\downarrow)			Success Rate (\uparrow)		
	Train	Unseen	Unseen (OOD)	Train	Unseen	Unseen (OOD)
SD	1.048	1.361	3.832	0.36	0.48	0.05
SP	1.028	1.407	3.825	0.53	0.40	0.05
CD-W	1.436	1.535	5.006	0.95	0.63	0.03
CP-W	1.418	1.555	48.760	0.97	0.70	0.08
CD-NAC	2.467	2.560	2.650	0.25	0.10	0.10
CP-NAC	4.148	4.237	4.114	0.00	0.00	0.00
RD	2.978	2.987	3.057	0.53	0.60	0.50
RP	1.360	1.417	1.432	0.47	0.65	0.48
TAX3D-CD (Ours)	1.413	1.532	1.610	0.95	0.98	0.73
TAX3D-CP (Ours)	1.400	1.568	1.608	0.94	0.85	0.70
DP3	-	-	-	0.97	0.45	0.00

Table 9: HangProcCloth-multimodal: Random Cloth Geometry (2-Hole)

	Coverage RMSE (\downarrow)			Precision RMSE (\downarrow)		
	Train	Unseen	Unseen (OOD)	Train	Unseen	Unseen (OOD)
SD	0.075	0.740	3.500	1.114	1.379	4.214
SP	0.061	0.773	3.462	1.093	1.432	4.062
CD-W	0.054	0.426	4.269	0.132	0.477	4.351
CP-W	0.047	0.446	33.868	0.099	0.504	47.372
CD-NAC	1.484	1.741	1.812	1.785	2.011	2.167
CP-NAC	3.593	3.720	3.603	3.770	3.914	3.786
RD	2.978	2.987	3.057	1.296	1.087	1.114
RP	1.360	1.417	1.432	1.305	1.192	1.154
TAX3D-CD (Ours)	0.040	0.457	0.543	0.084	0.403	0.558
TAX3D-CP (Ours)	0.038	0.453	0.540	0.087	0.495	0.631
DP3	-	-	-	-	-	-

Table 10: HangProcCloth-multimodal: Random Cloth Geometry (2-Hole)

D.4 HangBag Results

	RMSE (\downarrow)			Success Rate (\uparrow)		
	Train	Unseen	Unseen (OOD)	Train	Unseen	Unseen (OOD)
SD	0.079	0.346	2.041	0.00	0.00	0.00
SP	0.062	0.125	1.886	0.00	0.00	0.00
CD-W	0.053	0.178	3.337	1.00	0.78	0.00
CP-W	0.046	0.173	4.204	1.00	0.80	0.03
CD-NAC	1.254	1.301	1.309	0.56	0.38	0.33
CP-NAC	1.485	1.487	1.486	0.00	0.03	0.00
TAX3D-CD (Ours)	0.183	0.204	0.245	0.94	0.83	0.78
TAX3D-CP (Ours)	0.173	0.192	0.233	0.94	0.93	0.78
DP3	-	-	-	1.00	0.73	0.00

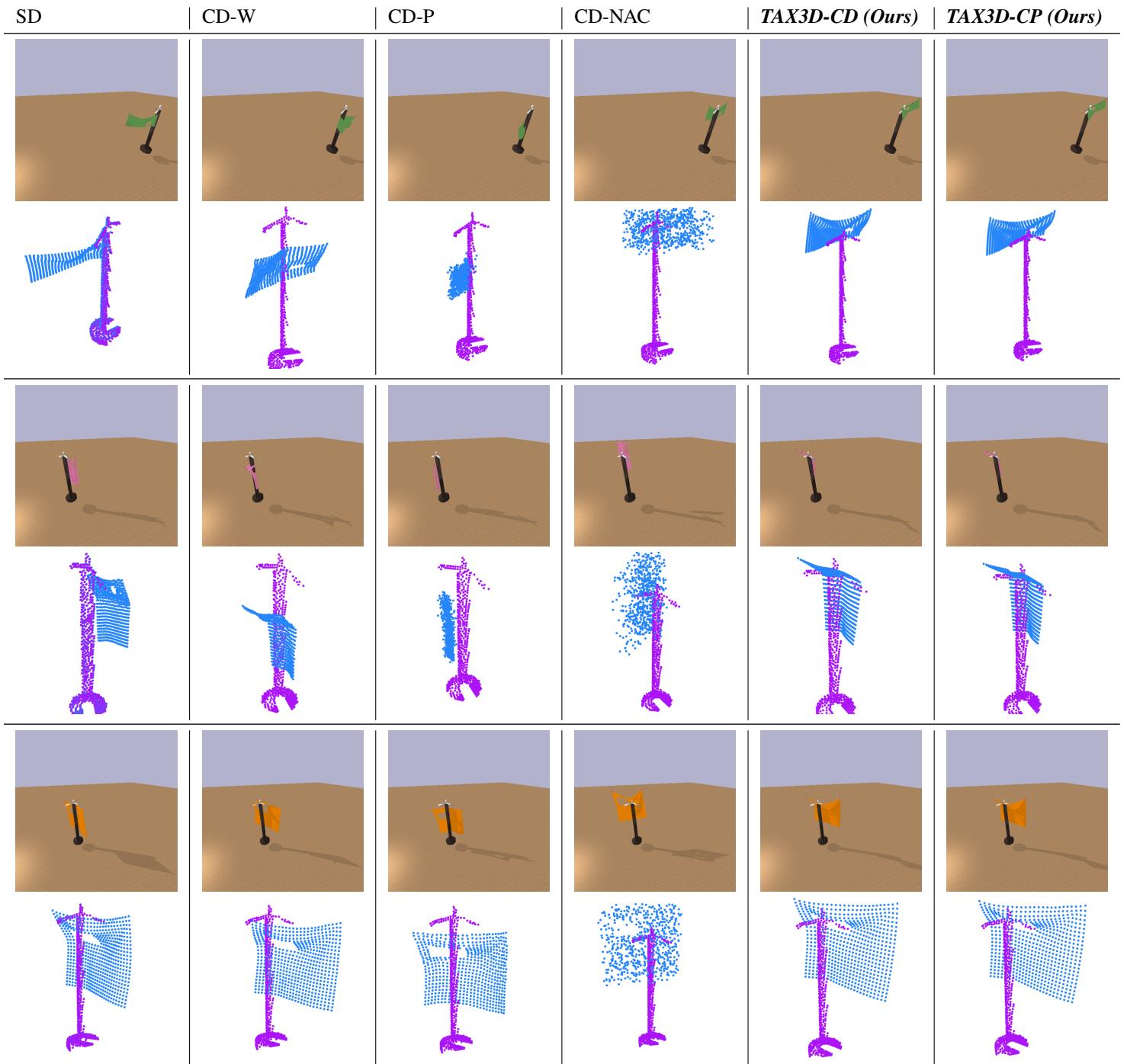
Table 11: HangBag: Fixed Cloth Geometry

D.5 Additional Visualizations

The following pages contain visualizations of our method (as well as all baseline methods) across classes of cloth geometry (single- and double-hole) on out-of-distribution scene configurations. For every visualized prediction, both the cloth and configuration were unseen by the model during training.

Within each table row, the top displays the result of the evaluation policy rollout on the corresponding model’s predicted cloth configuration; the bottom displays the predicted configuration itself. Zooming in may be necessary to properly view the policy executions. For more visualizations, including videos of the policy rollout and the full reverse diffusion process, see our [project page](#).

D.5.1 Single-Hole Cloth, Unseen Configuration (Out-of-distribution)



D.5.2 Double-Hole Cloth, Unseen Configuration (Out-of-distribution)

