APT: Action Expert Pretraining Improves Instruction Generalization of Vision-Language-Action Policies

Kechun Xu[1], **Zhenjie Zhu**[1], **Anzhe Chen**[1], **Rong Xiong**[1,2], **Yue Wang**[1]
[1]Zhejiang University, [2]Zhejiang Humanoid Robot Innovation Center

https://xukechun.github.io/APT

Abstract: Vision-Language-Action (VLA) models that couple pretrained Vision-Language Models (VLMs) with continuous action experts have achieved strong manipulation performance, yet generalization to out-of-distribution (OOD) language instructions remains poor. A known challenge is the structural imbalance in VLA data, where language is far less diverse than visual and action content, making policies prone to visual shortcuts. While discrete-action methods mitigate this through vision-language co-training, continuous action experts lack such protection: they start from random initialization and learn entirely from imbalanced data, producing noisy gradients that corrupt the VLM and fail to exploit its language capability. We address this from a Bayesian perspective, factorizing the policy into a language-agnostic Vision-Action (VA) prior and a language-conditioned VLA likelihood, and propose **APT**, a two-stage training method emphasizing **A**ction expert **P**re**T**raining. In Stage 1, the action expert is pretrained as a VA prior on vision-action pairs from a frozen VLM, bypassing the language imbalance. In Stage 2, language tokens are injected through a gated fusion mechanism that integrates VLM features while preserving the learned visuomotor prior. APT applies to mainstream VLA architectures, including the π and GR00T-style architectures. Comprehensive experiments validate that APT achieves consistent gains on unseen instructions and compositional tasks.

Keywords: VLA, Language Generalization, Manipulation

1 Introduction

Enabling robots to follow diverse task instructions across varied environments is a long-standing goal of generalizable robot policies. Vision-Language-Action (VLA) models have emerged as a promising paradigm toward this goal, leveraging pretrained Vision-Language Models (VLMs) to ground language instructions in visual observations and generate executable robot actions [1, 2, 3]. A critical requirement for real-world deployment is out-of-distribution (OOD) language generalization, *i.e.*, reliably following instructions and task compositions unseen during training. Despite significant recent progress, this capability remains largely unresolved [4, 5]: policies often fail under unseen paraphrased commands, novel object references, or compositional task specifications.

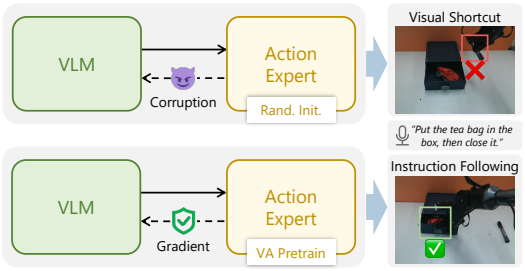

Figure 1: Action expert pretraining (APT) enables effective instruction following.

To achieve language generalization, a prerequisite is that the model genuinely attends to language instructions. However, prior work has identified a structural *imbalance* in VLA data [6, 7]: most trajectories pair many vision-action frames with a single task instruction, making policies prone to visual shortcuts that bypass language. The discrete-action paradigm, which extends a pretrained

VLM with discretized action tokens [8, 9, 10, 11, 12], mitigates this through VL co-training on large-scale reasoning data, effectively preserving the VLM's language capability, and can demonstrate good instruction following [10]. However, discrete representations struggle with the continuous, dynamic nature of dexterous manipulation. The continuous-action paradigm addresses this by coupling the VLM with a generative action expert [1, 2, 13, 14], substantially improving action quality. However, the action expert lacks both pretraining and availability of co-training data, and must learn entirely from imbalanced VLA data. Recent work further shows that gradients from this randomly initialized action expert can be harmful to the VLM backbone, and proposes stopping such gradients [15, 16, 17]. We explain that this gradient harm originates from training a randomly initialized action expert on imbalanced VLA data, where it gravitates toward visual shortcuts, thus producing noisy gradients that corrupt the VLM (Figure 1). By analogy with the discrete-action paradigm, where VL pretraining protects against such shortcuts, we hypothesize that properly pretraining the action expert can relieve this shortcut tendency and improve instruction following. This raises a question: *how can we pretrain the action expert using only the VLA data?*

We answer this through a Bayesian factorization of the VLA policy, $\pi(\mathbf{a}|\mathbf{v}, \ell) \propto \pi^p(\mathbf{a}|\mathbf{v}) \cdot \mathcal{L}(\ell|\mathbf{v}, \mathbf{a})$, which separates action generation into a language-agnostic Vision-Action (VA) prior $\pi^p(\mathbf{a}|\mathbf{v})$ and a language-conditioned VLA likelihood $\mathcal{L}(\ell|\mathbf{v}, \mathbf{a})$. The key observation is that although full VLA triplets suffer from language-vision imbalance, vision-action pairs alone are *well-balanced* and do not create shortcut incentives. Pretraining the action expert as a VA prior on vision-action pairs therefore provides a principled initialization, before any language is introduced. Building on this prior, the VLA likelihood is obtained by finetuning the action expert through VLM language token injection, starting from a point where the action expert already captures coherent visuomotor control.

We propose **APT**, a simple yet effective two-stage training method, emphasizing **A**ction expert **P**re-**T**raining. As shown in Figure 2, in Stage 1, a diffusion-based action expert is pretrained as a VA prior conditioned solely on visual tokens from a frozen VLM backbone. In Stage 2, language tokens are injected through newly introduced attention layers, forming the VLA likelihood that aligns the pretrained action distribution with task instructions. We propose a novel action expert design, characterized by a gated fusion mechanism that integrates layer-wise VLM features into the action expert, enabling the model to inherit the VLM's representational capacity while preserving the pretrained visuomotor priors. We find that jointly finetuning action expert with VLM from a pretrained VA prior yields substantially better language generalization than training from random action initialization, confirming the effectiveness of action expert pretraining. Notably, the two-stage training applies to mainstream continuous-action VLA architectures, such as the π [1, 15] and GR00T [2, 17]-style architectures, consistently improving their language generalization. Extensive experiments across simulation and real-world settings validate improved generalization to unseen instructions and unseen compositional tasks. To summarize, the main contributions are:

- We propose APT, a Bayesian factorization based method that enables principled action expert pretraining on vision-action pairs within existing VLA datasets.
- We propose a novel action expert design with a layer-wise gated fusion mechanism for effective VLM feature fusion.
- We demonstrate that action expert pretraining generalizes across architectures and achieves consistent gains on OOD language generalization in extensive simulation and real-world experiments.

2 Related Work

Vision-Language-Action Models. Vision-Language-Action (VLA) models have emerged as a dominant paradigm for generalist robot policies [18, 19, 20]. Early works [21, 22, 23, 24] train transformer-based policies from scratch on large-scale robot datasets [25, 26, 27, 28, 29, 30, 31]. Subsequent advances finetune pretrained VLMs [32, 33] with action discretization [8, 9, 12, 34], retaining language capability through VL co-training [10], but discrete representations struggle with continuous dexterous manipulation. To improve action quality, diffusion- or flow-based continuous action experts have been widely adopted [35, 1, 13, 36, 37], and dual-system architectures

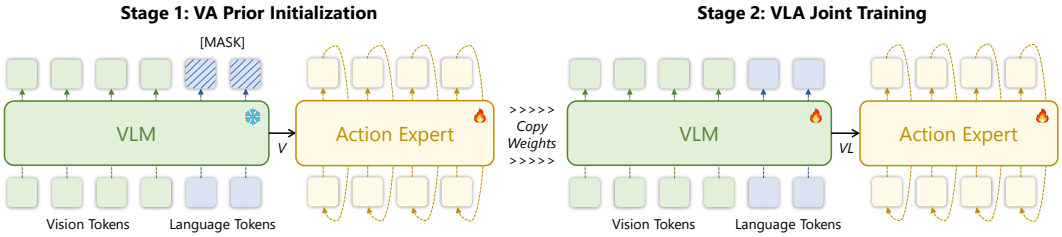

Figure 2: Overview of APT. In Stage 1, the action expert is pretrained as a VA prior conditioned solely on visual tokens from a frozen VLM backbone. In Stage 2, language tokens are injected, training the full VLA policy to align the pretrained action distribution with the task instruction.

that combine VLM and continuous motor control become prevalent [1, 2, 38, 39, 14, 3, 40]. Recent advances further scale this paradigm through VLM reasoning co-training [15, 17, 16, 41] and world-model auxiliary objectives [42, 43, 44, 45, 46]. Despite strong in-distribution performance, these models struggle to generalize to unseen instructions, objects, and compositional task variations [47, 4, 48, 49, 5, 50]. This is because the action experts are randomly initialized and co-trained with the VLM backbone, distorting pretrained language representations. This paper identifies this initialization gap as an important reason for language generalization failure and addresses it directly.

Generalization on Task Instructions. Benchmarking studies [4, 51, 5, 52, 53] reveal that most VLA models treat language instructions as task identifiers rather than grounded semantic descriptions: high success on seen instructions collapses on novel objects, skills, or compositional variations. This reflects a structural shortcut where joint training exploits visual correlations while discarding language semantics [6, 54]. A typical line to address this is knowledge insulation: stopping gradients from the action expert to the VLM backbone so that robot finetuning does not corrupt pretrained language representations [16, 17]. A complementary remedy is co-training on VL reasoning corpora to further regularize the VLM backbone [15, 55, 56]. Architectural approaches such as text-aware feature extraction [57] and inference-time language steering [58, 59, 60, 7, 61] offer complementary gains. BayesVLA [6] provides a principled diagnosis via Bayesian factorization, separating a vision-action prior from a language-conditioned likelihood to prevent visual shortcut learning, and demonstrates generalization to novel instructions, but its pre-/post-contact decomposition limits scalability to heterogeneous datasets. LangForce [54] maximizes conditional mutual information between actions and instructions but focuses on seen-task following, paying less attention on OOD task generalization. This paper extends the Bayesian framework to an action pretraining perspective, identifying action expert initialization as a key factor for OOD language generalization, and addressing it through large-scale pretraining on balanced vision-action pairs.

3 Method

3.1 Problem Statement

We formulate the VLA policy as $\pi(\mathbf{a} \mid \mathbf{v}, \ell)$, where $\mathbf{a}$ is the robot action, $\mathbf{v}$ is the visual observations, and ℓ is the language instruction. We denote a dataset D_{VLA} that contains triplets $(\mathbf{a}, \mathbf{v}, \ell)$ for training the VLA policy. The standard VLA training objective is:

$$\min \; -\mathbb{E}_{(\mathbf{a},\mathbf{v},\ell)\sim\mathcal{D}_{\text{VLA}}}\big[\log \pi(\mathbf{a} \mid \mathbf{v}, \ell)\big]. \tag{1}$$

where $\mathbb{E}$ is the expectation on the dataset D_{VLA}. In the dominant paradigm of continuous action generation [1, 2], a large pretrained VLM encodes visual and language tokens, which an action expert then consumes to generate continuous robot actions. Typically, there is no pretrained action expert, so it is trained from random initialization jointly with the VLM backbone.

Instruction Generalization Bottleneck. Current continuous-action VLA policies perform well on seen tasks, but struggle with unseen instructions and compositional variations [1, 2, 9, 34]. Prior work traces this failure to a structural imbalance in VLA data: a trajectory of T vision-action pairs shares a single language instruction, making visual-action diversity at least T times richer than language [6, 7]. A randomly initialized action expert trained on such data learns to predict actions

from visual cues alone, forming shortcuts that bypass language [6, 54](See Appendix B for analysis). These shortcuts further produce noisy gradients that corrupt the VLM backbone, harming its language representations [16, 17]. Just as VLMs gain generalizable capability by pretraining on balanced vision-language data [10, 15, 32, 33], we argue that the action expert should likewise be pretrained on balanced action data to build coherent visuomotor priors free of shortcut incentives.

3.2 Action Pretraining under Bayesian Formulation

To isolate the effect of language from action generation, we decompose the VLA policy via Bayesian factorization into a Vision-Action (VA) prior and a Vision-Language-Action (VLA) likelihood:

$$\pi(\mathbf{a} \mid \mathbf{v}, \ell) \;\propto\; \pi^{p}(\mathbf{a} \mid \mathbf{v}) \cdot \mathcal{L}(\ell \mid \mathbf{v}, \mathbf{a}). \tag{2}$$

VA Prior. The VA prior $\pi^{p}(\mathbf{a} \mid \mathbf{v})$ models the multimodal distribution of robot actions from visual observation, without any language input. It can be trained exclusively on vision-action pairs from the original VLA dataset. Every visual frame is paired with a unique action annotation, so the data is inherently balanced. Then, the action expert can develop an action manifold that captures diverse manipulation behaviors, without the risk of shortcut learning. Crucially, this produces a meaningful pretraining of the action expert before any language signal is introduced, directly addressing the bottleneck identified above.

VLA Likelihood. With a well-initialized action expert from $\pi^{p}(\mathbf{a} \mid \mathbf{v})$, the VLA likelihood $\mathcal{L}(\ell \mid \mathbf{v}, \mathbf{a})$ introduces language understanding to steer the action distribution to specific tasks. Rather than learning action generation from scratch alongside language grounding, the likelihood model only needs to align the pretrained action distribution with the specific task instruction, a significantly easier learning problem. This decoupling preserves the action expert's learned priors while enabling effective language conditioning.

Training Recipe. The factorization in Eq. (2) induces a natural two-stage pretraining procedure (Figure 2), followed by task-specific post-training: **Pretraining Stage 1: VA Prior Pretraining.** The action expert is trained on one half of the pretraining data without language input, learning $\pi^{p}(\mathbf{a} \mid \mathbf{v})$. All VLM parameters are fixed. **Pretraining Stage 2: VLA Likelihood Alignment.** The pretrained action expert is further conditioned on language input, then the full model is trained on the whole pretraining data, jointly training $\pi^{p}(\mathbf{a} \mid \mathbf{v})$ and $\mathcal{L}(\ell \mid \mathbf{v}, \mathbf{a})$. **Post-Training: Task-specific Finetuning.** Similar as Pretraining Stage 2, the full policy is finetuned on task-specific data to adapt to target-domain distributions.

3.3 Action Expert Design

Figure 3 shows the action expert design.

Multimodal Self-Attention. The action expert is a Transformer-based diffusion model. At each denoising step, visual tokens $\mathbf{v}$, language tokens ℓ, and action tokens $\mathbf{a}$ are concatenated into a single sequence and processed via block-wise causal self-attention. The action token sequence is composed of three parts: $\mathbf{a} = [\mathbf{a}^{\text{hist}}, \mathbf{s}^{\text{prop}}, \mathbf{a}^{\text{noisy}}]$, where $\mathbf{a}^{\text{hist}}$ encodes executed action history, $\mathbf{s}^{\text{prop}}$ encodes the current proprioceptive state, and $\mathbf{a}^{\text{noisy}}$ contains the noisy action tokens to be denoised. The denoising timestep is injected into each attention layer via Feature-wise Linear Modulation (FiLM) [62].

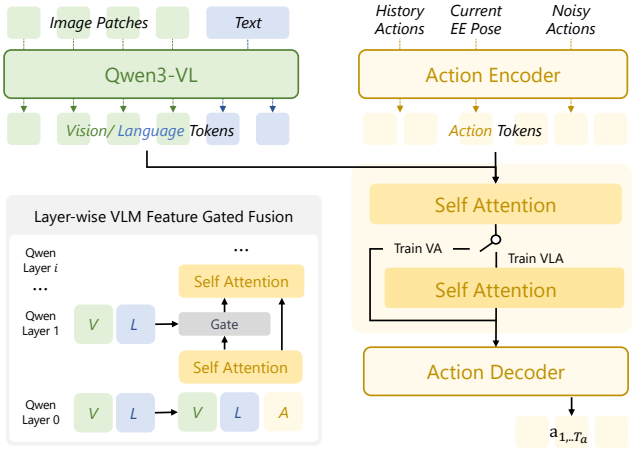

Figure 3: Action Expert Design. VLM features are injected into action expert via gated fusion. The action expert processes multimodal tokens by self-attention.

Layer-wise VLM Feature Gated Fusion. To leverage the semantic representations of large-scale VLMs, we use Qwen3-VL [32] as our VLM backbone and inject its intermediate features into every self-attention layer of the action expert. For an action expert with N attention layers, we uniformly sample N intermediate features from Qwen3-VL at equal depth intervals. The vision and language input tokens $\mathbf{h} = [\mathbf{h}_v, \mathbf{h}_\ell]$ of the $(i{+}1)$-th attention layer of the action expert is:

$$\mathbf{h}_{\text{in}}^{(i+1)} = \mathbf{h}_{\text{out}}^{(i)} + \sigma(\hat{w}_i) \cdot \phi_i^{\text{Qwen3-VL}}(\mathbf{v}, \ell), \tag{3}$$

$\hat{w}_i$ is a learnable scalar and $\sigma(\cdot)$ is the sigmoid function, acting as a gate that modulates the influence of each VLM layer on the action expert. For the first attention layer of the action expert, the input is directly the VLM's input embedding: $\mathbf{h}_{\text{in}}^0 = \phi_0^{\text{Qwen3-VL}}(\mathbf{v}, \ell)$. This layer-wise gated fusion enables the action expert to assimilate both shallow spatial features and deep semantic features from Qwen3-VL, while maintaining its own vision-language pathway through the self-attention layers.

Two Stage Pretraining. The two training stages are realized within a single network via two complementary mechanisms. **Stage 1 VA Prior**: In this stage, we only activate $N/2$ attention layers of action expert for training. The language tokens from Qwen3-VL are completely masked from all self-attention computations in the action expert, reducing the network to a pure vision-action function: $[\mathbf{h}_v, \mathbf{h}_a] = \text{SelfAttn}([\mathbf{v}, \mathbf{a}])$. **Stage 2 VLA Likelihood**: After Stage 1 pretraining, we expand to the action expert from $N/2$ to N attention layers by inserting an interleaved attention layer after each of the original $N/2$ layers. The language mask is removed, and all tokens participate in block-wise causal attention: $[\mathbf{h}_v, \mathbf{h}_\ell, \mathbf{h}_a] = \text{SelfAttn}([\mathbf{v}, \ell, \mathbf{a}])$. The original $N/2$ layers are initialized from the Stage 1 checkpoint, leveraging the learned VA prior. The newly inserted $N/2$ layers then specialize in language-conditioned alignment. Unlike BayesVLA [6] that freezes the prior in Stage 2, APT jointly optimizes all N layers with the full pretraining dataset, allowing the prior and likelihood to co-adapt toward a better equilibrium under large-scale data.

4 Experiments

We conduct comprehensive experiments to validate: 1) the effectiveness of action expert pretraining; 2) to evaluate the instruction generalization of our policy; 3) to validate the architecture designs.

4.1 Simulation Experiments

4.1.1 Benchmarks

We evaluate APT mainly on two simulation benchmarks of language generalization difficulty:

LIBERO-PRO [4] introduces two perturbation types on top of LIBERO [47]: (1) Pos, which swaps object positions while fixing the instruction, testing whether the policy uses language rather than positional priors to locate targets; and (2) Task, which replaces the manipulated object in the instruction with a different object in the same scene, forming an unseen task for OOD language generalization. Notably, simply replicating training trajectories fails under both perturbations, preventing dataset-level shortcuts.

Rigid Object Pick-Place. Following [6], we evaluate on a language-conditioned pick-and-place benchmark with four suites: Seen Object (SO), Unseen Object (UO), Unseen Container (UC), and Unseen Object & Unseen Environment (UOUE). Object layouts, camera viewpoints, and instructions are randomized across all suites, requiring genuine language following even in the SO setting.

4.1.2 Baselines

Our baselines include representative end-to-end VLA policies pretrained on large-scale datasets and finetuned on benchmark-specific data. These include **OpenVLA** [9, 34], π_0 [1], and $\pi_{0.5}$ [15], where OpenVLA and π_0 jointly train VLM and action expert, and $\pi_{0.5}$ apply Knowledge Insulation (KI) [16] to stop the gradient from action expert to VLM during training. We also compare to **LangForce** [54], a concurrent work that enforces language following by maximizing the mutual information between actions and instructions via a dual-branch architecture, and **CaP-X**, which

Method	Spatial		Object		Goal		Long		Avg
	Pos	Task	Pos	Task	Pos	Task	Pos	Task	
OpenVLA [9]	0	0	0	0	0	0	0	0	0
π_0 [1]	0	0	0	0	0	0	0	0	0
$\pi_{0.5}$ [15]	20	1	17	1	**38**	0	8	1	11
LangForce [54]	11	48	10	10	4	11	2	**15**	14
CaP-X [63]	12	14	22	**18**	26	17	–	–	–
APT	44	48	7	10	23	11	6	3	19
APT (Ft VLM)	**62**	**62**	**24**	17	10	**20**	**12**	12	**27**

Table 1: Results on LIBERO-PRO (success rate %).

Method	KI	2-Stage	Ft VLM	SO	UO	UC	UE
π_0			✓	42	30	26	16
$\pi_{0.5}$	✓		✓	84	70	86	50
APT			✓	88	56	66	34
	✓			90	58	40	40
	✓	✓		96	74	90	**62**
		✓	✓	**98**	**84**	**92**	58

Table 2: Results on Pick-Place (rate %).

generates structured task programs via a language model and executes them through learned robot primitives, providing an agent-level baseline. See Appendix C for more details. For brevity, we use **APT** to refer to both the action pretraining method and our action expert design.

4.1.3 Main Results

LIBERO-PRO. Table 1 reports results on LIBERO-PRO. OpenVLA and π_0 achieve 0% success under both perturbations, confirming that direct joint training collapses to visual shortcuts and fails once language matters. $\pi_{0.5}$ recovers on Pos via KI, indicating effective preservation of in-distribution language following. However, it remains near zero on Task, showing KI alone cannot bridge the gap to OOD language generalization. LangForce improves over $\pi_{0.5}$ on Task, but degrades sharply on Pos across all suites, where the language is unchanged but object layouts shift. This may be because aggressively enforcing the action-language coupling makes the policy overly rely on language signals, suppressing the visual cues required for layout adaptation. This exposes a trade-off: $\pi_{0.5}$ preserves visuomotor robustness but fails on novel language, while LangForce gains language sensitivity but loses visual adaptability. APT resolves this trade-off from the initialization side: Stage 1 VA prior training endows the action expert with visuomotor competence to handle layout variation, while Stage 2 VLA training adds language alignment without disrupting the pre-trained distribution. As a result, APT substantially outperforms both $\pi_{0.5}$ and LangForce, validating the effectiveness of action pretraining. APT (Ft VLM) further improves across most suites, showing that stop-gradient is not a necessary condition for language generalization: given a well-initialized action expert, jointly finetuning the VLM yields additional gains. One exception is Goal-Pos, where APT scores below $\pi_{0.5}$: for most cases, APT correctly follows the language, but may fail in obstacle avoidance, which is emphasized in this subsuite. Compared to CaP-X, APT (Ft VLM) achieves a higher average without an agent framework, showing that action pretraining enables a unified policy to match or exceed modular agent systems.

Rigid Object Pick-Place. We report results in Table 2 across three design dimensions to trace the source of generalization gains: **KI** (Knowledge Insulation [16]), **2-Stage** (action expert pretraining), and **Ft VLM** (joint training of VLM and action expert). π_0 (no KI, Ft VLM) yields the weakest generalization. $\pi_{0.5}$ improves substantially by combining KI with large-scale VL data co-training. Among our four APT variants trained solely on VLA data, adding KI without 2-Stage does not give obvious gains over the joint training variant (Ft VLM), showing that stopping gradients alone does not resolve generalization failure. Combining KI with 2-Stage surpasses $\pi_{0.5}$ without any VL reasoning data, directly attributing the gain to action expert pretraining. Finally, replacing KI with joint VLM finetuning while retaining 2-Stage achieves the best overall result, confirming that KI is not a necessary condition. Instead, a well-initialized action prior allows joint VLM training to further improve rather than degrade language generalization.

4.1.4 More Ablation Studies

Action Pretraining on More Architectures. To verify that action pretraining generalizes beyond our architecture design, we apply it to two representative architectures to combine VLM and the action expert: (1) **π-style** [1, 15], where VLM and action expert tokens interact via block-wise causal self-attention at every attention layer and (2) **GR00T-style** [2, 17], where only the final VLM layer feature conditions the action expert via cross-attention. Figure 4 shows that 2-Stage train-

Figure 4: Action expert pretraining applies to diverse architectures.

ing improves generalization across almost all settings. The gain is most pronounced for APT and GR00T-style compared to π-style. We attribute this to the VLM fusion design: APT inserts VLM features by gated fusion, and GR00T-style uses only the final-layer features, both of which better preserve the VA prior learned in Stage 1. In contrast, π-style uses original VLM features at every self-attention layer, which less effectively preserves the pretrained action representations.

Effectiveness of Large-Scale Pretraining. To disentangle large-scale pretraining, we evaluate a variant applying two-stage training only on task-specific datasets, *i.e.*, without any pretraining on large-scale datasets (**w/o Pretraining**). As shown in Figure 5, this variant still recovers meaningful generalization, confirming that action pretraining provides an inductive bias even in the low-data regime. However, it significantly underperforms APT with pretraining on UO and UOUE, where generalizing to unseen categories and environments requires diverse action priors that can only be acquired from large-scale heterogeneous pretraining data.

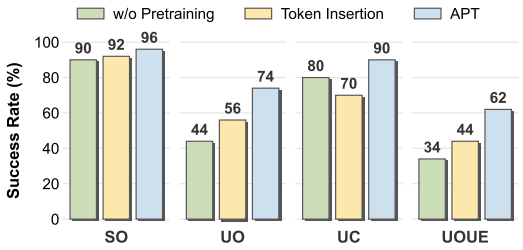

Figure 5: Ablation on large-scale pretraining and language injection mechanism.

Language Injection Mechanism. We compare our language injection mechanism via gated fusion against **Token Insertion**, which trains the VA prior with N self-attention layers in Stage 1, and then plugs language tokens directly into the attention layers of the pretrained VA prior in Stage 2. Figure 5 shows that, our mechanism outperforms across all dimensions, with the largest gaps on UO and UOUE. **Token Insertion** abruptly modifies the pretrained VA distribution, risking partial forgetting of VA prior. Instead, through gated fusion that modulate language conditioning, the VA prior can be better preserved while enabling language-guided refinement of the action distribution.

4.2 Real-World Experiments

4.2.1 Experiment Setup

In this section, we evaluate APT in real-world setups. For each task, we collect 30 demonstrations to finetune each policy. We compare against $\pi_{0.5}$ [15] across all tasks. Our real-world evaluation covers two aspects: (1) Single Task Generalization, which tests OOD language and object generalization within individual tasks; and (2) Compositional Task Generalization, which evaluates sequential multi-task instruction following via task coaching (per-task instructions issued in sequence) and task chaining (a single concatenated instruction). See Appendix E for more details.

4.2.2 Single Task Generalization

Pick-Place Task: The robot picks a specified object among several objects and places it on a specified container. We evaluate 110 trials across four settings of increasing OOD difficulty: seen objects (SO), unseen objects (UO), unseen objects with unseen containers (UOUC), and further with unseen environments (UOUCUE). Each unseen setting demonstrates OOD language generalization. Table 3 shows that both methods achieve strong SO performance, confirming reliable seen language following. For OOD generalization, $\pi_{0.5}$ degrades significantly as task complexity increases. It sometimes fails to pick up the unseen target (Figure 7 (a)), or grasps a non-target object. In contrast,

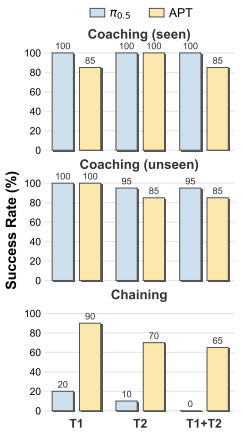

Figure 6: Results on compositional task.

Method	Pick-Place				Clutter Pick-Place			
	SO	UO	UOUC	UOUCUE	SO	UC	UO	UOUE
$\pi_{0.5}$ [15]	27/30	11/20	9/20	16/40	18/30	18/30	4/10	3/10
APT	**29/30**	**17/20**	**16/20**	**28/40**	**25/30**	**22/30**	**7/10**	**6/10**

Table 3: Real-world task generalization results (successes/trials).

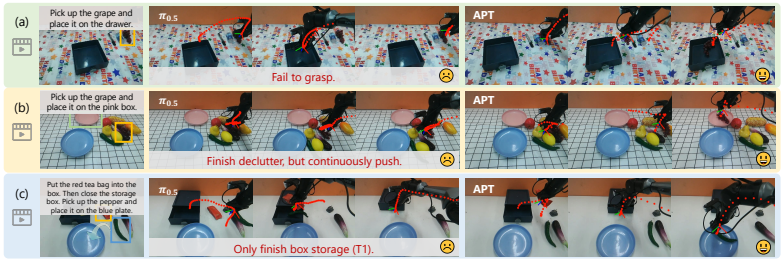

Figure 7: Real-world cases. (a) pick-place task, (b) clutter pick-place task, (c) compositional task chaining.

APT maintains robust performance across all OOD levels. This is because the action expert is well initialized for instruction alignment, thus better inherits language generalization from VLM.

Clutter Pick-Place Task: The target object is tightly surrounded by distractors. The robot must first push to clear space before executing pick-and-place, requiring long-horizon planning across push, pick, and place sub-tasks [64, 65] and semantic grounding to identify the correct target. We evaluate over 80 trials across four settings (SO, UC, UO, UOUE). Results in Table 3 show that APT outperforms $\pi_{0.5}$ across all settings, especially in OOD language generalization. $\pi_{0.5}$ sometimes struggles to transfer from push to grasp, or mistakenly pushes off the target. For unseen object settings, $\pi_{0.5}$ frequently grounds the wrong object when objects share similar colors (*e.g.*, misidentifying "eggplant" as "grape"), and hesitates to grasp the target (Figure 7 (b)). APT makes few such errors: validates that our action pretraining provides performance gain even in heavily cluttered long-horizon conditions.

4.2.3 Compositional Task Generalization

We evaluate compositional instruction following across two tasks: Table Storage (T1), where the robot picks specified items from the table, places them in a storage box, and closes the box; and Pick-Place (T2) from Section 4.2.2. Both $\pi_{0.5}$ and APT achieve above 80% success on each task in the seen setting, confirming sufficient individual task competence for the compositional evaluation. Notably, for the following evaluation, all the objects are seen during training.

Task Coaching: Instructions for each task are issued sequentially as separate prompts (seen or unseen language), with half in T1→T2 order, half in T2→T1 order. As shown in Figure 6, both methods perform comparably under seen language coaching, confirming that both retain strong language following when instructions are issued one at a time. For unseen language coaching, we perturb instructions via verb substitution, noun substitution, and cross-task object swapping. Both policies demonstrate robustness with nearly zero degradation. This indicates that single-task instruction understanding for seen objects is not the bottleneck for either method.

Task Chaining: A single concatenated prompt specifies both tasks jointly (*e.g.*, "Put the red tea bag into the box. Then close the storage box. Pick up the pepper and place it on the blue plate."). The results diverge dramatically: $\pi_{0.5}$ nearly collapses, while APT maintains strong performance. This contrast reveals a key limitation of $\pi_{0.5}$: although it can follow individual instructions reliably, it fails to parse and execute multi-task instructions in a single prompt. It exhibits a typical failure mode of over-executing the first task: it attempts to place all target objects into the storage box, and fails to transition to the second task (Figure 7 (c)). Instead, APT successfully executes chained tasks without explicit task segmentation, indicating better preservation of VLM's language understanding. Our failures arise from two sources: the policy occasionally advances to another task before the current task is fully completed, or it confuses the placement target with visually similar distractors.

5 Conclusion

We propose action expert pretraining on balanced vision-action data to improve OOD language generalization in continuous-action VLA policies. We decompose the policy into a VA prior and a VLA likelihood, inducing a two-stage pretraining procedure. Comprehensive experiments validate that APT consistently improves instruction generalization, and applies to diverse architectures.

Limitations. Our current design does not explicitly model long-horizon memory, limiting generalization on tasks that require tracking multi-step progress. Besides, our evaluation focuses on tabletop manipulation. Extending to locomotion or mobile manipulation remains to be explored.

References

[1] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[2] J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

[3] G. A. Team. Gen-0: Embodied foundation models that scale with physical interaction. *Generalist AI Blog*, 2025. https://generalistai.com/blog/preview-uqlxvb-bb.html.

[4] X. Zhou, Y. Xu, G. Tie, Y. Chen, G. Zhang, D. Chu, P. Zhou, and L. Sun. Libero-pro: Towards robust and fair evaluation of vision-language-action models beyond memorization. *arXiv preprint arXiv:2510.03827*, 2025.

[5] J. Gao, S. Belkhale, S. Dasari, A. Balakrishna, D. Shah, and D. Sadigh. A taxonomy for evaluating generalist robot policies. *arXiv preprint arXiv:2503.01238*, 2025.

[6] K. Xu, Z. Zhu, A. Chen, S. Zhao, Q. Huang, Y. Yang, H. Lu, R. Xiong, M. Tomizuka, and Y. Wang. Seeing to act, prompting to specify: A bayesian factorization of vision language action policy. *arXiv preprint arXiv:2512.11218*, 2025.

[7] Y. Fang, Y. Feng, D. Jing, J. Liu, Y. Yang, Z. Wei, D. Szafir, and M. Ding. When vision overrides language: Evaluating and mitigating counterfactual failures in vlas. *arXiv preprint arXiv:2602.17659*, 2026.

[8] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning (CoRL)*, pages 2165–2183. PMLR, 2023.

[9] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. P. Foster, P. R. Sanketi, Q. Vuong, et al. Openvla: An open-source vision-language-action model. In *Conference on Robot Learning (CoRL)*, pages 2679–2713, 2024.

[10] K. Pertsch, K. Stachowicz, B. Ichter, D. Driess, S. Nair, Q. Vuong, O. Mees, C. Finn, and S. Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

[11] G. R. Team, S. Abeyruwan, J. Ainslie, J.-B. Alayrac, M. G. Arenas, T. Armstrong, A. Balakrishna, R. Baruch, M. Bauza, M. Blokzijl, et al. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

[12] Q. Zhao, Y. Lu, M. J. Kim, Z. Fu, Z. Zhang, Y. Wu, Z. Li, Q. Ma, S. Han, C. Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1702–1713, 2025.

[13] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu. RDT-1B: A diffusion foundation model for bimanual manipulation. In *International Conference on Learning Representations (ICLR)*, 2025.

[14] J. Wen, Y. Zhu, M. Zhu, Z. Tang, J. Li, Z. Zhou, X. Liu, C. Shen, Y. Peng, and F. Feng. Diffusionvla: Scaling robot foundation models via unified diffusion and autoregression. In *International Conference on Machine Learning (ICML)*, pages 66558–66574. PMLR, 2025.

[15] P. Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

[16] D. Driess, J. T. Springenberg, L. Yu, A. Li-Bell, K. Pertsch, A. Z. Ren, H. Walke, Q. Vuong, L. X. Shi, S. Levine, et al. Knowledge insulating vision-language-action models: Train fast, run fast, generalize better. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[17] J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, et al. Gr00t n1.5: An improved open foundation model for generalist humanoid robots. 2025.

[18] Y. Zhong, F. Bai, S. Cai, X. Huang, Z. Chen, X. Zhang, Y. Wang, S. Guo, T. Guan, K. N. Lui, et al. A survey on vision-language-action models: An action tokenization perspective. *arXiv preprint arXiv:2507.01925*, 2025.

[19] X. Li, P. Li, L. Qian, M. Liu, D. Wang, J. Liu, B. Kang, X. Ma, X. Wang, D. Guo, et al. What matters in building vision-language-action models for generalist robots. *Nature Machine Intelligence*, pages 1–15, 2026.

[20] C. Cui, P. Ding, W. Song, S. Bai, X. Tong, Z. Ge, R. Suo, W. Zhou, Y. Liu, B. Jia, et al. Openhelix: A short survey, empirical analysis, and open-source dual-system vla model for robotic manipulation. *arXiv preprint arXiv:2505.03912*, 2025.

[21] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. RT-1: Robotics transformer for real-world control at scale. *Robotics: Science and Systems (RSS)*, 2023.

[22] O. M. Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

[23] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *Robotics: Science and Systems (RSS)*, 2023.

[24] Y. Jiang, A. Gupta, Z. Zhang, G. Wang, Y. Dou, Y. Chen, L. Fei-Fei, A. Anandkumar, Y. Zhu, and L. Fan. Vima: Robot manipulation with multimodal prompts. In *International Conference on Machine Learning (ICML)*, pages 14975–15022, 2023.

[25] O. X.-E. Collaboration. Open X-Embodiment: Robotic learning datasets and RT-X models. https://arxiv.org/abs/2310.08864, 2023.

[26] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, et al. DROID: A large-scale in-the-wild robot manipulation dataset. In *Robotics: Science and Systems (RSS)*, 2024.

[27] H.-S. Fang, H. Fang, Z. Tang, J. Liu, C. Wang, J. Wang, H. Zhu, and C. Lu. Rh20t: A comprehensive robotic dataset for learning diverse skills in one-shot. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 653–660. IEEE, 2024.

[28] K. Wu, C. Hou, J. Liu, Z. Che, X. Ju, Z. Yang, M. Li, Y. Zhao, Z. Xu, G. Yang, et al. Robomind: Benchmark on multi-embodiment intelligence normative data for robot manipulation. *arXiv preprint arXiv:2412.13877*, 2024.

[29] Q. Bu, J. Cai, L. Chen, X. Cui, Y. Ding, S. Feng, S. Gao, X. He, X. Hu, X. Huang, et al. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025.

[30] T. Chen, Z. Chen, B. Chen, Z. Cai, Y. Liu, Z. Li, Q. Liang, X. Lin, Y. Ge, Z. Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

[31] H. Geng, F. Wang, S. Wei, Y. Li, B. Wang, B. An, C. T. Cheng, H. Lou, P. Li, Y.-J. Wang, et al. Roboverse: Towards a unified platform, dataset and benchmark for scalable and generalizable robot learning. *arXiv preprint arXiv:2504.18904*, 2025.

[32] S. Bai, Y. Cai, R. Chen, K. Chen, X. Chen, Z. Cheng, L. Deng, W. Ding, C. Gao, C. Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[33] L. Beyer, A. Steiner, A. S. Pinto, A. Kolesnikov, X. Wang, D. Salz, M. Neumann, I. Alabdulmohsin, M. Tschannen, E. Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

[34] M. J. Kim, C. Finn, and P. Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

[35] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems (RSS)*, 2023.

[36] M. Zhu, Y. Zhu, J. Li, J. Wen, Z. Xu, N. Liu, R. Cheng, C. Shen, Y. Peng, F. Feng, et al. Scaling diffusion policy in transformer to 1 billion parameters for robotic manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 10838–10845. IEEE, 2025.

[37] J. Liu, H. Chen, P. An, Z. Liu, R. Zhang, C. Gu, X. Li, Z. Guo, S. Chen, M. Liu, et al. Hybridvla: Collaborative diffusion and autoregression in a unified vision-language-action model. *arXiv preprint arXiv:2503.10631*, 2025.

[38] Q. Bu, H. Li, L. Chen, J. Cai, J. Zeng, H. Cui, M. Yao, and Y. Qiao. Towards synergistic, generalized, and efficient dual-system for robotic manipulation. *arXiv preprint arXiv:2410.08001*, 2024.

[39] M. Shukor, D. Aubakirova, F. Capuano, P. Kooijmans, S. Palma, A. Zouitine, M. Aractingi, C. Pascal, M. Russi, A. Marafioti, et al. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

[40] Y. Jiang, S. Huang, S. Xue, Y. Zhao, J. Cen, S. Leng, K. Li, J. Guo, K. Wang, M. Chen, et al. Rynnvla-001: Using human demonstrations to improve robot manipulation. *arXiv preprint arXiv:2509.15212*, 2025.

[41] W. Wu, F. Lu, Y. Wang, S. Yang, S. Liu, F. Wang, Q. Zhu, H. Sun, Y. Wang, S. Ma, et al. A pragmatic vla foundation model. *arXiv preprint arXiv:2601.18692*, 2026.

[42] J. Cen, C. Yu, H. Yuan, Y. Jiang, S. Huang, J. Guo, X. Li, Y. Song, H. Luo, F. Wang, et al. Worldvla: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025.

[43] W. Zhang, H. Liu, Z. Qi, Y. Wang, X. Yu, J. Zhang, R. Dong, J. He, H. Wang, Z. Zhang, et al. DreamVLA: A vision-language-action model dreamed with comprehensive world knowledge. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[44] J. Sun, W. Zhang, Z. Qi, S. Ren, Z. Liu, H. Zhu, G. Sun, X. Jin, and Z. Chen. VLA-JEPA: Enhancing vision-language-action model with latent world model. *arXiv preprint arXiv:2602.10098*, 2026.

[45] P. Intelligence, B. Ai, A. Amin, R. Aniceto, A. Balakrishna, G. Balke, K. Black, G. Bokinsky, S. Cao, T. Charbonnier, et al. $\pi_{0.7}$: a steerable generalist robotic foundation model with emergent capabilities. *arXiv preprint arXiv:2604.15483*, 2026.

[46] L. Li, Q. Zhang, Y. Luo, S. Yang, R. Wang, F. Han, M. Yu, Z. Gao, N. Xue, X. Zhu, et al. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026.

[47] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 36:44776–44791, 2023.

[48] S. Fei, S. Wang, J. Shi, Z. Dai, J. Cai, P. Qian, L. Ji, X. He, S. Zhang, Z. Fei, et al. Libero-plus: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025.

[49] S. Nasiriany, A. Maddukuri, L. Zhang, A. Parikh, A. Lo, A. Joshi, A. Mandlekar, and Y. Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. *arXiv preprint arXiv:2406.02523*, 2024.

[50] J. Guo, Z. Wu, C. Tu, Y. Ma, X. Kong, Z. Liu, J. Ji, S. Zhang, Y. Chen, K. Chen, et al. On robustness of vision-language-action model against multi-modal perturbations. *arXiv preprint arXiv:2510.00037*, 2025.

[51] S. Orjuela et al. Robust skills, brittle grounding: Diagnosing restricted generalization in vision-language action policies via multi-object picking. *arXiv preprint arXiv:2602.24143*, 2026.

[52] S. L. Wanna, A. Luhtaru, R. Barron, J. Salfity, J. Moore, C. Matuszek, and M. Pryor. Let's talk about language! investigating linguistic diversity in embodied ai datasets. In *1st Workshop on Safely Leveraging Vision-Language Foundation Models in Robotics: Challenges and Opportunities*.

[53] I. Fang, J. Zhang, S. Tong, and C. Feng. From intention to execution: Probing the generalization boundaries of vision-language-action models. *arXiv preprint arXiv:2506.09930*, 2025.

[54] S. Lian, B. Yu, X. Lin, L. T. Yang, Z. Shen, C. Wu, Y. Miao, C. Huang, and K. Chen. LangForce: Bayesian decomposition of vision language action models via latent action queries. *arXiv preprint arXiv:2601.15197*, 2026.

[55] S. Yang, H. Li, Y. Chen, B. Wang, Y. Tian, T. Wang, H. Wang, F. Zhao, Y. Liao, and J. Pang. Instructvla: Vision-language-action instruction tuning from understanding to manipulation. *arXiv preprint arXiv:2507.17520*, 2025.

[56] C. Cheang, S. Chen, Z. Cui, Y. Hu, L. Huang, T. Kong, H. Li, Y. Li, Y. Liu, X. Ma, et al. Gr-3 technical report. *arXiv preprint arXiv:2507.15493*, 2025.

[57] H. Huang, F. Liu, L. Fu, T. Wu, M. Mukadam, J. Malik, K. Goldberg, and P. Abbeel. Otter: A vision-language-action model with text-aware visual feature extraction. In *International Conference on Machine Learning (ICML)*.

[58] M. Nakamoto, O. Mees, A. Kumar, and S. Levine. Steering your generalists: Improving robotic foundation models via value guidance. In *Conference on Robot Learning (CoRL)*, pages 4996–5013. PMLR, 2025.

[59] Y. Wu, R. Tian, G. Swamy, and A. Bajcsy. From foresight to forethought: Vlm-in-the-loop policy steering via latent alignment. *arXiv preprint arXiv:2502.01828*, 2025.

[60] Y. Zhang, C. Wang, O. Lu, Y. Zhao, Y. Ge, Z. Sun, X. Li, C. Zhang, C. Bai, and X. Li. Align-then-steer: Adapting the vision-language action models through unified latent guidance. *arXiv preprint arXiv:2509.02055*, 2025.

[61] Z. Zhan, Y. Chen, J. Zhou, Q. Lv, H. Liu, K. Wang, L. Lin, and G. Wang. Stable language guidance for vision-language-action models. *arXiv preprint arXiv:2601.04052*, 2026.

[62] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 32, pages 3942–3951, 2018.

[63] M. Fu, J. Yu, K. El-Refai, E. Kou, H. Xue, H. Huang, W. Xiao, G. Wang, F.-F. Li, G. Shi, et al. Cap-x: A framework for benchmarking and improving coding agents for robot manipulation. *arXiv preprint arXiv:2603.22435*, 2026.

[64] Z. Xu, Z. He, J. Wu, and S. Song. Learning 3d dynamic scene representations for robot manipulation. In *Conference on Robot Learning (CoRL)*, pages 126–142. PMLR, 2021.

[65] Z. Chen, Q. Yan, Y. Chen, T. Wu, J. Zhang, Z. Ding, J. Li, Y. Yang, and H. Dong. ClutterDex-Grasp: A Sim-to-Real system for general dexterous grasping in cluttered scenes. In *Conference on Robot Learning (CoRL)*, pages 885–905. PMLR, 2025.

[66] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5738–5746, 2019.

[67] A. Chen, Y. Yang, Z. Zhu, K. Xu, Z. Zhou, R. Xiong, and Y. Wang. Toward embodiment equivariant vision-language-action policy. *arXiv preprint arXiv:2509.14630*, 2025.

[68] R. Li, B. Yi, J. Liu, H. Gao, Y. Ma, and A. Kanazawa. Cameras as relative positional encoding. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[69] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[70] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:6840–6851, 2020.

[71] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*.

[72] Y. Tian, Y. Yang, Y. Xie, Z. Cai, X. Shi, N. Gao, H. Liu, X. Jiang, Z. Qiu, F. Yuan, et al. Interndata-a1: Pioneering high-fidelity synthetic data for pre-training generalist policy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 976–985, 2026.

[73] X. Chen, Y. Chen, Y. Fu, N. Gao, J. Jia, W. Jin, H. Li, Y. Mu, J. Pang, Y. Qiao, et al. Internvla-m1: A spatially guided vision-language-action framework for generalist robot policy. *arXiv preprint arXiv:2510.13778*, 2025.

[74] S. Karamcheti, S. Nair, A. Balakrishna, P. Liang, T. Kollar, and D. Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. In *International Conference on Machine Learning (ICML)*, pages 23123–23144, 2024.

[75] Q. Bu, Y. Yang, J. Cai, S. Gao, G. Ren, M. Yao, P. Luo, and H. Li. Univla: Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2505.06111*, 2025.

[76] S. Tan, K. Dou, Y. Zhao, and P. Krähenbühl. Interactive post-training for vision-language-action models. *arXiv preprint arXiv:2505.17016*, 2025.

[77] J. Zheng, J. Li, Z. Wang, D. Liu, X. Kang, Y. Feng, Y. Zheng, J. Zou, Y. Chen, J. Zeng, et al. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025.

[78] NVIDIA. Isaac Sim. URL https://github.com/isaac-sim/IsaacSim.

[79] K. Gao, D. Lau, B. Huang, K. E. Bekris, and J. Yu. Fast high-quality tabletop rearrangement in bounded workspace. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1961–1967. IEEE, 2022.

[80] K. Xu, H. Yu, Q. Lai, Y. Wang, and R. Xiong. Efficient learning of goal-oriented push-grasping synergy in clutter. *IEEE Robotics and Automation Letters*, 6(4):6337–6344, 2021.

A Implementation Details

Action Representation. Actions are defined on the SE(3) manifold: 3D translation, 6D continuous rotation [66], and normalized gripper width (-1 fully closed, $+1$ fully open), yielding a 10-dimensional action vector. Following [67], all end-effector poses and trajectories are expressed in the camera coordinate frame (Figure 8). This representation disentangles the embodiment-relative information and provides embodiment equivariance across the heterogeneous platforms used for pretraining.

Network Architecture. The VLM backbone is Qwen3-VL-2B-Instruct [32]. Visual tokens and language tokens are processed by separate MLP projectors before being injected into the action expert's embedding space. The action expert consists of $N = 20$ attention layers. Both the action encoder and decoder are 2-layer MLPs with hidden dimension 768. We use an action chunk length of $T_a = 32$ and a history action length of 1. Input images are resized to 256×256 for the wrist and third-view cameras. Proprioception is encoded as a 10-dimensional vector following the action representation above.

Positional Encodings. The two stages employ different positional encodings suited to their respective objectives. In Stage 1, visual and action tokens use Projective Positional Encoding (PRoPE) [68], which embeds camera extrinsics directly into the positional embedding and enables multi-view spatial reasoning. In Stage 2, the inherited $N/2$ layers retain PRoPE, while the newly inserted $N/2$ layers use Multimodal Rotary Position Embedding (mRoPE) [69, 32] that natively handles interleaved vision and language tokens. This design lets the inherited layers maintain action understanding while the inserted layers focus on language alignment. Figure 9 illustrates the attention masks of the two layer types across both stages.

Training Hyperparameters. We use the AdamW optimizer, with action expert learning rate 10^{-4} and weight decay 10^{-2}. When jointly finetuning the VLM, we use a VLM learning rate of 10^{-5} and weight decay of 10^{-10}. Models are trained with a batch size of 256. Stage 1 and Stage 2 are each trained for 100k iterations. We adopt DDPM [70] with 100 diffusion steps for training, and DDIM [71] with 20 denoising steps for inference.

Pretraining Datasets. We pretrain on four large-scale datasets covering diverse embodiments, scenes, and skills.

- *DROID* [26] contains 78,544 real-world Franka trajectories (about 350 hours) across 564 scenes and 84 task categories with multi-view RGB-D and end-effector pose labels.
- *AgiBotWorld-Alpha* [29] provides over one million real-world dual-arm trajectories spanning 217 tasks across five deployment scenarios.
- *InternData-A1* [72] is a high-fidelity synthetic dataset with about 630k trajectories across 4 embodiments, 18 skills, 70 tasks, and 227 scenes, spanning rigid, articulated, deformable, and fluid manipulation.
- *InternVLA-M1* [73] contributes 244k generated pick-place trajectories across 200 tasks and over 3,000 object instances, providing dense language-object correspondences.

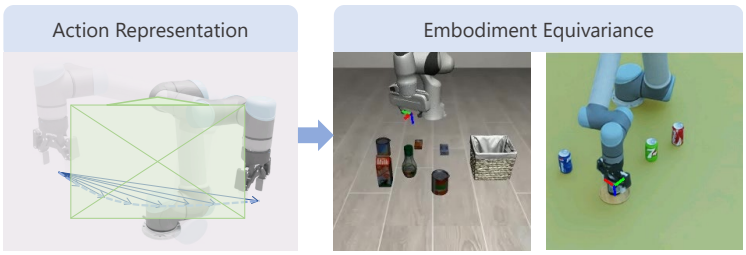

Figure 8: Action representation. We use relative end-effector poses projected in the camera frame, which is shared across embodiments.

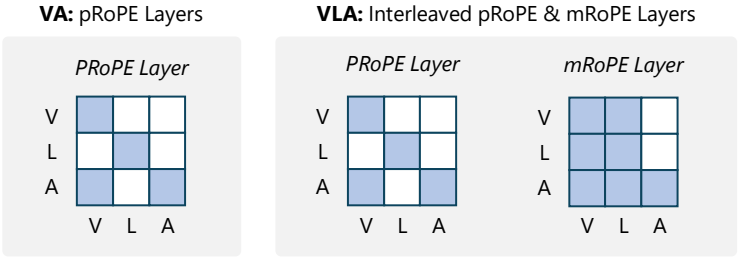

Figure 9: Attention masks of the two-stage training. Stage 1 uses PRoPE layers over vision and action tokens only. Stage 2 interleaves the inherited PRoPE layers with newly inserted mRoPE layers that jointly attend over vision, language, and action tokens.

During pretraining, we sample the four datasets with weights 5:5:4:1 for DROID, AgiBotWorld-Alpha, InternData-A1, and InternVLA-M1 respectively, balancing real-robot diversity with synthetic skill coverage.

B Visual Shortcut Analysis

Following [6], we use information theory to validate that the standard VLA training objective admits a shortcut solution that largely ignores language, and that our two-stage training mitigates this issue. The analysis is consistent with our main claim: a randomly initialized action expert, trained jointly with the VLM, falls into the shortcut degrading OOD language generalization.

Formally, minimizing the VLA objective corresponds to minimizing the conditional negative log-likelihood of actions given vision and language, which lower-bounds at the conditional entropy:

$$\min -\mathbb{E}[\log \pi(\mathbf{a}|\mathbf{v}, \ell)] \approx \mathcal{L}_{\text{VLA}} = H(\mathbf{a}|\mathbf{v}, \ell) \tag{4}$$

When training converges, the negative log-likelihood objective approximates this entropy lower bound.

Dataset Imbalance. In most VLA datasets, language diversity is much smaller than vision-action diversity. A trajectory typically consists of many visual frames associated with a single language prompt. Although two trajectories with different prompts may share a small number of visually similar frames, such overlaps occur only rarely. As a result, for most visual observations, the corresponding language instruction is nearly deterministic. Formally, denoting $P(\ell|\mathbf{v})$ as the probability of language prompt conditioned on the visual frame, we state the assumption that $P(\ell|\mathbf{v})$ is effectively one-hot for the majority of frames. In other words, the language can often be inferred directly from the visual observation. Consequently, the conditional entropy is upper-bounded by a small constant, $H(\ell|\mathbf{v}) \leq \epsilon$. For example, if a drawer is visible in a frame, the associated instruction is highly likely to be opening or closing the drawer. Such imbalance is prevalent in existing embodied datasets and implies that language provides limited additional information beyond what is already encoded in the visual input.

B.1 Shortcut Learning

Under this assumption, we show that the standard VLA training admits a shortcut solution. Defining the conditional mutual information I as

$$\begin{aligned} I(\mathbf{a}; \ell|\mathbf{v}) &= H(\mathbf{a}|\mathbf{v}) - H(\mathbf{a}|\mathbf{v}, \ell) \\ &= H(\ell|\mathbf{v}) - H(\ell|\mathbf{v}, \mathbf{a}) \end{aligned} \tag{5}$$

where $H(\cdot) \geq 0$. Combining with the dataset assumption gives

$$0 \leq I(\mathbf{a}; \ell|\mathbf{v}) \leq H(\ell|\mathbf{v}) \leq \epsilon \tag{6}$$

leading to

$$0 \leq H(\mathbf{a}|\mathbf{v}) - H(\mathbf{a}|\mathbf{v}, \ell) \leq \epsilon \tag{7}$$

which means that the inclusion of the language prompt contributes only marginally to action prediction.

Now consider an alternative vision-only model trained with the objective

$$\min -\mathbb{E}[\log \pi(\mathbf{a}|\mathbf{v})] \tag{8}$$

whose optimal lower bound $\mathcal{L}_{\text{VA}}$ is the conditional entropy $H(\mathbf{a}|\mathbf{v})$. Then we have

$$\mathcal{L}_{\text{VLA}} \leq \mathcal{L}_{\text{VA}} \leq \mathcal{L}_{\text{VLA}} + \epsilon \tag{9}$$

This bound shows that a vision-only policy achieves only marginally larger loss than the full VLA policy. Since the vision-only policy is simpler (it does not depend on ℓ), gradient descent on the VLA objective from a random action expert is biased toward this vision-only solution. We refer to this as the *visual shortcut*, and it explains why standard VLA training is weak at OOD instruction following.

B.2 Two-stage Conditioning

Bayesian factorization induces a two-stage training procedure that avoids this shortcut. In Stage 1, we learn the prior $\pi^p(\mathbf{a}|\mathbf{v})$ following Eqn. 8. Because the prior does not condition on language by construction, it is exactly the desired vision-action policy rather than a shortcut. Denoting the learned parameters as θ_{VA},

$$\mathcal{L}_{\text{VA}} \approx \min -\mathbb{E}[\log \pi^p_{\theta_{\text{VA}}}(\mathbf{a}|\mathbf{v})] \tag{10}$$

Let $f_{\theta_{\text{VA}}}(\mathbf{v})$ denote the intermediate visual embedding produced by the pretrained prior. In Stage 2, we train the language-conditioned policy on top of this embedding:

$$\mathcal{L}_{\text{VLA}} \leq \min -\mathbb{E}[\log \pi(\mathbf{a}|f_{\theta_{\text{VA}}}(\mathbf{v}), \ell)] \leq \mathcal{L}_{\text{VA}} \tag{11}$$

where θ_{VA} is initialized from Stage 1 and the newly added language-conditioning parameters are randomly initialized. If language were entirely ignored, the right inequality in Eqn. 11 would be tight at $\mathcal{L}_{\text{VA}}$. Since the loss can be further optimized, the gradient incentivizes the network to use language for action generation. Therefore, the resulting policy can effectively ground the language prompt.

C Baseline Details

OpenVLA [9] is built on the Prismatic 7B VLM backbone [74], pretrained on diverse datasets including Open X-Embodiment [25], and finetuned using the OpenVLA-OFT recipe [34] with continuous action representations.

π_0 [1] combines PaliGemma [33] with a diffusion-based action expert, pretrained on a subset of OXE and the π dataset, and finetuned on DROID [26].

$\pi_{0.5}$ [15] extends π_0 with a hybrid pretraining procedure that jointly trains on internet-scale reasoning data and large-scale robot demonstration data, representing the current state of the art in generalist VLA policies.

LangForce [54] introduces latent action queries into Qwen3-VL [32] and enforces language following by estimating vision-only and language-conditioned action distributions via dual branches and maximizing the mutual information between actions and instructions.

CaP-X [63] is a code-as-policies framework that generates structured task programs via a language model and executes them through learned robot primitives, providing a complementary language-grounding baseline.

The following are additional baselines involved in the appendix:

π_0**-FAST** [10] is a discrete-token variant of π_0 that replaces the diffusion action expert with a frequency-space action tokenization, enabling faster autoregressive action prediction.

UniVLA [75] learns a unified task-centric latent action space from heterogeneous human videos and robot data, supporting cross-embodiment pretraining with a discrete latent action codebook.

WorldVLA [42] couples a VLA policy with a world model that predicts future frames as an auxiliary objective to enhance visual representation learning.

RIPT-VLA [76] introduces a reinforcement-learning-based interactive post-training stage that fine-tunes pretrained VLA models using sparse binary success rewards via dynamic rollout sampling and leave-one-out advantage estimation.

X-VLA [77] is a flow-matching-based VLA architecture that uses soft prompts with embodiment-specific learnable embeddings to handle heterogeneous cross-embodiment data, built on standard transformer encoders for scalability.

D Simulation Experiment Details

D.1 Benchmark Details

We evaluate APT on simulation benchmarks of increasing language-generalization difficulty. Figure 10 summarizes the benchmark settings.

LIBERO [47]. We use all four task suites of LIBERO. LIBERO-Spatial tests spatial relational understanding under fixed object sets; LIBERO-Object evaluates the ability to discriminate object instances under fixed scene layouts; LIBERO-Goal measures goal-directed instruction following; and LIBERO-Long contains 10 long-horizon sequential tasks. Each suite contains 10 tasks with 50 demonstrations each, yielding 500 trajectories per suite. Notably, LIBERO is an in-distribution evaluation rather than a generalization benchmark: training and evaluation share the same task set, object set, and language template.

LIBERO-Plus [48]. LIBERO-Plus extends LIBERO with seven axes of controlled perturbation: object layout, camera viewpoint, robot initial state, language instruction paraphrase, lighting conditions, background textures, and sensor noise. Five of them (camera viewpoint, robot initial state, lighting conditions, background textures, and sensor noise) test robustness to visual and configuration changes while keeping the task and language fixed. The language axis paraphrases the original instruction without changing the target task, and primarily measures robustness to instruction wording. The object layout axis randomizes the initial positions of the manipulated objects, which probes both robustness to spatial distractors and the policy's ability to follow language under varied object configurations. This axis is therefore partially aligned with the language-following dimension we emphasize. Overall, however, LIBERO-Plus does not evaluate OOD task generalization, since the target task semantics remain unchanged across all seven perturbations.

LIBERO-PRO [4]. To explicitly evaluate language generalization under controlled perturbations, we adopt LIBERO-PRO, which introduces two structured perturbation types on top of LIBERO: (1) Pos, which swaps the initial positions of manipulated objects while holding the task instruction fixed, testing whether the policy genuinely uses language to locate the target rather than relying on positional priors; and (2) Task, which replaces the manipulated object in the instruction with

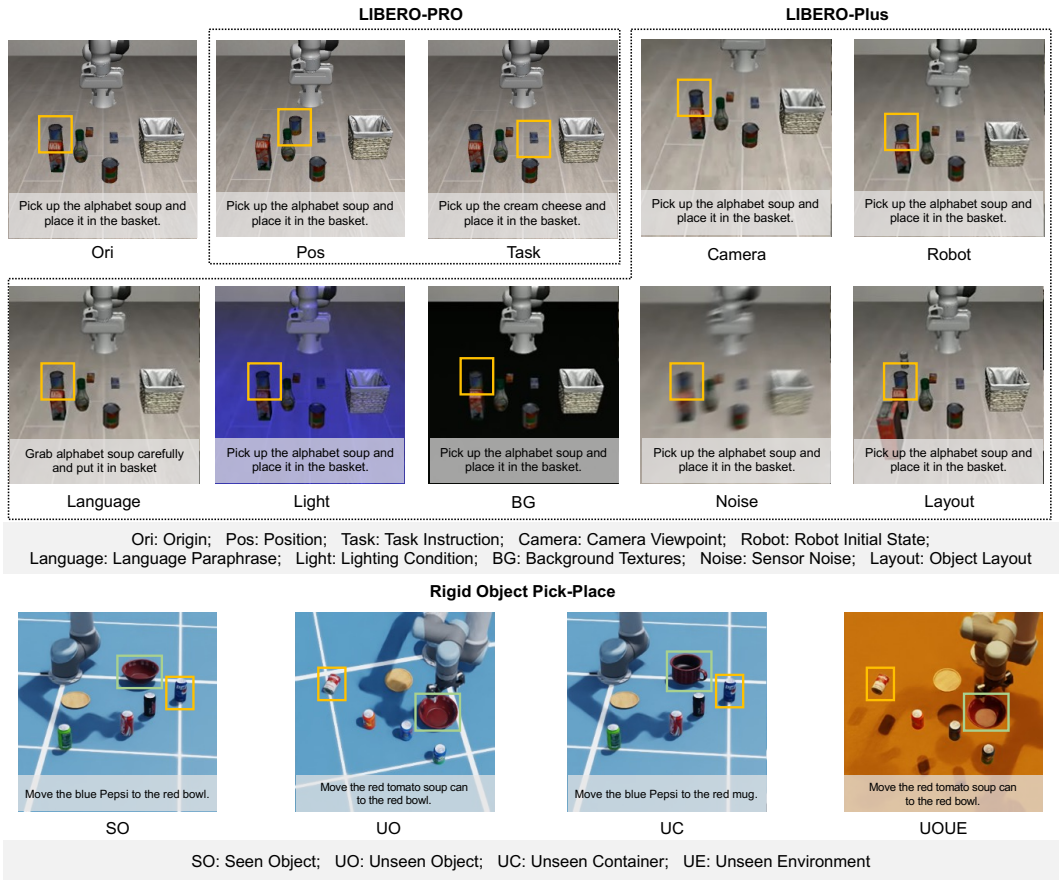

Figure 10: Overview of the simulation benchmarks.

a different object in the same scene, forming an unseen task for OOD language generalization. Notably, simply replicating training trajectories fails under both perturbations, preventing dataset-level shortcuts.

Rigid Object Pick-Place. Following [6], we evaluate on a simulation benchmark built in Isaac-Sim [78] for diverse language-conditioned pick-and-place tasks using a UR5 robot arm. Each scene randomly samples 4 of 25 objects covering diverse colors, shapes, sizes, and categories, with a randomly selected pick target. The place target is randomly chosen between a yellow plate and a red bowl. There are 10k pick-place trajectories across 500 scenarios for multi-task training, each with randomized camera viewpoints and object layouts. There are four evaluation suites: Seen Object (SO), Unseen Object (UO), Unseen Container (UC), and Unseen Object & Unseen Environment (UOUE). SO uses seen pick objects and containers but with randomly sampled instructions, viewpoints, and layouts. UO samples 4 of 10 unseen objects (with unseen colors, shapes, sizes, and categories) while retaining seen containers, testing generalization across object categories. UC uses seen objects with unseen containers, testing placement generalization. UOUE applies unseen backgrounds and lighting on top of UO. Language instructions, camera viewpoints, and object layouts are randomly sampled across all evaluations.

D.2 Results on Original LIBERO

Table 4 reports results on the original LIBERO benchmark. APT achieves an average success rate of 96.1%, with 98.4% on Spatial, 99.4% on Object, 96.4% on Goal, and 90.2% on Long. Since LIBERO primarily measures in-distribution task completion rather than OOD generalization, these results are not intended to claim superiority; rather, they confirm that APT does not sacrifice

Method	Spatial	Object	Goal	Long	Avg
UniVLA [75]	96.5	96.8	95.6	<u>92.0</u>	95.2
WorldVLA [42]	87.6	96.2	83.4	60.0	81.8
π_0 [1]	96.8	<u>98.8</u>	95.8	85.2	94.2
$\pi_{0.5}$ [15]	**98.8**	98.2	**98.0**	**92.4**	**96.9**
APT	<u>98.4</u>	**99.4**	<u>96.4</u>	90.2	<u>96.1</u>

Table 4: Results on LIBERO (success rate %). **Bold**: best. <u>Underline</u>: second best.

Model	Camera	Robot	Lang.	Light	BG	Noise	Layout	Total
π_0 [1]	13.8	6.0	58.8	85.0	81.4	**79.0**	68.9	53.6
π_0-FAST [79]	**65.1**	21.6	61.0	73.2	73.2	74.4	68.8	61.6
OpenVLA [34]	<u>55.6</u>	21.7	**81.0**	<u>92.7</u>	91.0	<u>78.6</u>	68.7	67.9
UniVLA [75]	1.8	46.2	69.6	69.0	81.0	21.2	31.9	42.9
RIPT-VLA [76]	55.2	31.2	<u>77.6</u>	88.4	91.6	73.5	<u>74.2</u>	68.4
X-VLA [77]	23.4	**89.7**	75.7	88.2	**96.0**	62.7	71.8	<u>71.4</u>
APT	31.8	<u>63.1</u>	<u>77.6</u>	**93.6**	<u>92.3</u>	76.0	**80.1**	**71.6**

Table 5: Results on LIBERO-Plus (success rate %) across seven robustness perturbation axes.

task-solving capability. APT remains competitive with $\pi_{0.5}$, and exceeds π_0, UniVLA on training-distribution tasks while substantially improving language generalization on the more challenging benchmarks reported in the main paper (Table 1).

D.3 Results on LIBERO-Plus

Table 5 reports results on LIBERO-Plus across the seven perturbation axes. APT achieves the highest average success rate, slightly above X-VLA, and clearly ahead of π_0 and UniVLA. APT obtains the best result on object layout (80.1%) and lighting (93.6%), and the second-best result on language (77.6%), background (92.3%), and robot initial state (63.1%). The combination results of layout (best) and language (second-best) axes indicate that APT retains reliable instruction following under perturbations that typically expose visual-shortcut policies, which further supports the effectiveness of action expert pretraining.

D.4 Case Studies on Rigid Object Pick-Place

Figure 11 presents qualitative examples from the rigid object pick-place benchmark across all four evaluation splits. In the SO split, both $\pi_{0.5}$ and APT follow the seen instruction reliably and grasp the correct object, indicating that in-distribution language following is largely solved for both methods. In the UO split, $\pi_{0.5}$ grasps a distractor whose color is similar to the language-specified target, while APT correctly grounds the unseen object, showing that action expert pretraining preserves the VLM's semantic grounding for novel object categories. In the UC split, $\pi_{0.5}$ reaches the correct pick object but hesitates during placement and drops the object outside the unseen container, whereas APT produces a precise placement trajectory aligned with the unseen container. In the most challenging UOUE split, $\pi_{0.5}$ frequently misgrasps distractors that share color or shape with the target, while APT grasps the correct object more often. Together, these cases are consistent with the quantitative results in the main paper: action expert pretraining narrows the gap between the VLM's semantic generalization and the policy's grounded action, particularly under compound shifts in object identity and environment.

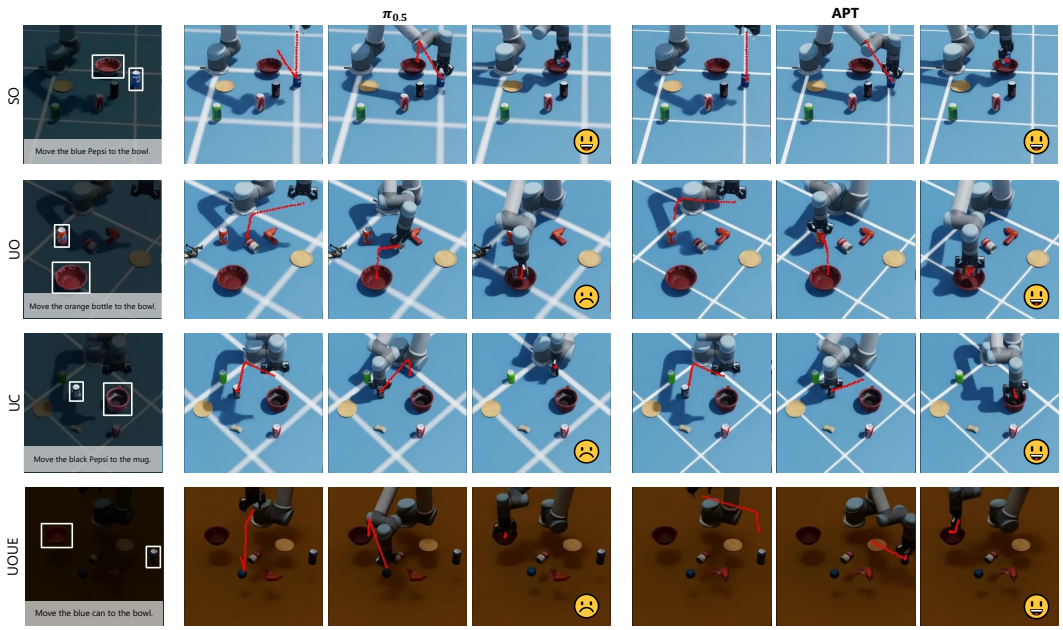

Figure 11: Qualitative case studies on rigid object pick-place across the four evaluation settings. Red dotted lines visualize end-effector trajectories. Annotated failure causes are highlighted.

Figure 12: Real-world platform, containers, objects, and background variations used in our real-world evaluations.

E Real-world Experiment Details

E.1 Real-world Setup

We use the Agilex Cobot platform, equipped with a Piper robot arm and two ORBBEC DaBai cameras for third-view and wrist-view, both capturing RGB-D images at 640×480 resolution. Figure 12 shows the platform together with the tested objects, containers, and background variations.

For each task, we collect 30 tele-operated demonstrations covering a diversity of language instructions and object layouts. We compare APT with $\pi_{0.5}$, which is the strongest baseline in our simulation experiments. In all real-world experiments, $\pi_{0.5}$ is finetuned using joint-space actions, consistent with its original pretraining setup, while APT uses the camera-frame action representation from Section A.

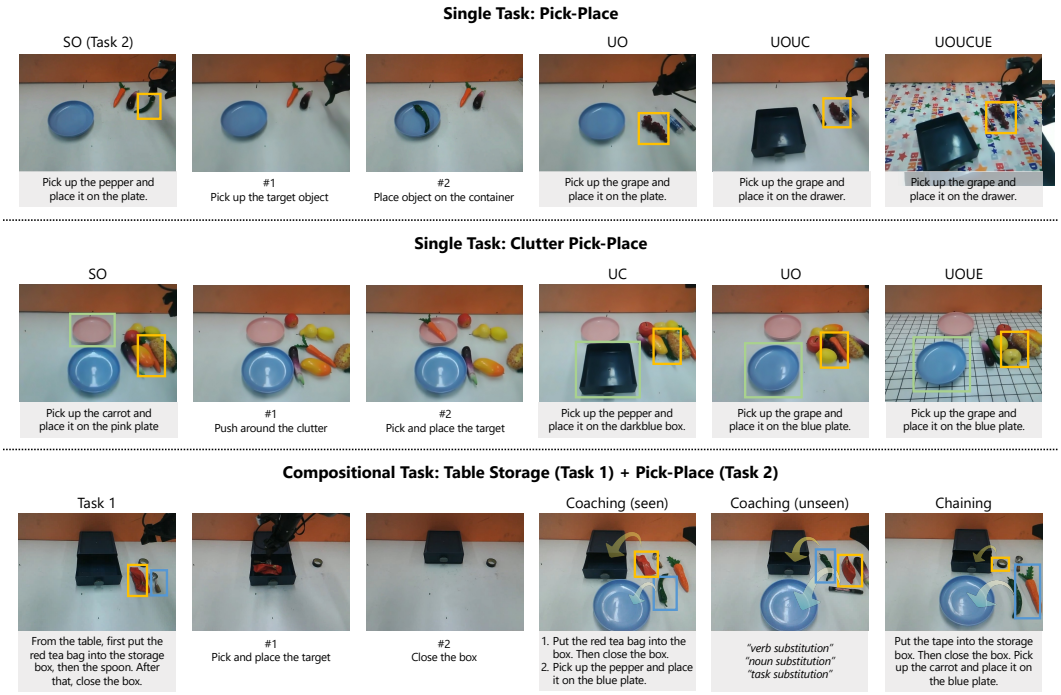

Figure 13: Real-world task and generalization setting overview. Top: single pick-place task with SO, UO, UOUC, and UOUCUE splits. Middle: single clutter pick-place task with SO, UC, UO, and UOUE splits. Bottom: compositional task combining table storage (T1) and pick-place (T2), evaluated under seen and unseen language coaching as well as task chaining.

E.2 Single Task Detailed Settings

Pick-place Task. For this task, the policy is supposed to pick up the specified object into a container. The language instruction for this task is: "Pick up the {object} and place it on the plate". Representative successful rollout and example cases of different generalization dimensions are shown in Figure 13. Each policy is evaluated over a total of 110 trials. In the first 30 trials, we evaluate the pick-place performance with the seen objects and container (SO). For each target object, we conduct 10 trials, where we vary object positions, with other objects serving as distractors. This setup evaluates the language-following capability of policies. The remaining 80 trials evaluate the policy's generalization performance across multiple dimensions. With two unseen objects as target items, we gradually increase the level of generalization: unseen object with seen container (UO), unseen object with unseen container (UOUC), and unseen object with unseen container under two distinct unseen backgrounds (UOUCUE). Each configuration is tested 10 times.

Clutter Pick-place Task. In this task, the robot cannot directly grasp the object, as it is tightly surrounded by distractors. Therefore, it must first perform push actions to create sufficient clearance for gripper insertion before executing the pick-place actions [80, 65]. This requires long-horizon planning across push, pick, and place sub-tasks. Additionally, the task demands identifying the target object among multiple objects and selecting the correct container among several options, emphasizing semantic grounding. The language prompt for this task is: "Pick up the {object} and place it on the {container}," where the {object} denotes the target object to pick, and {container} specifies the placement target. Representative successful rollout and example cases of different generalization dimensions are shown in Figure 13. Each policy is evaluated over 80 trials in total. In the first 30 trials, the target objects and distractors are seen during training. For the rest 50 trials, we test the generalization of: seen object with unseen container (UC), unseen object with seen container (UO), and unseen object with unseen background (UOUE). Each configuration is tested 10 times.

Method	Red Tea Bag	Spoon	Box	Avg.
$\pi_{0.5}$	19/20	19/20	16/20	16/20
APT	20/20	18/20	17/20	17/20

Table 6: Real-world table storage results (successes / trials).

E.3 Compositional Task Detailed Settings

We evaluate compositional instruction following and generalization across two real-world tasks:

- **Task 1: Table Storage (T1).** The robot picks specified items from the table, places them sequentially in a storage box, and finally pushes the box closed. The training instruction template is "From the table, put the {objects} into the storage box. Then close the box.", where {objects} are drawn from a fixed set of seen items including a marker, a spoon, a tape, and a red tea bag.
- **Task 2: Pick-Place (T2).** Same as the single pick-place task in the previous subsection, using only seen objects and seen containers.

We first test **Task 1** of seen prompt "From the table, put the red tea bag, and the spoon into the storage box. Finally, close the box." Combining results in Table 6 and Table 3 (Pick-Place SO), we can confirm that the two policies have sufficient single-task competence. For the compositional evaluation, all objects are drawn from the seen training set; only the instruction structure or wording, and the object layouts are varied.

We design three evaluation protocols for 2 compositional tasks of increasing difficulty:

Seen Language Coaching. Instructions for the two tasks are issued sequentially as separate prompts using the training-distribution wording. For example, the user first prompts T1 with "From the table, put the red tea bag into the storage box. Then close the box.", and after completion, prompts T2 with "Pick up the pepper and place it on the blue plate." We test 20 trials in total, 10 with T1 issued before T2, and 10 with T2 issued before T1.

Unseen Language Coaching. The same two-prompt protocol, but each prompt is rewritten along one of three OOD axes: (i) *Verb substitution*, replacing the training verb while keeping the task semantics, *e.g.*, "Place the red tea bag into the box. Then push the storage box closed."; (ii) *Noun substitution*, replacing the object or container noun with a synonym, *e.g.*, "Put the red tea pouch into the bin. Then close the bin."; (iii) *Task substitution*, swapping the manipulated objects between the two tasks, *e.g.*, T1 becomes "Put the pepper into the box. Then close the storage box." and T2 becomes "Pick up the red tea bag and place it on the blue plate." These perturbations probe the policy's ability to compose seen objects into novel task assignments.

Task Chaining. The two tasks are combined into a single concatenated prompt, for instance "Put the red tea bag into the box. Then close the storage box. Pick up the pepper and place it on the blue plate.". The policy must parse and execute the multi-task instruction without an explicit segmentation signal.

E.4 Detailed Results

Pick-Place Task. Table 7 shows per-object results on the single pick-place task. In the SO split, both methods reliably handle the carrot and eggplant. The gap widens on unseen objects: in UO, $\pi_{0.5}$ achieves only $5/10$ and $6/10$ on grape and bottle, whereas APT reaches $9/10$ and $8/10$. Under UOUC and UOUCUE, $\pi_{0.5}$'s success rates further degrade, particularly on the bottle ($6/20$ in UOUCUE), where its narrow shape under unseen backgrounds frequently leads to misgrounding or misgrasping. APT maintains a clear margin across all splits ($15/20$ and $13/20$ on grape and bottle in UOUCUE), indicating that the pretrained VA prior together with gated language fusion is robust to compound shifts in object identity, container, and environment.

Clutter Pick-Place Task. Table 8 shows per-object results on the clutter pick-place task. APT outperforms $\pi_{0.5}$ on almost every object and every split, with the largest gap on the seen-object pepper

Method	SO			UO		UOUC		UOUCUE		Avg.
	Pepper	Carrot	Eggplant	Grape	Bottle	Grape	Bottle	Grape	Bottle	
$\pi_{0.5}$ [15]	7/10	**10/10**	**10/10**	5/10	6/10	4/10	5/10	10/20	6/20	63/110
APT	**9/10**	**10/10**	**10/10**	**9/10**	**8/10**	**9/10**	**7/10**	**15/20**	**13/20**	**90/110**

Table 7: Real-world pick-place per-object results (successes / trials).

Method	SO			UC			UO	UOUE	Avg.
	Pepper	Carrot	Eggplant	Pepper	Carrot	Eggplant	Grape	Grape	
$\pi_{0.5}$ [15]	4/10	7/10	7/10	4/10	**8/10**	6/10	4/10	3/10	43/80
APT	**9/10**	**8/10**	**8/10**	**7/10**	**8/10**	**7/10**	**7/10**	**6/10**	**60/80**

Table 8: Real-world clutter pick-place per-object results (successes / trials).

case in SO ($9/10$ vs. $4/10$) and on the unseen-object grape under unseen backgrounds in UOUE ($6/10$ vs. $3/10$). The pepper case is notable: although the object is seen, $\pi_{0.5}$ often hesitates during the push-to-grasp transition when the pepper shares a similar height with surrounding distractors, a behavior that the pretrained VA prior largely avoids. Across UC, UO, and UOUE, $\pi_{0.5}$ is most prone to misgrounding when the target shares color with distractors (grape vs. eggplant), while APT maintains correct grounding and produces a coherent push-pick-place trajectory.

E.5 Case Studies

Figure 14 presents additional real-world cases across the single-task and compositional-task settings.

Grasping on Unseen Objects (a, b). On UOUC and UOUCUE pick-place cases with the unseen grape, $\pi_{0.5}$ either hesitates without closing the gripper or slips off on contact, while APT grasps the target on the first attempt. We also observe that when a non-target object is initially placed inside the container (a), $\pi_{0.5}$ often remains completely still, likely misinterpreting the task as already completed.

Sub-task Transition in Clutter (c, d). The clutter pick-place task requires a push-grasp-place sequence, and $\pi_{0.5}$ fails in two distinct modes at this transition. In (c), the target pepper is still partially surrounded by distractors and $\pi_{0.5}$ keeps issuing push actions without creating enough space for grasping. In (d), the policy successfully declutters and the target grape is fully exposed, yet it continues to push rather than switching to grasp, suggesting that the sub-task boundary is missed even when the perceptual condition for grasping is clearly met. In contrast, APT transitions from push to grasp at the appropriate time in both cases. These failures show that the bottleneck is not perceptual access to the target, but the policy's ability to commit to the next sub-task once the current one's precondition is satisfied, which is harder to learn when the action expert is jointly trained from random initialization rather than pretrained as a structured prior.

Compositional Execution (e, f). On chained instructions combining T1 (storage) and T2 (pick-place), $\pi_{0.5}$ fails in two distinct modes. In (e), it completes T1 correctly but during T2 places the wrong object into the plate, indicating that the chained instruction affects its language grounding. In (f), it executes T1, but places both the T1 and T2 target objects into the box and then stops. This suggests that the concatenated prompt is treated as a single task whose completion is judged by the T1 sub-goal alone. In contrast, APT executes both sub-tasks in the correct order and with the correct targets. These two failure modes show that the gap is not about understanding individual instructions, since each sub-task is reliable in isolation, but about parsing and switching between sub-instructions within a single prompt. This is consistent with the compositional-task results in Figure 6 and indicates that chained multi-task prompts are where the language-grounding gap is most visible.

Figure 14: More real-world cases comparing $\pi_{0.5}$ and APT. (a, b) Grasping on unseen objects. (c, d) Sub-task transition in clutter pick-place. (e, f) Compositional task execution. Red dotted lines visualize end-effector trajectories; annotated text highlights the failure cause for $\pi_{0.5}$.

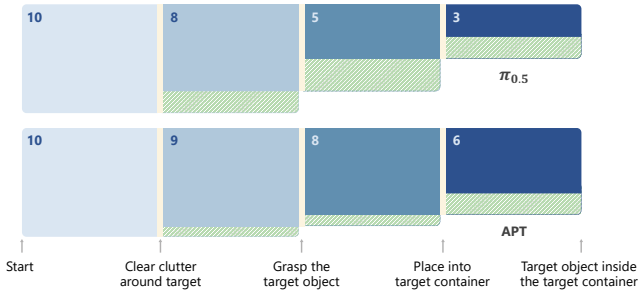

Figure 15: Failure breakdown on the UOUE setting of the clutter pick-place task. The Sankey diagram tracks the flow of success (solid) and failure (shadow) across the start, clear clutter, grasp, and place sub-tasks.

E.6 Failure Studies

Failure Analysis. Figure 15 visualizes the failure breakdown on the UOUE setting of the clutter pick-place tasks. The four checkpoints are: clear clutter around the target, grasp the target object, place into the target container, and target object inside the target container. Starting from 10 trials, $\pi_{0.5}$ retains $10 \rightarrow 8 \rightarrow 5 \rightarrow 3 \rightarrow 3$ rollouts across the four checkpoints, while APT retains $10 \rightarrow 9 \rightarrow 8 \rightarrow 6 \rightarrow 6$. The largest gap appears at the push-to-grasp transition. Common causes for $\pi_{0.5}$ are: (i) confusing the grape with an eggplant when their colors overlap, and (ii) continuing to push after the target is already exposed, both of which point to weak target-object grounding under unseen visual conditions. APT retains correct grounding in most rollouts; its remaining failures are concentrated at the contact and placement stages, which are physical execution issues rather than language-following issues.

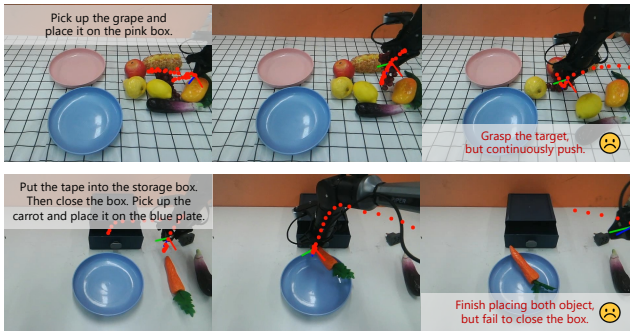

Figure 16: Typical failure cases of APT. Top: continued pushing after successfully grasping the target on clutter pick-place. Bottom: skipped "close the box" action at the end of T1 on compositional task chaining.

Failure Cases of APT. We show the typical failure modes of APT in Figure 16. On clutter pick-place, APT occasionally exhibits the same push-after-grasp behavior described earlier for $\pi_{0.5}$, where the policy successfully grasps the target but continues to issue push actions instead of transitioning to placement. The frequency of this failure is much lower than for $\pi_{0.5}$, but it has not been fully eliminated. On compositional task chaining, APT occasionally over-attends to the pick-place sub-task and skips the "close the box" action at the end of T1, completing the placement portion of the chained prompt while leaving the box open. Both failures indicate a remaining gap for APT on detecting sub-task termination.