RAFT: REASONING ALIGNED FINE TUNING IMPROVES ROBUSTNESS OF VLA POST-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Adapting a generalist vision-language-action (VLA) policy to a target task with limited demonstrations can improve nominal task success while reducing robustness to variations in appearance, objects, and instructions. We introduce reasoning-aligned fine-tuning (RAFT), a two-stage post-training procedure: RAFT first trains low-rank adapters on task-relevant reasoning annotations, then freezes them while fine-tuning the policy on action demonstrations. On a physical DROID robot, RAFT achieves about 75% out-of-distribution success, compared with about 28% for plain fine-tuning; on a multi-stage bimanual YAM task, it reaches 70% mean success, compared with 36% for the strongest baseline. RAFT also improves success under visual, object, and instruction shifts in simulated pick-and-place and drawer-opening tasks. Compared with plain fine-tuning, RAFT policies place more attention on annotated task regions and yield higher linear-probe accuracy for object identity in later layers, and a broader reasoning-data mixture further improves downstream success across demonstration budgets.

1 INTRODUCTION

Generalist vision-language-action (VLA) policies, built on pretrained vision-language models and large robot datasets Anil et al. (2023); O'Neill et al. (2024); Khazatsky et al. (2024); Kim et al. (2024); Black et al. (2025); Bjorck et al. (2025); Collaboration et al. (2023), typically still require adaptation to reach the success rates needed to deploy the policy on a real-world target task Black et al. (2024); Kim et al. (2025). Because collecting task-specific demonstrations is expensive, this post-training often relies on a small dataset collected in a narrow set of environments. Fine-tuning can improve nominal success while reducing robustness to visual and semantic variations absent from those demonstrations Gao et al. (2026); Zhang et al. (2025); Wang (2025); Xiang et al. (2025); Kaplanis et al. (2019); Xing et al. (2025). Existing approaches address this problem through parameter merging, as in RETAIN Yadav et al. (2026), or regularization during adaptation Kachaev et al. (2025); Huang et al. (2026). We study a complementary approach: aligning a pretrained policy with task-relevant reasoning supervision before adapting it to the target actions.

Reasoning supervision can teach policies to identify task-relevant objects, spatial relations, and motion structure Zawalski et al. (2024); Fang et al. (2026); Zha et al. (2026); Li et al. (2026a). We study how reasoning capabilities in pretrained VLMs can support low-data adaptation of generalist VLAs to a target skill. To the best of our knowledge, reasoning supervision has not been studied for low-data post-training of a pretrained generalist VLA. We introduce reasoning-aligned fine-tuning (RAFT), a two-stage adaptation procedure. Stage I learns low-rank adapters in the language and vision modules from task-relevant reasoning annotations while keeping pretrained parameters fixed. Stage II freezes these adapters and fine-tunes the language model, action expert, and projection layers on action demonstrations. Action gradients therefore do not directly update the reasoning-adapter weights. We find that how reasoning supervision is incorporated matters: jointly co-training reasoning and actions underperforms plain fine-tuning (63.2% vs. 79.6% mean success on simulated pick and place), and updating the reasoning adapter during action learning loses much of the benefit (74.0% vs. 83.2%) (Secs. B.1.3 and B.1.2). Staging the two, with the adapter frozen, yields these robustness gains. RAFT policies also place more attention on annotated task regions and yield higher linear-probe accuracy for object identity and episode-time position than plainly fine-tuned policies (Sec. 4.3).

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.

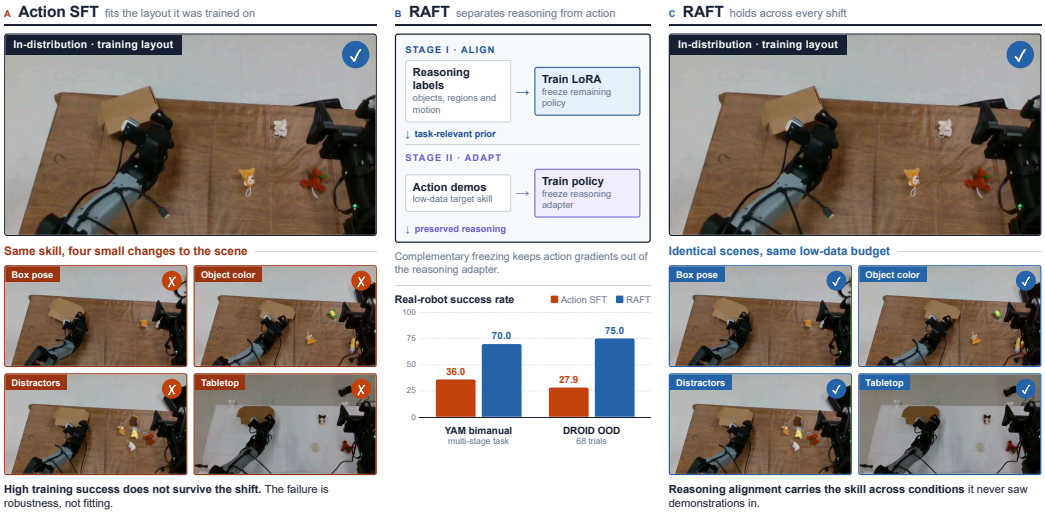

Figure 1: **Reasoning-aligned fine-tuning for robust VLA adaptation.** (a) A policy fine-tuned only on actions succeeds in the training layout but fails under small scene changes. (b) RAFT first learns task-relevant reasoning through LoRA adapters, then freezes them while fine-tuning the policy on action demonstrations. (c) With the same demonstrations, RAFT succeeds under these shifts. On physical robots, RAFT raises success over action-only fine-tuning from 35% to 70% on YAM and from 28% to 75% on out-of-distribution DROID trials.

We evaluate RAFT on two simulated tasks and two physical platforms: the single-arm DROID setup Khazatsky et al. (2024) and the bimanual YAM setup Fang et al. (2026). Evaluations test visual, object, instruction, and geometric variations of the adapted tasks. RAFT achieves the highest mean success on simulated pick and place and improves drawer-opening success, with the largest gains on the physical robots: 75% successful out-of-distribution DROID trials versus 28% for plain fine-tuning, and 70% mean success on the multi-stage YAM task versus 36% for the strongest baseline. Our contributions are threefold: (1) a post-training procedure that separates task-relevant reasoning supervision from action fine-tuning through complementary parameter freezing; (2) an evaluation of target-task robustness in simulation and on two physical robot platforms; and (3) measurements of attention, linearly decodable task information, and reasoning-data scaling in the resulting policies.

2 Related Work

Generalist robot policies have made substantial progress in learning transferable skills and supporting deployment through post-training Bousmalis et al. (2023); Kim et al. (2024); Team et al. (2024); Guo et al. (2025); Li et al. (2026b); Zang et al. (2025); Pan et al. (2026). Within this setting, work on VLA fine-tuning studies reliable task execution Kim et al. (2025); Zhang et al. (2025), while FAST improves action tokenization for autoregressive VLAs Pertsch et al. (2025). ControlVLA uses object-centric conditioning for few-shot adaptation Li et al. (2025). These advances make specialization more effective, but nominal task success can still mask failures under distribution shift Gao et al. (2026); Zhou et al. (2025); Fei et al. (2026). This motivates studying robustness to variations absent from the target demonstrations.

Robust post-training methods seek to adapt generalist policies while retaining useful pretrained capabilities. Weight-space approaches interpolate or merge pretrained and adapted parameters Wortsman et al. (2022); Jang et al. (2024); Dey et al. (2025), while regularization and steerability-preserving adaptation constrain changes during fine-tuning Kachaev et al. (2025); Grover et al. (2025); Sharma et al. (2023); Huang et al. (2026). RETAIN Yadav et al. (2026) interpolates pretrained and fine-tuned policy weights to improve generalization to variations of a new skill and retain broader capabilities. DeLock Huang et al. (2026) regularizes visual-encoder weight drift during post-training and uses

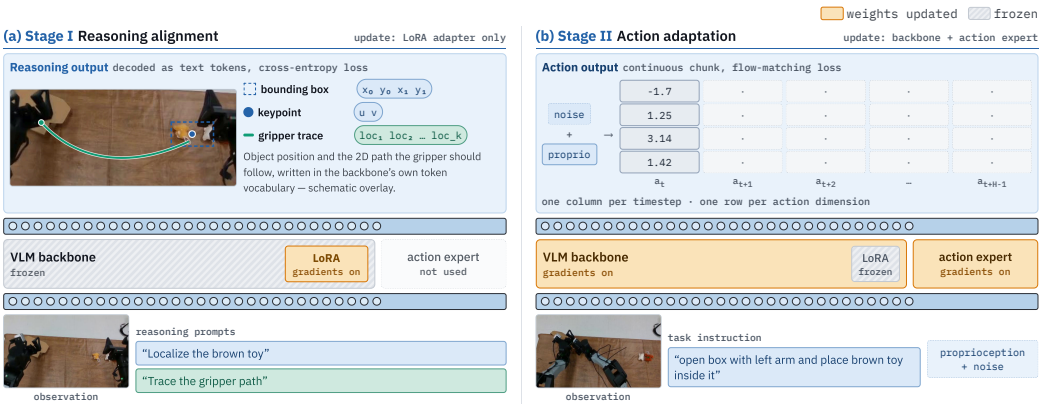

Figure 2: **Overview of RAFT.** Stage I updates only the reasoning adapters, using reasoning annotations. Stage II freezes the adapters and updates the language model, action expert, and projection layers on action demonstrations, so action gradients do not reach the reasoning adapters.

contrastive prompt guidance at inference. RAFT instead introduces task-relevant reasoning alignment before action fine-tuning, with complementary parameter freezing across the two stages.

Reasoning and grounding supervision provide a foundation for this approach. Embodied Chain-of-Thought interleaves sub-goal, spatial, and motion reasoning with action prediction Zawalski et al. (2024). Related work incorporates spatial representations, affordances, keypoints, and manipulable parts Fang et al. (2026); Zha et al. (2026); Li et al. (2026a), or object- and language-grounded supervision Black et al. (2025). Steerable Policies connect commands at multiple abstraction levels to control Chen et al. (2026), while VLS uses VLM-generated rewards to steer frozen policies at inference Liu et al. (2026). Our contribution concerns the post-training stage: adapting an already pretrained generalist VLA using limited target-task demonstrations. Reasoning supervision is used to prepare that policy for specialization, and the learned adapters are held fixed during subsequent action learning. We evaluate this adaptation schedule through downstream robustness, supplementary training controls, and analyses of task-relevant representations.

3 Method

3.1 Overview and Problem Setup

We consider task adaptation from a pretrained generalist policy using a small demonstration dataset $\mathcal{D}_\eta = \{(s_t, a_t, T)\}$ collected in a narrow set of environments. Here s_t contains the policy observations, a_t is the demonstrated action, and T is the task instruction. We denote the Stage-II trainable non-adapter parameters (the language model, action expert, and projection layers) by θ, initialized from θ_{pre}, and the LoRA parameters by ϕ Hu et al. (2022). In RAFT, the pretrained SigLIP weights remain frozen throughout adaptation, while LoRA modules in both the vision encoder and language model are trained in Stage I. Stage II freezes these learned adapters and updates θ. Thus, vision adaptation in RAFT occurs through Stage-I LoRA rather than updates to the pretrained SigLIP weights. The resulting action policy is $\pi_{\theta,\phi}(a_t \mid s_t, T)$. Figure 2 summarizes the two stages; Sec. A gives the adapter placement and optimization settings.

3.2 Stage I: Reasoning Alignment

Stage I trains the adapter to predict task-relevant reasoning annotations. A VLM annotator (Gemini 3.1 Pro) produces task-relevant reasoning targets from observations and instructions on $\mathcal{D}_\eta$, yielding $\mathcal{D}_{\text{reason}} = \{(s, r, T)\}$. These annotations describe the objects and interactions relevant to the task. The annotation choices include bounding boxes, keypoints, traces, spatial descriptions, and visual question answering. Section B.1.1 compares their effects. Each annotation is represented as a target token sequence $r = (r_1, \ldots, r_{|r|})$. We train the LoRA adapter using standard autoregressive

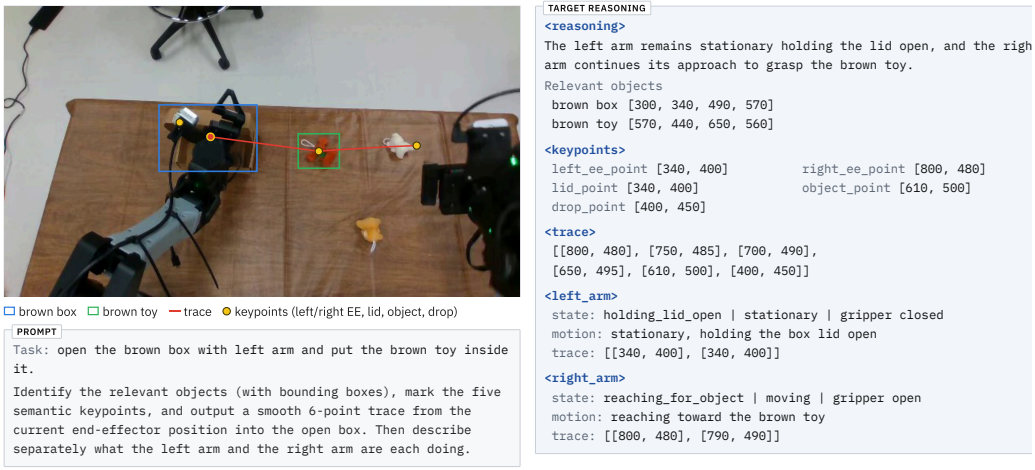

Figure 3: **Example Stage-I reasoning annotation.** In this YAM example, the left arm holds the lid open while the right arm reaches for the toy. The target combines language, object boxes, keypoints, motion traces, and per-arm state information. Blue and green mark the box and toy, yellow marks keypoints, and red marks the active arm's trace.

cross-entropy:

$$\mathcal{L}_{\text{reason}}(\phi) = -\mathbb{E}_{(s,r,T)\sim\mathcal{D}_{\text{reason}}} \sum_{j=1}^{|r|} \log p_{\theta_{\text{pre}},\phi}(r_j \mid s, T, r_{<j}). \tag{1}$$

Only ϕ is updated in this stage; all remaining parameters are frozen. This supervision encourages task-relevant grounding before action adaptation.

3.3 Stage II: Action Adaptation

Stage II turns the reasoning-aligned initialization into an action policy for the target task. We retain the learned adapter $\phi^\star$ and freeze its parameters. The Stage-II parameter set θ is then updated by supervised fine-tuning (SFT) on the task demonstrations Kim et al. (2024; 2025):

$$\theta^\star = \arg\min_\theta \underbrace{\mathbb{E}_{(s,a,T)\sim\mathcal{D}_\eta}\left[\ell_{\text{SFT}}(\pi_{\theta,\phi^\star}; s, a, T)\right]}_{\mathcal{L}_{\text{BC}}(\theta;\mathcal{D}_\eta,\phi^\star)}. \tag{2}$$

Here ℓ_{SFT} is the underlying policy's action-supervision objective. Stage II updates the language model, action expert, and projection layers while the adapter and SigLIP vision tower remain frozen. At deployment, the resulting policy is $\pi_{\theta^\star,\phi^\star}$. The supplementary adapter ablation (Sec. B.1.2) evaluates the benefit of freezing the reasoning adapter during action learning. Attention and linear-probe analyses (Sec. 4.3) provide empirical evidence of retained task grounding in the resulting policy.

4 Experiments

The goal of our experiments is to evaluate RAFT's ability to retain task-relevant information and improve downstream robustness. Concretely, we aim to answer: (1) does RAFT generalize better than baselines under distribution shift, in simulation and on physical robots? (2) do RAFT's representations retain more task-relevant information than a plainly fine-tuned policy, and does this track downstream success? (3) how does downstream performance scale with the amount of reasoning data used in Stage I? and (4) which design choices matter: does staged training outperform co-training reasoning and actions, and should the reasoning adapter stay frozen during action learning?

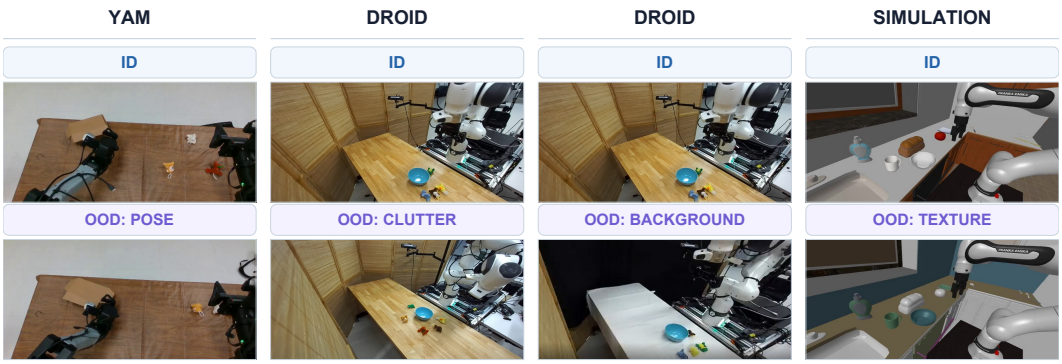

Figure 4: **Evaluation scenes.** Representative in-distribution (ID) and out-of-distribution (OOD) observations for simulation pick and place, simulation drawer opening, physical DROID pick and place, and the physical YAM two-stage task. The OOD panels show texture changes in simulation and background changes on the physical robots. The full set of evaluated shifts is shown in Fig. 20.

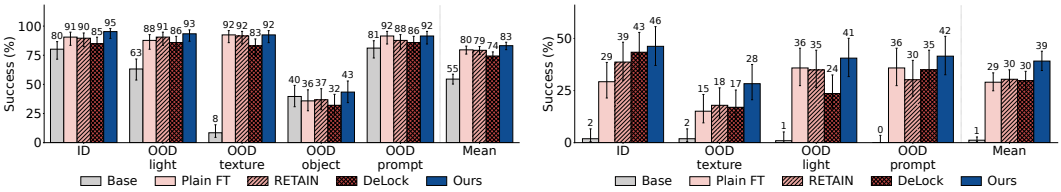

Figure 5: **Simulation results under distribution shift.** Success rates for pick and place (left) and drawer opening (right), with 106 trials per condition and 95% confidence intervals. RAFT uses full reasoning for pick and place and bounding-box supervision for drawer opening (Sec. B.1.1). RAFT has the highest mean success on both tasks: 83% versus 80% for Plain FT on pick and place, and 39% versus 29% on drawer opening.

4.1 Experimental setup

Baselines We compare RAFT with standard supervised fine-tuning (*Plain FT*), *RETAIN* Yadav et al. (2026), which merges pretrained and fine-tuned weights, and *DeLock* Huang et al. (2026), which regularizes visual-encoder weight drift during fine-tuning and applies contrastive prompt guidance at inference. We additionally report *Base*, the pretrained generalist policy evaluated without post-training. All fine-tuned policies are evaluated after 5,000 action-training steps. Sec. A.3 gives the baseline settings; Sec. 4.5 ablates the training design, with full adapter and co-training results in the supplementary material (Secs. B.1.2 and B.1.3).

4.2 RAFT generalizes under distribution shift

We evaluate RAFT on pick-and-place and drawer-opening tasks in simulation and on two real-world platforms: the single-arm DROID setup Khazatsky et al. (2024) and the dexterous, dual-arm YAM setup Fang et al. (2026), under the visual, object, and instruction or geometric shifts described in Sec. 4.1.

Simulation We evaluate MolmoB0t Deshpande et al. (2026) in the MolmoSpaces simulator Kim et al. (2026) on pick-and-place and drawer-opening tasks. For each task, we use 106 successful demonstrations from one layout provided in the MolmoSpaces benchmark for post-training. We measure robustness using perturbations absent from the demonstrations. *OOD-light* changes the number, position, and intensity of light sources; *OOD-texture* replaces table and scene textures; *OOD-object* substitutes a semantically similar manipulation object unseen during post-training and is evaluated only for pick and place; and *OOD-prompt* paraphrases the instruction without changing its meaning. *ID* uses the demonstration layout. Each evaluation split separately contains $n = 106$ trials with matched seeds across methods.

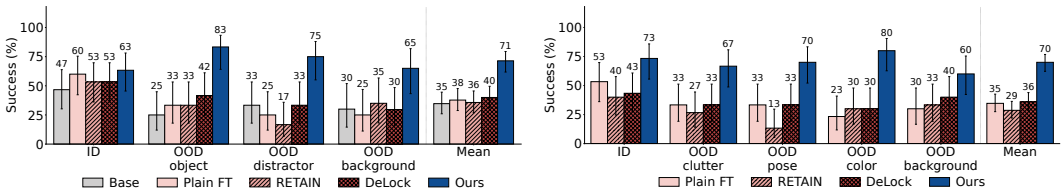

Figure 6: **Real-robot results under distribution shift.** Left: DROID pick and place, with 30 ID trials and 24, 24, and 20 trials under object, distractor, and background shifts. Right: YAM box opening and object placement, with 30 trials per condition. Error bars show Wilson 95% confidence intervals; means pool all trials per policy. RAFT has the highest success on both platforms: 75% versus 28% for Plain FT on out-of-distribution DROID trials, and 70% versus 35% on YAM.

RAFT has the highest simulated success, with the largest margin on novel objects On pick and place, RAFT achieves about 83% mean success across the five evaluation splits, compared with 80% for Plain FT, 79% for RETAIN, and 74% for DeLock (Fig. 5). The largest margin is under object shift: about 43% for RAFT versus 36% for Plain FT and 37% for RETAIN. On open drawer, RAFT reaches about 46% success in ID and 28% under texture shift, versus 29% and 15% for Plain FT, and it also leads under lighting and instruction shifts. Applying RETAIN to a trace-supervised RAFT checkpoint on open drawer gives about 33% mean success, versus 27% when RETAIN is applied to Plain FT and 35% for the RAFT checkpoint alone (Sec. B.1.4).

Real-world: DROID We post-train MolmoB0t Deshpande et al. (2026) on 50 pick-and-place demonstrations and evaluate on a physical DROID robot under four conditions: the demonstration setup (ID; blue, yellow, and brown toys), novel object colors (OOD-Object, e.g., a white toy in place of the brown one), added distractors (OOD-Distractor), and a changed tabletop surface (OOD-Background). On out-of-distribution DROID trials, RAFT succeeds about 75% of the time versus 28% for Plain FT (Fig. 6): 83% versus 33% with novel object colors, 75% versus 25% with distractors, and 65% versus 25% on the changed tabletop. In the demonstration setup, the two are close (63% versus 60%).

Real-world: YAM For the dual-arm YAM platform, we post-train $\pi_{0.5}$ Black et al. (2025) on 100 demonstrations of a long-horizon task: open a hinged box, pick up an object, and place it inside. Because placement depends on the box being open, the task separates completing the first interaction from completing the whole sequence. We evaluate in the demonstration setup and under four shifts: clutter, box pose, object color, and background. On the bimanual YAM task, RAFT completes the full sequence in about 70% of trials pooled over the five conditions, versus 36% for DeLock, 35% for Plain FT, and 29% for RETAIN (Fig. 6), and it has the highest success in every condition (about 73% in-distribution and 60%–80% under the four shifts). Plain FT usually opens the box but then fails to place the object. We hypothesize that the Stage-I annotations, which label each arm's phase and target separately (Fig. 3), help the policy track its progress through the sequence.

4.3 Representation Analysis

We examine representation dimensionality, linearly decodable task information, and visual attention to assess what RAFT preserves during post-training. The measurement protocol is in Sec. B.3.

Late-layer dimensionality correlates with downstream success We measure the effective dimensionality of the backbone's hidden states with the participation ratio (PR) Gao et al. (2017); Recanatesi et al. (2022); Jing et al. (2022):

$$\mathrm{PR} = \frac{(\sum_i \lambda_i)^2}{\sum_i \lambda_i^2}, \tag{3}$$

where λ_i are the eigenvalues of the covariance of mean-pooled hidden states across frames. Across six Stage-I reasoning variants, late-layer PR correlates with success on each task-matched evaluation axis; at layer 15, absolute PR has mean Spearman $\rho = 0.76$ (Fig. 19).

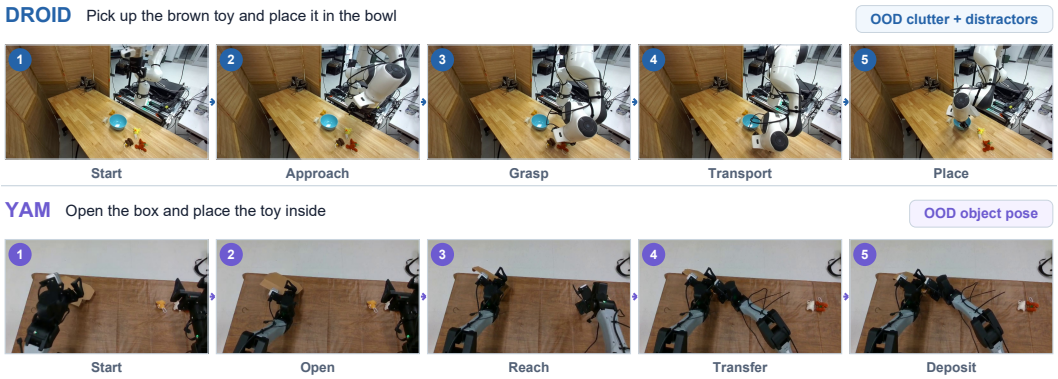

Figure 7: **Successful physical-robot rollouts.** Ordered frames show DROID picking up the brown toy and placing it in the bowl (top) and YAM opening the box and placing the toy inside (bottom). Numbers indicate temporal order; circles and arrows mark the manipulated object and motion.

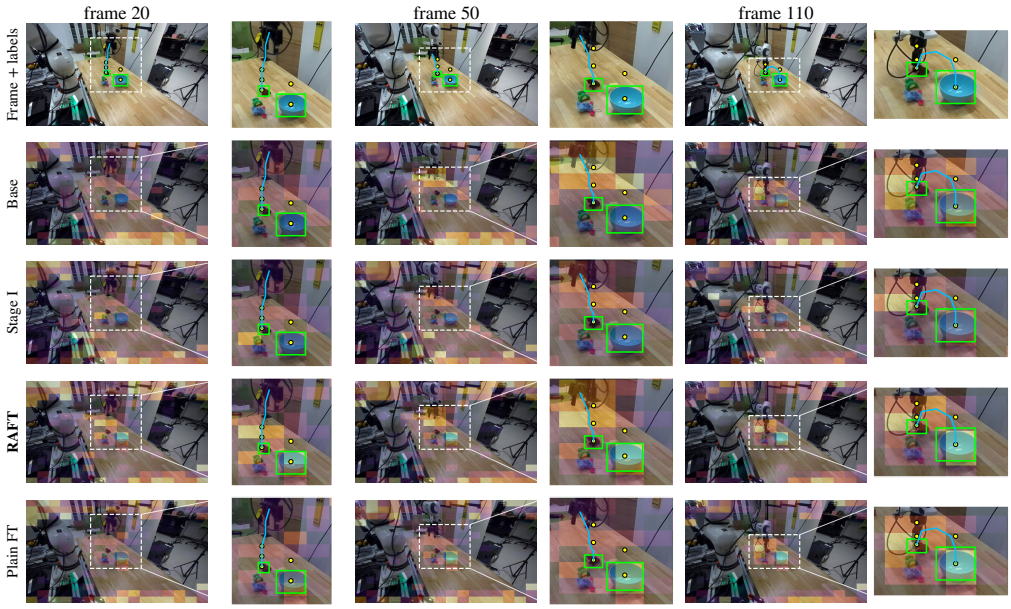

Figure 8: **Attention maps along one trajectory.** Columns show successive observations; rows compare Base, Stage I, RAFT, and Plain FT. Overlays identify annotated task regions, and color scales are shared within each column. RAFT places more attention on the annotated regions than Plain FT; Fig. 10 quantifies this across layers.

Object identity and task progress are more linearly decodable from RAFT's late layers We fit linear probes on the same late layers to predict object identity (seven classes, chance $1/7$) and episode-time quartile (four bins, chance 0.25). In layers 14–17, probes on RAFT reach 0.51–0.53 balanced accuracy for object identity, against 0.45 for Plain FT and 0.40 for the base policy, and 0.62–0.65 for episode quartile, against 0.55 and 0.51 (Fig. 9). Object-identity decoding also rises sharply between layers 13 and 15 for RAFT, but barely changes for Plain FT or the base policy. The layers where PR tracks success are the same layers where this information becomes decodable.

RAFT has higher attention enrichment on annotated task regions We measure how much of the policy's visual attention falls on annotated task regions, such as the manipulated object, the target, and the gripper, relative to their share of the image (enrichment; values above one mean attention is concentrated there). At layer 11, RAFT gives these regions 3.2–3.4 times their uniform share of attention, against 1.9–2.1 for Plain FT, while attention to the background stays near uniform (Figs. 8

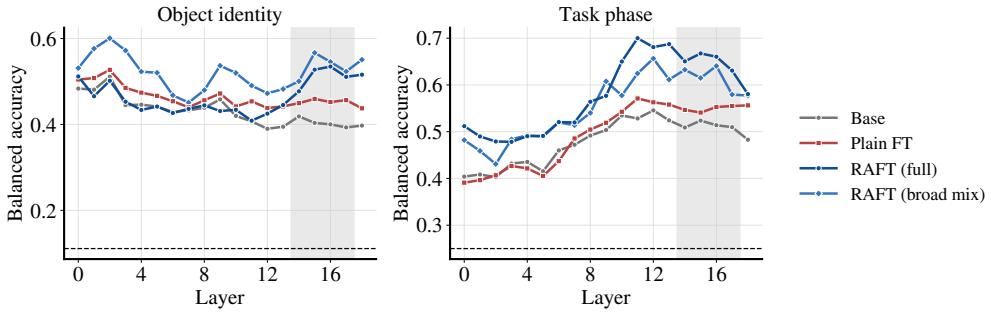

Figure 9: **Layer-wise linear probes.** Balanced accuracy for object identity (left; chance $1/7$) and episode-time quartile (right; chance $1/4$). Policies use the same 414 frames; episodes are held out from probe fitting, not policy training. Shading marks layers 14–17. In these layers, RAFT reaches 0.51–0.53 object-identity accuracy versus 0.45 for Plain FT, and 0.62–0.65 quartile accuracy versus 0.55.

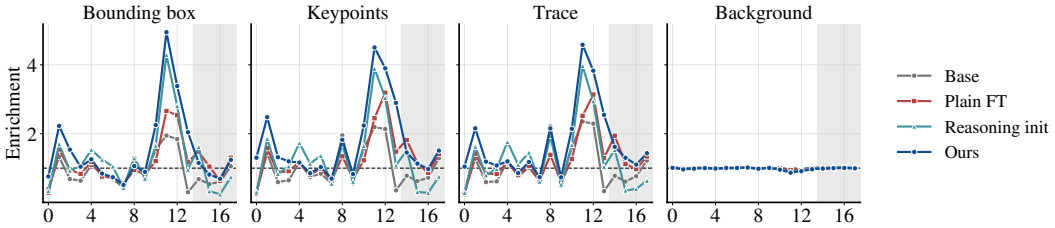

Figure 10: **Layer-wise attention enrichment.** Enrichment for object boxes, keypoints, traces, and background; values above one indicate attention exceeding a region's image-area share. Curves compare Base, Plain FT, Stage I, and RAFT; shading marks layers 14–17. At layer 11, RAFT gives task regions 3.2–3.4 times their area share, versus 1.9–2.1 for Plain FT, while background attention stays near uniform.

and 10); the separation narrows in later layers. The concentration persists through Stage-II action training and stays above that of plain fine-tuning.

The size of these effects depends on the reasoning modality used in Stage I; Sec. B.1.1 reports the full modality ablation.

4.4 Effect of reasoning data on held-out loss and downstream success

More reasoning data lowers held-out loss On a common 560-record held-out split, cross-entropy on full-reasoning targets falls as the number of reasoning records grows, in both frame- and episode-subsampled series (Fig. 11, left). How the corpus is built matters as much as its size: 628 frame-subsampled records reach a loss of 0.77, while 680 episode-subsampled records reach only 1.48. The broader mixture, which adds about 60,000 records to the 5,000-record in-domain corpus, gives the lowest loss (0.29, versus 0.62 in-domain) and still reaches 0.39 with target-simulator records removed. Increasing Stage-I reasoning data lowers held-out prediction loss.

Broader reasoning data also helps downstream Initializing Stage II from the broader-mixture adapter instead of the in-domain one improves success by 3–5 points at each of the three demonstration budgets (Fig. 11, middle). Against Plain FT at matched budgets (right panel), the two methods tie with 25% of the demonstrations (49 of 106 successes each), and RAFT leads modestly at intermediate budgets. With all demonstrations, RAFT succeeds in 101/106 ID trials against 96/106 for Plain FT (Fig. 5).

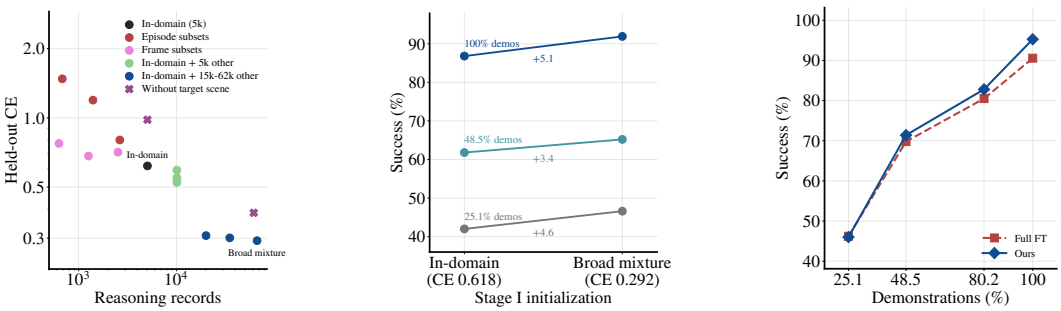

Figure 11: **Effect of reasoning data and demonstration budget.** Left: held-out reasoning loss across corpus sizes and constructions. Middle: downstream success from in-domain and broader-mixture Stage-I initializations at matched demonstration budgets. Right: RAFT versus Plain FT across demonstration budgets, with 106 evaluation episodes per budget. The broader mixture lowers held-out loss (0.29 versus 0.62) and improves downstream success by 3–5 points at every budget.

4.5 Staging and a frozen adapter matter

We ablate two design choices of RAFT on simulated pick and place: how reasoning and action supervision are combined, and whether the reasoning adapter is updated during action learning (Fig. 12). *Co-train* trains on reasoning and action examples jointly, sampling them equally and weighting their losses 1:1. *Adapter trainable* runs both stages but keeps the Stage-I adapter trainable in Stage II, and *Stage I only* applies the adapter without action fine-tuning.

Across five matched splits with 106 episodes each, co-training reaches 63.21% mean success, below both Plain FT (79.62%) and RAFT (83.21%). Updating the adapter in Stage II lowers success to 73.96%, and Stage I alone reaches 48.30%. Plain FT with a frozen vision encoder also stays at 79.62%, so freezing the pretrained vision weights alone does not account for the gain. Split-level results are in Tables 4 and 3.

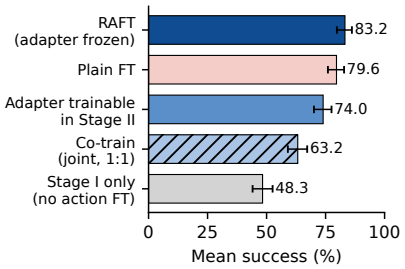

Figure 12: **Training-scheme ablation on simulated pick and place.** Mean success over five matched splits (106 trials each) with 95% Wilson intervals. Staged training with a frozen adapter achieves the highest success (83.2%), above Plain FT (79.6%), a trainable adapter (74.0%), and co-training (63.2%).

5 Conclusion

We presented RAFT, which fine-tunes a pretrained VLA on actions only after training, and then freezing, a low-rank adapter on task-relevant reasoning annotations. Across two simulated tasks and two real robots, this staging improved success under distribution shift relative to plain fine-tuning, RETAIN, and DeLock. Three questions remain open: whether RAFT preserves earlier skills over sequential task adaptations, how to choose the reasoning modality for a new task automatically, and how to reduce the motion lock-in, where a policy repeats a demonstrated motion despite changes in the scene, that underlies some remaining failures.

Reproducibility Statement

We describe the two-stage training procedure in Sec. 3.2 and provide the MolmoB0t training configuration, adapter placement, optimizer settings, hardware, and baseline implementations in Sec. A. The Gemini annotation response format is provided in Sec. A.1, with modality-specific target examples in Sec. A.2. Evaluation protocols specify matched simulation episode identities, physical-robot trial counts, resets, and success-rate uncertainty (Sec. A.4). The supplementary material reports the adapter, co-training, and frozen-vision controls (Secs. B.1.2 and B.1.3). Reported confidence intervals are 95% Wilson intervals over evaluation trials.

Ethics Statement

This work studies post-training for robot manipulation. Improved success on the evaluated tasks does not establish safety in unrestricted environments or around people. Generated reasoning annotations can contain incorrect object labels, locations, or motion descriptions, and the resulting policies can still fail under distribution shift. Deployment therefore requires application-specific validation, human oversight, and appropriate robot safety controls. Use of pretrained models, datasets, and annotation services should respect their licenses and applicable data-use terms; images sent to external annotation services should be checked for private or identifying information. The reported robustness and representation measurements should not be interpreted as a safety guarantee.

Use of Large Language Models

We used large language models to assist with writing, specifically editing text for clarity and grammar. All technical content, experiments, and results are the authors' own, and the authors reviewed every suggested edit and take full responsibility for the paper. Separately, as part of the method, a vision-language model (Gemini 3.1 Pro) generates the Stage-I reasoning annotations (Sec. 3.2).

References

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. PaLM 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

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.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert 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. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 27–30 Sep 2025. URL https://proceedings.mlr.press/v305/black25a.html.

Konstantinos Bousmalis, Giulia Vezzani, Dushyant Rao, Coline Devin, Alex X Lee, Maria Bauzá, Todor Davchev, Yuxiang Zhou, Agrim Gupta, Akhil Raju, et al. Robocat: A self-improving generalist agent for robotic manipulation. *arXiv preprint arXiv:2306.11706*, 2023.

William Chen, Jagdeep Singh Bhatia, Catherine Glossop, Nikhil Mathihalli, Ria Doshi, Andy Tang, Danny Driess, Karl Pertsch, and Sergey Levine. Steerable vision-language-action policies for embodied reasoning and hierarchical control. *arXiv preprint arXiv:2602.13193*, 2026.

Open X-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, Albert Tung, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anchit Gupta, Andrew Wang, Andrey Kolobov, Anikait Singh, Animesh Garg, Aniruddha Kembhavi, Annie Xie, Anthony Brohan, Antonin Raffin, Archit Sharma, Arefeh Yavary, Arhan Jain, Ashwin Balakrishna, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Blake Wulfe, Brian Ichter, Cewu Lu, Charles Xu, Charlotte Le, Chelsea Finn, Chen Wang, Chenfeng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Christopher Agia, Chuer Pan, Chuyuan Fu, Coline Devin, Danfei Xu, Daniel Morton, Danny Driess, Daphne

Chen, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dinesh Jayaraman, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Ethan Foster, Fangchen Liu, Federico Ceola, Fei Xia, Feiyu Zhao, Felipe Vieira Frujeri, Freek Stulp, Gaoyue Zhou, Gaurav S. Sukhatme, Gautam Salhotra, Ge Yan, Gilbert Feng, Giulio Schiavi, Glen Berseth, Gregory Kahn, Guangwen Yang, Guanzhi Wang, Hao Su, Hao-Shu Fang, Haochen Shi, Henghui Bao, Heni Ben Amor, Henrik I Christensen, Hiroki Furuta, Homanga Bharadhwaj, Homer Walke, Hongjie Fang, Huy Ha, Igor Mordatch, Ilija Radosavovic, Isabel Leal, Jacky Liang, Jad Abou-Chakra, Jaehyung Kim, Jaimyn Drake, Jan Peters, Jan Schneider, Jasmine Hsu, Jay Vakil, Jeannette Bohg, Jeffrey Bingham, Jeffrey Wu, Jensen Gao, Jiaheng Hu, Jiajun Wu, Jialin Wu, Jiankai Sun, Jianlan Luo, Jiayuan Gu, Jie Tan, Jihoon Oh, Jimmy Wu, Jingpei Lu, Jingyun Yang, Jitendra Malik, João Silvério, Joey Hejna, Jonathan Booher, Jonathan Tompson, Jonathan Yang, Jordi Salvador, Joseph J. Lim, Junhyek Han, Kaiyuan Wang, Kanishka Rao, Karl Pertsch, Karol Hausman, Keegan Go, Keerthana Gopalakrishnan, Ken Goldberg, Kendra Byrne, Kenneth Oslund, Kento Kawaharazuka, Kevin Black, Kevin Lin, Kevin Zhang, Kiana Ehsani, Kiran Lekkala, Kirsty Ellis, Krishan Rana, Krishnan Srinivasan, Kuan Fang, Kunal Pratap Singh, Kuo-Hao Zeng, Kyle Hatch, Kyle Hsu, Laurent Itti, Lawrence Yunliang Chen, Lerrel Pinto, Li Fei-Fei, Liam Tan, Linxi "Jim" Fan, Lionel Ott, Lisa Lee, Luca Weihs, Magnum Chen, Marion Lepert, Marius Memmel, Masayoshi Tomizuka, Masha Itkina, Mateo Guaman Castro, Max Spero, Maximilian Du, Michael Ahn, Michael C. Yip, Mingtong Zhang, Mingyu Ding, Minho Heo, Mohan Kumar Srirama, Mohit Sharma, Moo Jin Kim, Muhammad Zubair Irshad, Naoaki Kanazawa, Nicklas Hansen, Nicolas Heess, Nikhil J Joshi, Niko Suenderhauf, Ning Liu, Norman Di Palo, Nur Muhammad Mahi Shafiullah, Oier Mees, Oliver Kroemer, Osbert Bastani, Pannag R Sanketi, Patrick "Tree" Miller, Patrick Yin, Paul Wohlhart, Peng Xu, Peter David Fagan, Peter Mitrano, Pierre Sermanet, Pieter Abbeel, Priya Sundaresan, Qiuyu Chen, Quan Vuong, Rafael Rafailov, Ran Tian, Ria Doshi, Roberto Mart'in-Mart'in, Rohan Baijal, Rosario Scalise, Rose Hendrix, Roy Lin, Runjia Qian, Ruohan Zhang, Russell Mendonca, Rutav Shah, Ryan Hoque, Ryan Julian, Samuel Bustamante, Sean Kirmani, Sergey Levine, Shan Lin, Sherry Moore, Shikhar Bahl, Shivin Dass, Shubham Sonawani, Shubham Tulsiani, Shuran Song, Sichun Xu, Siddhant Haldar, Siddharth Karamcheti, Simeon Adebola, Simon Guist, Soroush Nasiriany, Stefan Schaal, Stefan Welker, Stephen Tian, Subramanian Ramamoorthy, Sudeep Dasari, Suneel Belkhale, Sungjae Park, Suraj Nair, Suvir Mirchandani, Takayuki Osa, Tanmay Gupta, Tatsuya Harada, Tatsuya Matsushima, Ted Xiao, Thomas Kollar, Tianhe Yu, Tianli Ding, Todor Davchev, Tony Z. Zhao, Travis Armstrong, Trevor Darrell, Trinity Chung, Vidhi Jain, Vikash Kumar, Vincent Vanhoucke, Vitor Guizilini, Wei Zhan, Wenxuan Zhou, Wolfram Burgard, Xi Chen, Xiangyu Chen, Xiaolong Wang, Xinghao Zhu, Xinyang Geng, Xiyuan Liu, Xu Liangwei, Xuanlin Li, Yansong Pang, Yao Lu, Yecheng Jason Ma, Yejin Kim, Yevgen Chebotar, Yifan Zhou, Yifeng Zhu, Yilin Wu, Ying Xu, Yixuan Wang, Yonatan Bisk, Yongqiang Dou, Yoonyoung Cho, Youngwoon Lee, Yuchen Cui, Yue Cao, Yueh-Hua Wu, Yujin Tang, Yuke Zhu, Yunchu Zhang, Yunfan Jiang, Yunshuang Li, Yunzhu Li, Yusuke Iwasawa, Yutaka Matsuo, Zehan Ma, Zhuo Xu, Zichen Jeff Cui, Zichen Zhang, Zipeng Fu, and Zipeng Lin. Open X-Embodiment: Robotic learning datasets and RT-X models. `https://arxiv.org/abs/2310.08864`, 2023.

Abhay Deshpande, Maya Guru, Rose Hendrix, Snehal Jauhri, Ainaz Eftekhar, Rohun Tripathi, Max Argus, Jordi Salvador, Haoquan Fang, Matthew Wallingford, et al. Molmob0t: Large-scale simulation enables zero-shot manipulation. *arXiv preprint arXiv:2603.16861*, 2026.

Sombit Dey, Jan-Nico Zaech, Nikolay Nikolov, Luc Van Gool, and Danda Pani Paudel. ReVLA: Reverting visual domain limitation of robotic foundation models. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

Haoquan Fang, Jiafei Duan, Donovan Clay, Sam Wang, Shuo Liu, Weikai Huang, Xiang Fan, Wei-Chuan Tsai, Shirui Chen, Yi Ru Wang, et al. Molmoact2: Action reasoning models for real-world deployment. *arXiv preprint arXiv:2605.02881*, 2026.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, et al. Libero-plus: A progressive robustness benchmark for visual-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 38574–38583, 2026.

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.

Peiran Gao, Eric Trautmann, Byron Yu, Gopal Santhanam, Stephen Ryu, Krishna Shenoy, and Surya Ganguli. A theory of multineuronal dimensionality, dynamics and measurement. *bioRxiv*, 2017. doi: 10.1101/214262.

Shresth Grover, Akshay Gopalkrishnan, Bo Ai, Henrik I Christensen, Hao Su, and Xuanlin Li. Enhancing generalization in vision-language-action models by preserving pretrained representations. *arXiv preprint arXiv:2509.11417*, 2025.

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. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 15665–15672. IEEE, 2025.

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.

Suning Huang, Jiaqi Shao, Ke Wang, Qianzhong Chen, Jiankai Sun, Yanjiang Guo, Mac Schwager, and Jeannette Bohg. Breaking lock-in: Preserving steerability under low-data VLA post-training. *arXiv preprint arXiv:2604.23121*, 2026.

Dong-Hwan Jang, Sangdoo Yun, and Dongyoon Han. Model stock: All we need is just a few fine-tuned models. In *European Conference on Computer Vision*, pp. 207–223. Springer, 2024.

Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding dimensional collapse in contrastive self-supervised learning. In *International Conference on Learning Representations (ICLR)*, 2022.

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.

Christos Kaplanis, Murray Shanahan, and Claudia Clopath. Policy consolidation for continual reinforcement learning. *arXiv preprint arXiv:1902.00255*, 2019.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

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.

Yejin Kim, Wilbert Pumacay, Omar Rayyan, Max Argus, Winson Han, Eli VanderBilt, Jordi Salvador, Abhay Deshpande, Rose Hendrix, Snehal Jauhri, et al. Molmospaces: A large-scale open ecosystem for robot navigation and manipulation. *arXiv preprint arXiv:2602.11337*, 2026.

Fuhao Li, Wenxuan Song, Han Zhao, Jingbo Wang, Pengxiang Ding, Donglin Wang, Long Zeng, and Haoang Li. Spatial forcing: Implicit spatial representation alignment for vision-language-action model. In *International Conference on Learning Representations (ICLR)*, 2026a.

Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaohui Yang, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, et al. SimpleVLA-RL: Scaling VLA training via reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2026b.

Puhao Li, Yingying Wu, Ziheng Xi, Wanlin Li, Yuzhe Huang, Zhiyuan Zhang, Yinghan Chen, Jianan Wang, Song-Chun Zhu, Tengyu Liu, et al. Controlvla: Few-shot object-centric adaptation for pre-trained vision-language-action models. *arXiv preprint arXiv:2506.16211*, 2025.

Shuo Liu, Ishneet Sukhvinder Singh, Yiqing Xu, Jiafei Duan, and Ranjay Krishna. Vls: Steering pretrained robot policies via vision-language models. *arXiv preprint arXiv:2602.03973*, 2026.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

Mingjie Pan, Siyuan Feng, Qinglin Zhang, Xinchen Li, Jianheng Song, Chendi Qu, Yi Wang, Chuankang Li, Ziyu Xiong, Zhi Chen, et al. Sop: A scalable online post-training system for vision-language-action models. *arXiv preprint arXiv:2601.03044*, 2026.

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.

Stefano Recanatesi, Serena Bradde, Vijay Balasubramanian, Nicholas A. Steinmetz, and Eric Shea-Brown. A scale-dependent measure of system dimensionality. *Patterns*, 3(8):100555, 2022. doi: 10.1016/j.patter.2022.100555.

Mohit Sharma, Claudio Fantacci, Yuxiang Zhou, Skanda Koppula, Nicolas Heess, Jon Scholz, and Yusuf Aytar. Lossless adaptation of pretrained vision models for robotic manipulation. *arXiv preprint arXiv:2304.06600*, 2023.

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.

Yixiao Wang. Generalization capability for imitation learning. *arXiv preprint arXiv:2504.18538*, 2025.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, et al. Robust fine-tuning of zero-shot models. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7949–7961. IEEE, 2022.

Tian-Yu Xiang, Ao-Qun Jin, Xiao-Hu Zhou, Mei-Jiang Gui, Xiao-Liang Xie, Shi-Qi Liu, Shuang-Yi Wang, Sheng-Bin Duan, Fu-Chao Xie, Wen-Kai Wang, et al. Parallels between vla model post-training and human motor learning: Progress, challenges, and trends. *arXiv preprint arXiv:2506.20966*, 2025.

Youguang Xing, Xu Luo, Junlin Xie, Lianli Gao, Hengtao Shen, and Jingkuan Song. Shortcut learning in generalist robot policies: The role of dataset diversity and fragmentation. *arXiv preprint arXiv:2508.06426*, 2025.

Yajat Yadav, Zhiyuan Zhou, Andrew Wagenmaker, Karl Pertsch, and Sergey Levine. Robust fine-tuning of vision-language-action robot policies via parameter merging. In *International Conference on Learning Representations (ICLR)*, 2026.

Hongzhi Zang, Mingjie Wei, Si Xu, Yongji Wu, Zhen Guo, Yuanqing Wang, Hao Lin, Peihong Wang, Liangzhi Shi, Yuqing Xie, et al. Rlinf-vla: A unified and efficient framework for reinforcement learning of vision-language-action models. *arXiv preprint arXiv:2510.06710*, 2025.

Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. In *Conference on Robot Learning (CoRL)*, 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.

Wenbo Zhang, Yang Li, Yanyuan Qiao, Siyuan Huang, Jiajun Liu, Feras Dayoub, Xiao Ma, and Lingqiao Liu. Effective tuning strategies for generalist robot manipulation policies. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7255–7262. IEEE, 2025.

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.

Table 1: **Training configuration for the two stages of RAFT.** Values are taken from the saved run configurations and training logs. Effective batch sizes include gradient accumulation and, for Stage II, distributed workers.

	Stage I: reasoning alignment	**Stage II: action fine-tuning**
Trainable components	LoRA adapters only	Language model, action expert, and projection layers; SigLIP vision tower and Stage-I LoRA frozen
LoRA placement	All 18 language-model layers: q/k/v/o attention projections and gate/up/down MLP projections (126 modules); all 27 vision layers: q/k/v projections (81 modules)	The Stage-I LoRA remains attached and frozen; no adapter parameters are optimized in Stage II
LoRA configuration	Rank $r = 64$, scaling $\alpha = 128$ ($\alpha/r = 2.0$), dropout 0.05	—
Trainable parameters	90.4M (78.4M language-model and 11.9M vision-adapter parameters; component values rounded)	3.09B of 3.50B parameters
Optimizer	AdamW, $\beta = (0.9, 0.999)$, weight decay 10^{-6}, gradient clipping 1.0	AdamW, $\beta = (0.9, 0.95)$, weight decay 10^{-10}, gradient clipping 1.0
Learning rate	Peak 2×10^{-4}; 100 warmup steps; cosine schedule configured over 2,010 steps, with training stopped at step 1,000 before the schedule completed	Peak 5×10^{-5}; 1,000 warmup steps; thereafter effectively constant at 5×10^{-5}
Effective batch size	12 samples/GPU $\times$ accumulation $4 = 48$	16 samples/GPU $\times$ 4 GPUs $\times$ accumulation $8 = 512$
Optimization steps	1,000	5,000
Numerics	bfloat16 with gradient checkpointing	bfloat16 with gradient checkpointing
Hardware	1$\times$ NVIDIA L40 (46 GB)	4$\times$ NVIDIA L40 (46 GB), one node with distributed data parallel training

A Training and implementation details

Table 1 reports the saved configuration and logged settings for the MolmoB0t implementation of RAFT used in simulation and on DROID (SigLIP vision encoder and an 18-layer Gemma-2B decoder). The YAM experiments instead adapt $\pi_{0.5}$; the architecture and numerical settings in this table should not be read as the YAM configuration. In RAFT, the pretrained SigLIP weights remain frozen in both stages. Stage I optimizes reasoning LoRA modules in the language model and vision encoder; these adapter parameters are distinct from the pretrained weights. Stage II freezes the learned adapters and optimizes the language model, action expert, and projection layers. The fine-tuned baselines use a trainable SigLIP encoder; they do not share RAFT's restriction to vision adaptation through LoRA. The pretrained Base policy is evaluated without adaptation.

A.1 Gemini annotation response format

For each observation and task instruction, Gemini 3.1 Pro is asked to return a JSON object in the following format. Coordinates are normalized to the $[0, 1000]$ image frame; boxes are $[x_1, y_1, x_2, y_2]$ and points are (x, y). The format below is for pick-and-place; open-drawer uses the same structure with `handle_point`, `pre_grasp_point`, `grasp_point`, `pull_start_point`, and `pull_end_point` as keypoints.

```
{
  "reasoning": "<one or two sentences about relevant objects and positions>",
  "objects": [
    {"name": "<label>", "bbox": [x1, y1, x2, y2]}
  ],
  "keypoints": {
    "grasp_point":     [x, y],
```

```
    "target_point":    [x, y],
    "lift_point":      [x, y],
    "pre_grasp_point": [x, y],
    "pre_place_point": [x, y]
  },
  "trace": [[x, y], ...]
}
```

Bimanual (YAM). For the bimanual box-loading task, one arm holds the lid open while the other loads the object, so the keypoints are split by arm rather than by motion stage, and the response adds a per-arm block:

```
{
  "reasoning": "<one or two sentences about relevant objects and positions>",
  "objects": [
    {"name": "<label>", "bbox": [x1, y1, x2, y2]}
  ],
  "keypoints": {
    "left_ee_point":  [x, y],
    "lid_point":      [x, y],
    "right_ee_point": [x, y],
    "object_point":   [x, y],
    "drop_point":     [x, y]
  },
  "trace": [[x, y], ...],
  "arms": {
    "left":  {"motion": "<what the LEFT arm is doing in this frame>",
              "trace": [[x, y], ...]},
    "right": {"motion": "<what the RIGHT arm is doing in this frame>",
              "trace": [[x, y], ...]}
  }
}
```

The prompt asks for one short clause per arm (e.g., approaching, holding the lid open, grasping, releasing, idle) and a short trace from that arm's current gripper position; the top-level trace is the manipulating arm's path for the object. The per-arm `state` line in the Stage-I target (phase, moving/stationary, gripper) is not predicted by Gemini: it is computed from the joint encoders and also given to Gemini as context, so its per-arm descriptions agree with the measured robot state.

A.2 Stage-I supervision targets by modality

Table 2 illustrates the target formats for a mug-into-bowl pick-and-place example. It is a separate example from the bimanual box-and-toy scene in Fig. 3. Every row uses the same pick-and-place observation and instruction; only the response format changes. VQA and spatial supervision contain language only, while geometric variants use the corresponding coordinates and, where applicable, object or keypoint labels.

A.3 Baseline implementation details

Plain FT. Plain fine-tuning, also labeled Full FT in the figures, optimizes the SigLIP vision encoder, language model, action expert, and projection layers without Stage-I reasoning alignment. It uses the Stage-II optimizer, learning-rate schedule, effective batch size, training steps, precision, gradient clipping, and checkpointing settings in Table 1; the table's trainable-parameter counts describe RAFT, not this baseline.

RETAIN and DeLock. We follow the methods as defined in their original papers Yadav et al. (2026); Huang et al. (2026). For our evaluations, RETAIN uses the merge coefficient $\alpha = 0.9$ reported by its authors. Both baselines use trainable SigLIP weights during fine-tuning. DeLock regularizes the drift

Table 2: **Modality-specific Stage-I targets for a mug-into-bowl pick-and-place example.** Coordinates use the normalized $[0, 1000]$ image frame.

Variant	Target response
VQA	The robot should grasp the glossy beige ceramic mug on the left side of the counter and place it into the white bowl on its right.
Spatial	The beige mug is left of the white bowl. The gripper begins above and to the right of the mug, moves to the mug, lifts it, then moves rightward to the bowl.
Bounding box	`glossy beige ceramic mug: [310, 600, 380, 740]; white bowl: [410, 570, 520, 720]`
Keypoints	`grasp_point: [340, 670]; target_point: [460, 650]; lift_point: [340, 500]; pre_grasp_point: [340, 550]; pre_place_point: [460, 500]`
Trajectory trace	`[[600, 400], [470, 470], [340, 550], [340, 670], [400, 580], [460, 650]]`
Full reasoning	The VQA plan together with the two object boxes, five semantic keypoints, and six-point trajectory trace shown above.

of those weights from their pretrained values with coefficient 10^{-2}. We tune its contrastive-prompt coefficient over the values described below.

Contrastive prompt. We sweep the contrastive-prompt coefficient over $\{0.5, 2, 3\}$ and report the setting with the highest average success across evaluation conditions.

Unless required otherwise by the baseline method, all comparisons use the same action demonstrations, augmentations, optimization steps, action loss, and fixed final-checkpoint evaluation after 5,000 action-training steps.

A.4 Evaluation protocol

Simulation task definitions, reset procedure, matched initial states, and success criteria follow the MolmoSpaces evaluation protocol Kim et al. (2026). Within each evaluation condition, methods are compared on matched episode IDs. Physical-robot resets preserve the task configuration and start state. Trials for different policies were interleaved across resets, and all trials for each platform were collected on the same day. DROID used 30 ID trials and 24, 24, and 20 trials for the object, distractor, and background shifts; YAM used 30 trials per condition. Wilson confidence intervals summarize uncertainty in trial success rates for the evaluated policies.

B Qualitative real-robot comparisons

Figures 13–15 compare policies on matched DROID evaluation cases in which RAFT completes the task and the comparison policies fail before completion. Each method is shown at four ordered points in its rollout. Episode names and initial object configurations are matched across methods; small viewpoint differences result from physical camera placement.

Figures 16–18 show matched YAM evaluations under three OOD shifts. Rows correspond to RAFT, Plain FT, RETAIN, and DeLock; columns follow each rollout in temporal order.

B.1 Ablations

B.1.1 Effect of reasoning modality

We compare Stage-I supervision using full reasoning, VQA, spatial descriptions, bounding boxes, keypoints, and trajectory traces. Each variant changes only the Stage-I target format; Stage-II action training and evaluation remain fixed. Figure 19 reports results for open drawer and pick and place, together with the layer-wise participation-ratio analysis. Examples of each target format are provided in Table 2. The main-text results use full reasoning for pick and place and bounding-box supervision

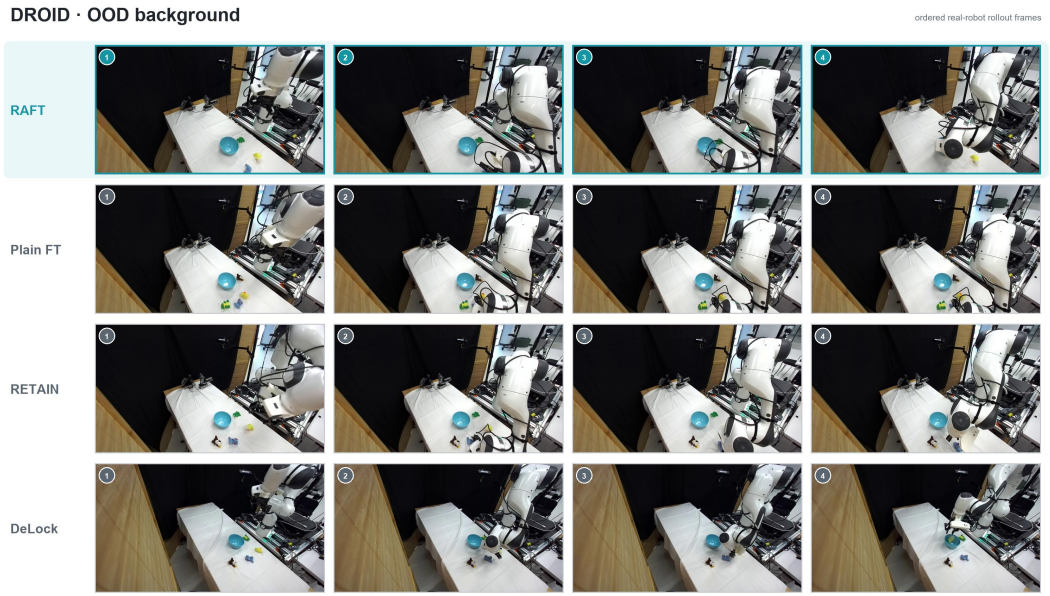

Figure 13: **DROID rollout comparison under an OOD background.** Rows show RAFT, Plain FT, RETAIN, and DeLock at four ordered points in approximately matched real-robot episodes. RAFT completes the manipulation, while the comparison policies fail before task completion.

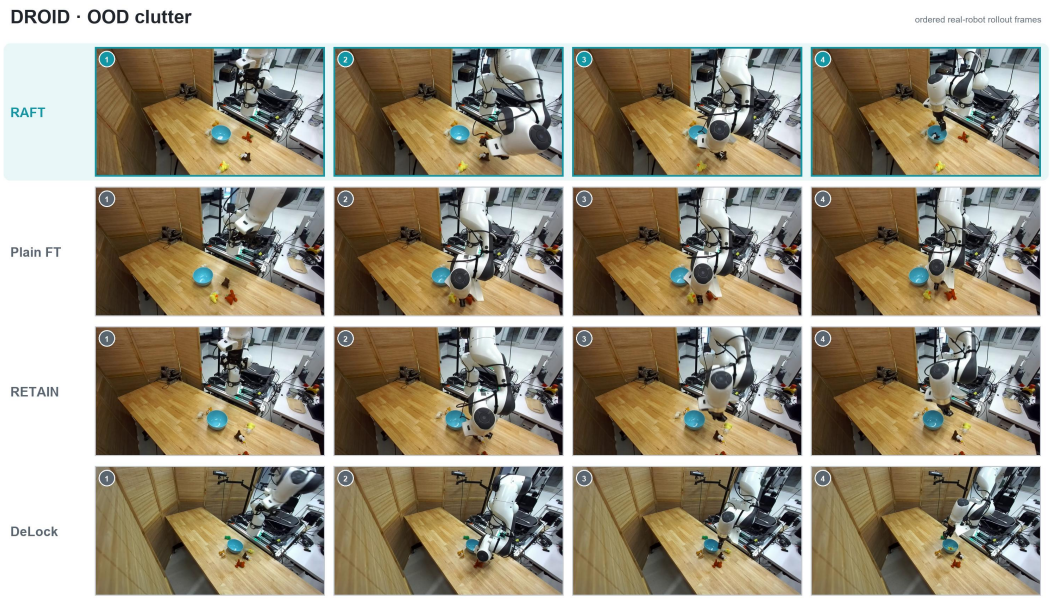

Figure 14: **DROID rollout comparison under OOD clutter.** Rows show RAFT, Plain FT, RETAIN, and DeLock at four ordered points in approximately matched real-robot episodes. RAFT completes the manipulation, while the comparison policies fail before task completion.

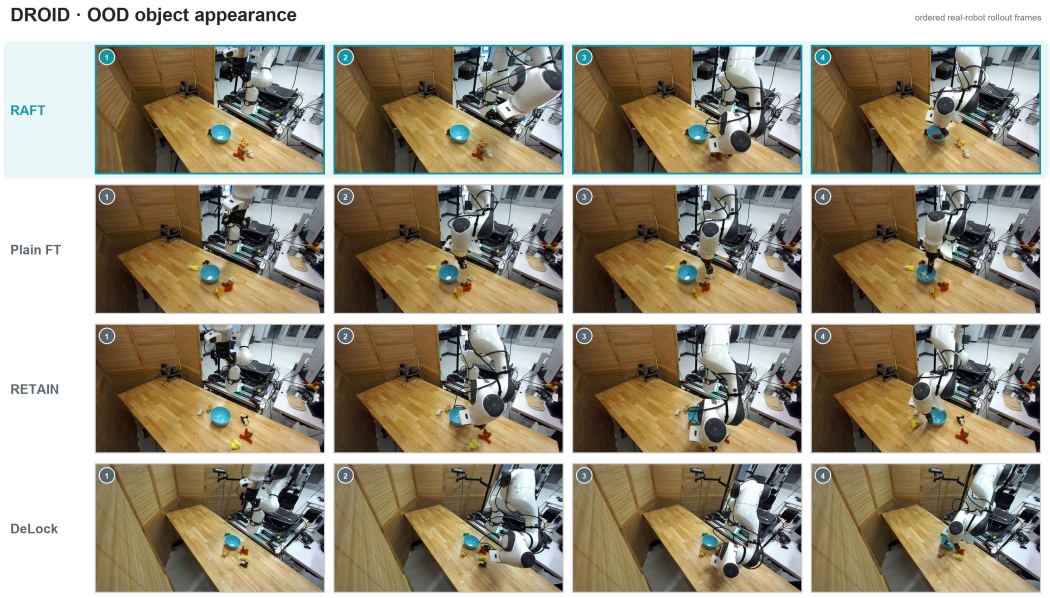

Figure 15: **DROID rollout comparison with an OOD object.** Rows show RAFT, Plain FT, RETAIN, and DeLock at four ordered points in approximately matched real-robot episodes. RAFT completes the manipulation, while the comparison policies fail before task completion.

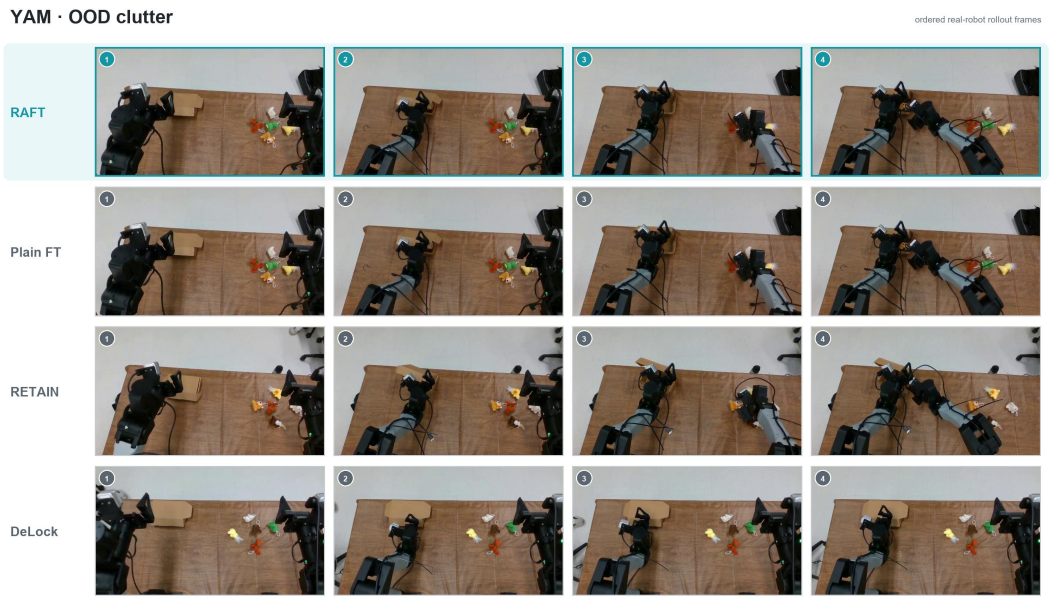

Figure 16: **YAM rollout comparison under OOD clutter.** Rows show RAFT, Plain FT, RETAIN, and DeLock on matched task instances; columns show ordered rollout frames. RAFT completes the two-stage task, while the comparison policies do not.

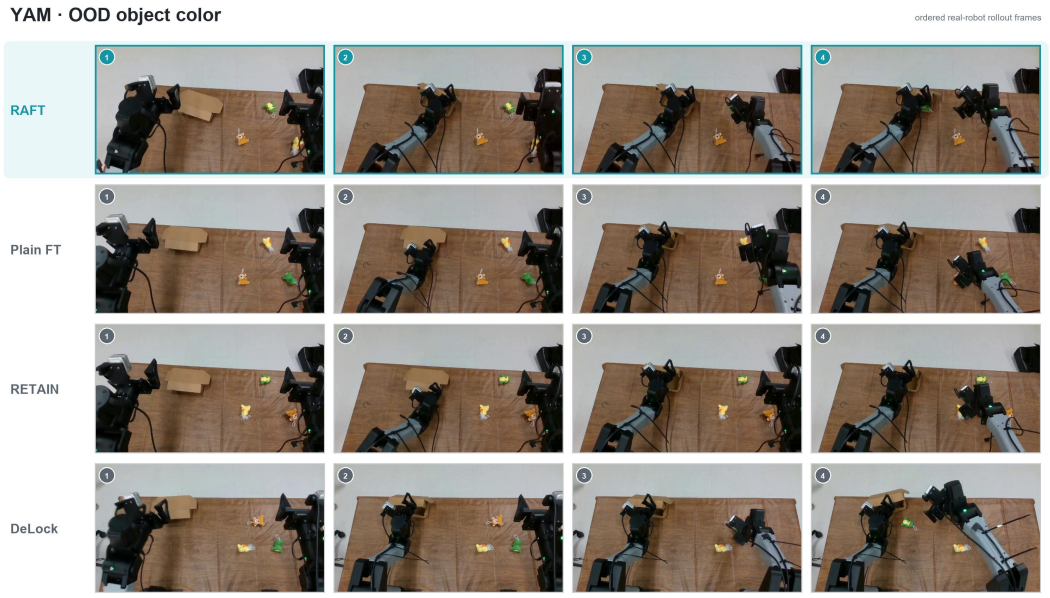

Figure 17: **YAM rollout comparison under an OOD color shift.** Rows show RAFT, Plain FT, RETAIN, and DeLock on matched task instances; columns show ordered rollout frames. RAFT completes the two-stage task, while the comparison policies do not.

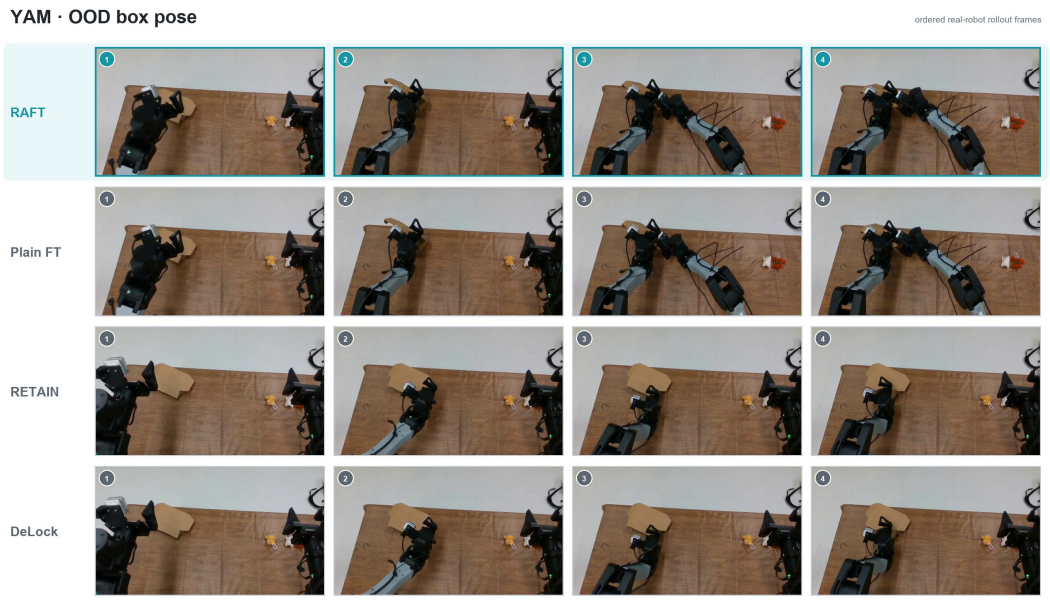

Figure 18: **YAM rollout comparison under an OOD initial pose.** Rows show RAFT, Plain FT, RETAIN, and DeLock on matched task instances; columns show ordered rollout frames. RAFT completes the two-stage task, while the comparison policies do not.

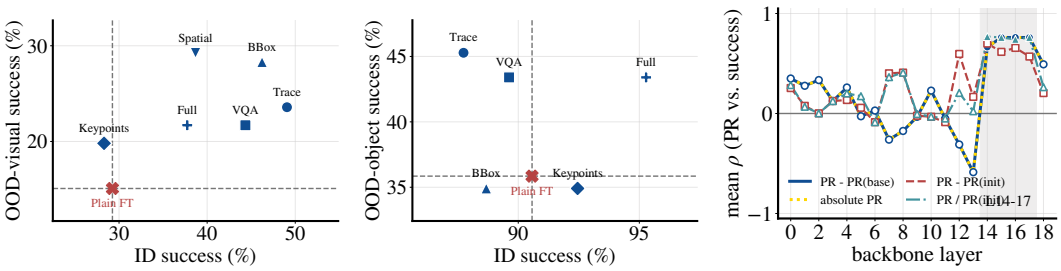

Figure 19: **Reasoning modality and representation dimensionality.** Left: ID and OOD-visual success on open drawer for six reasoning modalities. Center: ID and OOD-object success on pick and place for five modalities. Red crosses denote no-reasoning controls; dashed lines mark Plain FT. Each policy uses 106 trials per condition. Right: layer-wise mean Spearman correlation between participation ratio and success across three task-matched evaluation axes and six reasoning variants, using four participation-ratio measures. Filled markers indicate significance after FDR correction; shading marks layers 14–17.

Table 3: **Stage-II reasoning-adapter ablation on pick and place.** Success rate (%) when the Stage-I reasoning LoRA is applied without Stage II, updated during Stage II, or frozen during Stage II. Each split contains 106 episodes.

Adapter setting	ID	OOD-light	OOD-texture	OOD-object	OOD-prompt	Mean
LoRA on base (no Stage II)	72.64	60.38	9.43	33.02	66.04	48.30
LoRA trainable in Stage II	90.57	86.79	65.09	34.91	**92.45**	73.96
LoRA frozen in Stage II (RAFT)	**95.28**	**93.40**	**92.45**	**43.40**	91.51	**83.21**

for open drawer, with the modality held fixed across evaluation conditions within each task. The other modalities are evaluated separately here.

On *open drawer*, trajectory-trace supervision gives the highest ID success at 49.06%, compared with 29.25% for Plain FT. Under the visual shift, spatial descriptions reach 29.25%, bounding boxes reach 28.30%, and trajectory traces reach 23.58%.

On *pick and place*, full reasoning gives the highest ID success at 95.28%. Trajectory traces give the highest OOD-object success at 45.28%, followed by VQA and full reasoning, both at 43.40%. Bounding-box and keypoint supervision each reach 34.91%, compared with 35.85% for Plain FT. The best target format therefore varies across tasks and evaluation conditions.

B.1.2 Stage-II adapter ablation

We evaluate the same Stage-I reasoning LoRA in three settings: attached to the base policy without Stage-II action fine-tuning, updated during Stage II, and frozen during Stage II as in RAFT. Table 3 reports success over 106 episodes per split.

The LoRA-on-base policy reaches 48.30% mean success. Updating the LoRA during Stage II increases mean success to 73.96%, while freezing it reaches 83.21%. The largest difference between the trainable and frozen settings occurs under the texture shift: 65.09% versus 92.45%.

B.1.3 Co-training and frozen-vision controls

We also compare the staged procedure with co-training, which applies reasoning and action supervision in a joint training run. Reasoning and action examples are sampled in a 50/50 ratio, and their losses have equal weight (1:1). Equal weights favor neither objective and avoid giving co-training a tuned hyperparameter that Plain FT and RAFT do not have. The primary Plain FT and co-training controls use a trainable SigLIP encoder. We additionally evaluate Plain FT with the pretrained vision encoder frozen throughout action fine-tuning. In RAFT, the pretrained SigLIP weights remain frozen and vision adaptation is performed through Stage-I LoRA, which is held fixed during Stage II. Table 4 reports pick-and-place success on the same 106 evaluation episodes per split.

Table 4: **Co-training and frozen-vision controls on simulated pick and place.** Success rate (%) across five matched evaluation splits ($n = 106$ per cell). Mean pools the 530 trials per method. The co-train percentages correspond to 84, 75, 77, 19, and 80 successes, respectively.

Method	ID	OOD-light	OOD-texture	OOD-object	OOD-prompt	Mean
Plain FT	90.57	87.74	92.45	35.85	91.51	79.62
Plain FT (frozen vision)	91.51	86.79	87.74	40.57	91.51	79.62
Co-train	79.2	70.8	72.6	17.9	75.5	63.21
RAFT	**95.28**	**93.40**	**92.45**	**43.40**	**91.51**	**83.21**

Table 5: **RETAIN applied to Plain FT and trace-supervised RAFT on Franka open drawer.** Success rate (%) and successes/trials for each condition. Overall pools the four conditions ($n = 424$ per policy). The standalone trace-supervised RAFT full-texture cell comes from its earlier September 2 evaluation; its other three cells were evaluated alongside the merging controls.

Policy	ID	Lighting	Lighting + textures	Full texture	Overall
RETAIN on Plain FT ($\alpha = 0.9$)	39.6 (42/106)	38.7 (41/106)	12.3 (13/106)	17.0 (18/106)	26.9 (114/424)
RETAIN on trace RAFT ($\alpha = 0.9$)	41.5 (44/106)	37.7 (40/106)	22.6 (24/106)	**29.2** (31/106)	32.8 (139/424)
Trace RAFT alone	**44.3** (47/106)	**41.5** (44/106)	**28.3** (30/106)	25.5 (27/106)	**34.9** (148/424)

Co-training reaches 63.21% mean success, compared with 79.62% for Plain FT and 83.21% for RAFT. The largest co-train deficit is under the object shift (17.9% versus 43.40% for RAFT). Freezing vision in Plain FT changes its split-level results but leaves its pooled mean unchanged at 79.62% ($422/530$); its OOD-object result is 40.57% ($43/106$), compared with 35.85% ($38/106$) for the primary Plain FT and 43.40% ($46/106$) for RAFT.

B.1.4 Compatibility with weight merging on open drawer

We test whether RAFT can serve as the adapted checkpoint in RETAIN weight merging. This composability matters for continual-adaptation workflows that combine task specialization with merging toward a pretrained policy. Both merged policies interpolate with the same pretrained base at $\alpha = 0.9$: one uses Plain FT as the adapted endpoint, while the other uses trace-supervised RAFT. We also evaluate the standalone trace-supervised RAFT policy. Table 5 reports the completed Franka open-drawer evaluation: all 12 cells contain 106 trials, with no omitted scenes.

Using RAFT rather than Plain FT as RETAIN's adapted endpoint increases pooled success from $114/424$ (26.9%) to $139/424$ (32.8%), a 5.9-point difference. The largest differences are under lighting plus texture changes ($24/106$ versus $13/106$) and full texture randomization ($31/106$ versus $18/106$). Standalone trace-supervised RAFT reaches $148/424$ (34.9%). Thus RAFT can be used as RETAIN's adapted endpoint and yields a stronger merged policy than Plain FT on this evaluation.

B.2 Evaluation environments

Figure 20 summarizes the ID and OOD conditions used across simulation and physical-robot experiments. Simulation task construction, resets, initial-state matching, and success criteria follow MolmoSpaces Kim et al. (2026). Methods are compared on the same episode identities within each condition. Physical resets preserve the task configuration and start state.

B.3 Representation and attention analysis protocol

Backbone and captured frames. The analyzed policy uses a SigLIP vision encoder producing 256 patch tokens and an 18-block Gemma-2B decoder with hidden width 2048. We use the same probe prompt and image records for every policy arm. The participation-ratio (PR) analysis reported in Fig. 19 uses the first 400 records of the held-out iThor FP13 validation split, spanning five episodes.

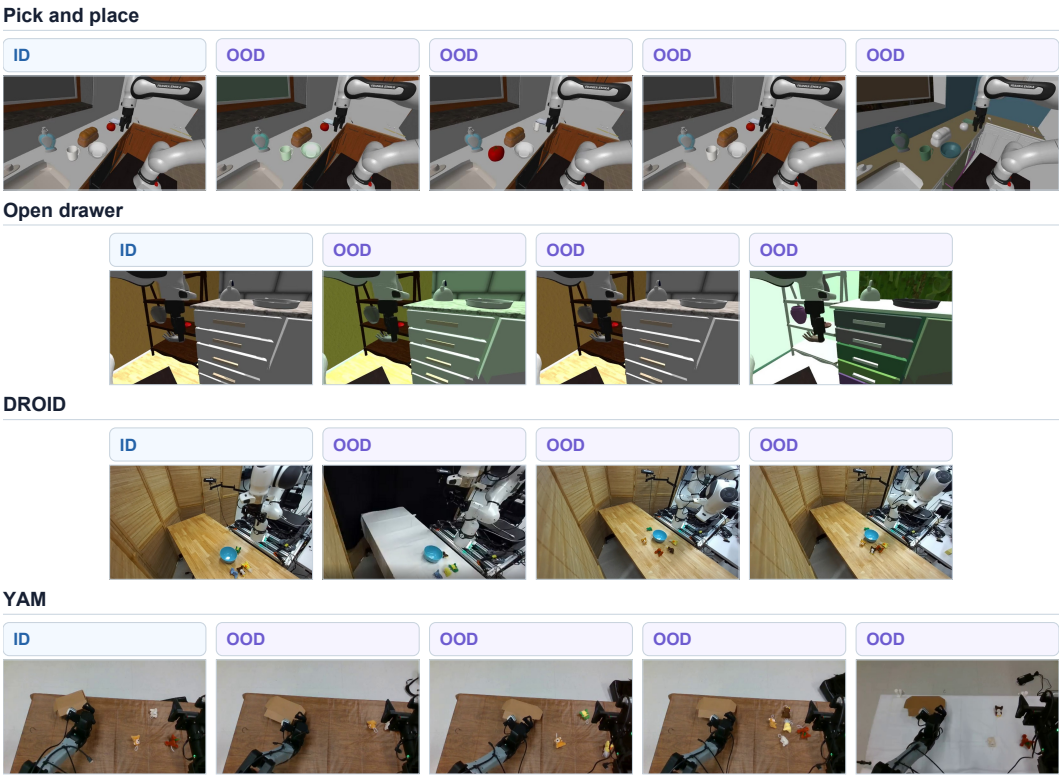

Figure 20: **Evaluation conditions across four robot tasks.** Each row begins with an in-distribution observation; the remaining panels show OOD variants. Pick and place varies lighting, target object, instruction, and scene texture. Open drawer varies lighting, instruction, and scene texture. DROID varies background, clutter, and object appearance. YAM varies box pose, object color, clutter, and background. Instruction shifts retain the ID image because only the task prompt changes.

The captured input frames are asserted bit-identical across arms using the base-frame hash H_{base}. The linear-probe results in Fig. 9 use a separate, wider capture of 414 frames: nine evenly spaced frames from each of 46 policy-training episodes. Although that capture is stored in a split named `val`, its episodes come from the policy-training set. Leave-one-episode-out cross-validation holds out each episode from fitting the probe, not from policy training.

Participation ratio and correlations. For each frame and decoder layer ℓ, we mean-pool `hidden_states[`ℓ`]` over non-padding positions, producing a 400×2048 matrix X_ℓ. After centering columns, we obtain the covariance eigenvalues λ_i from squared singular values and compute $\mathrm{PR}_\ell = (\sum_i \lambda_i)^2 / \sum_i \lambda_i^2$. This is effective dimensionality of *across-frame* variance, not within-sequence token spread. Each evaluation axis pairs one downstream benchmark with one PR probe domain. The three task-matched axes are pooled in Fig. 19; the partially matched next-to axis and unmatched close axis serve as excluded controls. Per layer and axis, we compute Spearman correlation across the same six Stage-I arms. Its two-sided exact permutation test enumerates all $6! = 720$ assignments, and Benjamini–Hochberg FDR correction is applied across the 19 layers.

Linear probes. The wide capture contains nine object categories before filtering; the reported object classifier retains seven categories with at least three episodes each, so chance balanced accuracy is $1/7$. Episode-time labels are $\min(3, \lfloor 4\,\mathrm{frame_idx}/\mathrm{total_frames} \rfloor)$, with chance balanced accuracy $1/4$. The names reach, grasp, transport, and place are readability labels for positional quartiles, not semantic action annotations. We fit `StandardScaler`, `PCA(64)`, and `LogisticRegression(C=0.5)` within each leave-one-episode-out fold. We summarize layer-band uncertainty and differences from Plain FT with a paired bootstrap over episodes ($B = 1000$).

Visual attention and region masks. At each layer we extract image-key attention from text queries as $A[:, 256:, :256]$, uniformly average over heads and text queries, and L_1-normalize across the 256 image patches. Thus there is no selected head or last-token readout. We average normalized maps across layers for band displays; this normalization removes the total image-attention mass, which is reported separately as `img_mass`. For visualization, patch values are expanded by nearest-neighbor tiling; all arms share a color maximum at the 99th percentile of interior-patch attention. The border ring is excluded from that scale because it exhibits sink/register behavior. Bounding boxes are mapped to patches with floor/ceiling boundaries and can include up to one extra patch on each edge; each keypoint or trace point maps to its containing patch. Background is the complement of annotated patches. For each respective region analysis, a frame must contain at least two bounding-box patches, one keypoint, or one trace point, as applicable. For a patch mask M and normalized attention p_j, enrichment is $(256/|M|)\sum_{j\in M} p_j$. Bounding-box, keypoint, and trace masks may overlap and therefore do not partition the image; only background is defined by complementation.

C Extended late-layer probe analysis

Figure 21 expands the main-text probe comparison to all nine model arms. VQA and trace show the largest late-layer recovery in object-identity decoding and the strongest novel-object results; spatial, bounding-box, and keypoint variants show no comparable recovery and are the weakest on that shift.

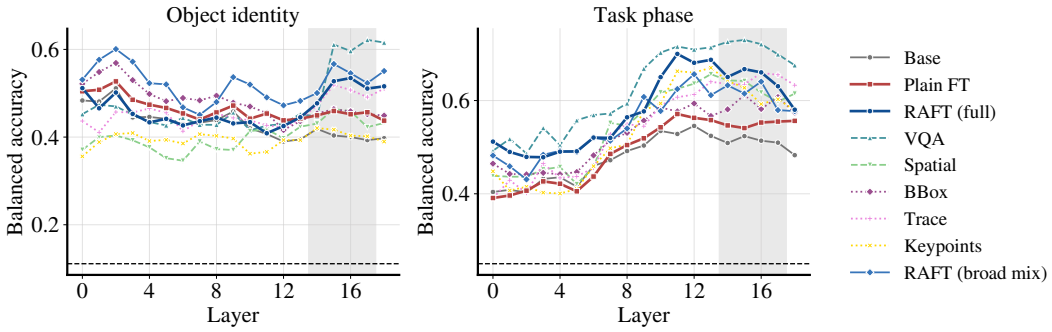

Figure 21: **Late-layer probes for all nine model arms.** Leave-one-episode-out balanced accuracy for object identity and episode-time quartile across backbone layers. The shaded band marks layers 14–17. This expands the four-arm comparison in the main text to the full set of reasoning-modality variants.