FINE-TUNING VLAS WITH SELF-DEMONSTRATED GENERATIVE CONTROL FOR MULTI-TASK MANIPULATION

Anonymous authors
Paper under double-blind review

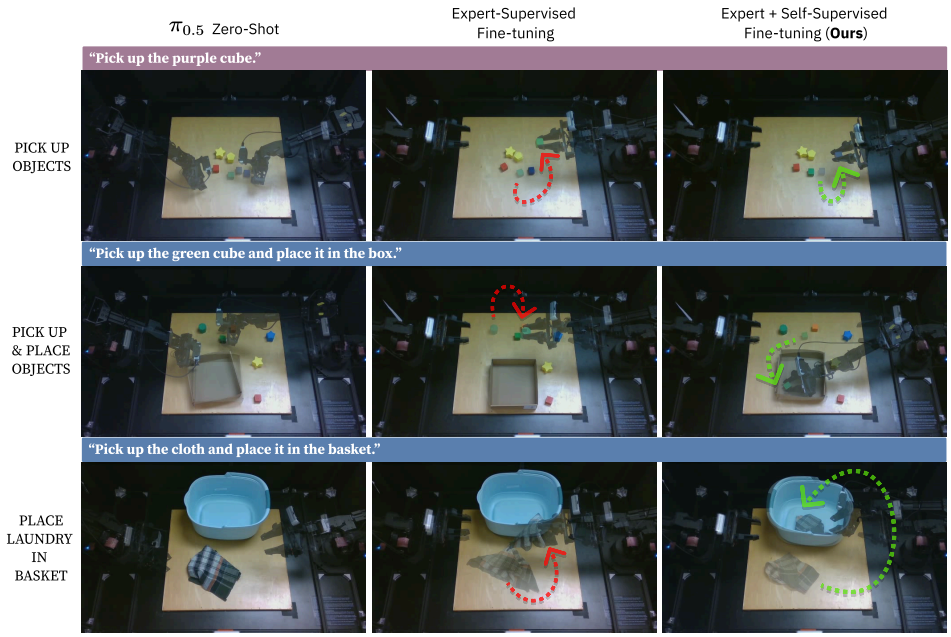


Figure 1: **Left**: Zero-shot $\pi_{0.5}$ policy follows instructions but cannot complete grasps on our robot. **Center**: Fine-tuning on expert "pick-up" data alone loses instruction following, reaching for the wrong object (rows 1–2), and loses the place motion after pick (rows 2–3). **Right (Ours)**: A single multi-task policy, jointly trained on 2.2 mins of *expert-supervised* teleoperation for "pick-up" and *self-supervised* generative rollouts from $\pi_{0.5}$ for "pick-and-place" succeeds across all task families.

ABSTRACT

State-of-the-art vision-language-action (VLA) models such as $\pi_{0.5}$ exhibit strong semantic understanding, instruction following and task behavior. However, when deployed on new robots, even minor mismatches in hardware configuration relative to pretraining can cause severe performance drops. Finetuning the VLA on in-domain expert data from the new embodiment improves performance on the expert task but leads to a loss in its original instruction following and behavioral priors. In this paper, we propose a self-supervised method that generates online interaction rollouts from the zero-shot VLA as additional training data for finetuning. Our experiments show this yields strong multi-task manipulation policies *on the target robot* that (1) inherit pretrained tasks distilled from the zero-shot VLA without expert demos for them, (2) improve success and generalist instruction following beyond expert-only fine-tuning on expert-teleoperated tasks, and (3) mitigate forgetting of held-out tasks from the pretrained policy, not part of fine-tuning. We demonstrate the success of our approach across test sets probing generalization on a real ALOHA robot and a new simulation benchmark in RoboTwin. Video results are available at `https://anonymous-sgc-vla.pages.dev/`.

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

1 INTRODUCTION

Vision-language-action (VLA) policies (Genesis AI Team, 2026; NVIDIA GEAR Lab, 2025; Physical Intelligence et al., 2025; 2026; Team et al., 2025) learn to map a language instruction and camera images to robot actions, offering a promising route to task generalization in robotics. The catch is that this generalization is hard to inherit: a VLA policy trained on one robot platform and deployed on another of the same type can struggle with exact physical grounding and grasping due to minor variations in grippers, camera configurations or other embodiment gaps (Song et al., 2026; Wagenmaker et al., 2025). Fine-tuning improves grasping behavior but erodes generalist instruction following and behavioral priors that made it valuable in the first place (Kachaev et al., 2025; Xu et al., 2025; Zha et al., 2026). *How can we adapt a VLA to a new robot while retaining its instruction following ability and behavioral priors?*

When we deploy $\pi_{0.5}$ (Physical Intelligence et al., 2025) to our ALOHA robotic platform (Zhao et al., 2023) without tuning, the robot attempts to follow instructions, moving towards objects it should pick up, but fails to grasp them. For e.g., zero-shot policy localizes and attempts to "pick up the purple cube" in Figure 1. If we fine-tune this VLA with "expert-supervised" teleoperated demonstrations from the "pick-up objects" task family, this last mile fine-grained control is mostly restored, but instruction following degrades (Figure 1, center) as it picks up the green cube instead of the purple one. Additionally, expert supervision forgets place motion in the "pick-up-place" tasks.

Our key observation is that, even when it cannot complete a task, the base policy's action predictions stay semantically correlated with the instruction across its online rollout trajectories (Figure 1, left). We exploit this by adding "self-supervised" demonstrations. Concretely, we roll out the frozen base policy on a wider range of pick-and-place tasks similar to its pretraining, and train on its own predicted actions, a self-generated form of rehearsal that requires no access to the original pretraining data. Because these rollouts are executed on our target robot hardware, in our scenes, under our prompts, the rehearsal data carries no domain gap from the pretraining robot(s). Although the policy often fails to fully succeed on these rollouts, training on them with a small number of expert supervised rollouts improves performance over expert-only fine-tuning across our tasks, including the pick-up task with expert supervision (Figure 1, right). Self-supervised demonstrations extend task coverage and distill the semantic understanding, instruction following and pretrained behavior from the base policy.

We demonstrate the effectiveness of jointly training on expert-supervised and self-supervised demonstrations in both real and simulated environments. On the ALOHA robot, we evaluate expert tasks, self-supervised tasks, and held-out tasks; generalizing to novel objects and randomized object layouts throughout. We also contribute a new simulation benchmark and evaluation protocol in RoboTwin (Chen et al., 2025) that learns new tasks and retains old ones through self-demonstrations without access to the base policy's training data, testing generalization to novel objects and task compositions. Hence, we propose a more comprehensive way of evaluating generalization. Our **core insights** are that self-demos improve performance on (1) self-supervised tasks, (2) expert-supervised tasks, and (3) other held-out tasks not part of the fine-tuning data:

- *Self-supervision distills behavioral priors of pretraining tasks without expert data for them.* Fine-tuning on expert "pick" demos completely forgets "place" behavior (0% success rate). Jointly training on expert "pick" and self-supervised "place" data recovers "place" behavior from 0% to 55% success , without any expert "place" demos. In Robotwin simulation, it recovers 10 severely forgotten mid-training tasks from 16.6% to 70.6%.
- *Training jointly on self- and expert-demos improves success on expert tasks, including instruction following on out-of-distribution objects beyond expert-only fine-tuning.* SR rises from 65% to 90% on the expert picking up cubes task, and from 40% to 90% on a new contact-rich bimanual gear insertion task (Figure 3) not present in pretraining over expert-only fine-tuning. Moreover, sub-optimal self-demos are competent w.r.t. an oracle that re-collects expert demos for those same tasks, so there is little reason to pay the price of data re-collection.
- *Self-demonstrations even preserves held-out skill families the policy was never fine-tuned on.* We conducted a user study for "push" and "pour/turn over" rollouts on ALOHA robot, ours attempts the correct push motion in 82% vs 38% rollouts and pour/turn in 94% vs 43% compared to an oracle given expert teleoperated demos for all tasks (expert and self-supervised). Six blind raters preferred our motion in 68% and 75% of paired comparisons ($p < 0.001$).

2 Related Works

VLA post-training and continual learning. Fine-tuning foundation VLMs on embodied action data frequently causes catastrophic forgetting, degrading semantic understanding and instruction following (Driess et al., 2026; Gao et al., 2026; Hancock et al., 2025; Kachaev et al., 2025). While recent VLA pre-training frameworks attempt to preserve these capabilities through multi-modal co-training (Wen et al., 2025; Yang et al., 2025; Driess et al., 2026; Physical Intelligence et al., 2025) or Bayesian factorization (Xu et al., 2025), downstream post-training to a target robotic domain remains prone to forgetting. Recent continual adaptation approaches explore visual representation alignment in simulation (Kachaev et al., 2025), parameter expansion and adapter routing (Römer et al., 2026; Wu et al., 2025), weight merging (Yadav et al., 2025; Fu et al., 2025), and on-policy reinforcement fine-tuning (Hu et al., 2026; **?**). Yet, experience replay (Rolnick et al., 2019; Chaudhry et al., 2019), which stores and rehearses past rollouts remains the most effective (Syed et al., 2025; Liu et al., 2026a;b). Existing rehearsal works assume access to original pre-training datasets (Kim et al., 2024) or conduct intermediate fine-tuning on mid-scale benchmarks which completely overwrites pretrained representations (Liu et al., 2023; Zhou et al., 2025). In contrast, we address the practical setting where downstream users receive a pretrained VLA (e.g., $\pi_{0.5}$) without access to its proprietary pre-training data or task list. In a first, we propose distilling semantic priors and task behaviors directly from online rollouts of the zero-shot base policy during post-training, enabling seamless policy transfer to target user's setup.

In the continual learning literature, our method is similar to **generative replay** (Shin et al., 2017), here applied to text-, vision-, and state- conditioned generative control; drawing a close parallel to prior-preserved fine-tuning of text-to-image diffusion models such as DreamBooth (Ruiz et al., 2023; Kumari et al., 2023; Masip et al., 2025) and dataset distillation (Wang et al., 2018). Unlike DreamBooth which samples the pretrained model's prior *offline*, we roll out the base policy *online* on the target robot to generate action chunks. DreamBooth also preserves the prior for the same class being customized (like dog), whereas we replay pretraining tasks disparate from those receiving expert supervision, to distill them. Both differences follow from generating the replay data through physical interaction on a target robot that may differ in hardware, making our approach a *self-supervised* instantiation of generative replay: the policy's own rollouts serve as pseudo-targets for distillation. Here, "**self-supervision**" denotes using the base policy's generative control rollouts as target supervision for rehearsal, distinguishing our approach from classical robotics self-supervision aimed at learning reward or dynamics models through physical interaction (Pinto & Gupta, 2016; Lynch et al., 2020; Oh et al., 2018). Unlike trajectory-based generative replay that trains a separate generator or teacher network to synthesize synthetic data (Yue et al., 2024), we use the frozen pretrained VLA directly as its own trajectory generator to perform *self-supervised self-distillation*.

3 Method

Our goal is to finetune a pre-trained VLA to solve a variety of tasks on our robot in our environments. VLAs do not perform well on different robots in a zero-shot manner and need to be finetuned on expert data from the specific deployment scenario. Existing finetuning methods are ineffective and also lead to forgetting. We propose a modification for how to finetune with expert data in section 3.2 and also develop a new self-supervised methodology in section 3.3.

3.1 Preliminaries

Our base model is a pretrained continuous action chunking VLA policy $\pi_{0.5}$ (Physical Intelligence et al., 2025). For a trajectory with task prompt p specified in natural language, the policy takes in a multi-modal observation $\mathbf{o}_t = (\mathbf{I}_t, p, \mathbf{q}_t)$ at each decision-making time step t, where $\mathbf{I}_t$ denotes camera images and $\mathbf{q}_t$ is the robot's proprioceptive state configuration. $\mathbf{I}_t$ and $\mathbf{q}_t$ vary with t while p is shared across the trajectory. The policy outputs a continuous action chunk $\mathbf{a}_t = a_{t:t+H}$ and models the probability distribution $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$.

The policy architecture includes a specialized action expert and a VLM backbone. The action expert takes in a sequence of continuous noise action tokens, $\mathbf{a}_t^{\tau,\omega} = \tau\mathbf{a}_t + (1-\tau)\omega, \omega \sim \mathcal{N}(0, \mathbf{I})$, where $\tau \in [0, 1]$ is the flow matching time index and the action expert is trained to predict the flow velocity vector $\omega - \mathbf{a}_t$. The output action space is absolute joint angles $\mathbf{a}_t \in \mathbb{R}^{H\times D}$, with an action horizon

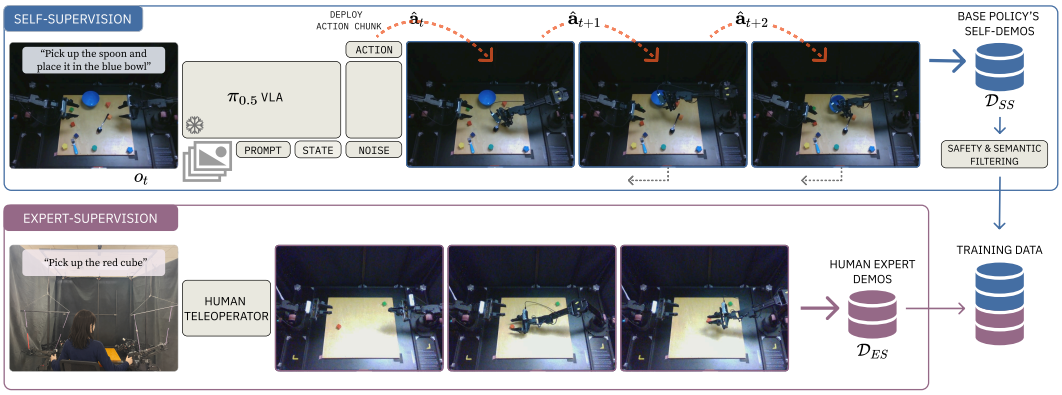


Figure 2: **A new way to fine-tune pretrained VLAs.** We combine two multi-task data sources: (1) *Bottom row:* expert-supervised demonstrations on a new task (E.g. "Pick up the red cube"), and (2) *Top row:* real-world interaction data generated by rolling out the frozen base policy on self-supervised tasks drawn from its pretraining distribution on the target robot (E.g. "Pick up the spoon and place it in the blue bowl"). Rehearsing the expert demonstrations alongside the base policy's self-demonstrations preserves its semantic understanding, instruction following, and behavioral priors; yielding a multi-task policy that can be deployed to perform both new and pretrained tasks on the target robot. Red arrows (⇢) denote action chunks generated by the base VLA policy.

H and action dimension of D. The VLM backbone takes in image patch tokens encoded by the SigLIP vision encoder and a set of text tokens encoded through the PaliGemma (Beyer et al., 2024) vocabulary. The text tokens consist of an ordered sequence of prompt tokens p, discretized state $\mathbf{q}_t$, and an additional set of discrete action tokens called FAST $\mathbf{a}^{\text{FAST}}_{1:M}$. FAST tokens are only used during training and mapped from ground truth actions via a discrete cosine transform quantization (Pertsch et al., 2025).

3.2 Expert-Supervised Imitation Learning

We define a multi-task fine-tuning setup for imitation learning. Let $\mathcal{D}_{\text{ES}} = \{(\mathbf{o}_t, \mathbf{a}_t)\}$, with $\mathbf{o}_t = (\mathbf{I}_t, p_{\text{ES}}, \mathbf{q}_t)$, be a dataset of observation–action pairs drawn from N *expert supervised* (ES) trajectories collected on our embodiment using teleoperation or a motion planner (Figure 2 bottom row), where each trajectory is associated with a task prompt $p_{\text{ES}} \in \mathcal{P}_{\text{ES}}$. $\mathcal{P}_{\text{ES}}$ is selected such that the task behavior (which object to interact with and how) should be identifiable through vision and text prompts, and not just rely on vision.

The model is optimized through a dual training objective, combining cross-entropy on the discretized FAST tokens and conditional flow matching on the continuous actions. For an observation $\mathbf{o}_t$ and target action chunk $\mathbf{a}$, we define the per-sample loss:

$$\ell(\mathbf{o}_t, \mathbf{a};\, \theta) = \mathbb{E}_{\tau,\,\omega}\Big[\, L_{\text{CE}}\big(\mathbf{a}^{\text{FAST}}_{1:M},\; f^{\ell}_{\theta}(\mathbf{o}_t, \mathbf{a}^{\text{FAST}}_{<m})\big) \;+\; \alpha \left\| (\omega - \mathbf{a}) - f^{a}_{\theta}(\mathbf{a}^{\tau,\omega}, \mathbf{o}_t) \right\|^2 \Big], \tag{1}$$

where $L_{\text{CE}}(\cdot,\cdot)$ is the cross-entropy loss applied only over the FAST action token positions and tends to train and converge the VLM backbone better on actions. f^{a}_{θ} is the action expert's predicted velocity, and $\alpha \in \mathbb{R}$ is a trade-off coefficient balancing the discrete and continuous objectives. The expert-supervised objective is the expectation of this loss over $\mathcal{D}_{\text{ES}}$:

$$\mathcal{L}_{\text{ES}}(\theta) = \mathbb{E}_{(\mathbf{o}_t, \mathbf{a}_t)\sim\mathcal{D}_{\text{ES}}}\big[\, \ell(\mathbf{o}_t, \mathbf{a}_t;\, \theta)\,\big]. \tag{2}$$

The original $\pi_{0.5}$ paper used both FAST actions and sub-task generation, but these two components are not released as part of the openpi code. We re-implement the FAST conditioning for training, as it tends to help generalization. We skip the sub-task generation head, as the goal of our work is to understand the properties and behaviors of the underlying vision-, text-, and state-conditioned flow matching policy. We find that the best performance on new tasks is achieved by fully fine-tuning both the VLM and the action expert and use that as default, with more ablations shown in Table 8.

3.3 Self-Supervised Imitation Learning

In addition to the expert imitation data, we propose to collect the pretrained policy's self-demonstrations on familiar tasks. Concretely, we roll out the frozen base VLA checkpoint ${\pi_{0.5}}^{\text{base}}$ prompting it on a set of self-supervised task instructions $p_{\text{SS}} \in \mathcal{P}_{\text{SS}}$ that are within the base model's pretraining family of tasks, *on our robot*. This generates trajectory motion behavior that correlates with the task prompt from the base policy's priors on our specific robot. At each decision step t along the rollout, we sample a teacher action chunk from the base policy's conditional flow distribution,

$$\hat{\mathbf{a}}_t \sim {\pi_{0.5}}^{\text{base}}\big(\cdot \mid \mathbf{I}_t, p_{\text{SS}}, \mathbf{q}_t\big), \tag{3}$$

execute these actions to take steps in the environment, and record the resulting observation–action pair. Aggregating these pairs over all trajectory rollout steps and self-supervised tasks yields a rehearsal dataset $\mathcal{D}_{\text{SS}} = \{(\mathbf{o}_t, \hat{\mathbf{a}}_t)\}$, with $\mathbf{o}_t = (\mathbf{I}_t, p_{\text{SS}}, \mathbf{q}_t)$ as shown in the top row of Figure 2. Since both observations and actions are collected along the teacher's own rollouts, every state is reached by executing the base policy's actions on our robot, in our scenes, under our prompts, so there is no embodiment or domain gap with respect to deployment. The self-supervised objective is the expectation of the per-sample loss in Eq. 1 over $\mathcal{D}_{\text{SS}}$, with the teacher's sampled predictions $\hat{\mathbf{a}}_t$ (Eq. 3) serving as target actions:

$$\mathcal{L}_{\text{SS}}(\theta) = \mathbb{E}_{(\mathbf{o}_t, \hat{\mathbf{a}}_t) \sim \mathcal{D}_{\text{SS}}}\big[\, \ell(\mathbf{o}_t, \hat{\mathbf{a}}_t;\, \theta)\,\big]. \tag{4}$$

The model is then jointly optimized on both objectives:

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{ES}}(\theta) + \lambda\, \mathcal{L}_{\text{SS}}(\theta), \tag{5}$$

where $\lambda \in \mathbb{R}$ controls the relative weight of the self-supervision term. Interestingly, despite often unsuccessful in completing the task, the self-supervised demonstrations improve learning by increasing task coverage with the new robot/environment and retaining instruction following and pretrained behavior through distillation.

Self-demo filtering. We conduct semantic and safety filtering on the self-demos. A rollout is discarded if it never approaches the specified object even once (Figure 4 shows self-demo distribution after filtering). We also filter out any self-demos with undesired behavior before distillation, e.g., if it makes unsafe contact such as wedging the grippers against the basket's rim in laundry task or the arm and eye-in-hand camera going inside the basket and getting stuck.

Data collection and mixture. We collect trajectories with different task instructions from the same initial scene, to ensure the model continues to use the text instruction rather than only the visual observation to identify the task. Training with a balance of expert- and self-supervised data performs best, using a 1:1 mixture and sampling uniformly in each. Task-weighted sampling hurts performance. In experiments with a physical robot, training with strongly imbalanced ratios leads to artifacts and erratic behavior.

4 Proposed Benchmarks and Protocol

We experiment on bimanual robots: the stationary ALOHA-1 (Zhao et al., 2023) in the real world and the ALOHA-Agilex embodiment in RoboTwin 2.0 (Chen et al., 2025) in simulation. In real we post-train the pretrained $\pi_{0.5}$ directly; in sim we mid-train first, then post-train. As summarized in Table 1, our benchmarks evaluate generalization across four test distributions: $\mathbf{T_{ES}}$: tasks from the distribution of expert supervised training; $\mathbf{T_{SS}}$: tasks from the distribution of self-supervised training; $\mathbf{T_{NO}}$: tasks with similar skills to T_{ES} but with novel objects not seen during training; and $\mathbf{T_{NC}}$: new tasks composed of skills from T_{ES}, e.g. stacking three blocks instead of two (RoboTwin only). All our test sets use random and diverse initial object positions. Simulated task setups are illustrated in Figure 5. The two settings are complementary. Simulation gives systematic, repeatable evaluation with no embodiment gap between stages, isolating task-incremental learning. Real exposes what simulation cannot: the embodiment gap that blocks grasps on our robot, the zero-shot instruction following that exists only because the VLA is pretrained on real data, and the need to work from small expert datasets.

5 EXPERIMENTS

Implementation Details. The ALOHA has 7 DoFs in each arm (6 target joint angles and 1 parallel-jaw gripper position). $\pi_{0.5}$ predicts action chunks $\mathbf{a}_t \in \mathbb{R}^{H \times D}$, with $H = 50, D = 32$. In both real and sim, $H/2 = 25$ actions are executed open loop on the robot before querying the VLA for the next observation from the environment. All real experiments are at a control frequency of 30-50 Hz.

Metrics. ***Success rate (SR)*** measures if the policy interacts with the specified object, demonstrates the correct behavior, and completes the task. A policy may largely do the right thing, exhibit task relevant behavior but ultimately fail leading to a low success rate. Hence, we also report ***Instruction Following (IF)***: ***IF (pre-grasp)***, measuring how often the robot approaches and attempts to grasp the specified object. For multi-stage tasks, we additionally report ***IF (post-grasp)***, i.e. how often the arm moves to the correct container or insertion peg after an attempted pick-up.

5.1 ZERO-SHOT VLA BEHAVIOR

Real. **Base policy follows instructions zero-shot but cannot grasp on our robot (embodiment gap).** For Benchmark 1 in Table 1, $\pi_{0.5}$ *zero-shot* localizes and reaches the prompted object 80% on picking red/green cubes, 70% on picking novel objects but *grasps none* (0% SR). We attribute this to potential gaps in exact hardware configuration between our target robot and the pretraining data collection system, such as grippers used and top camera's pose. On the self-supervised pick-and-place tasks, it exhibits pretraining behavior: it moves to pick correct object 70% and laundry 65% times, fails grasp, moves towards container "Y" 25% and laundry basket 30% attempting to place, but always fails (0% SR; the 5% SR on laundry is an accidental gripper slip).

Sim. $\pi_{0.5}$ *zero-shot* is not reported to contain simulation data in its pretraining (Physical Intelligence et al., 2025), scoring 0% on all tasks. We therefore mid-train on these 10 tasks with expert supervision from RoboTwin's motion planner to obtain a *RoboTwin base policy* (90.8%, Table 2). Unlike prior benchmarks that assume persistent access to mid-training data, we restrict access to it during post-training, mirroring real-world deployment.

5.2 CATASTROPHIC FORGETTING IN VLA POST-TRAINING

Real. **Fine-tuning to close embodiment gap causes forgetting of generalist instruction following, multi-task better than single-task.** Fine-tuning on 30 expert demos of a single task, "Pick up the green cube" (*Single Task ES*) degrades instruction following, relying more on visual observations collapsing the text conditioning. This ignorance of text has been called 'visual-action shortcut learning' (Xu et al., 2025). Splitting the same budget across two tasks in a shared scene with both objects in each scene (15 "red" + 15 "green") recovers it: *Multi-Task ES* improves IF from 40% to 50% on T_{NO} and 45% to 75% on T_{SS}. We also ablate this choice of multiple instructions in the same scene (requires text to specify the task) by training *Multi-Task ES, Single Object/Scene* with 15 "red" and 15 "green" demos having only one object in each scene and find it has worse IF.

Real. **Expert supervision alone forgets pretraining task motion priors.** Despite better instruction following, *Multi-Task ES* overfits the motion to the expert task family ("Pick up X") at the expense of the pretrained pick-and-place behavior (25% and 30% zero-shot post-grasp IF for placing)

Sim. **Multi-Task expert fine-tuning forgets catastrophically.** Similar to our real benchmark, expert post-training is done on two stacking tasks in a shared scene ("Stack red block on green block" and "Stack green block on red block"). Naive *Multi-Task ES* fine-tuning learns T_{ES} well (93.0%), but catastrophically forgets mid-training tasks (T_{SS}), dropping by 74.2% on average.

5.3 REHEARSING SELF-DEMOS IMPROVES BOTH SELF-SUPERVISED AND EXPERT TASKS

Real. **Benchmark 1, pick-up and pick-and-place.** *Multi-Task ES+SS* (i) distills the place behavior *purely* from the base policy's self-demos, with no expert place demos, reaching 90% post-grasp IF and 55% SR on pick-and-place as compared to 0% for Multi-Task ES; (ii) closes the hardware embodiment gap through expert data, reaching 90% SR on the expert cubes and 55% on novel objects; (iii) attains the best IF on T_{ES} and T_{SS} test sets; achieving the best SR on T_{NO}, grasping

Table 1: **ALOHA Robot.** Multi-task policy results across a variety of *pick up-only* and *pick-and-place* tasks. Our method not only outperforms $\pi_{0.5}$ zero-shot and fine-tuning baselines for tasks that are expert demonstrated T_{ES} like "Pick up red/green cube", but also enables task success on various pick and place tasks purely distilled through the zero-shot policy's self-supervised demos. Our approach enables fine-tuning to get a MT policy ready to deploy on our robot with only 2.2 minutes of expert teleoperation data (30 episodes, 6121 frames), while retaining strong generalization priors from the base VLA. Each row's results are from a single model. All values are percentages over 20 trials, except *Single Task ES* on T_{ES} which uses 10 trials.

	T_{ES}: Expert Supervised Tasks ("Pick up red/green cube")		T_{NO}: Novel Object Tasks ("Pick up X")		T_{SS} Self-Supervised Tasks ("Pick up X and place it in Y")			("Pick up laundry and place in basket")		
Method	IF (pre-grasp)	SR	IF (pre-grasp)	SR	IF (pre-grasp)	IF (post-grasp)	SR	IF (pre-grasp)	IF (post-grasp)	Partial SR
$\pi_{0.5}$ Zero-Shot	80	0	**70**	0	70	25	0	65	30	5
$\pi_{0.5}$ Single Task ES	90	50	40	25	45	25	0	—	—	—
$\pi_{0.5}$ Multi-Task ES, Single Object/Scene	65	65	45	30	55	35	10	—	—	—
$\pi_{0.5}$ Multi-Task ES	**100**	65	50	50	75	0	0	**100**	50	10
$\pi_{0.5}$ **Multi-Task ES+SS (Ours)**	**100**	**90**	55	**55**	**90**	**90**	**55**	**100**	**95**	**40**

Table 2: **RoboTwin Simulation.** In mid-training, policies are fine-tuned on 10 tasks. Post-training on new expert tasks (T_{ES}) while rehearsing self-demos from mid-training stage (T_{SS}) prevents catastrophic forgetting in expert-only. Success rates (%, ↑) averaged across 50 seeds per task.

	T_{SS}: Mid-training tasks (old)											T_{ES}: Post-training tasks (new)		
Experiment	Place Shoes	Beat Hammer	Pick Bottles	Place Plate	Handover Mic	Place Cup	Place Basket	Lift Pot	Open Micro.	Put Dustbin	**Avg.** T_{SS}	Stack $R \rightarrow G$	Stack $G \rightarrow R$	**Avg.** T_{ES}
$\pi_{0.5}$ Zero-Shot	–	–	–	–	–	–	–	–	–	–	0	–	–	–
Mid-Training Experiments														
$\pi_{0.5}$ RoboTwin Base Policy	76	94	98	100	100	98	82	98	72	90	**90.8**	–	–	–
Post-Training Experiments: All models initialized from RoboTwin Base Policy														
Multi-Task ES	0	0	0	60	0	36	34	36	0	0	16.6	90	96	93
Multi-Task ES+SS (Ours)	**42**	**82**	**48**	**92**	**62**	**92**	**58**	**86**	**66**	**78**	**70.6**	**98**	**98**	**98**

every object it localizes. (iv) Achieves 40% partial SR on the harder laundry task where self-demo priors are noisy and the cloth needs to be lifted over the baskets rim to place.

Real. **Benchmark 2, caterpillar gear insertion.** Caterpillar gear insertion (Table 10) is a precise, contact-rich bimanual task: the policy must localize the specified gear, grasp it, move to the correct gear color, and align it onto the specified peg shaft. It is a peg-in-hole insertion with tight alignment tolerances, made harder by a non-fixed peg board (the other arm adjusts the board if required). We even vary board positions across test sets. The 0% and 30% $\pi_{0.5}$ *zero-shot* instruction following indicates the task and its objects lie largely outside the base policy's pretraining. Adding self-supervision raises SR from 40% to 90% on the expert-demonstrated gear colors (T_{ES}), with post-grasp IF rising from 40% to 90%, Figure 3. On three unseen colors (T_{NO}), SR improves from 10% to 20% and post-grasp IF from 30% to 40%.

Table 3: **ALOHA Robot Caterpillar.** This is a complex, long-horizon bimanual task of colored peg insertion on a non-stationary board. This task is out-of-distribution w.r.t. base policy's pretraining. Using expert- and self-supervised fine-tuning (*Multi-task ES+SS*) improves all metrics across in-distribution gear colors (T_{ES}) and out-of-distribution gear colors (T_{NO}). (%, 10 trials)

	T_{ES}: Expert Supervised Tasks ("Pick up gear X and insert it into gear shaft X")			T_{NO}: Novel Object Tasks ("Pick up gear X and insert it into gear shaft X", novel gears)		
Method	IF (pre-grasp)	IF (post-grasp)	SR	IF (pre-grasp)	IF (post-grasp)	SR
$\pi_{0.5}$ Zero-Shot	0	0	0	20	0	0
$\pi_{0.5}$ Multi-Task ES	100%	40%	40%	70%	30%	10%
$\pi_{0.5}$ **Multi-Task ES+SS (Ours)**	**100%**	**90%**	**90%**	**80%**	**40%**	**20%**

Sim. **Robotwin Benchmark.** Fine-tuning with only 10 self-demos per task in the simulator *Multi-Task ES+SS* recovers 72.8% of the forgotten T_{SS} performance (reaching 70.6%), recovery spanning diverse skills (hammering, opening microwaves, pick-and-place, etc). It also achieves peak plasticity on expert stacking tasks (98%) against 93% with *Multi-Task ES*.

5.4 Comparison with upper bounds

Real. The naive alternative to our method is to teleoperate expert demos for all tasks, including those in T_{SS}. This *MT ES+ES* setting removes the data-availability constraint our method is designed for, and serves as an upper-bound reference point. The upper bound matches or slightly exceeds ours,

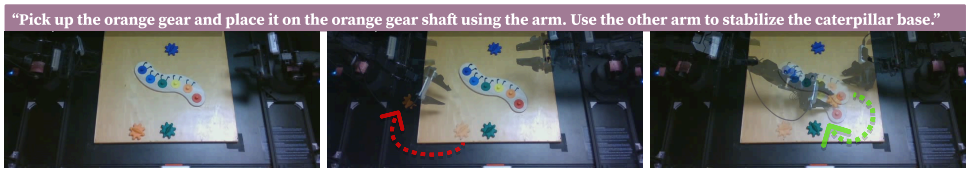


Figure 3: A rollout from the caterpillar test set showing successful alignment of orange gear on peg.

Table 4: **ALOHA Robot: comparison with full expert supervision.** *ES+ES* is an upper-bound reference that collects expert human teleoperation on *our* robot for *all* tasks, both T_{ES} and T_{SS}, replacing our self-supervised demonstrations with expert data on the same scenes and task list. *Multi-Task ES+SS (Ours)* is the same checkpoint from Table 1. (%, 20 trials)

	T_{ES}: Expert Supervised ("Pick up red/green cube")		T_{NO}: Novel Object ("Pick up X")		T_{SS}: Self-Supervised Tasks ("Pick up X and place it in Y")			("Pick up laundry and place in basket")		
Method	IF (pre-grasp)	SR	IF (pre-grasp)	SR	IF (pre-grasp)	IF (post-grasp)	SR	IF (pre-grasp)	IF (post-grasp)	Partial SR
$\pi_{0.5}$ Multi-Task ES+ES (Recollect Expert Oracle)	100	95	65	60	90	90	65	100	100	90
$\pi_{0.5}$ **Multi-Task ES+SS (Ours)**	100	90	55	55	90	90	55	100	95	40

Table 4: 95% vs. 90% SR on T_{ES}, 65% vs. 55% on T_{SS} pick-and-place, with instruction following tied on 3/4 task families. The gap is larger on the laundry task (90% vs. 40% SR). As self-demos are noisier, the *MT ES+SS* policy releases the cloth near the edge of the basket and it often falls out.

Sim. **Oracle baselines** (Table 5). These assume access to stored mid-train expert data, which our setup avoids. *Rehearsal (Oracle)* fully fine-tunes on stored mid-train data plus new expert data. *Rehearsal + LoRA (Oracle)* tests the claim from Liu et al. (2026a) that PEFT with replay mitigates forgetting. Ours instead generates 10 self-demos per task, trains on this mix of success and failure rollouts, without storing any old data.

Table 5: **RoboTwin Simulation.** Oracle baselines rehearse stored expert data from mid-training. *Ours:* rehearsing 10 self-demos per task recovers 82.5% of the rehearsal oracle's T_{SS} performance, landing 0.2 points below it Overall and above Rehearsal + LoRA. Average test set SR (%, ↑).

Experiment	T_{SS} (old)	T_{ES} (new)	T_{NO}	T_{NC}	**Overall**
Multi-Task ES+ES (Rehearsal Oracle)	85.6	97.0	30.0	15.3	57.0
Multi-Task ES+ES (Rehearsal + LoRA Oracle)	84.6	77.0	29.5	11.3	50.6
Multi-Task ES	16.6	93.0	**57.0**	6.7	43.3
Multi-Task ES+SS (Ours)	**70.6**	**98.0**	44.5	**14.0**	**56.8**

5.5 Self-demos retain held-out skill families which expert oracle forgets

Real. In Table 6, we evaluate the *MT ES+ES* versus *MT ES+SS (Ours)* checkpoints from Table 4 on two held-out skill families that are absent from both expert- or self-supervised data. We find self-supervision is more effective at retaining the motion priors of both "Push objects" and "Pour/turn over" skills. Since the presence of a motion prior in a rollout is subjective, we assess it with a blind user study rather than a fixed success criterion. While IF is comparable across the 2 policies, motion prior is far better retained by our method. Ours attempts the requested push on 82% of rated rollouts vs. 38% for *Multi-Task ES+ES*, and the pour/turn on 94% vs. 43% ($p < 0.001$). On push, the oracle instead picks the container up in 43% of rollouts.

6 Ablation and Analysis

Augmenting self-supervision helps even when expert data already covers the task. Table 7 isolates the contribution of augmenting pick-place self-demos in addition to expert pick-place teleoperation demonstrations for fine-tuning only on pick-place task family. Here, we conducted blind A/B testing of the two policies on all 20 scenes from T_{SS} test set.

Qualitative Results. Overall, our method yields smooth rollouts personalized to our target robot's desired motion range, as compared to the jerkier rollouts from the base policy which often raises the two arms higher than the task requires. Our method post-trains the VLA to stay sensitive to text instructions (which object to go to and how to interact with it), and has better recovery behavior when objects sit at table edges; it turns to look for objects behind the wrist camera unlike baselines.

Scaling self-demos (sim). We scaled up self-demos in simulation from 5 to 50 per task without filtering. Average performance does not change substantially as we scale up self-demos. But reten-

Table 6: **Blind user study on held-out skills.** *Multi-Task ES+ES (Oracle)* → *Multi-Task ES+SS (Ours)*; 6 raters × 20 scenes per task (%). All differences except IF have $p < 0.001$ (exact McNemar; sign test for preference, ties omitted).

Metric	Push	Pour / turn over
IF (correct object)	82 → 84	91 → 97
Attempted the requested motion	38 → **82**	43 → **94**
Picks up instead ↓	43 → **16**	—
SR (correct object)	22 → **71**	18 → **51**
Preferred by raters	9 → **68**	11 → **75**

Table 7: **Self-demos help even when expert data covers the task.** Adding pick-place self-demos to expert pick-place demos ("Pick up X and place it in Y"); blind A/B on the 20 T_{SS} scenes. (%)

Method	IF (pre-grasp)	IF (post-grasp)	SR
Multi-Task ES (pick-place)	75	**85**	55
Multi-Task ES+SS (pick-place)	**95**	80	**65**

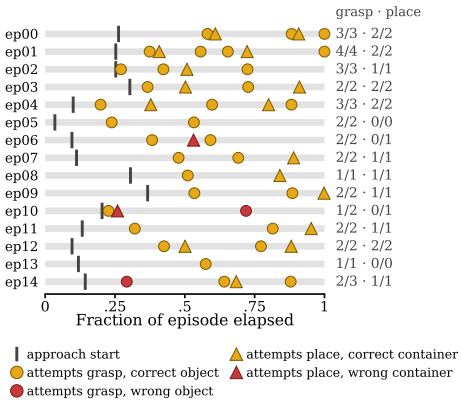


Figure 4: **Self-demo analysis** for the 15 rollouts from $\pi_{0.5}$ used for training in Table 1. 94% of grasp and 89% of place attempts target the prompted object, 13/15 episodes retry (mean 2.1 grasp and 1.2 place attempts per episode), but no grasp succeeds resulting in task failure.

tion is better on tasks where base Robotwin policy does well and regresses on tasks where the base policy does not do well. The prior's presence matters more than the quantity.

Self-demo rollouts supply states, not just targets. We conduct an ablation on the *Multi-Task ES+ES* experiment where we query the frozen base $\pi_{0.5}$ on expert pick-place demo states, and fine-tune on these offline chunks instead of trajectory actions. We find 0% SR. This could be due to the expert rollout states being out-of-distribution for the base policy and per-timestep action chunks being temporally inconsistent.

Failure modes. Our method can sometimes inherit artifacts from the data it rehearses. For example, since neither the teleoperation nor the generative prior data shows the scene *after* a successful place, the policy is never taught when to stop, sometimes re-picks the placed object.

Limitations. We find two experiments where the embodiment gap propagated to the fine-tuned policy: (i) on the caterpillar checkpoint, *Multi-Task ES+SS* retains instruction following, but neither policy closes the grasp gap for smaller pick-place objects, possibly because the gears (from expert data) have larger radius than the standard pick-place objects. (ii) we conducted a control experiment on *Multi-task ES (pick-place) + SS (pick-place)* from Table 7 with lower quality self-demos (poor semantic filtering) and the performance dropped as compared to the expert-only. Currently, selecting prompts $\mathcal{P}_{\text{SS}}$ for self-supervised data collection and filtering on the target robot requires manual curation, and can be automated in the future for larger-scale self-supervised learning. We evaluate only $\pi_{0.5}$, but our post-training recipe can easily be extended to other robotic foundation models.

7 Conclusion and Future Work

Conclusion. Pretrained VLAs suffer from embodiment gaps on a new robot even under minor differences in camera placement, gripper, etc, and re-collecting data for every pretrained skill on that robot is expensive. We show the base policy can supply priors from its own pretraining: rolled out on the target robot, its own generated actions serve as rehearsal targets that need no access to pretraining data and the resultant states carry no domain gap. Self-supervised imitation learning helps mitigate forgetting across both simulation and real world results. This is a particularly surprising result in the real world. The self-supervised data mostly has failures, and yet training on them recovers the self-supervised tasks, improves the expert tasks, and preserves held-out skills. VLA pretraining thus serves not only as initialization but as a prior generator. Future works can extend this into a data flywheel, distilling successive rounds of policy-generated demos mixed with expert data. We hope this is a step towards making pretrained VLAs practical to adapt without losing generalization, especially for the broader robotics community that cannot pretrain one from scratch.

AI USE STATEMENT

In this work, we used generative AI tools for finding relevant papers in parallel literatures, in code debugging, and generating visualizations and plots during project development.

REFERENCES

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, S A deliberate, Marc'Aurelio Ranzato, and Philip HS Torr. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

Danny Driess, Jost Springenberg, Brian Ichter, Lili Yu, Adrian Li-Bell, Karl Pertsch, Allen Ren, Homer Walke, Quan Vuong, Lucy Xiaoyang Shi, et al. Knowledge insulating vision-language-action models: Train fast, run fast, generalize better. *Advances in Neural Information Processing Systems*, 38:102867–102888, 2026.

Yuxia Fu, Zhizhen Zhang, Yuqi Zhang, Zijian Wang, Zi Huang, and Yadan Luo. Mergevla: Cross-skill model merging toward a generalist vision-language-action agent. *arXiv preprint arXiv:2511.18810*, 2025.

Jensen Gao, Suneel Belkhale, Sudeep Dasari, Ashwin Balakrishna, Dhruv Shah, and Dorsa Sadigh. A taxonomy for evaluating generalist robot manipulation policies. *IEEE Robotics and Automation Letters*, 2026.

Genesis AI Team. GENE-26.5: Advancing Robotic Manipulation to Human-Level. https://www.genesis.ai/blog/gene-26-5-advancing-robotic-manipulation-to-human-level, May 2026. Accessed: 2026-05-26.

Asher J Hancock, Xindi Wu, Lihan Zha, Olga Russakovsky, and Anirudha Majumdar. Actions as language: Fine-tuning vlms into vlas without catastrophic forgetting. *arXiv preprint arXiv:2509.22195*, 2025.

Jiaheng Hu, Jay Shim, Chen Tang, Yoonchang Sung, Bo Liu, Peter Stone, and Roberto Martin-Martin. Simple recipe works: Vision-language-action models are natural continual learners with reinforcement learning. *arXiv preprint arXiv:2603.11653*, 2026.

Nikita Kachaev, Mikhail Kolosov, Daniil Zelezetsky, Alexey K Kovalev, and Aleksandr I Panov. Don't blind your vla: Aligning visual representations for ood generalization. *arXiv preprint arXiv:2510.25616*, 2025.

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.

Nupur Kumari, Bingliang Zhang, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Multi-concept customization of text-to-image diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1931–1941, 2023.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Huihan Liu, Changyeon Kim, Bo Liu, Minghuan Liu, and Yuke Zhu. Pretrained vision-language-action models are surprisingly resistant to forgetting in continual learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2026a.

Yuan Liu, Haoran Li, Shuai Tian, Yuxing Qin, Yuhui Chen, Yupeng Zheng, Yongzhen Huang, and Dongbin Zhao. Towards long-lived robots: Continual learning vla models via reinforcement fine-tuning. *arXiv preprint arXiv:2602.10503*, 2026b.

Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on robot learning*, pp. 1113–1132. Pmlr, 2020.

Sergi Masip, Pau Rodriguez, Gido van de Ven, V Lomonaco, S Melacci, T Tuytelaars, S Chandar, R Pascanu, et al. Continual learning of diffusion models with generative distillation. In *Proceedings of The 3rd Conference on Lifelong Learning Agents*, volume 274, pp. 431–456. PMLR, 2025.

NVIDIA GEAR Lab. GR00T N1.6: An improved open foundation model for generalist humanoid robots. https://research.nvidia.com/labs/gear/gr00t-n1_6/, December 2025. NVIDIA Technical Report.

Junhyuk Oh, Yijie Guo, Satinder Singh, and Honglak Lee. Self-imitation learning. In *International conference on machine learning*, pp. 3878–3887. PMLR, 2018.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Physical Intelligence, Bo Ai, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Greg Balke, Kevin Black, George Capsis Bokinsky, Shihao Cao, Thomas Charbonnier, et al. $\pi_{0.7}$: A steerable generalist robotic foundation model with emergent capabilities. *arXiv preprint arXiv:2604.15483*, 2026.

Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *2016 IEEE international conference on robotics and automation (ICRA)*, pp. 3406–3413. IEEE, 2016.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy P Lillicrap, and Greg Wayne. Experience replay for continual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

Ralf Römer, Yi Zhang, Yuming Li, and Angela P Schoellig. Clare: Continual learning for vision-language-action models via autonomous adapter routing and expansion. *IEEE Robotics and Automation Letters*, 2026.

Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 22500–22510, 2023.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017.

Yunzhou Song, Long Le, Yong-Hyun Park, Jie Wang, Junyao Shi, Lingjie Liu, Jiatao Gu, Eric Eaton, Dinesh Jayaraman, and Kostas Daniilidis. Omniguide: Universal guidance fields for enhancing generalist robot policies. *arXiv preprint arXiv:2603.10052*, 2026.

Shahram Najam Syed, Yatharth Ahuja, Arthur Jakobsson, and Jeff Ichnowski. Expres-vla: Specializing vision-language-action models through experience replay and retrieval. *arXiv preprint arXiv:2511.06202*, 2025.

Gemini Robotics Team, Abbas Abdolmaleki, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Ashwin Balakrishna, Nathan Batchelor, Alex Bewley, Jeff Bingham, et al. Gemini robotics 1.5: Pushing the frontier of generalist robots with advanced embodied reasoning, thinking, and motion transfer. *arXiv preprint arXiv:2510.03342*, 2025.

Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. *arXiv preprint arXiv:2506.15799*, 2025.

Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

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.

Yuxuan Wu, Guangming Wang, Zhiheng Yang, Maoqing Yao, Brian Sheil, and Hesheng Wang. Continually evolving skill knowledge in vision language action model. *arXiv preprint arXiv:2511.18085*, 2025.

Kechun Xu, Zhenjie Zhu, Anzhe Chen, Shuqi Zhao, Qing Huang, Yifei Yang, Haojian Lu, Rong Xiong, Masayoshi Tomizuka, and Yue Wang. Seeing to act, prompting to specify: A bayesian factorization of vision language action policy. *arXiv preprint arXiv:2512.11218*, 2025.

Yajat Yadav, Zhiyuan Zhou, Andrew Wagenmaker, Karl Pertsch, and Sergey Levine. Robust finetuning of vision-language-action robot policies via parameter merging. *arXiv preprint arXiv:2512.08333*, 2025.

Shuai Yang, Hao Li, Bin Wang, Yilun Chen, Yang Tian, Tai Wang, Hanqing Wang, Feng Zhao, Yiyi Liao, and Jiangmiao Pang. Instructvla: Vision-language-action instruction tuning from understanding to manipulation. *arXiv preprint arXiv:2507.17520*, 2025.

William Yue, Bo Liu, and Peter Stone. t-dgr: A trajectory-based deep generative replay method for continual learning in decision making. *arXiv preprint arXiv:2401.02576*, 2024.

Lihan Zha, Asher J Hancock, Mingtong Zhang, Tenny Yin, Yixuan Huang, Dhruv Shah, Allen Z Ren, and Anirudha Majumdar. Lap: Language-action pre-training enables zero-shot cross-embodiment transfer. *arXiv preprint arXiv:2602.10556*, 2026.

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.

Xueyang Zhou, Yangming Xu, Guiyao Tie, Yongchao Chen, Guowen Zhang, Duanfeng Chu, Pan Zhou, and Lichao Sun. Libero-pro: Towards robust and fair evaluation of vision-language-action models beyond memorization. *arXiv preprint arXiv:2510.03827*, 2025.