What Can RL Bring to VLA Generalization? An Empirical Study

Jijia Liu[1*] **Feng Gao**[2*] **Bingwen Wei**[1]
Xinlei Chen[1] **Qingmin Liao**[1] **Yi Wu**[2] **Chao Yu**[1†] **Yu Wang**[3†]

[1]Shenzhen International Graduate School, Tsinghua University
[2]Institute for Interdisciplinary Information Sciences, Tsinghua University
[3]Department of Electronic Engineering, Tsinghua University

Figure 1: Overview of our study for evaluating how RL enhances VLA generalization in terms of *Vision*, *Semantics*, and *Execution*: in out-of-distribution tests, RL yields substantial gains in *Execution*, moderate improvements in *Semantics*, and performance on par with SFT for *Vision*.

Abstract

Large Vision-Language Action (VLA) models have shown significant potential for embodied AI. However, their predominant training via supervised fine-tuning (SFT) limits generalization due to susceptibility to compounding errors under distribution shifts. Reinforcement learning (RL) offers a path to overcome these limitations by optimizing for task objectives via trial-and-error, yet a systematic understanding of its specific generalization benefits for VLAs compared to SFT is lacking. To address this, our study introduces a comprehensive benchmark for evaluating VLA generalization and systematically investigates the impact of

*Equal contribution.
†Corresponding authors: {yuchao, yu-wang}@mail.tsinghua.edu.cn

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

RL fine-tuning across diverse visual, semantic, and execution dimensions. Our extensive experiments reveal that RL fine-tuning, particularly with PPO, significantly enhances generalization in semantic understanding and execution robustness over SFT, while maintaining comparable visual robustness. We identify PPO as a more effective RL algorithm for VLAs than LLM-derived methods like DPO and GRPO. We also develop a simple recipe for efficient PPO training on VLAs, and demonstrate its practical utility for improving VLA generalization. The project page is at `https://rlvla.github.io`.

1 Introduction

Vision-Language-Action (VLA) models represent an emerging class of foundation models [Ma et al., 2024, Firoozi et al., 2023] that unify perception, language understanding, and embodied control. By leveraging vision-language models pretrained on internet-scale data and further training on large, heterogeneous robot demonstration datasets [Collaboration et al., 2023, Khazatsky et al., 2024], VLAs can interpret sensor observations and natural language instructions to directly map them to robot actions. This paradigm has demonstrated promising generalization across diverse tasks—including single-arm, bimanual, and mobile manipulation [Team et al., 2024, Kim et al., 2024, Liu et al., 2024, Wen et al., 2025], navigation [Shah et al., 2022], and even complex long-horizon activities like kitchen or bedroom cleaning in unseen scenarios [Black et al., 2024, Intelligence et al., 2025].

Despite this promise, VLA model training predominantly relies on supervised fine-tuning (SFT) through behavioral cloning of demonstration labels [Kim et al., 2025]. This approach, whether in pretraining or few-shot adaptation, is inherently susceptible to compounding errors under distribution shift: minor deviations from expert trajectories can accumulate, steering the policy into unfamiliar states and undermining robust performance [Ross and Bagnell, 2010, De Haan et al., 2019, Belkhale et al., 2023, Foster et al., 2024]. This mismatch between training and testing distributions fundamentally limits robustness, with research highlighting issues like quadratic regret growth relative to the task horizon under such conditions [Ross and Bagnell, 2010].

In contrast, reinforcement learning (RL) offers a paradigm that directly optimizes cumulative task rewards through trial-and-error, enabling policies to explore beyond narrow expert data and learn corrective behaviors. Crucially, in the broader foundation model landscape, particularly for Large Language Models (LLMs) and Vision-Language Models (VLMs), recent studies have underscored RL's advantages for generalization [Ouyang et al., 2022, Zhai et al., 2024, Huang et al., 2025]. Compelling evidence suggests that while SFT tends to memorize training data, RL fine-tuning can lead to substantially better out-of-distribution performance and unlock greater reasoning capabilities [Chu et al., 2025, Huang et al., 2025, Ma et al., 2025]. Drawing from RL's established success in robotics and these encouraging results from other large-scale models, RL fine-tuning is increasingly being applied to VLAs [Collaboration et al., 2023, Walke et al., 2023, Khazatsky et al., 2024], with approaches ranging from incorporating human feedback [Chen et al., 2025] to using offline RL updates [Zhang et al., 2024b] or algorithms like PPO [Schulman et al., 2017], sometimes in multi-stage processes with imitation learning [Guo et al., 2025b].

However, despite these pioneering efforts, a systematic understanding of what specific generalization benefits RL fine-tuning confers upon VLAs, especially in direct comparison to SFT baselines, and how their respective strengths differ, remains insufficiently developed [Hu et al., 2024, Mark et al., 2024]. For instance, while recent work like FLaRe [Hu et al., 2024] demonstrated PPO's utility for fine-tuning VLAs, a comprehensive analysis of the resulting model's generalization capabilities was not its primary focus.

This paper aims to address this critical gap. We undertake a systematic study to dissect the generalization properties of VLAs fine-tuned with RL versus SFT.

> *Specifically, we empirically investigate: What unique benefits can RL bring to VLA generalization compared to supervised fine-tuning?*

To answer this, we center our investigations on the representative *pick-and-place* task. On this task, we first evaluate mainstream RL algorithms for large-scale models (PPO [Schulman et al., 2017, Ouyang et al., 2022], DPO [Rafailov et al., 2023, Zhang et al., 2024b], GRPO [Shao et al., 2024, Guo et al., 2025a]) to identify effective VLA fine-tuning strategies. We then conduct a broad

comparative evaluation, also on pick-and-place, pinpointing where RL fine-tuning outperforms SFT. As previewed in Fig. 1, this comprehensive evaluation rigorously examines generalization across three key dimensions: 1) *Vision*: challenging generalization with novel backgrounds (unseen tables) and by overlaying unseen textures on the foreground or the entire image. 2) *Semantics*: testing understanding via unseen objects, novel receptacles, and varied instruction phrasings to probe language sensitivity and object recognition. 3) *Execution*: probing robustness by varying initial robot states, object/receptacle positions, and introducing dynamic disturbances like random object reposition within episodes.

Through extensive experiments and deep analyses, we:

- Establish a rigorous and challenging benchmark to evaluate how VLA fine-tuning methods affect generalization across diverse visual, semantic, and execution dimensions.
- Identify PPO as the preferred RL algorithm for VLA fine-tuning over GRPO and DPO, while also discuss key challenges in adapting these RL algorithms from LLM/VLM paradigms to the distinct requirements of VLAs.
- Develop an efficient PPO-based VLA fine-tuning recipe, enabled by a shared actor-critic backbone, VLA model warm-up, and minimal PPO training epochs.
- Demonstrate RL's superior generalization over SFT in semantic understanding and embodied execution for VLAs, while maintaining comparable visual robustness.

2 Related works

2.1 Vision-Language-Action (VLA) models

In robotic tasks that utilize both vision and language inputs, recent research demonstrates that leveraging knowledge from pre-trained LLMs or VLMs [Chen et al., 2023, Driess et al., 2023, Karamcheti et al., 2024, Beyer et al., 2024] leads to better policy generalization compared to smaller models [Zhao et al., 2023, Chi et al., 2023]. By harnessing the capabilities of these foundation models and utilizing large-scale robotic datasets, VLA models [Brohan et al., 2023b,a, Team et al., 2024, Kim et al., 2024, Liu et al., 2024, Wu et al., 2023, Cheang et al., 2024, Black et al., 2024, Intelligence et al., 2025] provide a strong basis for embodied agent. Our work builds upon VLA and seeks to further improve its generalization abilities.

2.2 RL fine-tuning for large-scale (Vision-)Language Models

Fine-tuning large language models has shown promising results [Stiennon et al., 2020, Ouyang et al., 2022, Achiam et al., 2023], especially when using high-quality supervised data [Zhang et al., 2024a, Zhu et al., 2023, Zhang et al., 2023b]. Some work aims to align LLMs with human preferences for helpfulness and safety [Kaufmann et al., 2023, Bai et al., 2022], using online RL algorithms [Schulman et al., 2017, Ramamurthy et al., 2023, Zheng et al., 2023] or offline updates [Rafailov et al., 2023, Ivison et al., 2024, Xu et al., 2024]. Other studies have shown that online RL fine-tuning on math and coding tasks [Shao et al., 2024, Guo et al., 2025a] can significantly improve the reasoning abilities of LLMs [Guo et al., 2025a, Jaech et al., 2024, Team et al., 2025]. Although RL has been extensively explored in LLMs [Chu et al., 2025], its generalization potential for similar-scale VLA models remains underexplored.

2.3 Fine-tuning policies for robotic tasks

VLA fine-tuning largely relies on supervised data [Kim et al., 2025, Liu et al., 2025, Zhao et al., 2025, Zheng et al., 2024]. However, the scarcity of extensive and high-quality data often hampers the ability of VLAs to generalize to unseen scenarios or perturbations [Collaboration et al., 2023, Walke et al., 2023, Khazatsky et al., 2024]. To move beyond imitation and enable reward-driven learning, RL fine-tuning has been explored in various settings, from small-scale models [Ren et al., 2024] to large models with human interventions [Chen et al., 2025], or through offline updates [Zhang et al., 2024b]. Guo et al. [2025b] proposed using PPO [Schulman et al., 2017] to fine-tune VLA models, incorporating imitation learning in a two-stage process. Despite these efforts, the generalization ability of RL fine-tuning in VLA models remains insufficiently studied [Hu et al., 2024, Mark et al., 2024]. Moreover, we find that VLAs can be directly fine-tuned with online PPO without additional training stages or imitation learning, underscoring RL's scalability for robot tasks.

3 Preliminaries

3.1 Problem formulation

We model each language-conditioned robotic task $T \in \mathcal{T}$ as a Partially Observable Markov Decision Process (POMDP), defined as a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \mathcal{O}, \mathcal{L}, P(s_0), \gamma)$. Here, the state space $\mathcal{S}$ includes robot and environment states, $\mathcal{A}$ is the action space of control commands, and $\mathcal{O}$ the observation space of sensor outputs. Transitions follow $s_{t+1} \sim \mathcal{P}(\cdot \mid s_t, a_t)$, and γ is the discount factor. Each task T has natural language instructions $\mathcal{L}_T \subset \mathcal{L}$ and a reward $R(s, l) \in \mathbb{R}$ when state s completes task stages or satisfies instruction l. Episodes start with an instruction $l \sim \mathcal{L}_T$ and initial state $s_0 \sim P(s_0)$, which defines the robot's initial pose and environment. The policy π_θ uses the last H observations $o_{t-H+1:t}$ and the language instruction l to output actions $a_t \sim \pi_\theta(a_t \mid o_{t-H+1:t}, l)$, constructing trajectory $\tau = (o_0, a_0, \dots)$.

Supervised fine-tuning (SFT). SFT learns from an expert-collected demonstration dataset $\mathcal{D}_T = \{(\boldsymbol{\tau}^{(i)}, l^{(i)})\}_{i=1}^{N}$, where each trajectory is $\boldsymbol{\tau}^{(i)} = (o_0^{(i)}, a_0^{(i)}, \dots, o_{K_i-1}^{(i)}, a_{K_i-1}^{(i)})$ and N is the total number of trajectories. The VLA policy π_θ minimizes:

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = \sum_{(\tau^{(i)}, l^{(i)}) \in \mathcal{D}_T} \sum_{t=0}^{K_i-1} \ell_{\mathrm{SFT}}\big(\hat{a}_t^{(i)}, a_t^{(i)}\big), \quad \hat{a}_t^{(i)} = \pi_\theta\big(o_{t-H+1:t}^{(i)}, l^{(i)}\big),$$

where ℓ_{SFT} is a loss function (e.g., next-token prediction, L_1 regression, diffusion) based on VLA architecture and action representation [Kim et al., 2025].

RL fine-tuning. RL fine-tuning maximizes scalar rewards or preference signals through direct interactions with environments. With rewards, it minimizes the negative discounted return over an episode of length M:

$$\mathcal{L}_{\mathrm{RL}}(\theta) = -\mathbb{E}_{\boldsymbol{\tau} \sim \pi_\theta}\Big[\sum_{t=0}^{M-1} \gamma^t R(s_t, l)\Big],$$

often via a policy-gradient surrogate:

$$\mathcal{L}_{\mathrm{PG}}(\theta) = -\mathbb{E}_{\tau \sim \pi_\theta}\Big[\sum_{t=0}^{M-1} A_t^\pi \log \pi_\theta\big(a_t \mid o_{t-H+1:t}, l\big)\Big],$$

using an advantage estimator A_t^π. With preferences, methods like Direct Preference Optimisation (DPO) [Rafailov et al., 2023] use pairwise/ranked feedback to optimize π_θ for preferred trajectories.

3.2 Vision-Language-Action models

We base our study on OpenVLA [Kim et al., 2024] (Fig. 2), an open-source model that achieves state-of-the-art performance by pairing a fused visual encoder of SigLIP [Zhai et al., 2023] and DINOv2 [Oquab et al., 2023] with a Llama-2 7B language backbone [Touvron et al., 2023], built on the Prismatic VLM [Karamcheti et al., 2024]. At each time step the policy receives a single RGB image o_t and an instruction l, i.e., the history length $H = 1$. The image is embedded into visual tokens, the instruction is tokenised with Llama 2's tokenizer, and the resulting token sequence is fed to the causal transformer decoder.

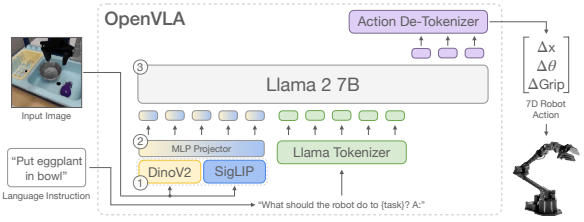
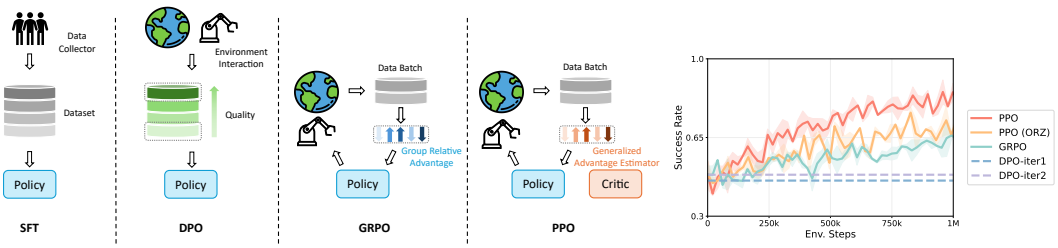


Figure 2: Architecture of the OpenVLA model [Kim et al., 2024], reproduced from the official open-source code and checkpoints.

OpenVLA follows the RT-2 discretisation recipe [Brohan et al., 2023a]: each scalar in the continuous command $a_t \in \mathbb{R}^{d_a}$ is mapped to one of 256 bins that uniformly split the range between its 1^{st} and 99^{th} percentiles in the training set, producing an action–token vector $\mathbf{u}_t \in \{0, \dots, 255\}^{d_a}$. These action tokens overwrite the 256 least-used tokens in the Llama-2 vocabulary, so the language model can emit them directly. The network is then trained with the usual next-token cross-entropy objective, and the cross-entropy loss is computed solely on the predicted action tokens.



(a) Overview of different fine-tuning methods (b) Performance comparison

Figure 3: Overview of VLA fine-tuning methods: SFT learns from offline demonstrations, whereas DPO, GRPO, and PPO use RL-based updates—employing preference alignment, group-relative advantage estimation, and standard actor-critic PPO with generalized advantage estimation (GAE), respectively; and performance comparison between different RL fine-tuning algorithms.

4 Effective RL fine-tuning of VLA models

We begin our study by exploring how to fine-tune VLA models with reinforcement learning. As these models inherit a similar scale and structure from their pre-trained LLM/VLM backbones, we first test whether common RL algorithms for large-scale models like LLMs/VLMs transfer effectively to VLAs. In Sec. 4.1, we will show that Proximal Policy Optimization (PPO) [Schulman et al., 2017] consistently improves performance, whereas Direct Preference Optimization (DPO) [Rafailov et al., 2023, Zhang et al., 2024b] and Group Relative Policy Optimization (GRPO) [Shao et al., 2024] struggle to learn something for POMDP robotic tasks. Guided by this finding, we then ablate key design choices within PPO to distill an effective fine-tuning recipe for VLA models in Sec. 4.2.

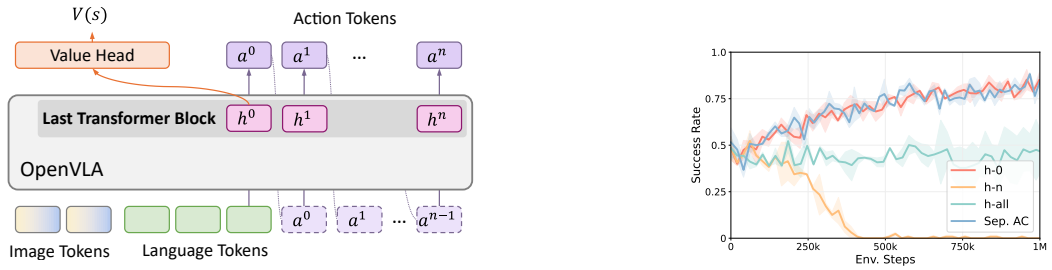
4.1 RL algorithms: PPO, GRPO, DPO

Here, we consider three representative RL algorithms: PPO [Schulman et al., 2017], GRPO [Shao et al., 2024] and DPO [Rafailov et al., 2023, Zhang et al., 2024b]. All models are fine-tuned using Low-Rank Adaptation (LoRA) [Hu et al., 2022] with rank = 32.

- **PPO** performs on-policy updates, computing policy gradients with clipped importance ratios and Generalised Advantage Estimation (GAE) from freshly collected rollouts. Recent work Open-Reasoner-Zero (ORZ) by [Hu et al., 2025] shows that disabling GAE (setting $\gamma = 1$ and $\lambda = 1$) can boost performance when fine-tuning large models. Accordingly, we evaluate both the standard algorithm (**PPO**) and this variant (**PPO-ORZ**) in our experiments.
- **GRPO** estimates the baseline using a group of samples, allowing for direct computation of advantages without explicit value estimation. While natural language generation and robotic multi-step MDP tasks are inherently different, we follow the GRPO setup [Shao et al., 2024] by sampling trajectories from a common initial state, with each group containing 8 trajectories.
- **DPO** exploits offline datasets annotated with pairwise preferences. In robotic tasks, however, obtaining contrastive trajectories from the same initial state is difficult. Following [Zhang et al., 2024b], we infer trajectory-level preferences from reward signals. To ensure a fair comparison, we employ only the stage-based sparse reward described in Sec. 5.1, rather than the richer reward scheme used in the original implementation by [Zhang et al., 2024b].

These algorithms are illustrated in Fig. 3a, with further details and implementation specifics provided in Sec. A.1. We evaluate the performance on the *pick-and-place* task described in Sec. 5.1.

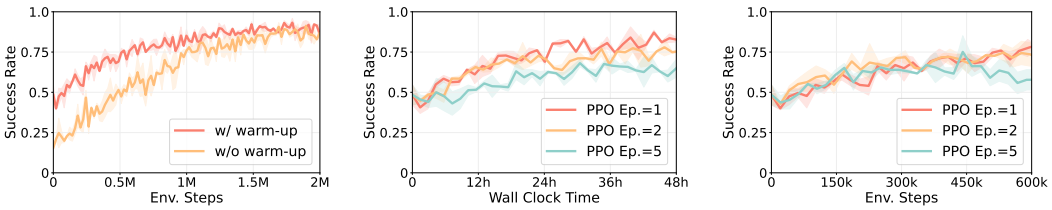
The results are presented in Fig. 3b, with each experiment conducted using two different random seeds. Our findings indicate that PPO consistently outperforms GRPO. We attribute this to fundamental differences in environment dynamics between natural language tasks and robotic tasks [Li et al., 2024b]. We hypothesis that, in robotic tasks' POMDP, each action sequentially alters the environment state in a non-stationary manner, which may destabilizes GRPO's advantage estimates. Additionally, PPO surpasses DPO performance. We hypothesize that this is due to the sparse reward structure, which makes it challenging to distinguish between the quality of different trajectories, as well as significant distribution shifts between the offline dataset and interactive execution [Prudencio et al., 2023].



(a) Model architecture of actor-critic OpenVLA

(b) Performance comparison

Figure 4: PPO with shared actor-critic backbone, where $V(s)$ is predicted by a three-layer MLP. We compare the performance of different critic designs, as well as a separate actor-critic architecture.



(a) VLA warm-up

(b) PPO epoch (wall-clock time)

(c) PPO epoch (env. steps)

Figure 5: Performance comparison to verify our efficient training designs.

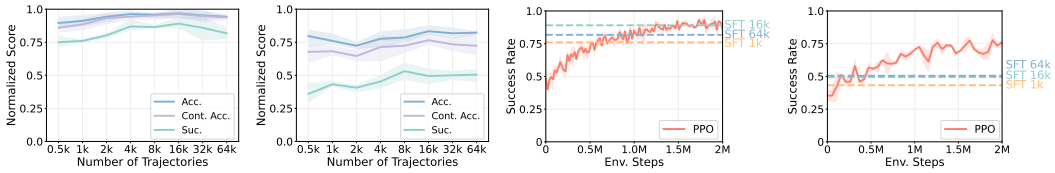
4.2 Design factors of PPO

We employ several key design choices to enable PPO to work effectively with OpenVLA. With these designs, our main experiments require about 42 hours on a single NVIDIA A100 GPU to converge. In the following, we introduce these design choices and present ablation studies to assess their impact. We conduct each experiment with two different random seeds.

Shared actor-critic backbone. Since PPO—and most modern RL methods—use an actor–critic formulation [Konda and Tsitsiklis, 1999], we treat the pretrained VLA policy as the *actor* and attach a lightweight *critic* that estimates the state value $V(s)$. To keep the architecture compact, the actor and critic share the entire Transformer backbone; a three-layer MLP value head (see Fig. 4a) takes the hidden vector h^0 produced by the final Transformer block [Vaswani et al., 2017] at the first action-token position and regresses it to a scalar value. To verify its efficacy, we evaluated several critic designs. Feeding the value head with the first action-token embedding h^0 produced the highest and most stable returns, outperforming the last-token input h^n and the concatenation $[h^0, \ldots, h^n]$ (see Fig. 4b). A critic that uses its own Transformer backbone achieved comparable rewards but trained $35\,\%$ slower and consumed $83\,\%$ more VRAM (81.3 GB vs. 44.4 GB). These results confirm that a shared backbone with h^0 is the most efficient choice.

VLA warm-up. We run all experiments with the official OpenVLA checkpoint [Kim et al., 2024] pretrained on the OXE dataset [Collaboration et al., 2023]. Due to the poor performance of this checkpoint on our benchmark, we *warm it up* with 140 demonstration trajectories gathered by Octo-Small [Team et al., 2024] and a motion planner (see Sec. A.2). As shown in Fig. 5a, the warm-up model reaches convergence with roughly $50\,\%$ fewer environment steps, while both initialisations attain comparable asymptotic returns given sufficient interaction. Unless stated otherwise, all subsequent experiments use the warmed-up OpenVLA model for initialization.

Minimal PPO epoch. We show that the update–to–data ratio is a key factor for efficient fine-tuning. In PPO this ratio is set by the *epoch* hyper-parameter, which specifies how many gradient passes each batch receives. Fig. 5b and Fig. 5c reveal that increasing the PPO *epoch* beyond one yields no gains in return or sample efficiency, yet lengthens wall-clock time almost linearly. Accordingly, we fix epoch $= 1$ in all remaining experiments, achieving the fastest training without sacrificing performance.



(a) SFT@Training (b) SFT@OOD Eval (c) RL@Training (d) RL@OOD Eval

Figure 6: Effect of data scale on SFT performance. The reported metrics include grasp accuracy, continuous grasp accuracy, and task success rate.

5 Evaluating fine-tuning methods on VLA generalization

5.1 Environments and datasets

To thoroughly investigate the generalization capabilities of VLA, we focus on a typical *pick-and-place* task [Kim et al., 2025, Li et al., 2024a], where the agent is instructed to place an object from the table into a receptacle. Inspired by prior works [Fan et al., 2025, Stone et al., 2023] and the concept of Vision-Language-Action models, we define three dimensions of generalization: *Vision*, *Semantics (Language)*, and *Execution (Action)*, as illustrated in Fig. 1.

- *Vision*: We include both foreground (*Dynamic Textures*) and background (*Unseen Table*) changes, as well as image-level *Dynamic Noise*, applied with either *weak* or *strong* intensity.
- *Semantics*: We consider previously unseen variations in *Objects*, *Receptacles*, and *Instruction Phrasings*. Additionally, we design new tasks where one of two objects must be picked (*Multi-Object*), using either seen or unseen object sets. We also include a task with a *Distractive Receptacle*, and a task requiring the object to be placed in one of two unseen receptacles (*Multi-Receptacle*).
- *Execution*: We investigate changes in the initial positions of both the object and the receptacle, as well as the robot initial pose. Furthermore, we introduce a new scenario in which the object's position changes during the task (*Mid-Episode Object Reposition*).

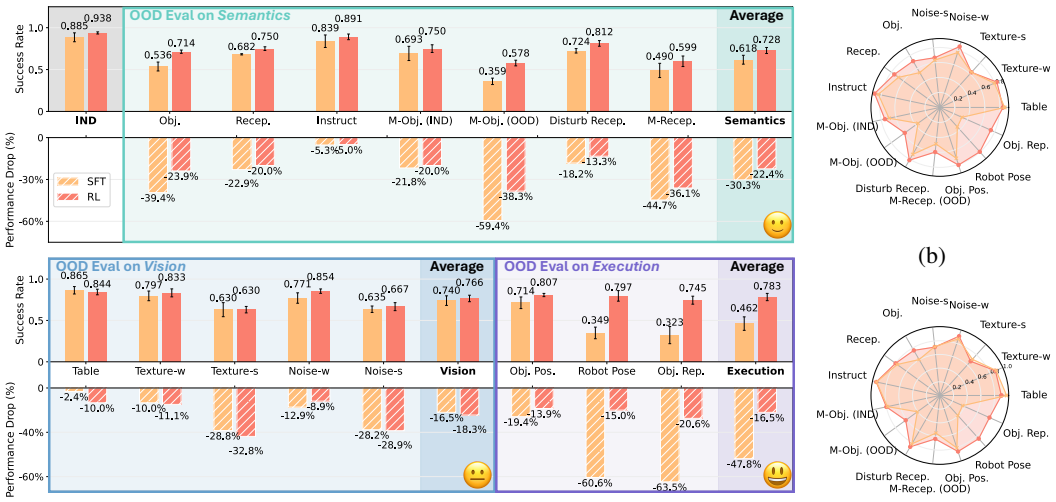
To probe generalisation, we randomise each task along three axes during training: *Vision* (16 tables), *Semantics* (16 objects), and *Execution* (perturbations of object and receptacle poses). At test time we hold at least one of these factors out of distribution, introducing nine novel objects, sixteen unseen receptacles, five new table surroudings, and sixteen distractor textures. Assets are drawn from Objaverse [Deitke et al., 2023] and other public sources; additional table appearances are synthesised with Stable Diffusion [Rombach et al., 2022] and ControlNet [Zhang et al., 2023a], ensuring surface variation without changing the table's global pose. More details of asset acquisition and task specification are provided in Secs. A.3 and A.4. Built upon these assets, our tasks run in ManiSkill [Tao et al., 2024] with an 8-DoF WidowX-250S arm. At every step the agent observes a 640×480 RGB frame and a natural-language instruction, and outputs a Cartesian end-effector delta plus a binary gripper signal. Rewards are sparse: 0.1 for grasping and continuously holding the correct object, and 1.0 for placing it successfully. For supervised fine-tuning we collect demonstration trajectories using the MPLib motion planner [Guo et al., 2024] and fine-tuned using LoRA [Hu et al., 2022]; further dataset details can be found in Sec. A.2.

5.2 Effect of data scale on SFT performance

Following Lin et al. [2025], we study how supervised fine-tuning (SFT) scales with demonstration count. We train OpenVLA to convergence on datasets ranging from a few hundred to 64k expert trajectories (~ 1.26M transitions) and report average scores over three random seeds in-distribution and on unseen objects/tables (Fig. 6a and 6b). We observe that performance plateaus at roughly 16k trajectories in both settings, so we adopt the 16k-trajectory SFT checkpoint as the baseline for comparing RL fine-tuning methods.

5.3 Performance comparison between RL and SFT

Building on preceding analyses, we plot success rates during RL training for both the training distribution and the OOD object/table split in Fig. 6c and 6d, alongside SFT scores at various data



(a) (b) (c)

Figure 7: (a) Performance comparison between SFT and RL across different tasks. Both success rate and relative performance drop are reported. (b) Radar chart of success rate. (c) Radar chart of the performance ratio of out-of-distribution (OOD) to in-distribution (IND).

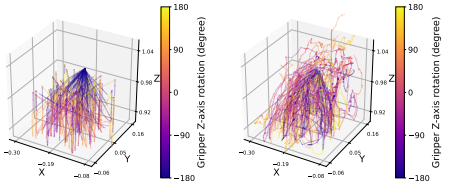
scales. RL overtakes the strongest SFT checkpoint (SFT-16k) after roughly 0.4 M environment steps on OOD task. At convergence, it performs comparably to SFT-16k in the training setting and $42.6\,\%$ better on unseen objects and tables, showing that reinforcement learning not only improves policy performance under the training distribution but also provides markedly stronger generalization.

We further evaluate the generalization performance of the converged RL policy and the best-performing SFT policy (SFT-16k, shortly denoted as SFT) on OOD tasks, as shown in Fig. 7. Here we report average success rates and average performance drop over three random seeds while full detailed results can be found in Sec. B.3. The relative performance drop is calculated as $P = \frac{\text{OOD}-\text{IND}}{\text{IND}}$. The results indicate that RL performs comparably to SFT in *Vision* tasks, noticeably better in *Semantics*, and significantly better in *Execution*.

For *Vision*, we hypothesize that neither RL nor SFT training induces visual robustness beyond the imposed visual randomness (i.e., random training tables), resulting in similar performance for both methods. In *Semantics* tasks, RL notably outperforms SFT when faced with OOD objects, both in single and multiple object scenarios. We hypothesize that, through trial-and-error, RL is able to learn the skill of "grasp" in a manner that is less dependent on object type, leading to improved generalization. For *Execution* tasks, RL surpasses SFT across all three evaluated scenarios: OOD object and receptacle positions, OOD robot initial positions, and mid-episode object repositioning.

5.4 Dissecting RL's contributions to generalization

To probe qualitative differences between the two approaches, we visualise policy roll-outs on four representative tasks (Fig. 9). In *Vision/Dynamic-Noise (strong)*, the SFT agent repeatedly drops the object immediately after grasping, indicating that heavy visual perturbations prevent it from localising the receptacle, whereas the RL agent completes the placement. With an *Unseen Object* in *Semantics*, SFT keeps attempting to grasp the item it already holds and stalls, while RL lifts it and finishes the task. This may be attributed to RL's extensive trial-and-error experience during training, enabling it to make finer-grained control when interacting with unseen objects. RL also learned to recover from failed grasps and mid-episode object shifts in two *Execution* tests, whereas SFT marches on in spite of position errors, likely because such cases never appear in the demonstration data. The trajectory distributions encountered during



(a) SFT (b) RL

Figure 8: Training trajectories of SFT (left) and RL (right); color encodes gripper Z-axis rotation.

Figure 9: Visualization of some concrete examples.

training (Fig. 8) echo this gap: RL trajectories span a broader workspace and a richer range of end-effector orientations, whereas SFT roll-outs cluster along the motion-planner paths present in the dataset. This wider coverage appears key to RL's superior generalisation on *Execution* tasks.

6 Conclusion

This study puts forward a demanding benchmark that probes how RL fine-tuning shapes the out-of-distribution generalization of VLA models. After adapting several RL methods to VLAs, we find that a streamlined PPO variant—featuring a shared actor-critic backbone, a simple warm-up, and only minimal PPO epochs—yields the strongest and most efficient gains. Relative to supervised fine-tuning, this RL approach substantially improves semantic grounding and action execution under distribution shifts, while matching its resilience to visual perturbations. These results underscore the promise of reinforcement learning as a route towards more generalizable embodied agents.

Limitations. Despite its contributions, our study has several limitations. First, we rely solely on motion-planner–generated demonstrations for supervised fine-tuning, which may not fully capture the variability present in human-collected data. Second, our evaluation is confined to pick-and-place tasks; scaling to a broader, more complex, multi-task setting remains an important direction for future work. Third, all experiments are conducted in simulation, so integrating RL fine-tuning with sim-to-real transfer to validate VLA generalization on physical robots is a crucial next step. We hope our findings will inform and accelerate these future efforts.

Acknowledgements

This research was supported by National Natural Science Foundation of China (No.62325405,62406159), Tsinghua University Initiative Scientific Research Program, Tsinghua-Efort Joint Research Center for EAI Computation and Perception, Beijing National Research Center for Information Science, Technology (BNRist), Beijing Innovation Center for Future Chips, State Key laboratory of Space Network and Communications, and Beijing Zhongguancun Academy Project C20250301.

We would like to express our sincere gratitude to Liangzhi Shi, who carried out the deployment on the Franka Panda robotic arm and provided valuable sim-to-real evaluation results. We are also grateful to Hongzhi Zang for her insightful advice on GRPO performance tuning, which significantly improved the quality of this work.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. Data quality in imitation learning. *Advances in neural information processing systems*, 36:80375–80395, 2023.

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, Thomas Unterthiner, Daniel Keysers, Skanda Koppula, Fangyu Liu, Adam Grycner, Alexey A. Gritsenko, Neil Houlsby, Manoj Kumar, Keran Rong, Julian Eisenschlos, Rishabh Kabra, Matthias Bauer, Matko Bosnjak, Xi Chen, Matthias Minderer, Paul Voigtlaender, Ioana Bica, Ivana Balazevic, Joan Puigcerver, Pinelopi Papalampidi, Olivier J. Hénaff, Xi Xiong, Radu Soricut, Jeremiah Harmsen, and Xiaohua Zhai. Paligemma: A versatile 3b vlm for transfer. *CoRR*, abs/2407.07726, 2024. URL `https://doi.org/10.48550/arXiv.2407.07726`.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2024. URL `https://arxiv.org/abs/2410.24164`.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023a.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale, 2023b. URL `https://arxiv.org/abs/2212.06817`.

Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, Hanbo Zhang, and Minzhao Zhu. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

Xi Chen, Josip Djolonga, Piotr Padlewski, Basil Mustafa, Soravit Changpinyo, Jialin Wu, Carlos Riquelme Ruiz, Sebastian Goodman, Xiao Wang, Yi Tay, Siamak Shakeri, Mostafa Dehghani, Daniel Salz, Mario Lucic, Michael Tschannen, Arsha Nagrani, Hexiang Hu, Mandar Joshi, Bo Pang, Ceslee Montgomery, Paulina Pietrzyk, Marvin Ritter, AJ Piergiovanni, Matthias Minderer, Filip Pavetic, Austin Waters, Gang Li, Ibrahim Alabdulmohsin, Lucas Beyer, Julien Amelot, Kenton Lee, Andreas Peter Steiner, Yang Li, Daniel Keysers, Anurag Arnab, Yuanzhong Xu, Keran Rong, Alexander Kolesnikov, Mojtaba Seyedhosseini, Anelia Angelova, Xiaohua Zhai, Neil Houlsby, and Radu Soricut. Pali-x: On scaling up a multilingual vision and language model, 2023. URL https://arxiv.org/abs/2305.18565.

Yuhui Chen, Shuai Tian, Shugao Liu, Yingting Zhou, Haoran Li, and Dongbin Zhao. Conrft: A reinforced fine-tuning method for vla models via consistency policy. *arXiv preprint arXiv:2502.05450*, 2025.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*, 2025.

Open X.-Embodiment Collaboration, Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alexander Herzog, Alex Irpan, Alexander Khazatsky, Anant Raj, Anikait Singh, Anthony Brohan, Antonin Raffin, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Brian Ichter, Cewu Lu, Charles Xu, Chelsea Finn, Chenfeng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Chuer Pan, Chuyuan Fu, Coline Devin, Danny Driess, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Federico Ceola, Fei Xia, Freek Stulp, Gaoyue Zhou, Gaurav S. Sukhatme, Gautam Salhotra, Ge Yan, Giulio Schiavi, Gregory Kahn, Hao Su, Haoshu Fang, Haochen Shi, Heni Ben Amor, Henrik I. Christensen, Hiroki Furuta, Homer Walke, Hongjie Fang, Igor Mordatch, Ilija Radosavovic, and et al. Open x-embodiment: Robotic learning datasets and rt-x models. *CoRR*, abs/2310.08864, 2023. URL http://dblp.uni-trier.de/db/journals/corr/corr2310.html#abs-2310-08864.

Pim De Haan, Dinesh Jayaraman, and Sergey Levine. Causal confusion in imitation learning. *Advances in neural information processing systems*, 32, 2019.

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13142–13153, 2023.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. Palm-e: An embodied multimodal language model, 2023. URL https://arxiv.org/abs/2303.03378.

Cunxin Fan, Xiaosong Jia, Yihang Sun, Yixiao Wang, Jianglan Wei, Ziyang Gong, Xiangyu Zhao, Masayoshi Tomizuka, Xue Yang, Junchi Yan, and Mingyu Ding. Interleave-vla: Enhancing robot manipulation with interleaved image-text instructions, 2025. URL https://arxiv.org/abs/2505.02152.

Roya Firoozi, Johnathan Tucker, Stephen Tian, Anirudha Majumdar, Jiankai Sun, Weiyu Liu, Yuke Zhu, Shuran Song, Ashish Kapoor, Karol Hausman, et al. Foundation models in robotics: Applications, challenges, and the future. *The International Journal of Robotics Research*, page 02783649241281508, 2023.

Dylan J Foster, Adam Block, and Dipendra Misra. Is behavior cloning all you need? understanding horizon in imitation learning. *arXiv preprint arXiv:2407.15007*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025a.

Runlin (Kolin) Guo, Xinsong Lin, Minghua Liu, Jiayuan Gu, and Hao Su. Mplib: a lightweight motion planning library, 2024. URL `https://github.com/haosulab/MPlib`. Accessed: 2024-05-13.

Yanjiang Guo, Jianke Zhang, Xiaoyu Chen, Xiang Ji, Yen-Jen Wang, Yucheng Hu, and Jianyu Chen. Improving vision-language-action model with online reinforcement learning. *arXiv preprint arXiv:2501.16664*, 2025b.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=nZeVKeeFYf9`.

Jiaheng Hu, Rose Hendrix, Ali Farhadi, Aniruddha Kembhavi, Roberto Martín-Martín, Peter Stone, Kuo-Hao Zeng, and Kiana Ehsani. Flare: Achieving masterful and adaptive robot policies with large-scale reinforcement learning fine-tuning. *arXiv preprint arXiv:2409.16578*, 2024.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025.

Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*, 2025.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL `https://arxiv.org/abs/2504.16054`.

Hamish Ivison, Yizhong Wang, Jiacheng Liu, Zeqiu Wu, Valentina Pyatkin, Nathan Lambert, Noah A Smith, Yejin Choi, and Hanna Hajishirzi. Unpacking dpo and ppo: Disentangling best practices for learning from preference feedback. *Advances in neural information processing systems*, 37: 36602–36633, 2024.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. In *International Conference on Machine Learning (ICML)*, 2024.

Timo Kaufmann, Paul Weng, Viktor Bengs, and Eyke Hüllermeier. A survey of reinforcement learning from human feedback. *arXiv preprint arXiv:2312.14925*, 10, 2023.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, Patrick Tree Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sungjae Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean Mercat, Abdul Rehman, Pannag R Sanketi, Archit Sharma, Cody Simpson, Quan Vuong, Homer Rich Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Z. Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Daphne Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan Paul Foster, Jensen Gao, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Donovon Jackson,

Charlotte Le, Yunshuang Li, Xinyu Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Khuong Nguyen, Abigail O'Neill, Rosario Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Abhishek Gupta, Dinesh Jayaraman, Joseph J Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. DROID: A large-scale in-the-wild robot manipulation dataset. In *RSS 2024 Workshop: Data Generation for Robotics*, 2024. URL https://openreview.net/forum?id=Ml2pTYLNLi.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

Vijay Konda and John Tsitsiklis. Actor-critic algorithms. *Advances in neural information processing systems*, 12, 1999.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, et al. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024a.

Ziniu Li, Tian Xu, Yushun Zhang, Zhihang Lin, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models. In *Forty-first International Conference on Machine Learning*, 2024b. URL https://openreview.net/forum?id=Stn8hXkpe6.

Fanqi Lin, Yingdong Hu, Pingyue Sheng, Chuan Wen, Jiacheng You, and Yang Gao. Data scaling laws in imitation learning for robotic manipulation, 2025. URL https://arxiv.org/abs/2410.18647.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2024.

Yuecheng Liu, Dafeng Chi, Shiguang Wu, Zhanguang Zhang, Yaochen Hu, Lingfeng Zhang, Yingxue Zhang, Shuang Wu, Tongtong Cao, Guowei Huang, et al. Spatialcot: Advancing spatial reasoning through coordinate alignment and chain-of-thought for embodied task planning. *arXiv preprint arXiv:2501.10074*, 2025.

Yan Ma, Steffi Chern, Xuyang Shen, Yiran Zhong, and Pengfei Liu. Rethinking rl scaling for vision language models: A transparent, from-scratch framework and comprehensive evaluation scheme. *arXiv preprint arXiv:2504.02587*, 2025.

Yueen Ma, Zixing Song, Yuzheng Zhuang, Jianye Hao, and Irwin King. A survey on vision-language-action models for embodied ai. *arXiv preprint arXiv:2405.14093*, 2024.

Max Sobol Mark, Tian Gao, Georgia Gabriela Sampaio, Mohan Kumar Srirama, Archit Sharma, Chelsea Finn, and Aviral Kumar. Policy agnostic rl: Offline rl and online rl fine-tuning of any class and backbone. *arXiv preprint arXiv:2412.06685*, 2024.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Hung Pham and Quang-Cuong Pham. A new approach to time-optimal path parameterization based on reachability analysis. *IEEE Transactions on Robotics*, 34(3):645–659, 2018. doi: 10.1109/TRO.2018.2819195.

Rafael Figueiredo Prudencio, Marcos ROA Maximo, and Esther Luna Colombini. A survey on offline reinforcement learning: Taxonomy, review, and open problems. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

Rajkumar Ramamurthy, Prithviraj Ammanabrolu, Kianté Brantley, Jack Hessel, Rafet Sifa, Christian Bauckhage, Hannaneh Hajishirzi, and Yejin Choi. Is reinforcement learning (not) for natural language processing: Benchmarks, baselines, and building blocks for natural language policy optimization, 2023. URL `https://arxiv.org/abs/2210.01241`.

Allen Z Ren, Justin Lidard, Lars L Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. *arXiv preprint arXiv:2409.00588*, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

Stephane Ross and Drew Bagnell. Efficient reductions for imitation learning. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 661–668, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL `https://proceedings.mlr.press/v9/ross10a.html`.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Dhruv Shah, Blazej Osinski, Brian Ichter, and Sergey Levine. LM-nav: Robotic navigation with large pre-trained models of language, vision, and action. In *6th Annual Conference on Robot Learning*, 2022. URL `https://openreview.net/forum?id=UW5A3SweAH`.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021, 2020.

Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Sean Kirmani, Brianna Zitkovich, Fei Xia, et al. Open-world object manipulation using pre-trained vision-language models. *arXiv preprint arXiv:2303.00905*, 2023.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-kai Chan, et al. Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai. *arXiv preprint arXiv:2410.00425*, 2024.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Homer Rich Walke, Kevin Black, Tony Z. Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, Abraham Lee, Kuan Fang, Chelsea Finn, and Sergey Levine. Bridgedata v2: A dataset for robot learning at scale. In *7th Annual Conference on Robot Learning*, 2023. URL `https://openreview.net/forum?id=f55MlAT1Lu`.

Junjie Wen, Yichen Zhu, Jinming Li, Zhibin Tang, Chaomin Shen, and Feifei Feng. Dexvla: Vision-language model with plug-in diffusion expert for general robot control. *arXiv preprint arXiv:2502.05855*, 2025.

Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation, 2023.

Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weilin Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. Is dpo superior to ppo for llm alignment? a comprehensive study. *arXiv preprint arXiv:2404.10719*, 2024.

Simon Zhai, Hao Bai, Zipeng Lin, Jiayi Pan, Peter Tong, Yifei Zhou, Alane Suhr, Saining Xie, Yann LeCun, Yi Ma, et al. Fine-tuning large vision-language models as decision-making agents via reinforcement learning. *Advances in neural information processing systems*, 37:110935–110971, 2024.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11975–11986, 2023.

Biao Zhang, Zhongtao Liu, Colin Cherry, and Orhan Firat. When scaling meets llm finetuning: The effect of data, model and finetuning method. *arXiv preprint arXiv:2402.17193*, 2024a.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3836–3847, 2023a.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*, 2023b.

Zijian Zhang, Kaiyuan Zheng, Zhaorun Chen, Joel Jang, Yi Li, Siwei Han, Chaoqi Wang, Mingyu Ding, Dieter Fox, and Huaxiu Yao. Grape: Generalizing robot policy via preference alignment. *arXiv preprint arXiv:2411.19309*, 2024b.

Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

Wei Zhao, Pengxiang Ding, Min Zhang, Zhefei Gong, Shuanghao Bai, Han Zhao, and Donglin Wang. Vlas: Vision-language-action model with speech instructions for customized robot manipulation. *arXiv preprint arXiv:2502.13508*, 2025.

Rui Zheng, Shihan Dou, Songyang Gao, Yuan Hua, Wei Shen, Binghai Wang, Yan Liu, Senjie Jin, Qin Liu, Yuhao Zhou, et al. Secrets of rlhf in large language models part i: Ppo. *arXiv preprint arXiv:2307.04964*, 2023.

Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. *arXiv preprint arXiv:2412.10345*, 2024.

Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly reflect the paper's contributions and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discussed the limitation of our work in Sec. 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: No theoretical assumptions or proofs are included in this paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The code provided in the supplementary materials contains details to ensure reproducibility.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We release the code for this paper in the supplementary materials.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The training and testing details are provided in both the experimental section and the appendix of this paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We clarify the statistical significance of the experiments in the experimental section of the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We introduce the computational resources used to carry out the experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our research complies with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the broader impacts in the appendix of this paper.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper does not pose significant risks for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We state all assets used in this work in the appendix of this paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: No new assets are released in this paper.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper contains no crowdsourcing or research involving human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper contains no research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Implementation Details

A.1 Implementation of different fine-tuning algorithm

LoRA finetuning Instead of fine-tuning the entire OpenVLA model, we use Low-Rank Adaptation (LoRA) [Hu et al., 2022] to reduce computational costs for all fine-tuning methods in this paper, including both RL and SFT. Specifically, we apply LoRA modules with rank $r = 32$ to all linear layers in the OpenVLA model, while fully fine-tuning the value head.

PPO update The policy is updated by maximizing the following objective [Schulman et al., 2017]

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t \left[\min \left(\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \hat{A}_t,\ \text{clip} \left(\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)},\ 1-\epsilon,\ 1+\epsilon \right) \hat{A}_t \right) \right] \tag{1}$$

where $\pi_\theta(a_t \mid s_t)$ is calculated as the product of the probabilities of each action token, and the advantage $\hat{A}_t$ is given by Generalized Advantage Estimator (GAE) [Schulman et al., 2015], where

$$\hat{A}_t = \sum_{l=0}^{T-t-1} (\gamma\lambda)^l \left[r_{t+l} + \gamma V(s_{t+l+1}) - V(s_{t+l}) \right] \tag{2}$$

GRPO implementation We follow the outcome supervision RL paradigm in GRPO [Shao et al., 2024], where the advantage is computed as the normalized outcome rewards within each group:

$$\hat{A}_t^i = \frac{r^i - \text{mean}(\mathbf{r})}{\text{std}(\mathbf{r})} \tag{3}$$

$$\mathbf{r} = \{r^1, r^2, \cdots, r^G\} \tag{4}$$

where r^i is the outcome reward for trajectory i, and G denotes the number of trajectories in a group.

To stabilize training, we use a total of 32 groups, each containing 8 trajectories. For successful placement episodes, the trajectory is truncated immediately after the first successful step. In contrast, unsuccessful episodes remain untruncated. The outcome reward is given only at the final step of each episode and is defined as the sum of the grasping, holding, and placement rewards.

DPO implementation We employ an improved version of DPO as proposed by [Zhang et al., 2024b], namely Trajectory-wise Preference Optimization (TPO), which fine-tunes the vision-language agent (VLA) by aligning its policy with human preferences over entire trajectories, rather than individual actions. This is achieved by comparing pairs of trajectories (one preferred, one rejected) and optimizing the following loss:

$$\mathcal{L}_{\text{TPO}} = -\mathbb{E}_{(\zeta_w, \zeta_l) \sim \mathcal{D}} \left[\log \sigma \left(\beta \left(\log \frac{\pi_\theta(\zeta_w)}{\pi_{\text{ref}}(\zeta_w)} - \log \frac{\pi_\theta(\zeta_l)}{\pi_{\text{ref}}(\zeta_l)} \right) \right) \right] \tag{5}$$

where ζ_w and ζ_l are the preferred and rejected trajectories, respectively.

In the original TPO paper [Zhang et al., 2024b], the total reward consists of a task success reward, a self-evaluated score, and an LLM-generated dense reward. In our task, only a sparse reward is available, as described in Sec. 5.1. Therefore, we use only this sparse reward for simplicity.

A.2 Details of motion planner and SFT dataset

We use the "plan_screw" function provided by [Guo et al., 2024]. In detail, we employ a task-space guided iterative inverse kinematics method to generate a locally feasible path that approximates a screw (helical) motion, using the robot's kinematic model (from URDF) and the current joint angles. This path is then post-processed using a Time-Optimal Path Parameterization algorithm (TOPP) [Pham and Pham, 2018], which assigns timing characteristics to the geometric path, resulting in an efficient and executable trajectory (in joint space) that satisfies the robot's kinematic constraints. We perform forward kinematics on the output joint positions and robot current joint positions to obtain the corresponding end-effector poses, and then compute their differences to obtain "delta_ee_pose".

However, data collected by the motion planner contains a fraction of idling actions. When trained with SFT on such data, the fine-tuned OpenVLA model often gets stuck during task execution. To

address this, we employ an action filtering technique: actions with a delta position norm less than 0.01 and a delta Euler angle norm less than 0.06 are discarded. Using this filter, approximately one-third of the actions are removed, which significantly alleviates the issue of the model getting stuck. We apply action filtering to all data collected by the motion planner.

A.3 Details of assets usage

As described in Sec. 5.1, we acquire digital assets from multiple sources. The 3D assets of objects and receptacles are sourced from the ManiSkill simulator [Tao et al., 2024], Objaverse [Deitke et al., 2023], or downloaded from Sketchfab[3]. Additional table appearances are synthesized using Stable Diffusion [Rombach et al., 2022] and ControlNet [Zhang et al., 2023a]; specifically, we use the online service provided by LibLibAI[4]. Distractor textures are downloaded from ambientCG[5]. We comply with the licenses of these assets accordingly.

A.4 Details of tasks for generalization evaluation

As described in Sec. 5.1, we build multiple tasks to test the generalization of VLA modes. The details of each task is detailed as below:

- *Training*: At the start of each episode, one object from the 16 training objects and one of the 16 training table appearances is selected. The object and the receptacle (yellow plate) are placed on the table, with their positions randomized within a rectangular area on the table. The language instruction is "put O on R", where O and R denote the names of the object and receptacle, respectively.
- *Unseen Table*: The table appearance is selected from 5 unseen appearances. Other settings are the same as in the *Training* setting.
- *Dynamic Texture (weak)*: In addition to selecting an object and a table appearance, a texture from the 16 textures is chosen at the start of each episode. This texture is cropped and resized differently at each step in the episode, and is overlaid on top of the object, receptacle, and robot arm in the image, with an image mixing transparency of 0.3.
- *Dynamic Texture (strong)*: The settings are the same as in the *Dynamic Texture (weak)* setting, except with an image mixing transparency of 0.5.
- *Dynamic Noise (weak)*: In addition to selecting an object and a table appearance, a texture from the 16 textures is chosen at the start of each episode. This texture is cropped and resized differently at each step in the episode, and is overlaid on whole image, with an image mixing transparency of 0.3.
- *Dynamic Noise (strong)*: The settings are the same as in the *Dynamic Noise (weak)* setting, except with an image mixing transparency of 0.5.
- *Unseen Objects*: The object is selected from the 9 unseen objects. Other settings are the same as in the *Training* setting.
- *Unseen Receptacles*: In addition to selecting an object and a table appearance, a receptacle from the 16 unseen receptacles is chosen at the start of each episode, replacing the default training receptacle (yellow plate). Other settings are the same as in the *Training* setting.
- *Unseen Instruction Phrasing*: In addition to selecting an object and a table appearance, a language instruction template from the 16 unseen language templates is chosen at the start of each episode, replacing the default language template ("put O on R"). Other settings are the same as in the *Training* setting. The unseen language templates are:
  - Place the O on the R
  - set O on R
  - move the O to the R
  - Take the O and put it on the R

[3] `https://sketchfab.com/`

[4] `https://liblib.art/`

[5] `https://ambientcg.com/`

  - pick up O and set it down on R
  - please put the O on the R
  - Put O onto R.
  - place the O onto the R surface
  - Make sure O is on R.
  - on the R, put the O
  - put the O where the R is
  - Move the O from the table to the R
  - Move O so it's on R.
  - Can you put O on R?
  - O on the R, please.
  - the O should be placed on the R.

  Note that there are variations in punctuation and capitalization in these instruction templates.
- *Multi-Object (both seen)* At the start of each episode, two different objects from the 16 training objects and one of the 16 training table appearances is selected. These two objects and the receptacle (yellow plate) are placed on the table, with their positions randomized within a rectangular area on the table. The language instruction is "put O on R", where O and R denote the names of the fist object and receptacle, respectively.
- *Multi-Object (both unseen)* The two objects are selected from the 9 unseen objects, and they are different from each other. Other settings are the same as in the *Multi-Object (both seen)* setting.
- *Distractive Receptacle*: In addition to selecting an object and a table appearance, a distractive receptacle from the 16 unseen receptacles is chosen at the start of each episode, and placed on the table without being used. Other settings are the same as in the *Training* setting.
- *Multi-Receptacle (both unseen)*: At the start of each episode, an objects from the 16 training objects, two different receptacles from the 16 unseen receptacles, and one of the 16 training table appearances is selected. The object and the two receptacles are placed on the table, with their positions randomized within a rectangular area on the table. The language instruction is "put O on R", where O and R denote the names of the object and the first receptacle, respectively.
- *Unseen Position (Object & Receptacle)*: The object and the receptacle (yellow plate) are placed on the table, with their positions randomized within a surrounding square frame, larger than the rectangular area, on the table. Other settings are the same as in the *Training* setting.
- *Unseen Robot Init Pose*: The initial pose of each articulation of the robot is randomized at the start of each episode, rather than being set to a fixed pose as in the *Training* setting. Other settings are the same as in the *Training* setting.
- *Mid-Episode Object Reposition*: The object is teleported to a new random position on the table at the 5-th timestep in an episode. Other settings are the same as in the *Training* setting.

B Additional Experiment Results

B.1 PPO performance on generation temperature

We evaluate the performance of PPO with different generation temperatures during training, as shown in Fig. 10. The results show that a very large temperature hinders the training process, while a moderate or small temperature leads to good performance. Therefore, we use a temperature of 1.0 in PPO.

B.2 PPO performance on LoRA rank

We evaluate the performance of PPO with different LoRA ranks, as shown in Fig. 11. The results show that a lower LoRA rank leads to slightly more efficient training. In all of our fine-tuning experiments, we use a LoRA rank of 32 to ensure sufficient model capacity. A more detailed study of different LoRA ranks is left for future work.

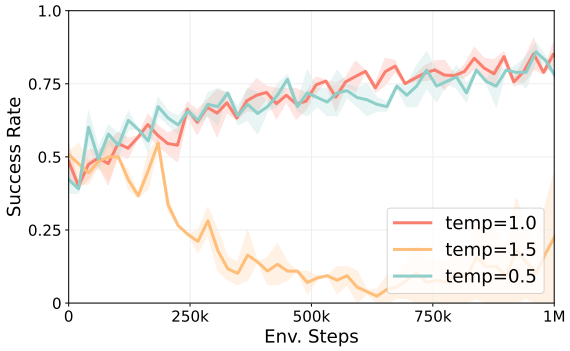


Figure 10: PPO performance on generation temperature during training.

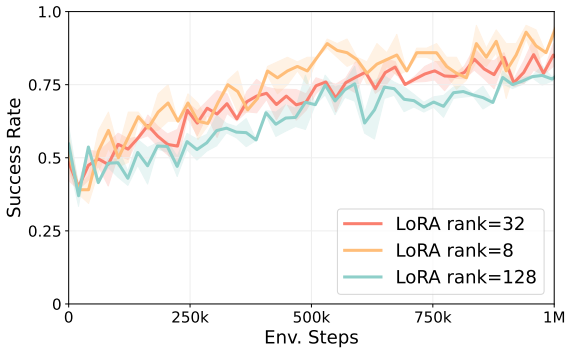


Figure 11: PPO performance on different LoRA rank.

B.3 Full results of OOD tasks

Detailed results of Fig. 7 are listed in Tab. 1, where grasp accuracy, continuous grasp accuracy and task success rate are reported.

B.4 Performance comparison on sim-to-real generalization

As discussed, all of our previous experiments are conducted in simulation for scalability and reproducibility when comparing RL and SFT. Nonetheless, real-world performance is a critical factor to validate. We therefore include a preliminary real-world experiment to assess sim-to-real generalizability.

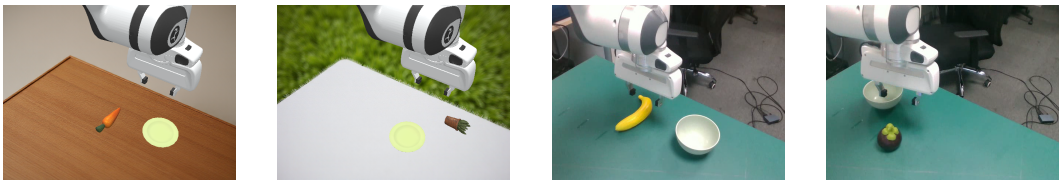
Specifically, we replaced the WidowX arm with a Franka Panda and aligned the camera views. Using the same pipeline as in our main experiments, we retrained VLA models with both RL and SFT under this configuration, and deployed them zero-shot on the real robot. This naturally introduces an OOD setting involving kinematic differences, varied backgrounds, and different real-world objects and receptacles.

In simulation, the training environment contains 16 objects (as described in Sec. 5.1), 7 backgrounds, and a plate as the receptacle. The real-world evaluation involves 6 new objects (pepper, cucumber, banana, mangosteen, kiwi, sponge), a table as the background, and a green bowl as the receptacle. In both simulation and real-world setups, the initial positions of objects and receptacles are randomized. For real-world evaluation, the initial conditions of each trail are aligned across SFT and RL to ensure a fair comparison. The training and evaluation environments are illustrated in Fig. 12.

We report the average performance over 30 real-world trials in Tab. 2. The results show that RL outperforms SFT in both grasp and pick-and-place success rates, demonstrating stronger generalization ability. Qualitatively, we observed that SFT often exhibits excessive movement and overshooting, leading to suboptimal grasp poses and failures. In contrast, RL shows some jitter but successfully iteratively adjusts the end-effector pose, leading to higher grasp success.

Table 1: Detailed results across different tasks, with the best results across SFT and RL highlighted in **bold**. Standard deviations are reported in parentheses.

	Acc.		Cont. Acc.		Suc.	
	SFT	RL	SFT	RL	SFT	RL
IND	0.922 (.013)	**0.974** (.007)	0.906 (.026)	**0.948** (.007)	0.781 (.013)	**0.938** (.013)
Table	0.880 (.032)	**0.979** (.007)	0.828 (.022)	**0.927** (.019)	0.719 (.051)	**0.844** (.034)
Texture-w	0.896 (.019)	**0.948** (.019)	0.870 (.039)	**0.906** (.046)	0.719 (.078)	**0.833** (.048)
Texture-s	**0.839** (.019)	0.802 (.039)	**0.750** (.034)	0.703 (.066)	0.557 (.041)	**0.630** (.039)
Noise-w	0.901 (.019)	**0.948** (.027)	0.854 (.041)	**0.911** (.027)	0.708 (.032)	**0.854** (.027)
Noise-s	0.781 (.013)	**0.823** (.041)	0.688 (.034)	**0.750** (.038)	0.505 (.027)	**0.667** (.048)
Obj.	0.818 (.027)	**0.875** (.000)	0.724 (.032)	**0.833** (.039)	0.453 (.044)	**0.714** (.019)
Recep.	0.844 (.013)	**0.911** (.039)	0.802 (.032)	**0.833** (.019)	0.615 (.019)	**0.750** (.022)
Instruct	0.885 (.045)	**0.969** (.013)	0.859 (.044)	**0.948** (.027)	0.672 (.034)	**0.891** (.034)
M-Obj. (IND)	0.797 (.013)	**0.823** (.027)	0.771 (.019)	**0.797** (.051)	0.615 (.052)	**0.750** (.046)
M-Obj. (OOD)	0.656 (.034)	**0.750** (.046)	0.583 (.019)	**0.688** (.038)	0.297 (.013)	**0.578** (.034)
Disturb Recep.	0.880 (.032)	**0.948** (.007)	0.844 (.044)	**0.896** (.027)	0.672 (.056)	**0.812** (.034)
M-Recep.	0.865 (.019)	**0.885** (.007)	**0.818** (.041)	0.797 (.034)	0.458 (.065)	**0.599** (.063)
Obj. Pos.	0.865 (.019)	**0.943** (.007)	0.802 (.007)	**0.917** (.019)	0.568 (.070)	**0.807** (.019)
Robot Pose	0.646 (.027)	**0.917** (.063)	0.568 (.019)	**0.844** (.071)	0.339 (.052)	**0.797** (.064)
Obj. Rep.	0.573 (.039)	**0.891** (.034)	0.474 (.060)	**0.807** (.045)	0.286 (.041)	**0.745** (.048)



(a) (b) (c) (d)

Figure 12: Visualization of the training and evaluation environments for the preliminary real-world evaluation: (a)-(b) examples of simulation training environments; (c)-(d) examples of real-world evaluation settings.

B.5 Performance comparison with action chunk

To evaluate the robustness of our conclusions with stronger architectures, we additionally test with OpenVLA-OFT [Kim et al., 2025], which enhance OpenVLA with action chunking.

We adopted an action chunking strategy with a chunk size of 4, enabling the model to predict sequences of actions rather than single steps. To initialize training, we first collected 400 trajectories through motion planning without applying any action filtering, and then trained a warm-up model using a checkpoint from Huggingface[6].

Building on this, we conducted RL fine-tuning in which each observation corresponds to a sequence of 4 actions and is optimized based on cumulative rewards on the following 4 steps. Given the increased dimensionality of the action space, we adjusted the PPO algorithm by introducing action-dimension-wise clipping, reducing the clipping ratio from 0.2 to 0.1, and tuning the hyperparameters with discount factor $\gamma = 0.96$ and GAE $\lambda = 0.85$ to improve training stability.

Results are summarized in Fig. 13 and Tab. 3, showing that RL maintains its advantage over SFT even under this more advanced architecture.

[6]Haozhan72/Openvla-oft-SFT-libero10-trajall

Table 2: Preliminary real-world evaluation results. RL fine-tuning in simulation helps sim-to-real transfer compared to SFT.

	SFT	RL
Grasp Success Rate	0.10	**0.43**
Pick-and-Place Success Rate	0.00	**0.27**

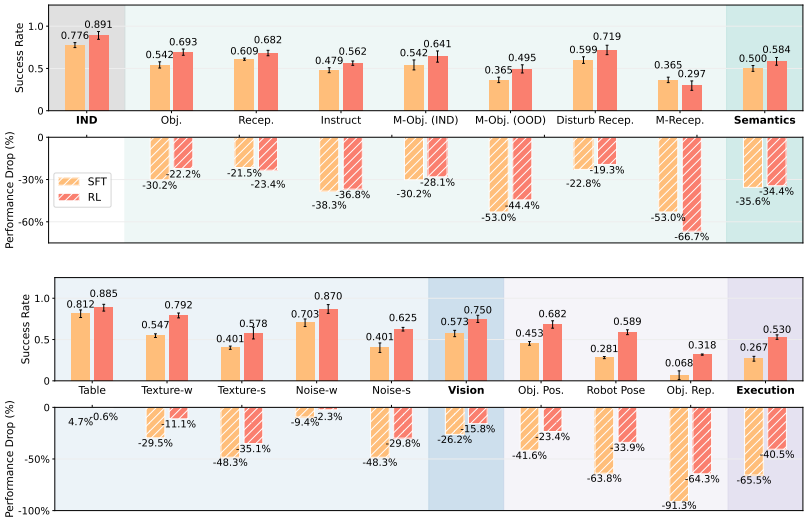


Figure 13: Performance comparison between SFT and RL across different tasks *with action chunking*. Both success rate and relative performance drop are reported.

B.6 Extended performance comparison on opening articulated objects

To move beyond simple pick-and-place and introduce richer manipulation dexterity, we design an *open articulated objects* task, and replace the WidowX robot arm with a Franka Panda arm to provide a larger working space. The agent must grasp an object's handle and open the door past $20°$. The reward function is shaped as follows: a reward of 0.1 is given when the handle is first grasped, an additional 0.1 is granted if the grasp is sustained for five consecutive steps, and a final 1.0 is provided upon successful door opening.

During training, we introduce randomization across vision, semantics, and execution to improve generalization. For vision, we randomize over 16 background assets (as described in Sec. 5.1). For semantics, we employ 8 articulated objects that vary in size, name, handle type (bar or boxed), and handle position (up, down, left, or right). For execution, we perturb the object frame with random translations up to ± 8 cm and rotations up to $\pm\pi/48$. The training and evaluation environments are illustrated in Fig. 14.

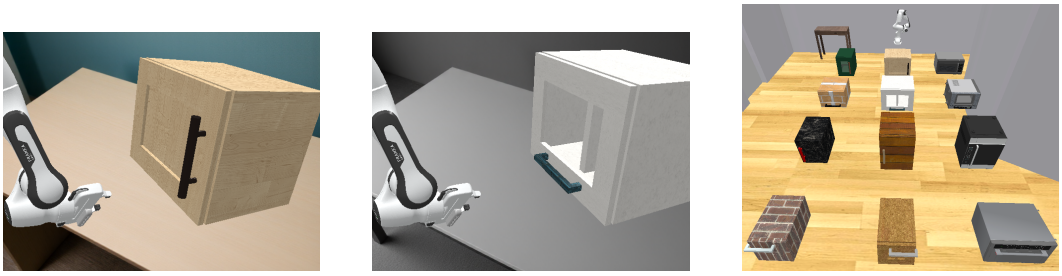
We then evaluate generalization on six out-of-distribution (OOD) splits:

- *Vision:* 5 novel background scenes,
- *Vision:* whole-scene dynamic noise with an image mixing transparency of 0.5,
- *Semantics:* 4 unseen articulated objects (microwave, dishwasher, mini-fridge, oven),
- *Semantics:* 16 re-phrased language instructions,
- *Execution:* translations up to 12 cm and rotations up to $\pm\pi/24$,
- *Execution:* random initial offsets on joints 0/1/2/4 of the robot arm.

As shown in Tab. 4, RL continues to outperform SFT on semantic and execution splits, consistent with our main pick-and-place results.

Table 3: Detailed results across different tasks *with action chunking*, with the best results across SFT and RL highlighted in **bold**. Standard deviations are reported in parentheses.

	Acc.		Cont. Acc.		Suc.	
	SFT	RL	SFT	RL	SFT	RL
IND	**0.984** (.013)	0.969 (.022)	**0.953** (.026)	**0.953** (.022)	0.776 (.029)	**0.891** (.046)
Table	**0.964** (.007)	0.948 (.019)	**0.953** (.013)	0.948 (.019)	0.812 (.046)	**0.885** (.041)
Texture-w	0.844 (.046)	**0.885** (.015)	0.807 (.032)	**0.870** (.007)	0.547 (.022)	**0.792** (.029)
Texture-s	0.719 (.034)	**0.781** (.056)	0.661 (.015)	**0.734** (.046)	0.401 (.019)	**0.578** (.071)
Noise-w	0.932 (.015)	**0.974** (.007)	0.880 (.019)	**0.958** (.027)	0.703 (.046)	**0.870** (.053)
Noise-s	0.755 (.039)	**0.839** (.041)	0.724 (.019)	**0.786** (.045)	0.401 (.058)	**0.625** (.022)
Obj.	0.807 (.007)	**0.901** (.019)	0.740 (.007)	**0.875** (.022)	0.542 (.037)	**0.693** (.037)
Recep.	0.927 (.027)	**0.932** (.045)	0.885 (.019)	**0.911** (.053)	0.609 (.013)	**0.682** (.032)
Instruct	0.620 (.007)	**0.641** (.013)	0.604 (.019)	**0.630** (.019)	0.479 (.029)	**0.562** (.026)
M-Obj. (IND)	0.661 (.075)	**0.745** (.019)	0.641 (.080)	**0.714** (.032)	0.542 (.059)	**0.641** (.066)
M-Obj. (OOD)	0.589 (.048)	**0.661** (.019)	0.552 (.041)	**0.620** (.019)	0.365 (.032)	**0.495** (.048)
Disturb Recep.	**0.901** (.027)	0.870 (.029)	**0.891** (.022)	0.849 (.048)	0.599 (.039)	**0.719** (.056)
M-Recep.	**0.891** (.034)	0.859 (.038)	**0.849** (.060)	0.828 (.058)	**0.365** (.032)	0.297 (.056)
Obj. Pos.	0.807 (.015)	**0.917** (.027)	0.771 (.015)	**0.896** (.029)	0.453 (.022)	**0.682** (.045)
Robot Pose	0.557 (.019)	**0.797** (.034)	0.484 (.034)	**0.734** (.046)	0.281 (.013)	**0.589** (.029)
Obj. Rep.	0.219 (.038)	**0.510** (.015)	0.177 (.041)	**0.453** (.013)	0.068 (.053)	**0.318** (.007)



(a) (b) (c)

Figure 14: Illustration of the open articulated objects task. (a)-(b) Two representative training scenarios. (c) Set of articulated objects used in training (left two rows) and in OOD testing (the most right row).

C Broader impacts

Our work advances the understanding and generalization capabilities of Vision-Language Action (VLA) models in embodied AI, which may accelerate the deployment of more robust and adaptive agents in real-world environments. Improved generalization can enhance the reliability of AI systems in applications such as assistive robotics, autonomous navigation, and household automation, potentially benefiting society by making these technologies safer and more accessible.

However, as VLAs become more capable and generalizable, there is also a risk of unintended consequences, such as misuse in surveillance, privacy violations, or deployment in safety-critical scenarios without adequate oversight. We encourage responsible use of our findings and recommend that future deployments of VLA-based systems consider safety, ethical, and privacy implications.

Table 4: Performance comparison on the open articulated objects task, with results averaged over 3 random seeds. Standard deviations are reported in parentheses.

	SFT		RL	
	Success Rate	Degradation Ratio	Success Rate	Degradation Ratio
IND	**0.745** (.015)	–	0.724 (.045)	–
New Backgrounds	**0.766** (.071)	**+0.028**	0.734 (.058)	+0.014
Dynamic Noise	**0.526** (.015)	**-0.294**	0.448 (.037)	-0.381
Vision Avg.	**0.646** (.043)	**-0.130**	0.591 (.048)	-0.184
Articulated Obj.	0.042 (.015)	-0.944	**0.151** (.019)	**-0.791**
Re-phrased Inst.	**0.703** (.077)	**-0.056**	0.620 (.041)	-0.144
Semantic Avg.	0.373 (.046)	-0.500	**0.386** (.030)	**-0.468**
Obj. Pose	0.448 (.032)	-0.399	**0.484** (.089)	**-0.331**
EE Pose	0.151 (.027)	-0.797	**0.312** (.013)	**-0.568**
Execution Avg.	0.300 (.030)	-0.600	**0.398** (.051)	**-0.450**

.