TEXT-TO-3D POLICY: FINE-GRAINED LANGUAGE-BEHAVIOR ALIGNMENT FOR UNSEEN SPECIFICATION GENERALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

3D visuomotor policies provide a strong foundation for spatially precise manipulation, yet current text-to-3D policies struggle to follow unseen fine-grained behavioral specifications beyond those covered by demonstrations. We study this challenge as *unseen specification generalization*, where language specifies behaviorally significant variations, such as target position, displacement, or articulated state, that are absent from policy training. We find that pretrained language representations and conventional global behavior-language alignment capture coarse task semantics but often blur nearby specifications that require distinct behaviors. We introduce **T3DP**, a **T**ext-to-**3D P**olicy framework for fine-grained language-behavior alignment. Rather than compressing each instruction and demonstration into a single global embedding, T3DP preserves their local structures and establishes bidirectional token-level correspondence between linguistic elements and behavioral segments. This directly grounds subtle linguistic variations in the behavior components they affect, preventing closely related specifications from collapsing in the representation space. The resulting specification-sensitive language representation conditions a point-cloud-based 3D diffusion policy, enabling more precise control over unseen behavioral specifications without modifying the underlying policy architecture. Across Meta-World, ManiSkill, and RoboTwin, T3DP improves average held-out-specification success over global language-behavior alignment by **+11.0-14.2 points**, with gains on all 15 task families; on real-robot tasks, it further raises average success from 47.5% to 65.0% (**+17.5 points**). Representation and action-probe analyses show that fine-grained alignment better preserves specification geometry and action-relevant variation, linking local behavior grounding to downstream control.

1 INTRODUCTION

Learning visuomotor policies from 3D observations has emerged as a powerful paradigm for robotic manipulation (Gervet et al., 2023; Ze et al., 2024). By making scene geometry explicit, point-cloud representations provide strong spatial grounding for precise control and support effective generalization from limited demonstrations (Ze et al., 2024). Meanwhile, natural language offers a flexible interface for specifying diverse robot behaviors, motivating a new generation of *text-to-3D policies* that translate human instructions into actions grounded in 3D observations (Ke et al., 2025). However, a capable text-to-3D policy should determine not only *what* task to perform, but also precisely *how* that task should be executed (Fig. 1). Whether 3D policies can follow such behaviorally precise instructions beyond the demonstrations observed during training remains an open challenge (Gong et al., 2023).

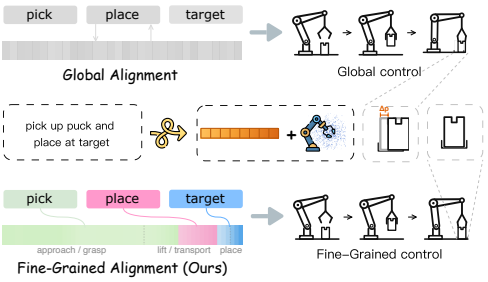

Figure 1: **Global alignment** matches entire sequences, yielding nearly uniform token-to-segment similarities. **Fine-grained alignment** learns distinct segment-to-token grounding, enabling more precise control with lower errors.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Consider a robot instructed to reach a particular position, push an object by a specified displacement, place it at a desired location, or open an articulated object to a target state. These instructions describe different *specifications* within the same task family: they share the same coarse task semantics but demand distinct behaviors (Gong et al., 2023; Zheng et al., 2022). Such specifications often form large, continuous, or compositional spaces, making exhaustive coverage through demonstration collection impractical (Gong et al., 2023; Zheng et al., 2022). Moreover, the same or similar initial observation may admit multiple valid behaviors, so the intended specification cannot always be inferred from the scene alone and must be communicated through language. Although prior work has studied generalization to unseen tasks, objects, scenes, and linguistic expressions (Jang et al., 2022; Shridhar et al., 2022; Gong et al., 2023), considerably less attention has been paid to unseen behavior specifications within a task family. We formalize this challenge as **unseen specification generalization**: a policy is trained on only a subset of language-specified behaviors and must execute held-out specifications without adaptation.

A natural approach is to condition the policy directly on representations from a pretrained language model (Jang et al., 2022). These representations encode rich semantic knowledge, but are learned independently of robot interaction and need not preserve how subtle linguistic variations correspond to changes in physical behavior (Zhang et al., 2025). Recent work therefore grounds language in behavior-relevant representations derived from robot transitions or trajectories (Ma et al., 2023; Myers et al., 2023; Li et al., 2024). Our preliminary study confirms that generic language representations alone provide insufficient actionable specification information. The alignment analysis in Appendix D.4 further shows that global alignment does not consistently preserve the geometry of continuous specifications. By compressing an entire instruction and demonstration into a pair of global embeddings, existing objectives primarily emphasize task- or trajectory-level compatibility and provide limited pressure to distinguish nearby specifications. This raises a critical question: *How can language be grounded in 3D robot behavior without losing the fine-grained structure that distinguishes closely related specifications?*

Our key insight is that fine-grained specifications are often distinguished by *local* rather than global variations. Closely related instructions share most of their linguistic content, while differing only in behavior-defining elements such as a target, direction, distance, or desired state. Likewise, their corresponding demonstrations may share most of the motion structure while differing only in the behavioral components affected by the specification. Global aggregation can obscure these sparse but decisive correspondences. We therefore hypothesize that directly aligning local linguistic elements with local behavior representations can better preserve the specification-dependent structure that global alignment tends to obscure.

Built on this insight, we propose **T3DP**, a **T**ext-to-**3D P**olicy framework for fine-grained language-behavior alignment. T3DP first encodes demonstration trajectories into a sequence of local behavior representations and preserves the token-level structure of their paired instructions. Rather than matching only globally pooled features, it establishes bidirectional token-level correspondence between the two modalities, allowing specification-dependent linguistic variations to interact directly with the behavioral components they affect. This simple alignment objective reshapes the language representation to preserve subtle distinctions among closely related specifications. The resulting specification-sensitive language representation conditions a point-cloud-based 3D diffusion policy, enabling more precise control over unseen behavioral specifications without redesigning the underlying policy architecture. Under strictly held-out specification splits, T3DP improves average success over global language-behavior alignment by **14.2, 13.5, and 11.0 points** on Meta-World, ManiSkill, and RoboTwin, respectively, with gains on all 15 task families. On an AgileX PiperX robot, it raises average success from **47.5% to 65.0%** evaluated over 20 specifications per task. Representation and action-probe analyses further show that its learned condition better preserves specification geometry and action variation. In summary, our main contributions are:

- We formulate **unseen specification generalization** for text-to-3D policies, studying whether they can execute fine-grained behavioral specifications absent from policy training.
- We identify the coarse granularity of global language-behavior alignment as a key bottleneck and propose **T3DP**, a simple framework that uses local cross-modal correspondence to learn specification-sensitive language representations for 3D policy learning.
- We demonstrate consistent gains across simulation and real-robot tasks, with representation and action-probe analyses showing better preservation of specification structure and action variation.

2 Related Work

Language-Conditioned 3D Visuomotor Policies. 3D visuomotor policies exploit structured geometric representations to provide spatial inductive biases for precise manipulation (Shridhar et al., 2023; Gervet et al., 2023; Ze et al., 2024). Language-conditioned variants, including PerAct, Act3D, 3DDA, and 3D-LOTUS, further condition action prediction on linguistic descriptions, enabling shared policies across diverse tasks (Shridhar et al., 2023; Gervet et al., 2023; Ke et al., 2025; Garcia et al., 2025). Our work considers a finer-grained setting in which instructions share the same task semantics but require distinct behaviors. We therefore focus on preserving instruction-dependent behavioral variations within a task for downstream control.

Generalization to Language-Specified Behaviors. Language-conditioned manipulation has studied generalization across unseen tasks, objects, scenes, and linguistic variations (Jang et al., 2022; Shridhar et al., 2022; Gong et al., 2023). Gong et al. (2023) further study unseen continuous goal states within the same task in ARNOLD, primarily varying the desired state along a task-specific dimension. Our setting additionally considers spatial specifications, such as target positions and relative displacements, where language determines the spatial structure of the required motion. We study generalization to such held-out specifications within the same task family.

Language-Behavior Alignment. Multimodal alignment has been widely studied in vision–language learning, from global image–text contrastive alignment in CLIP (Radford et al., 2021) to finer-grained correspondence in FILIP (Yao et al., 2022) and FG-CLIP (Xie et al., 2025). Similar ideas have been extended to robot learning, where LIV (Ma et al., 2023), GRIF (Myers et al., 2023), DecisionNCE (Li et al., 2024), and T2DA (Zhang et al., 2025) ground language in visual, transition, or behavior representations. These methods demonstrate the value of language–behavior alignment, but largely operate at the level of a task, transition, or aggregated trajectory. Fine-grained correspondence benefits local semantic distinctions in vision–language learning, yet remains less explored for grounding specification-bearing language in local robot behavior. T3DP addresses this gap through local language–behavior correspondence that preserves specification-dependent behavioral variations for downstream control.

3 Preliminary Study

Language-conditioned control remains strongly tied to visual target cues. We conduct a target-cue ablation on Meta-World Reach, where the target can be provided through either observation or language. As shown in Table 1(a), removing the language cue leaves success essentially unchanged (54.7% vs. 55.0%), whereas removing the visual cue reduces it to 12.7% and increases the prediction error. This indicates that language alone provides some actionable information but remains substantially less effective than explicit visual target cues, motivating settings where similar observations require distinct executions.

Table 1: Diagnostic study on Meta-World.

(a) Target information

Target	Lang.	Succ. ↑	Error ↓
✓	✓	54.7	0.093
✓	✗	**55.0**	**0.091**
✗	✓	12.7	0.158

(b) Specification geometry

Metric	Generic	Global	Ours
Spearman	0.373	0.591	**0.720**
kNN@3	0.367	0.400	**0.574**

Global alignment only partially preserves fine-grained specification structure. We compare generic language representations, global language-behavior alignment, and fine-grained alignment on held-out Reach, Push, and Door Open specifications. Table 1(b) shows that global alignment improves specification geometry over generic representations, while fine-grained alignment further improves both global ordering and local neighborhood preservation. It suggests that trajectory-level supervision can still obscure distinctions among closely related specifications, highlighting the necessity of our fine-grained alignment.

4 Method

Our preliminary study (Section 3) shows that generic language representations lack sufficient behaviorally actionable information, while conventional global language-behavior alignment does not consistently preserve the geometry of continuous specifications. This motivates grounding language

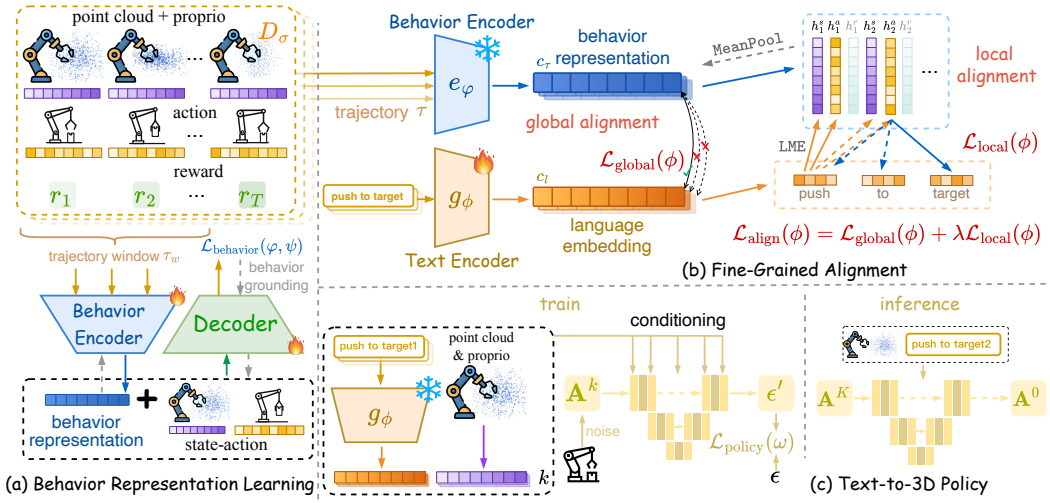

Figure 2: **Overview of T3DP.** (a) A behavior encoder learns specification-dependent representations from demonstration windows through reward prediction conditioned on state–action pairs. (b) With the behavior encoder frozen, the text encoder is trained using global language-behavior alignment and local alignment between state–action hidden states and instruction tokens. (c) The aligned text encoder is frozen to train a language-conditioned 3D diffusion policy using point-cloud observations and proprioceptive states. At inference time, the policy executes unseen specifications from language instructions and current observations, without demonstrations or further adaptation.

in local, specification-relevant behavioral structure rather than relying solely on global task-level summaries. Building on this insight, we introduce **T3DP**, which is trained in three stages, as illustrated in Figure 2. Algorithm pseudocodes are presented in Appendix C. First, we learn a behavior encoder to extract specification-dependent behavior representations from demonstrations (Section 4.2). Second, we ground the text encoder in these representations through fine-grained language-behavior alignment (Section 4.3). Finally, we freeze the aligned text encoder and train a language-conditioned 3D diffusion policy (Section 4.4).

4.1 Problem Formulation

We consider language-conditioned manipulation with point-cloud observations, where each specification σ from a task family $\mathcal{T}$ defines an MDP $\mathcal{M}_\sigma = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r_\sigma, \gamma)$. Specifications in a family share $\mathcal{S}$, $\mathcal{A}$, and $\mathcal{P}$, but differ in the specification-dependent reward r_σ; each state contains a point-cloud observation o and proprioceptive state p. Training uses paired instructions l_{σ_n} and expert datasets $\mathcal{D}_{\sigma_n} = \{\tau_{\sigma_n}^{(k)}\}_{k=1}^{K_n}$ for specifications $\{\sigma_n\}_{n=1}^{N}$, with $\tau = \{(s_t, a_t, r_t)\}_{t=1}^{T}$, where N, K_n, and T denote the number of training specifications, expert trajectories for specification σ_n, and the trajectory length, respectively. At test time, the policy receives only the language instruction for a held-out specification from the same family. Our goal is to learn a behavior-grounded language encoder that distinguishes related instructions and a policy $\pi(a_t \mid o_t, l)$ that generalizes zero-shot.

4.2 Behavior Representation Learning from Demonstrations

Table 1 shows that generic language representations alone do not carry enough behaviorally actionable information to specify a task. We therefore first learn a behavior encoder that extracts specification-dependent behavioral structure from demonstrations before aligning it with language.

Specifications within the same task family share the same state and action spaces and transition dynamics, and differ only in the specification-dependent reward r_σ. The behavioral differences that separate closely related specifications therefore originate from the reward rather than from dynamics or embodiment. This motivates training the behavior encoder with a reward-predictive objective: the behavior embedding of a trajectory segment should be able to predict the reward corresponding to

any state-action pair within the same specification. Succeeding at this requires the embedding to capture specification-dependent behavioral structure, rather than surface-level motion statistics.

Given a length-L window $\tau_w = \{(s_t, a_t, r_t)\}_{t=w}^{w+L-1}$, e_φ uses PointNet and modality-specific MLPs to tokenize each timestep, followed by a bidirectional Transformer and mean pooling. The point cloud and proprioceptive state form $\tilde{s}_t$, while the action and reward form $\tilde{a}_t$ and $\tilde{r}_t$. All tokens at same timestep share the same temporal positional encoding. The resulting $3L$ tokens are arranged in temporal order and compute the behavior embedding as

$$\mathbf{X}_w = \left[\tilde{s}_w, \tilde{a}_w, \tilde{r}_w, \ldots, \tilde{s}_{w+L-1}, \tilde{a}_{w+L-1}, \tilde{r}_{w+L-1}\right], \tag{1}$$

$$e_\varphi(\tau_w) = \mathrm{MeanPool}(\mathrm{Transformer}(\mathbf{X}_w)). \tag{2}$$

To train e_φ, a lightweight decoder d_ψ predicts the reward $\hat{r} = d_\psi(e_\varphi(\tau_w), s, a)$ for a state-action pair drawn from the same specification as the encoded trajectory. We select a trajectory $\tau \in \mathcal{D}_\sigma$, a segment $\tau_w \subset \tau$, and a transition $(s, a, r) \in \tau \setminus \tau_w$, ensuring that the target reward is not contained in the encoded segment, and jointly optimize the encoder and decoder with

$$\mathcal{L}_{\mathrm{behavior}}(\varphi, \psi) = \mathbb{E}_{\substack{\sigma\sim\mathcal{T},\ \tau\in\mathcal{D}_\sigma,\ \tau_w\subset\tau \\ (s,a,r)\in\tau\setminus\tau_w}} \left[\left(d_\psi(e_\varphi(\tau_w), s, a) - r\right)^2\right]. \tag{3}$$

4.3 Fine-Grained language-behavior alignment

The behavior encoder captures what each specification entails in physical execution, but demonstrations are generally unavailable for novel specifications at deployment. Language provides a scalable interface, but generic representations do not necessarily encode behavioral consequences. Language-behavior alignment transfers actionable structure from demonstrations into instructions, which is crucial when a single observation allows for multiple plausible behaviors and generic language lacks distinction. It is also important for multimodal policy learning because precise point-cloud observations can otherwise dominate generic language features, as shown in our preliminary study. We therefore seek an alignment that preserves distinctions among closely related specifications and provides an effective condition for the policy below.

Limitations of global alignment. A common approach to behavior grounding is to align globally pooled instruction and behavior embeddings (Radford et al., 2021; Zhang et al., 2025). Although global alignment improves over generic language representations (Appendix D.4), its gains are limited and inconsistent for closely related specifications. Such specifications share most linguistic and behavioral content, differing only in decisive elements such as target location, direction, or desired state. Pooling both modalities can obscure these differences, motivating finer-grained alignment.

Fine-grained alignment. To preserve these subtle distinctions, we retain local language and behavior representations and explicitly model their cross-modal correspondence. This grounds specification-defining linguistic elements in the behavioral components they affect and complements global alignment by preserving behaviorally consequential changes.

Implementation and loss function. We use a pretrained Transformer text encoder g_ϕ, similar to T5 and CLIP (Raffel et al., 2020; Radford et al., 2021). For an instruction l, the text encoder produces token representations $E(l) = g_\phi(l) = \{e_j\}_{j=1}^M$ and a pooled representation $c_l = \frac{1}{M}\sum_{j=1}^M e_j$. For its paired full trajectory $\tau = \{(s_t, a_t, r_t)\}_{t=1}^T$, let $\mathbf{H}_\varphi(\tau) = \left[h_1^s, h_1^a, h_1^r, \ldots, h_T^s, h_T^a, h_T^r\right]$ denote the output of the trained behavior Transformer before mean pooling. We use the pooled representation $c_\tau = e_\varphi(\tau) = \mathrm{MeanPool}(\mathbf{H}_\varphi(\tau))$ for global alignment and retain the state and action hidden states $Z(\tau) = \{z_i\}_{i=1}^{2T} = \left[h_1^s, h_1^a, \ldots, h_T^s, h_T^a\right]$ for fine-grained alignment.

Given a batch of B paired trajectories and instructions, global alignment uses c_τ and c_l in a contrastive objective with matched pairs as positives and mismatched pairs as negatives:

$$\mathcal{L}_{\mathrm{global}}(\phi) = -\frac{1}{2B}\sum_{b=1}^{B}\left[\log\frac{\exp(c_{\tau_b}^\top c_{l_b}/\gamma)}{\sum_{q=1}^B \exp(c_{\tau_b}^\top c_{l_q}/\gamma)} + \log\frac{\exp(c_{\tau_b}^\top c_{l_b}/\gamma)}{\sum_{q=1}^B \exp(c_{\tau_q}^\top c_{l_b}/\gamma)}\right], \tag{4}$$

where γ is the contrastive temperature. This brings paired embeddings closer and separates unpaired ones within the batch.

Because global pooling may dilute specification-defining elements, we additionally match state-action hidden states with instruction tokens. For each segment-token pair, we compute its cosine similarity $s_{ij} = \cos(z_i, e_j)$ and define two directional late-interaction scores:

$$S_{\tau\to l}(Z,E) = \frac{1}{N}\sum_{i=1}^{N} \mathrm{LME}_\beta\left(\{s_{ij}\}_{j=1}^{M}\right), \qquad S_{l\to\tau}(Z,E) = \frac{1}{M}\sum_{j=1}^{M} \mathrm{LME}_\beta\left(\{s_{ij}\}_{i=1}^{N}\right), \tag{5}$$

where $\mathrm{LME}_\beta(\{x_k\}_{k=1}^K) = \frac{1}{\beta}\log\left(\frac{1}{K}\sum_{k=1}^K \exp(\beta x_k)\right)$ is a smooth approximation to the maximum, with $\beta > 0$ controlling the matching sharpness. The two scores softly match segments to relevant tokens and tokens to the segments expressing their behavioral meaning. We apply the same bidirectional contrastive construction to these fine-grained scores:

$$\mathcal{L}_{\mathrm{local}}(\phi) = -\frac{1}{2B}\sum_{b=1}^{B}\left[\log\frac{\exp(S_{\tau\to l}(Z_b,E_b)/\gamma)}{\sum_{q=1}^{B}\exp(S_{\tau\to l}(Z_b,E_q)/\gamma)} + \log\frac{\exp(S_{l\to\tau}(Z_b,E_b)/\gamma)}{\sum_{q=1}^{B}\exp(S_{l\to\tau}(Z_q,E_b)/\gamma)}\right]. \tag{6}$$

The final objective combines global and local alignment with weight λ:

$$\mathcal{L}_{\mathrm{align}}(\phi) = \mathcal{L}_{\mathrm{global}}(\phi) + \lambda\mathcal{L}_{\mathrm{local}}(\phi). \tag{7}$$

During alignment, we freeze the behavior encoder e_φ and fine-tune the text encoder using LoRA.

4.4 Language-Guided 3D Policy Learning

Natural language provides a universal interface compatible with foundation models and expresses semantic or compositional goals that fixed numerical parameters cannot fully capture, such as "open the door". We instantiate the downstream policy with a point-cloud-based 3D diffusion policy (Ze et al., 2024), which encodes the current point-cloud observation o_t and proprioceptive state p_t into a compact 3D feature and combines them with language to denoise an action sequence. Because the observation specifies scene geometry but not the intended behavior when one scene admits multiple specifications, the policy requires complementary, specification-dependent language guidance.

Given an instruction l, we use the behavior-aligned text encoder from Section 4.3 to compute pooled language embedding c_l as the policy condition. Although the downstream policy consumes only this pooled representation, language-behavior alignment reshapes c_l to preserve fine-grained behavioral distinctions. We freeze g_ϕ during policy learning and train a single policy for all training specifications within each task family $\mathcal{T}$. Let $\mathbf{A}^0$ denote an expert action sequence and $\mathbf{A}^k$ its noisy version at diffusion step k. The denoising network ϵ_ω is conditioned on the point-cloud observation, proprioceptive state, and aligned instruction representation, and is optimized using

$$\mathcal{L}_{\mathrm{policy}}(\omega) = \mathbb{E}_{\mathbf{A}^0,\,\epsilon,\,k}\left[\left\|\epsilon - \epsilon_\omega\left(\mathbf{A}^k, k, o_t, p_t, c_l\right)\right\|_2^2\right], \tag{8}$$

where $\epsilon \sim \mathcal{N}(0, I)$ denotes the noise added in the forward diffusion process. Importantly, specification-dependent targets, such as desired positions, are not rendered in the point-cloud observation because such information is generally unavailable in real scenes and must instead be communicated through language. At inference time, the policy requires only the current observation, robot state, and instruction to execute held-out specifications without demonstrations or further adaptation.

5 Experiments

Our experiments are organized around three questions. First, does T3DP consistently improve generalization to unseen task specifications over competing baselines (Section 5.1)? Second, we isolate the contribution of fine-grained alignment by comparing it with raw language conditioning and global language-behavior alignment (Section 5.2). Third, does the advantage of fine-grained alignment transfer to a physical robot (Section 5.3)? Additional analyses of scalability, mixed-task training, RoboTwin results, and representation properties are provided in Appendix D.

Benchmarks. We evaluate T3DP on three robot manipulation benchmarks: Meta-World (Yu et al., 2021), ManiSkill (Mu et al., 2021), and RoboTwin (Chen et al., 2026). Together, these benchmarks cover diverse task families, including spatial goal reaching, object interaction, placement,

Table 2: Unseen-specification generalization on Meta-World and ManiSkill. We report the average success rate (%) and corresponding standard deviation over three independently trained seeds. "Avg." denotes the mean across the five task families within each benchmark. Best and second-best results are shown in **bold** and underlined, respectively.

Meta-World						
Method	**Reach**	**Push**	**Pick Place**	**Door Open**	**Drawer Open**	**Avg.**
DP3 (Ze et al., 2024)	9.7±2.3	19.3±4.0	12.7±9.3	15.3±5.0	24.0±4.4	16.2
3DDA (Ke et al., 2025)	23.3±18.9	31.7±20.2	26.7±11.5	16.7±2.9	6.7±7.6	21.0
3D-LOTUS (Garcia et al., 2025)	21.7±7.6	1.7±2.9	1.7±2.9	38.3±15.3	28.3±2.9	18.3
T2DA (Zhang et al., 2025)	24.3±4.0	45.7±8.6	46.0±1.7	41.3±8.3	23.3±2.1	36.1
T3DP (Ours)	**38.3**±2.9	**64.7**±17.9	**58.0**±2.6	**59.7**±4.2	**30.7**±12.7	**50.3**
Δ vs. DP3	+28.6	+45.4	+45.3	+44.4	+6.7	+34.1
ManiSkill						
Method	**Reach**	**Push Cube**	**Pull Cube**	**Pick Cube**	**Turn Valve**	**Avg.**
DP3 (Ze et al., 2024)	3.3±0.6	8.0±2.0	16.3±4.2	53.3±1.5	21.0±3.0	20.4
3DDA (Ke et al., 2025)	35.0±8.7	36.7±25.2	6.7±2.9	8.3±5.8	**56.7**±7.6	28.7
3D-LOTUS (Garcia et al., 2025)	**41.7**±2.9	**51.7**±5.8	35.0±8.7	0.0±0.0	53.3±2.9	36.3
T2DA (Zhang et al., 2025)	32.3±4.6	15.0±9.1	48.3±6.1	72.0±12.5	36.0±2.0	40.7
T3DP (Ours)	38.0±4.4	47.3±4.0	**58.7**±3.5	**77.7**±6.0	49.3±1.2	**54.2**
Δ vs. DP3	+34.7	+39.3	+42.4	+24.4	+28.3	+33.8

and articulated-object manipulation. For each task family, we construct disjoint training and test specification sets, and evaluate each baseline on the test specification set without further adaptation. The desired behavior is explicitly specified through language, yet such specifications may be semantically similar to one another and therefore hard to distinguish from raw instruction embeddings alone. Detailed benchmark and task descriptions are provided in Appendix A.

Baselines. We compare T3DP with baselines from three categories: i) 3D diffusion policy (Ze et al., 2024), a strong 3D visuomotor policy without language conditioning; ii) independent language-conditioned 3D policies, including 3DDA (Ke et al., 2025) and 3D-LOTUS (Garcia et al., 2025); and iii) behavior–instruction alignment methods, represented by T2DA (Zhang et al., 2025), which performs global language-behavior alignment. In contrast, T3DP introduces fine-grained local alignment between behavior and language representations. Following the data regime of DP3, all methods are trained with only a limited number of expert demonstrations. All methods use the same demonstrations, observation inputs, task instructions, training and test specification splits, and evaluation protocol. Detailed baseline implementations are provided in Appendix A.5.

5.1 Main Results

We evaluate unseen-specification generalization on Meta-World and ManiSkill. Policies are trained independently for each task family with different specifications. Table 2 reports the results of comparing T3DP against several baselines, with all results averaged over three training seeds.

Meta-World. T3DP achieves the highest average success at **50.3%**, improving on T2DA, the strongest baseline, by **14.2 percentage points**. The improvement is consistent across all five task families, with gains ranging from 7.4 to 19.0 points. On the more challenging Push and Pick Place tasks, which require precise object interaction and placement control, several baselines achieve substantially lower success rates. T3DP delivers particularly pronounced gains on these tasks, suggesting that fine-grained alignment enables instruction representations to capture the subtle distinctions between closely related behavioral specifications.

ManiSkill. Across the five ManiSkill task families, T3DP attains an average success rate of **54.2%**, a **13.5-point advantage** over T2DA. This improvement is positive on all five tasks, ranging from 5.7 to 32.3 points, and is largest on Push Cube. Push Cube requires precise object interaction, yet T2DA reaches only 15.0% success, compared with 47.3% for T3DP. Other baselines struggle on Pick Cube: 3DDA reaches only 8.3%, and 3D-LOTUS is difficult to train, whereas T3DP achieves 77.7%.

These results suggest that fine-grained alignment helps distinguish subtle behavioral requirements in demanding manipulation tasks.

Across Meta-World and ManiSkill, T3DP consistently outperforms the global-alignment baseline. This advantage persists under the limited-demonstration regime inherited from DP3, highlighting the value of fine-grained alignment for distinguishing subtle behavioral requirements.

5.2 Ablation Experiments

We isolate the contribution of T3DP's language-conditioning and alignment components on all five Meta-World task families. The four variants form a hierarchical comparison: `T3DP w/o language` removes the language input and is equivalent to DP3; `T3DP w/o alignment` directly uses the pretrained language representation without language-behavior alignment; `T3DP w/o fine-grained` retains only the global pooled alignment used by T2DA; and `T3DP` adds the proposed token-to-segment fine-grained alignment. Results are shown as success rates for each task family and their average in Figure 3.

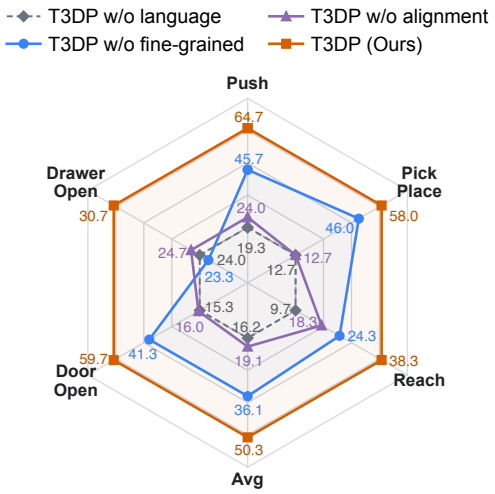

Figure 3: Hierarchical component ablation on Meta-World. Each spoke is scaled independently with the task-specific T3DP w/o language score as its inner reference and additional range above the maximum observed score. Avg denotes the mean over the five task families.

Does T3DP's alignment contribute to the improvement? Adding raw language conditioning to the no-language baseline increases average success from 16.2% to 19.1%, a gain of 2.9 points. Global language-behavior alignment raises the average to 36.1%, demonstrating the benefit of behavior grounding. Adding local fine-grained alignment (T3DP) achieves the strongest average success at **50.3%**, improving over the global-only variant by **14.2 points**. T3DP also performs best on every task family, with gains of 7.4–19.0 points over the global-only variant. Together, these results show that fine-grained language-behavior alignment consistently benefits over raw language conditioning and global alignment.

Does the learned language representation encode behavior-relevant information? We train a small predictor to recover the expert action from the learned language condition, holding the robot state fixed and varying only the language specification. A predictor given only the state cannot distinguish actions from different specifications and collapses to their average action, whereas one given T3DP's fine-grained language representation follows each specification's target action (Figure 4). On Reach, T3DP lowers the mean absolute action error (MAE) from 0.2567 to 0.1238, beating global alignment (T2DA, 0.1423). The full results on Reach and Push are in Appendix D.4.

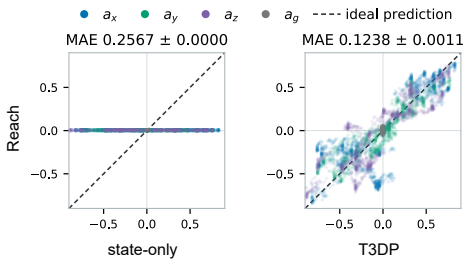

Figure 4: With the robot state fixed while only the language specification varies, a predictor given only the state collapses to the average action, whereas one given T3DP's fine-grained language representation follows each specification's target action.

5.3 Real-world Transfer

We further evaluate whether the learned language-behavior correspondence transfers to a physical robot. We use a PiPer X robotic arm and construct the point cloud exclusively from a fixed D435i depth camera mounted above the workspace. We construct two task families, both require moving an object from its initial location to a designated target location, as illustrated in Figure 5. A trial is

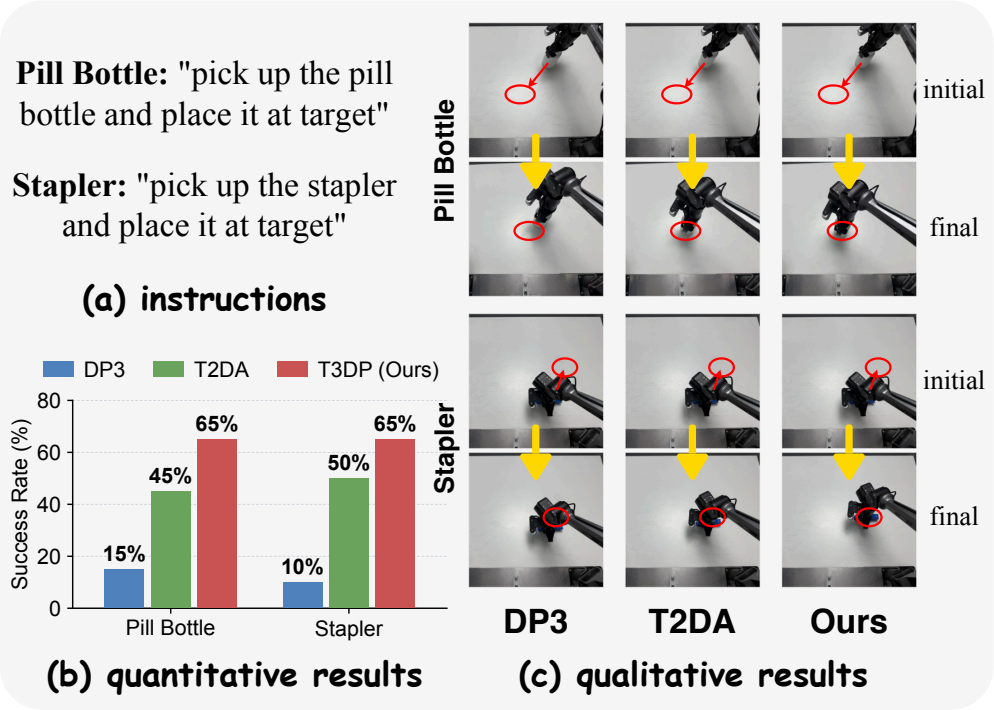

Figure 5: Real-world manipulation experiments on a PiPer X robotic arm. (a) Language instructions for two task families **Pill Bottle** and **Stapler**. (b) Quantitative success rates for DP3, T2DA, and T3DP. (c) Qualitative real-world control results showing the final object positions relative to the marked targets. T3DP achieves a higher success rate and higher final placement precision.

successful when the final object position is within 3 cm of the target. Each task family is evaluated on 20 distinct specifications. Additional details of the real-world setup are provided in Appendix E.

We compare DP3 without language conditioning, T2DA with global language-behavior alignment, and T3DP with the proposed fine-grained alignment under the same task and evaluation settings.

Quantitatively, global language–behavior alignment substantially improves the success rate of DP3 without language conditioning across both tasks. Adding fine-grained alignment on top of it yields a further clear gain, with T3DP achieving the highest success rates across both tasks. These results show that behavior grounding provides useful language guidance in the real-world setting, while fine-grained correspondence offers additional benefits beyond global alignment. **Qualitatively**, DP3 produces final object positions farthest from the target locations. Although T2DA brings the objects close to their targets, its placement remains less precise than that of T3DP.

Together, the quantitative and qualitative results indicate that fine-grained alignment improves both the reliability and the precision of language-conditioned manipulation beyond simulation.

6 Conclusion, Limitations and Future Work

We study unseen specification generalization in text-to-3D policies and introduce T3DP, which aligns instruction tokens with local behavior representations learned from demonstrations. This fine-grained alignment produces specification-sensitive conditions for a 3D diffusion policy without demonstrations at inference time. T3DP improves held-out specification execution across simulation benchmarks and real-robot tasks, and representation analyses show better preservation of specification structure and action variation. T3DP requires expert trajectories with reward signals, however, and our evaluation is limited to held-out specifications within known task families under templated instructions. Further limitations are discussed in Appendix F. Future work could learn specification-dependent behavior representations directly from state-action sequences and extend T3DP to unseen task families, free-form instructions, and diverse embodiments.

REPRODUCIBILITY STATEMENT

Section 4 describes the three-stage training procedure of T3DP and the demonstration-free inference setting. The appendix provides the remaining details required to reproduce our results, including task environments, language instruction templates, dataset construction and evaluation protocol, baseline implementations, training hyperparameters and computational resources, pseudocode for all three stages, and the real-robot setup (Appendices A–E). All reported numbers are averaged over three independently trained seeds, and every method is evaluated on identical held-out specifications, initial configurations, and success criteria. We will publicly release the implementation, training and evaluation configurations, and data-processing scripts upon acceptance.

AI USE STATEMENT

In this work, we used generative AI tools to edit and polish the language of this manuscript and to assist in preparing figures that present the experimental results. We have not used generative AI tools for any other research subtask that requires disclosure. We have reviewed all AI-assisted work: the AI-assisted edits were checked line by line against our original drafts by the authors, and every claim, number, figure, and artifact in this paper was produced and verified by the authors. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work involves no human subjects or personal data. Physical robot experiments were conducted in a controlled laboratory environment with appropriate safety precautions. Because learned robot policies may fail under untested conditions, they should be carefully evaluated and supervised before deployment. We adhere to the ICLR Code of Ethics.

REFERENCES

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, et al. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. In *Forty-third International Conference on Machine Learning*, 2026.

Ricardo Garcia, Shizhe Chen, and Cordelia Schmid. Towards generalizable vision-language robotic manipulation: A benchmark and LLM-guided 3D policy. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8996–9002, 2025.

Theophile Gervet, Zhou Xian, Nikolaos Gkanatsios, and Katerina Fragkiadaki. Act3D: 3D feature field transformers for multi-task robotic manipulation. In *Proceedings of the 7th Conference on Robot Learning*, volume 229, pp. 3949–3965, 2023.

Ran Gong, Jiangyong Huang, Yizhou Zhao, Haoran Geng, Xiaofeng Gao, Qingyang Wu, Wensi Ai, Ziheng Zhou, Demetri Terzopoulos, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. ARNOLD: A benchmark for language-grounded task learning with continuous states in realistic 3D scenes. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20426–20438, 2023.

Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. BC-Z: Zero-shot task generalization with robotic imitation learning. In *Proceedings of the 5th Conference on Robot Learning*, volume 164, pp. 991–1002, 2022.

Tsung-Wei Ke, Nikolaos Gkanatsios, and Katerina Fragkiadaki. 3D diffuser actor: Policy diffusion with 3D scene representations. In *Proceedings of The 8th Conference on Robot Learning*, volume 270, pp. 1949–1974, 2025.

Jianxiong Li, Jinliang Zheng, Yinan Zheng, Liyuan Mao, Xiao Hu, Sijie Cheng, Haoyi Niu, Jihao Liu, Yu Liu, Jingjing Liu, Ya-Qin Zhang, and Xianyuan Zhan. DecisionNCE: Embodied multimodal representations via implicit preference learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 29461–29488, 2024.

Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. LIV: Language-image representations and rewards for robotic control. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 23301–23320, 2023.

Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. ManiSkill: Generalizable manipulation skill benchmark with large-scale demonstrations. *arXiv preprint arXiv:2107.14483*, 2021.

Vivek Myers, Andre Wang He, Kuan Fang, Homer Rich Walke, Philippe Hansen-Estruch, Ching-An Cheng, Mihai Jalobeanu, Andrey Kolobov, Anca Dragan, and Sergey Levine. Goal representations for instruction following: A semi-supervised language interface to control. In *Proceedings of The 7th Conference on Robot Learning*, volume 229, pp. 3894–3908, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 8748–8763, 2021.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67, 2020.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. CLIPort: What and where pathways for robotic manipulation. In *Proceedings of the 5th Conference on Robot Learning*, volume 164, pp. 894–906, 2022.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Proceedings of the 6th Conference on Robot Learning*, volume 205, pp. 785–799, 2023.

Chunyu Xie, Bin Wang, Fanjing Kong, Jincheng Li, Dawei Liang, Gengshen Zhang, Dawei Leng, and Yuhui Yin. FG-CLIP: Fine-grained visual and textual alignment. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 68777–68793, 2025.

Lewei Yao, Runhui Huang, Lu Hou, Guansong Lu, Minzhe Niu, Hang Xu, Xiaodan Liang, Zhenguo Li, Xin Jiang, and Chunjing Xu. FILIP: Fine-grained interactive language-image pre-training. In *International Conference on Learning Representations*, 2022.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Avnish Narayan, Hayden Shively, Adithya Bellathur, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-World: A benchmark and evaluation for multi-task and meta reinforcement learning. *arXiv preprint arXiv:1910.10897*, 2021.

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3D diffusion policy: Generalizable visuomotor policy learning via simple 3D representations. *arXiv preprint arXiv:2403.03954*, 2024.

Shilin Zhang, Zican Hu, Wenhao Wu, Xinyi Xie, Jianxiang Tang, Chunlin Chen, Daoyi Dong, Yu Cheng, Zhenhong Sun, and Zhi Wang. Text-to-decision agent: Offline meta-reinforcement learning from natural language supervision. In *Advances in Neural Information Processing Systems*, volume 38, pp. 79073–79104, 2025.

Kaizhi Zheng, Xiaotong Chen, Odest Chadwicke Jenkins, and Xin Wang. VLMbench: A compositional benchmark for vision-and-language manipulation. In *Advances in Neural Information Processing Systems*, volume 35, pp. 665–678, 2022.

Appendix

A Task, Language Specifications, and Dataset Splits **13**
A.1 Dataset Splits . . . 13
A.2 Language Instructions and Task Specifications . . . 13
A.3 Dataset Construction and Sequence Sampling . . . 13
A.4 Evaluation Protocol . . . 14
A.5 Baseline Implementations and Fair Comparison . . . 15

B Implementation Details **16**
B.1 Behavior Encoder . . . 16
B.2 Text Encoder and Language–Behavior Alignment . . . 16
B.3 Language-Conditioned 3D Policy . . . 16
B.4 Computational Resources . . . 17

C Training and Inference Algorithms **18**
C.1 Behavior Representation Learning . . . 18
C.2 Global and Local Language–Behavior Alignment . . . 18
C.3 Language-Conditioned Policy Learning . . . 18
C.4 Inference without Demonstrations . . . 18

D Additional Experimental Results **20**
D.1 Effect of Specification Coverage . . . 20
D.2 Mixed-Task Training . . . 20
D.3 Additional RoboTwin Results . . . 21
D.4 Alignment Representation Analysis . . . 21

E Real-Robot Experimental Details **23**
E.1 Robot and Sensor Setup . . . 23
E.2 Point-Cloud Construction . . . 23
E.3 Tasks and Target Specifications . . . 23
E.4 Real-Robot Demonstration Collection . . . 23
E.5 Execution and Success Criteria . . . 24
E.6 Qualitative Results and Failure Cases . . . 24

F Limitations **24**

A TASK, LANGUAGE SPECIFICATIONS, AND DATASET SPLITS

- **Meta-World.** We evaluate *Push*, *Pick Place*, *Reach*, *Door Open*, and *Drawer Open*. Each observation consists of 512 XYZ points and a 9-dimensional proprioceptive state, and the policy predicts a 4-dimensional continuous action. The numerical task specification is excluded from the observation and provided only through language. Reach parameterizes the gripper target location; Push parameterizes a relative planar displacement; Pick Place parameterizes the object target location; and Door Open and Drawer Open parameterize normalized articulation progress over a restricted feasible range. Each rollout is capped at 200 environment steps.
- **RoboTwin.** We evaluate *Open Microwave*, *Open Cabinet Drawer*, *Move Pill Bottle*, *Move Playing Card*, and *Move Stapler*. The policy receives 512 XYZ points and a 14-dimensional robot joint state and predicts 14-dimensional joint-position actions. The articulated tasks parameterize target opening progress, while the remaining tasks parameterize planar placement locations sampled over their feasible workspaces. Episode horizons are 550 steps for *Open Microwave*, 410 for *Open Cabinet Drawer*, and 200 for the placement tasks. To control initial-state variation, each task uses 20 reproducibly sampled initial object configurations around its nominal pose.
- **ManiSkill.** We evaluate *Reach*, *Push Cube*, *Pull Cube*, *Pick Cube*, and *Turn Valve*. Each observation contains 512 XYZ points and a 32-dimensional proprioceptive state, while the numerical task specification is provided only through language. The policy predicts a 7-dimensional end-effector delta-pose action. The five domains parameterize, respectively, a Cartesian target, a forward displacement, a planar pulling target, a lifting extent, and a rotation extent. In *Pull Cube*, the expert grasps the cube, moves it to the specified planar target, and releases it.

Unless otherwise specified, simulation benchmark success rates are averaged over three independently trained seeds. Real-robot results use 20 held-out trials per task and method, while action-probe results are averaged over four trajectory splits.

A.1 DATASET SPLITS

All 15 task domains appear in both training and evaluation. Generalization is therefore measured over *held-out task specifications*, rather than unseen task categories. For each domain, we use 20 training specifications and 20 held-out test specifications, with one successful demonstration associated with each training specification.

Training and test specifications are disjoint within each task and are sampled to provide approximately uniform coverage of the corresponding feasible specification space. Test specifications are not used during representation alignment or policy training. Any optimization-time validation split is constructed only from the training demonstrations and is excluded from the reported held-out evaluation.

A.2 LANGUAGE INSTRUCTIONS AND TASK SPECIFICATIONS

We represent each task instance by a specification

$$\sigma = (f_1, f_2, \ldots, f_d), \tag{9}$$

where each factor corresponds to a behaviorally relevant property of the desired outcome, such as a target location, displacement, placement goal, or articulation extent. The specification is expressed through a task-specific natural language template whose values are instantiated from σ.

For each task, training and test instructions follow the same linguistic construction, while their underlying specifications are disjoint. Our evaluation therefore focuses on *generalization to unseen behavioral specifications*, rather than generalization to paraphrased language. Table 3 summarizes the task specifications and instruction templates used in each benchmark.

A.3 DATASET CONSTRUCTION AND SEQUENCE SAMPLING

Expert demonstrations. Training data consist exclusively of successful expert trajectories collected using the corresponding benchmark-specific controllers. Meta-World uses its parameter-conditioned expert controller, while RoboTwin and ManiSkill use their respective expert or motion-planning pipelines. Only successful trajectories are retained.

Table 3: Task specifications and language instruction templates across the evaluated benchmarks.

Meta-World		
Task	**Specification**	**Instruction Template**
Push	Relative displacement	"Push the object [lateral displacement] and [forward displacement]."
Pick Place	Placement target location	"Pick up the object and place it at [target location]."
Reach	End-effector target location	"Move the gripper to [target location]."
Door Open	Target opening extent	"Open the door to [opening extent]."
Drawer Open	Target opening extent	"Open the drawer to [opening extent]."

RoboTwin		
Task	**Specification**	**Instruction Template**
Open Microwave	Target opening extent	"Open the microwave door to [opening extent]."
Open Cabinet Drawer	Target opening extent	"Open the cabinet drawer to [opening extent]."
Move Pill Bottle	Placement target location	"Pick up the pill bottle and place it at [target location]."
Move Playing Card	Placement target location	"Pick up the playing card and place it at [target location]."
Move Stapler	Placement target location	"Pick up the stapler and place it at [target location]."

ManiSkill		
Task	**Specification**	**Instruction Template**
Reach	End-effector target location	"Move the gripper to [target location]."
Push Cube	Target displacement	"Push the cube by [target displacement]."
Pull Cube	Object target location	"Grasp the cube and pull it to [target location]."
Pick Cube	Target lifting extent	"Lift the cube by [lifting extent]."
Turn Valve	Target rotation extent	"Turn the valve by [rotation extent]."

Data processing. Each trajectory stores the point-cloud observation, proprioceptive state, action, reward, episode boundary, and task-specification metadata. Point clouds are converted to 512 XYZ points per frame, and normalization statistics are computed exclusively from the training demonstrations.

Sequence sampling. The same temporal sampling protocol is used across all three benchmarks. We condition on two observation steps and use a fixed action prediction horizon during policy training. Episode boundaries are handled using standard boundary padding. The same sampling and preprocessing protocol is used for all compared methods.

A.4 Evaluation Protocol

All methods are evaluated on exactly the same held-out specifications, initial configurations, and closed-loop rollout protocol. At each replanning step, the policy conditions on the latest two observations, predicts an action sequence over the configured horizon, executes the next eight actions, and then replans. The primary evaluation metric is success rate.

Because our setting evaluates fine-grained specification following rather than coarse task completion, we adopt stricter success thresholds than the native benchmark criteria. For example, in Meta-World, Reach requires the final end-effector position to be within $2.5\,\mathrm{cm}$ of the specified target, while Push requires the planar displacement error to be below $1\,\mathrm{cm}$. Articulated tasks are similarly evaluated using tight errors on the specified opening extent. All task-specific thresholds are fixed before evaluation and applied identically to all compared methods.

Unless otherwise specified, results are aggregated over three random seeds. Additional quantities such as final task error, return, and episode length are retained as diagnostic metrics where supported by the corresponding environment. Representation-level metrics, including Spearman correlation, kNN@3, and action-probe MAE, are reported separately in Section D.4.

A.5 Baseline Implementations and Fair Comparison

We compare against four baselines that span three levels of language use: i) DP3, a 3D diffusion policy without any language conditioning; ii) language-conditioned 3D policies that inject language directly into the policy, *3DDA* and *3D-LOTUS*; iii) and T2DA, which grounds language through global behavior–instruction alignment. All baselines are trained and evaluated on the same demonstrations, point-cloud observations, proprioceptive inputs, task instructions, and train–test specification splits, with identical held-out specifications, initial configurations, and success criteria as T3DP. We keep each baseline's model architecture, training objective, and inference procedure unchanged, and adapt only the input and action interfaces (observation format, end-effector pose targets, and waypoint-to-action conversion) needed to place the method in our protocol. Training budgets, training epochs, and batch sizes are aligned across methods.

- **DP3 (Ze et al., 2024).** DP3 is a point-cloud diffusion policy: a lightweight MLP encoder embeds the point cloud and the proprioceptive state, and a conditional U-Net denoises a future action sequence. It receives no language input, so specification-dependent targets are unobservable to the policy. We use the official DP3 code directly, retaining its encoder and diffusion configuration, and train it per task on the same demonstrations as every other method, with the same prediction horizon, observation history, executed action segment, and optimization budget as our policy.
- **3DDA (Ke et al., 2025).** 3D Diffuser Actor builds on a 3D feature-field transformer that lifts visual features into a 3D scene representation and applies relative-position attention over sampled action queries, and replaces single-step keypose selection with a diffusion process over end-effector pose trajectories. Its vision–language–trajectory attention conditions the denoising network on task instructions, but supervision is still a global instruction–trajectory pair. We adapt the official implementation only at the interface level: point clouds are mapped to the image grid expected by its 2D backbone, proprioception is converted into end-effector pose targets, the gripper command provides the openness channel, and the official task instruction is pre-encoded offline by its CLIP text tower. Its DDPM objective, position/rotation losses, and attention structure are unchanged. Following its native inference convention, the predicted pose segment is executed before replanning, and poses are converted into each benchmark's action space with gains calibrated from expert data.
- **3D-LOTUS (Garcia et al., 2025).** 3D-LOTUS is a point-cloud transformer policy (PTV3) that injects language through cross-attention between point features and word embeddings, and predicts actions by classification with per-axis position heatmaps, discretized rotation bins, and a binary gripper state. We use 3D-LOTUS rather than 3D-LOTUS++, since the LLM task-planning and VLM grounding stages of the latter are not applicable to our setting, where each instruction already denotes a single grounded behavioral specification. As with 3DDA, we adapt only the interfaces: XYZ-only point clouds without fabricated RGB channels, end-effector pose targets in a unified position–quaternion–openness format, and pose-to-increment conversion calibrated from expert data at evaluation time. The PTV3 backbone, the cross-attention language injection, and the classification heads are unchanged.
- **T2DA (Zhang et al., 2025).** T2DA learns a decision embedding of trajectories and aligns it with natural language through a global, pooled bidirectional contrastive objective with a learnable temperature, then conditions the downstream policy on the aligned text embedding. We reproduce its global alignment stage and combine it with the same DP3-based diffusion policy used by T3DP. Consequently, T2DA and T3DP share the behavior encoder and its frozen checkpoint, the text-encoder initialization, the alignment data, the downstream policy architecture, and the optimization budget, differing only in the alignment objective: T2DA optimizes the pooled global contrastive loss, whereas T3DP adds the proposed fine-grained local objective. This comparison isolates the effect of fine-grained

language–behavior alignment from changes in policy capacity, training data, or optimization.

B Implementation Details

B.1 Behavior Encoder

Stage I: behavior representation learning. The behavior encoder processes a trajectory window containing point-cloud and proprioceptive states, actions, and rewards. Point clouds are encoded by a PointNet encoder, while the remaining modalities are projected by modality-specific multilayer perceptrons. The state, action, and reward tokens at each timestep share a temporal positional encoding and are processed by a bidirectional Transformer. Mean pooling over the Transformer outputs produces the global behavior representation. The Transformer has two layers, four attention heads, a hidden dimension of 256, an output dimension of 256, and a dropout rate of 0.1.

A lightweight reward decoder is trained jointly with the behavior encoder. It contains separate projections for the action and observation features followed by a multilayer perceptron with two hidden layers of dimension 256. The decoder predicts the reward of a state–action pair drawn from the same task specification but outside the encoded trajectory window. This objective encourages the behavior representation to capture specification-dependent behavior rather than only surface motion.

The reward signal is used only when learning the behavior representation. Neither reward nor a demonstration is required by the deployed policy.

B.2 Text Encoder and Language–Behavior Alignment

Stage II: fine-grained Language–Behavior alignment. The language encoder is initialized from the CLIP ViT-B/32 text encoder. It produces token-level representations and a pooled instruction representation. The frozen behavior encoder provides a global behavior representation for global alignment and local behavior tokens for fine-grained alignment. Global alignment applies a bidirectional contrastive objective to the global behavior and instruction representations. Local alignment applies bidirectional late interaction between the local behavior tokens and instruction tokens. The final alignment objective combines the global and local losses.

During alignment, the behavior encoder remains frozen and the language encoder is adapted using LoRA on its query, key, and value projections. The LoRA rank is 16, its scaling parameter is 32, and its dropout rate is 0.1. Instructions are tokenized to a maximum length of 77. The local alignment dimension is 256. The initial global contrastive temperature is 0.02 and is learnable. The local matching temperature is 0.05, the local contrastive temperature is 0.07, and the local alignment loss weight is 0.25. The two alignment directions are weighted equally.

We use the terms *trajectory* for a complete demonstration, *behavior window* for the fixed-length encoder input, and *local alignment unit* for the state–action hidden states retained for token-level matching. The pooled behavior and language representations are the corresponding global embeddings.

B.3 Language-Conditioned 3D Policy

Stage III: language-guided 3D policy learning. The aligned language encoder is frozen during policy learning. The downstream point-cloud-based 3D diffusion policy encodes the current point-cloud observation and proprioceptive state and uses the pooled aligned language representation as the policy condition. The denoising network predicts the diffusion noise of a noisy action sequence, conditioned on the observation history, diffusion timestep, and aligned language representation. One policy is trained across all training specifications within each task family. Specification-dependent targets are not rendered in the point-cloud observation and must therefore be communicated through language.

Table 4: Principal training hyperparameters across the three benchmarks.

Stage	Configuration	Meta-World	RoboTwin	ManiSkill
I	Training epochs	500	500	500
	Batch size	128	128	128
II	Training epochs	375	375	375
	Batch size	128	128	128
III	Training epochs	300	600	600
	Batch size	512	128	512
	Prediction horizon	16	16	16
	Observation steps	2	2	2
	Executed action steps	8	8	8
Policy	Training timesteps	100	100	100
	Inference steps	10	10	10
	Checkpoint interval	25	50	50
	Optimizer	AdamW	AdamW	AdamW
	Learning rate	10^{-4}	10^{-4}	10^{-4}

B.4 Computational Resources

Table 4 lists the principal training hyperparameters used for Meta-World, RoboTwin, and ManiSkill. Batch sizes are reported per GPU.

All stages use AdamW with weight decay 10^{-6}. The policy uses a DDIM noise scheduler. Point-cloud observations contain 512 XYZ points and do not include RGB features.

We train our models on one NVIDIA GeForce RTX 3090 GPU with dual AMD EPYC 9654 CPUs and 512 GB RAM. Behavior-encoder pre-training, fine-grained language–behavior alignment, and language-guided policy training each cost about 1 hour, with slight variations depending on the complexity of the environment and data-loading overhead. Consequently, the complete three-stage training pipeline takes approximately 3 hours.

C Training and Inference Algorithms

T3DP is trained in three stages. The following pseudocodes make explicit which components are optimized in each stage and clarify that deployment uses only the aligned text encoder and the policy.

C.1 Behavior Representation Learning

Let a length-L trajectory window be $\tau_w = \{(s_t, a_t, r_t)\}_{t=w}^{w+L-1}$, where s_t contains the point-cloud observation and proprioceptive state. The behavior encoder tokenizes the state, action, and reward at every timestep, processes the resulting $3L$ tokens with a bidirectional Transformer, and mean-pools its outputs.

C.2 Global and Local Language–Behavior Alignment

For an instruction l, the text encoder produces token representations $E(l) = \{e_j\}_{j=1}^{M}$ and the pooled representation $c_l = M^{-1} \sum_{j=1}^{M} e_j$. For its paired trajectory τ, let $H_\varphi(\tau)$ denote the hidden states of the frozen behavior Transformer. The global behavior representation is $c_\tau = \mathrm{MeanPool}(H_\varphi(\tau))$, while $Z(\tau) = \{z_i\}_{i=1}^{N}$ retains its state and action hidden states for local matching. With $s_{ij} = \cos(z_i, e_j)$, define

$$S_{\tau \to l}(Z, E) = \frac{1}{N} \sum_{i=1}^{N} \mathrm{LME}_\beta\big(\{s_{ij}\}_{j=1}^{M}\big), \tag{10}$$

$$S_{l \to \tau}(Z, E) = \frac{1}{M} \sum_{j=1}^{M} \mathrm{LME}_\beta\big(\{s_{ij}\}_{i=1}^{N}\big), \tag{11}$$

where $\mathrm{LME}_\beta(\{x_k\}_{k=1}^{K}) = \beta^{-1} \log\Big(K^{-1} \sum_{k=1}^{K} \exp(\beta x_k)\Big)$. Both the global similarities and the two local scores are optimized with bidirectional contrastive objectives over paired trajectories and instructions.

C.3 Language-Conditioned Policy Learning

The aligned text encoder is frozen during policy learning. Following the method definition in the main paper, the downstream policy consumes only the pooled aligned representation c_l; the local objective shapes this pooled condition but does not pass token-level features directly to the policy.

C.4 Inference without Demonstrations

At deployment, the behavior encoder, rewards, and demonstrations are not used, and the aligned text encoder is not updated. The policy receives only the current point cloud, robot state, and language instruction; it samples a horizon-length action sequence, executes the configured action segment, and then replans from the next observation. Consequently, a held-out specification does not require a corresponding training demonstration at test time.

Algorithm 1: Point-cloud behavior representation learning

Input: Expert demonstration sets $\{\mathcal{D}_\sigma\}_\sigma$; behavior encoder e_φ; reward decoder d_ψ
Output: Trained behavior encoder e_φ

foreach *training epoch* **do**
  foreach *specification* σ*, trajectory* $\tau \in \mathcal{D}_\sigma$*, and sampled window* $\tau_w \subset \tau$ **do**
    Encode point cloud and proprioception as state tokens, and embed the corresponding actions and rewards
    Add temporal encodings and form the interleaved state–action–reward token sequence X_w
    $c_\tau \leftarrow e_\varphi(\tau_w)$ by processing X_w with the bidirectional Transformer and mean pooling
    Sample $(s, a, r) \in \tau \setminus \tau_w$
    $\hat{r} \leftarrow d_\psi(c_\tau, s, a)$
    Update (φ, ψ) using $\widehat{\mathcal{L}}_{\text{behavior}} = (\hat{r} - r)^2$
  Evaluate reward prediction on the validation split
Retain the selected behavior-encoder checkpoint

Algorithm 2: Global–local Language–Behavior alignment

Input: Frozen behavior encoder e_φ; paired trajectory–instruction minibatches $\{(\tau_b, l_b)\}_{b=1}^B$; text encoder g_ϕ with LoRA; text-side projections; local-loss weight λ
Output: Aligned text encoder g_ϕ and text-side projections

foreach *paired minibatch* $\{(\tau_b, l_b)\}_{b=1}^B$ **do**
  $\{(c_{\tau_b}, Z_b)\}_{b=1}^B \leftarrow e_\varphi(\{\tau_b\}_{b=1}^B)$
  Retain global behavior embeddings c_{τ_b} and local state–action representations Z_b
  $\{(c_{l_b}, E_b)\}_{b=1}^B \leftarrow g_\phi(\{l_b\}_{b=1}^B)$
  Retain pooled instruction embeddings c_{l_b} and token representations E_b
  Compute $\mathcal{L}_{\text{global}}$ from global language–behavior similarities
  Compute $\mathcal{L}_{\text{local}}$ from $S_{\tau\to l}(Z_b, E_q)$ and $S_{l\to\tau}(Z_q, E_b)$ using Eq. (11)
  $\mathcal{L}_{\text{align}} \leftarrow \mathcal{L}_{\text{global}} + \lambda\mathcal{L}_{\text{local}}$
  Update the LoRA parameters of g_ϕ and the text-side projections using $\mathcal{L}_{\text{align}}$

Algorithm 3: Language-conditioned diffusion policy learning and inference

Input: Policy training tuples $\mathcal{D}_\pi = \{(o_t, p_t, l, \mathbf{A}^0)\}$; frozen aligned text encoder g_ϕ; observation encoder F_η; denoiser ϵ_ω; DDIM scheduler
Output: Trained policy parameters (η, ω)

Training
foreach *minibatch* $(o_t, p_t, l, \mathbf{A}^0) \sim \mathcal{D}_\pi$ **do**
  $c_l \leftarrow \text{MeanPool}(g_\phi(l))$
  $h_t \leftarrow F_\eta(o_t, p_t)$
  Sample diffusion step k and $\epsilon \sim \mathcal{N}(0, I)$
  Construct $\mathbf{A}^k$ using the forward diffusion scheduler
  $\hat{\epsilon} \leftarrow \epsilon_\omega(\mathbf{A}^k, k, h_t, c_l)$
  Update (η, ω) using $\mathcal{L}_{\text{policy}} = \|\epsilon - \hat{\epsilon}\|_2^2$
Receding-horizon inference
Given the current (o_t, p_t, l), compute c_l and h_t as above
Initialize $\mathbf{A}^K \sim \mathcal{N}(0, I)$
for $k = K, K-1, \ldots, 1$ **do**
  $\hat{\epsilon} \leftarrow \epsilon_\omega(\mathbf{A}^k, k, h_t, c_l)$
  Obtain $\mathbf{A}^{k-1}$ with one DDIM reverse step
Unnormalize $\mathbf{A}^0$
Execute the configured action segment and replan from the next observation

Table 5: Effect of training specification coverage on unseen-specification generalization in Meta-World. Each training specification contains one expert demonstration, and success rates (%) are evaluated on a common held-out specification set.

Method	# Train Specs.	Reach	Push	Pick Place	Door Open	Drawer Open	Avg.
Global Alignment	10	22.3	25.3	26.7	28.7	16.7	23.9
	20	24.3	45.7	46.0	41.3	23.3	36.1
	40	36.1	63.3	75.0	100.0	71.7	69.2
Ours	10	30.3	37.0	34.0	35.3	25.3	32.4
	20	38.3	64.7	58.0	59.7	30.7	50.3
	40	47.2	75.0	76.7	100.0	83.3	76.4

D Additional Experimental Results

D.1 Effect of Specification Coverage

We further evaluate whether the advantage of fine-grained language–behavior alignment persists across different levels of specification coverage. For each task, we vary the number of training specifications among 10, 20, and 40 while keeping the held-out test specification set and evaluation protocol fixed. Each training specification is associated with exactly one expert demonstration, so increasing the training set expands coverage of the specification space rather than adding repeated demonstrations for previously observed specifications. At each scale, global language–behavior alignment and our fine-grained alignment model are independently trained from scratch using exactly the same training specifications and demonstrations.

Across all three specification scales, fine-grained alignment consistently outperforms global alignment. With 10, 20, and 40 training specifications per task, T3DP improves average success from 23.9%, 36.1%, and 69.2% to 32.4%, 50.3%, and 76.4%, respectively. The advantage therefore persists from sparse to denser specification coverage, suggesting that fine-grained alignment remains beneficial across different levels of demonstration availability.

D.2 Mixed-Task Training

Experimental setting. The main benchmark protocol trains one policy independently for each task family. We therefore consider a more heterogeneous setting in which a single policy is jointly trained across all five Meta-World task families, covering spatial targets, relative displacements, object placement, and articulated-state specifications. The observation space, demonstrations, and train–test specification splits remain unchanged; only demonstrations from different task families are combined into a shared training set. We compare global language-behavior alignment (T2DA) with our fine-grained alignment, and evaluate exclusively on held-out specifications within each task family.

Result analysis. Under joint training, T3DP obtains the best average success of 40.7%, compared with 35.5% for T2DA and 11.3% for DP3. Thus, fine-grained alignment improves over global alignment by 5.2 percentage points, while both language-conditioned methods substantially outperform the no-language baseline. T3DP is best on Push (56.7%), Door Open (47.3%), and Drawer Open (21.3%), whereas T2DA is slightly better on Pick Place (53.0% versus 42.7%) and Reach (36.3% versus 35.3%). The overall advantage therefore persists despite task-specific reversals, indicating that the proposed local alignment is beneficial when one policy must model heterogeneous task semantics rather than only when policies are trained separately.

Conclusion. Joint multi-task training confirms that fine-grained alignment extends beyond the single-task-family protocol: it provides the strongest overall specification generalization while sharing one policy across heterogeneous manipulation behaviors. The remaining per-task variation also suggests that the advantage of local alignment is complementary to, rather than a replacement for, task-specific modeling capacity.

Table 6: Joint multi-task training on Meta-World. A single policy is trained jointly across all five task families and evaluated on held-out specifications within each family. Success rates (%) are averaged over three seeds.

Method	Push	Pick Place	Reach	Door Open	Drawer Open	Avg.
DP3	14.3	12.0	1.0	10.3	18.7	11.3
T2DA	42.7	**53.0**	**36.3**	31.0	14.3	35.5
Ours	**56.7**	42.7	35.3	**47.3**	**21.3**	**40.7**

D.3 Additional RoboTwin Results

Experimental setting. We further evaluate unseen-specification generalization on five RoboTwin task families: *Open Microwave*, *Open Cabinet Drawer*, *Move Pill Bottle*, *Move Playing Card*, and *Move Stapler*. These tasks cover both articulated-object manipulation and spatial object placement. We compare DP3 without language conditioning, direct conditioning with pretrained CLIP representations (Raw CLIP), T2DA with global language–behavior alignment, and T3DP with the proposed fine-grained alignment. All methods use the same demonstrations, observation inputs, language instructions, train–test specification splits, and evaluation protocol. We evaluate exclusively on held-out specifications following the protocol described in Appendix A.

Table 7: Unseen-specification generalization on RoboTwin. We report success rates (%) and corresponding standard deviations over three independently trained seeds. "Avg." denotes the mean across the five task families. Best and second-best results are shown in bold and underlined, respectively.

Method	Open Microwave	Open Cabinet Drawer	Move Pill Bottle	Move Playing Card	Move Stapler	Avg.
DP3	15.7±8.6	5.0±1.3	3.7±1.3	9.0±2.2	2.7±2.4	7.2
Raw CLIP	37.7±9.5	12.3±10.9	19.3±3.8	28.7±3.4	22.0±2.8	24.0
T2DA	<u>46.0±4.5</u>	<u>16.7±16.3</u>	<u>34.3±1.3</u>	<u>32.7±2.5</u>	<u>24.3±1.7</u>	<u>30.8</u>
T3DP (Ours)	**52.0±13.5**	**37.3±0.6**	**39.3±0.5**	**52.3±1.5**	**28.0±1.6**	**41.8**

Result analysis. T3DP achieves an average success rate of 41.8%, compared with 30.8% for T2DA, 24.0% for Raw CLIP, and 7.2% for DP3. Fine-grained alignment therefore improves over the global-alignment baseline by 11.0 percentage points. Importantly, T3DP improves over T2DA on all five RoboTwin task families, including both articulated-object manipulation and object-placement tasks. The gains are particularly pronounced on *Open Cabinet Drawer* and *Move Playing Card*, where T3DP improves success from 16.7% to 37.3% and from 32.7% to 52.3%, respectively.

The consistent improvement over Raw CLIP and T2DA provides additional evidence that behavior grounding is beneficial beyond direct language conditioning, while fine-grained language–behavior correspondence further improves generalization beyond global alignment. Together with the Meta-World and ManiSkill results, these results show that the advantage of T3DP transfers across different manipulation environments, embodiments, and action spaces.

D.4 Alignment Representation Analysis

Beyond downstream success, we examine what the learned alignment preserves and how this structure relates to control. We focus on two complementary views: the geometry of the alignment space and the action relevance of the learned language condition.

Representation geometry. We evaluate whether the learned language representations preserve the underlying specification structure on Reach, Push, and Door Open, covering spatial targets, relative displacements, and articulated states. We report Spearman correlation between pairwise specification and representation distances, together with kNN@3 neighborhood preservation.

Table 8: Representation geometry on held-out Meta-World specifications.

Task	Metric	Generic	Global	Fine-grained
Reach	Spearman ↑	0.504	0.567	**0.612**
	kNN@3 ↑	0.417	0.450	**0.533**
Push	Spearman ↑	0.255	0.524	**0.708**
	kNN@3 ↑	0.283	0.317	**0.567**
Door Open	Spearman ↑	0.360	0.682	**0.841**
	kNN@3 ↑	0.400	0.433	**0.622**

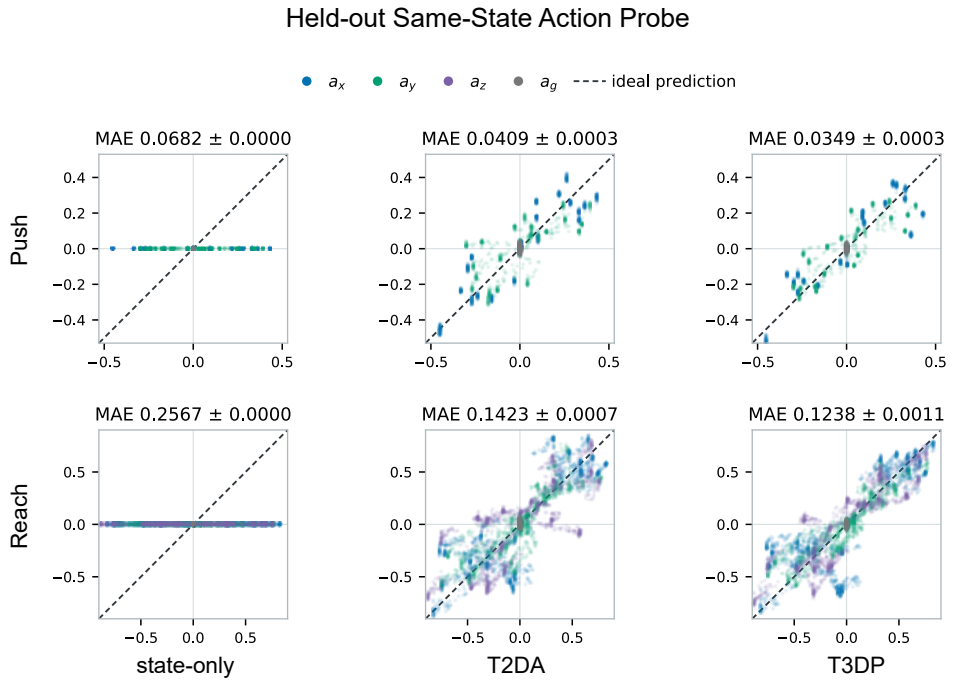

Figure 6: Full same-state action probe results on held-out Push and Reach specifications. Each point compares the predicted and target action under a fixed state while varying the language specification; the dashed diagonal denotes ideal prediction. State-only predictions collapse toward the mean action, whereas behavior–language alignment recovers specification-dependent action variation, with T3DP more closely following the target actions than global alignment (T2DA).

Global alignment improves specification structure over generic language features, while fine-grained alignment further improves both global ordering and local neighborhood preservation across all three task types. This supports our hypothesis that finer alignment better preserves specification-dependent behavioral variation.

Action relevance of the learned language condition. Representation geometry alone does not establish whether the learned structure is informative for control. We therefore construct a *same-state action probe*, where the robot and scene state are fixed while the language condition varies across held-out specifications. This removes state-dependent shortcuts, such that differences in the target actions must be explained by specification-dependent information encoded in the language representation. Probe training and evaluation use disjoint source trajectories, and results are averaged over four trajectory splits.

As shown in Figure 6, conditioning on an aligned language representation substantially reduces action error relative to the state-only probe, indicating that behavior–language alignment captures

information about how different specifications require different actions. Fine-grained alignment further reduces MAE over global alignment by 14.7% on Push and 13.0% on Reach. The consistent trend across relative-displacement and spatial-target specifications provides complementary evidence that T3DP better preserves specification-dependent action information in the condition consumed by the downstream policy. Given the limited number of held-out specifications, we treat this probe as diagnostic evidence rather than a standalone statistical claim.

E Real-Robot Experimental Details

E.1 Robot and Sensor Setup

We conduct real-robot experiments using a PiPer X manipulation platform. Although the platform supports dual-arm operation, all experiments reported here use a single PiPer X manipulator. An Intel RealSense D435i depth camera is mounted above the workspace and remains fixed throughout data collection and evaluation. Depth observations are transformed into the robot coordinate frame using a fixed extrinsic calibration.

The policy receives geometric point-cloud observations and robot proprioceptive states; RGB information is not used. We retain the same observation modality and end-effector action parameterization as in simulation. Specification-dependent target information is not explicitly encoded in the visual observation and is instead communicated to the policy through language.



Figure 7: Real-robot setup with a PiPer X manipulator and D435i depth camera.

E.2 Point-Cloud Construction

Depth images are back-projected to 3D and transformed into the robot coordinate frame. Points outside the task workspace are removed, after which the remaining cloud is subsampled to 512 XYZ points using the same input format as in simulation. No RGB features are provided to the policy. The same normalization convention is used for simulation and real-robot experiments.

E.3 Tasks and Target Specifications

We evaluate two placement tasks, *Move Pill Bottle* and *Move Stapler*. In both tasks, the specification is defined by a target placement location in the tabletop plane and communicated to the policy through a fixed language template.

For each task, we use 20 training target specifications and 20 disjoint held-out target specifications. Figure 8 visualizes their spatial distributions. The held-out targets cover the same feasible placement region as the training targets while remaining unseen during training, allowing us to evaluate generalization to unseen target specifications without introducing an additional shift in workspace coverage.

E.4 Real-Robot Demonstration Collection

Real-robot demonstrations are collected through a simulation-assisted pipeline. For each task specification, we first instantiate the corresponding task in a modified RoboTwin environment and use the simulator expert to generate a successful reference trajectory. The resulting trajectory is then replayed on the real PiPer X robot, during which real depth observations, robot proprioception, and executed actions are recorded to construct the training demonstration.

The simulator is used only to generate the expert motion sequence; policy training uses the observations collected during real-robot execution. Only successful real-world replays are retained in the training dataset.

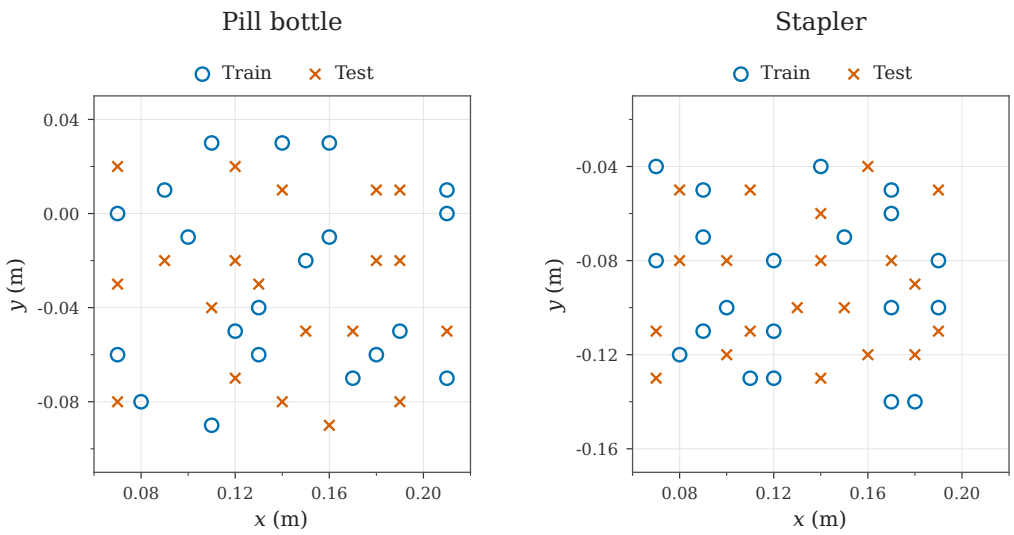

Figure 8: Spatial distribution of the target specifications used in the real-robot experiments. Left: *Move Pill Bottle*. Right: *Move Stapler*. Blue markers denote training specifications, orange markers denote held-out specifications. Each task uses 20 training and 20 held-out target specifications.

E.5 Execution and Success Criteria

We compare DP3 without language conditioning, T2DA with global language–behavior alignment, and T3DP under the same task and evaluation settings. All methods are evaluated on the same 20 held-out target specifications for each task. Each held-out specification is executed once, resulting in 20 real-robot trials per task and method.

A trial is considered successful when the final object position is within $3\,\mathrm{cm}$ of the specified target in the tabletop plane. Target locations are used only for evaluation and are not explicitly encoded in the policy observation. Quantitative success rates are reported in the main paper.

E.6 Qualitative Results and Failure Cases

DP3 produces final object positions farthest from the targets. T2DA typically brings objects closer, whereas T3DP yields the strongest placement precision and success rate across the two task families. The qualitative examples in Figure 5 illustrate the remaining failures as grasping or placement errors, including cases that approach but do not meet the 3 cm success threshold.

F Limitations

Our behavior encoder is trained with a reward-predictive objective and therefore assumes access to expert trajectories annotated with rewards, which are often unavailable or costly to define in real-world datasets, and many existing robot datasets provide only state–action sequences without a reliable reward signal. A plausible remedy is to learn specification-dependent behavior representations directly from state–action sequences through self-supervised objectives, which we leave to future work. Our evaluation focuses on held-out specifications within the task families seen during training, and our instructions are instantiated from fixed task-specific templates. We additionally study mixed-task training in Appendix D.2, where a single policy is trained jointly across five Meta-World task families, but that study is limited in scale and confined to one benchmark. Scaling joint training to a larger and more heterogeneous collection of tasks, and extending T3DP to unseen task families, longer-horizon behaviors, and more diverse embodiments, therefore remains an important direction for future work.