

What Matters in Quality-Aware VLA Pretraining: A Benchmark and Empirical Study

Mingxuan Yan¹, Litian Gong³, Dihong Huang^{2,4}, Zhixu Li³, Tianyu Zhang³, Wenqian Zhang³,
Hefeifei Jiang³, Peihao Li⁵, Jian Zhang⁶, Mengfei Zhao², Yikai Tang⁴, Hai Zhai²,
Zehao Wang¹, Ruijian Liang², Yanjia Huang⁶, Lin Shao⁷, Changliu Liu⁴,
Kaiyu Hang⁸, Zhiwen Fan⁶, Masayoshi Tomizuka⁵, Jianfei Yang⁹, and Jiachen Li^{1,†}

¹Georgia Institute of Technology ²Axis Robotics ³University of California, Riverside
⁴Carnegie Mellon University ⁵University of California, Berkeley ⁶Texas A&M University
⁷National University of Singapore ⁸Rice University ⁹Nanyang Technological University
[†]Corresponding author: jiachen_li@gatech.edu

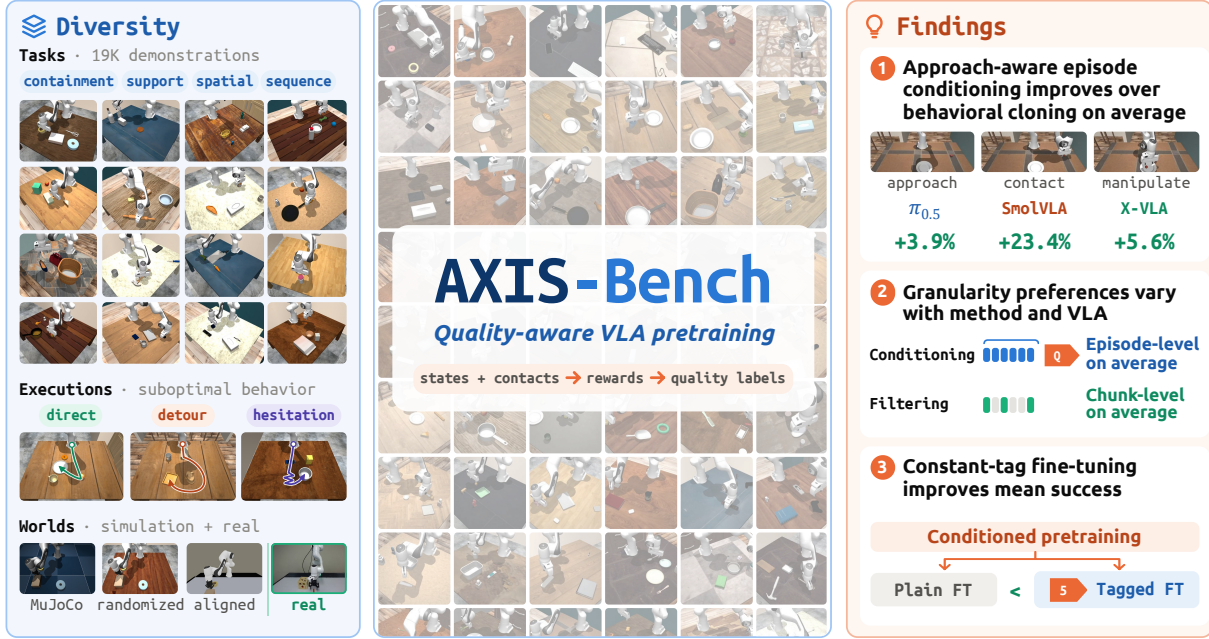


Fig. 1. **AXIS-Bench enables a systematic study of quality-aware VLA pretraining.** We use 5,000 simulation demonstrations for controlled comparisons and 19,250 for pretraining before real-world fine-tuning. The simulation study spans three VLAs, comparing reward definitions, label granularities, and learning methods within each VLA. Green values show mean relative gains over plain behavioral cloning.

Abstract—How should demonstration quality be used during vision-language-action (VLA) pretraining to improve performance after target-task fine-tuning? We compare weighting, filtering, and conditioning across reward definitions and label granularities using 5,000 naturally suboptimal human demonstrations and 24 target tasks. Within each VLA, we match demonstrations and training budgets. Additional comparisons examine pretraining data size and the roles of quality tags during fine-tuning and inference. Across three VLAs, we find that approach-aware episode conditioning improves mean success over plain behavioral cloning; conditioning and filtering favor episode and chunk labels, respectively, on average, with model-dependent reversals; and constant-tag fine-tuning improves mean success after conditioned pretraining with either tagged or untagged inference. We introduce **AXIS-Bench** to support these controlled comparisons. Built on crowdsourced simulation demonstrations, it provides recorded object and contact states for recomputing quality labels on shared trajectories. Replayable trajectories support workspace-aligned rendering and user-defined real-world target tasks. In our real-world experiment, simulation pretraining raises success from 9.6% to 31.7%. Project page: <https://axis-bench.github.io/>.

I. INTRODUCTION

Vision-language-action (VLA) policies are commonly pre-trained on multitask demonstrations and fine-tuned on target tasks before deployment [1], [2], [3], [4]. During a demonstration, an operator may hesitate or move an object along an unnecessary detour, producing action chunks that make little progress toward the goal. Quality-aware pretraining [5], [6] aims to learn effectively from naturally suboptimal demonstrations. We evaluate its effect on performance after target-task fine-tuning (FT).

Quality-based filtering, weighting, and conditioning have been studied in robot learning [7], [5], [8], [9], [6]. Yet a quality signal leaves several learning choices open. The same quality information can be used to select which training examples to retain [8], [10], determine how strongly individual actions contribute to the training loss [11], [10], or condition the policy through explicit quality tags [6]. Quality can also be defined at different granularities, describing local progress

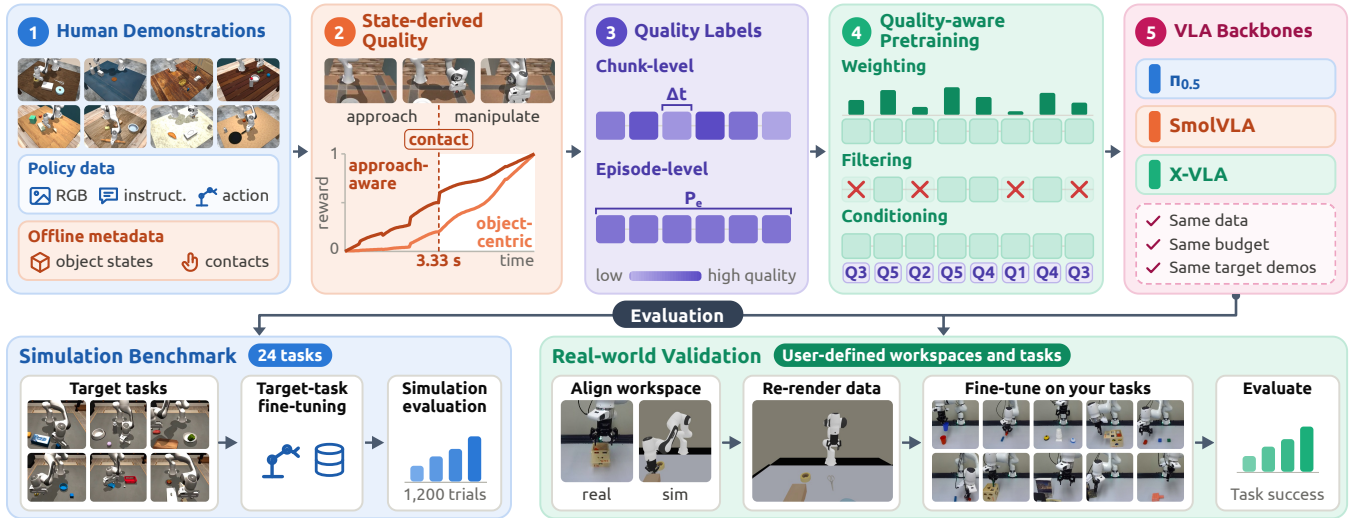


Fig. 2. **AXIS-Bench workflow for quality-aware VLA pretraining.** Human demonstrations provide policy observations and action targets, while recorded object and contact states enable offline computation of state-derived quality. Quality is represented at the chunk or episode level and incorporated into pretraining through weighting, filtering, or conditioning across three VLA backbones. The simulation benchmark evaluates performance after fine-tuning on 24 target tasks, with matched pretraining data, training budgets, and target-task demonstrations within each VLA comparison. The real-world workflow supports user-defined workspaces and tasks by aligning the simulation workspace, re-rendering demonstrations under the target workspace and camera configuration, fine-tuning on real-world target-task demonstrations, and evaluating task success. Photos illustrate our reference real-world setup.

over an action chunk or summarizing the quality of an entire episode. For conditioned policies, additional choices arise regarding how quality tags should be handled during subsequent fine-tuning and inference. However, systematic evidence comparing how these choices interact under shared data and evaluation protocols remains limited. To address this gap, we study how quality definitions, label granularities, and learning methods jointly affect target-task performance, and which findings are consistent across base VLA policies. We compare weighting, filtering, and conditioning across two progress-based reward definitions, two label granularities, and three VLAs ($\pi_{0.5}$ [4], SmolVLA [12], and X-VLA [13]). Using 5,000 pretraining demonstrations and 24 target tasks, we further examine the effects of pretraining data size and fine-tuning/inference protocols.

Three findings emerge (see Fig. 1). First, approach-aware episode conditioning improves mean success over plain behavioral cloning (plain BC) across the three evaluated VLAs. Second, label granularity interacts with both the learning method and base VLA: conditioning favors episode labels and filtering favors chunk labels on average, but the preferences can reverse. Third, after approach-aware episode conditioning, constant-tag target-task FT yields higher mean success than plain FT with tagged/untagged inference.

To support this study, we introduce AXIS-Bench (see Fig. 2), a simulation and real-world benchmark built on the crowdsourced AXIS simulation teleoperation dataset [14]. Recorded object and contact states let us recompute progress-based quality labels on the same trajectories. Replayable trajectories further enable researchers to re-render demonstrations to align with target workspace and camera configurations and evaluate policies on their own real-world target tasks. In our real-world reference experiment, simulation pretraining raises $\pi_{0.5}$ -base success rate from 9.6% to 31.7%.

Our main contributions are:

- 1) **A systematic empirical study of quality-aware VLA pretraining.** We identify how quality definitions, label granularities, learning methods, and adaptation protocols interact across base policies, with matched data and training budgets within each VLA.
- 2) **AXIS-Bench for controlled comparison and real-world validation.** We provide shared human demonstrations, configurable state-derived quality labels, and standardized training and evaluation protocols, together with replayable simulation trajectories and a real-world setup for workspace adaptation.

II. RELATED WORK

Robot demonstration datasets. Robot demonstration datasets have expanded the diversity of tasks, environments, and embodiments available for policy learning. Bridge-Data V2 and DROID collect real-world demonstrations across tasks and workspaces [15], [16], while Open X-Embodiment aggregates data across robot embodiments [1], supporting generalist policies such as Octo and OpenVLA [2], [3]. Simulation provides complementary approaches through crowdsourced teleoperation, as in RoboTurk [17], and demonstration-based or procedural generation, as in MimicGen and MolmoBOT [18], [19]. Real-world collections naturally capture variation in human execution but generally lack simulator states for directly recomputing progress-based rewards. Generated simulation trajectories provide privileged state information but need not preserve the natural execution variability of human demonstrations. AXIS unifies crowdsourced human teleoperation with recorded object states, contacts, and replayable trajectories [14].

Quality-aware VLA training. Studies of demonstration proficiency motivate learning selectively from demonstrations of varying quality [20]. Related data-curation methods estimate training utility using state-action mutual informa-

tion, target-task relevance, or rollout-based influence [7], [21], [22], [23]. These methods also operate at different granularities: Re-Mix optimizes dataset-level mixture weights [5], SCIZOR filters individual transitions for generalist policy training [8], and World Value Models (WVMs) use chunk-level value changes for weighting and filtering [10]. Conditioning supplies quality information as an input to the policy. Decision Transformer and RvS study return-conditioned supervised learning for offline RL [24], [25]. In VLA training, RECAP applies advantage conditioning during both pretraining and task-specific training [9], while $\pi_{0.7}$ conditions on quality, speed, and mistake metadata [6]. Related metadata- and quality-conditioning strategies have also been explored in language-model pretraining and vision-language learning [26], [27], [28]. We complement these studies with a controlled comparison of reward definitions, label granularities, and learning methods on shared demonstrations across three VLAs.

III. BENCHMARK DESIGN AND DATA RESOURCE

A. Human demonstrations and replayable data

AXIS-Bench pairs human demonstrations with scene assets and replayable simulation trajectories (see Fig. 2). Policy observations comprise images, task instructions, and proprioceptive states, paired with corresponding action targets. Recorded object geometry, contacts, task progress, episode boundaries, and validity masks support offline quality labeling (Sec. IV). We use two pretraining datasets: 5,000 simulation demonstrations (about 35.3 hours) for the simulation experiments and 19,250 simulation demonstrations (about 114.6 hours) for the real-world experiments. Both simulation and real-world experiments use a DROID-compatible interface [16], with actions comprising seven joint velocities and a gripper command at 15 Hz. Pretraining observations include both external-camera and wrist-camera views.

B. Controlled simulation benchmark

The simulation branch of Fig. 2 evaluates target-task performance on 24 tabletop tasks spanning containment, support placement, spatial relations, separation, and short sequences. Pretraining uses the shared 5,000-demonstration dataset, in which object meshes and poses, language goals, and visual conditions vary across demonstrations. Visual randomization varies room backgrounds, table textures, and lighting while keeping the external-camera pose fixed. Target-task FT uses 354 demonstrations (about 2.4 hours). The embodiment, renderer, and object library are shared between pretraining and evaluation, so the benchmark evaluates target tasks and initial object placements within this setting. Pretraining and target task IDs are disjoint, although three target instructions also occur under four pretraining task IDs (22 of the 5,000 demonstrations); the split therefore does not imply complete semantic separation. Within each VLA configuration, method comparisons use the same base-policy initialization, training budget, target-task demonstrations, and evaluation layouts.

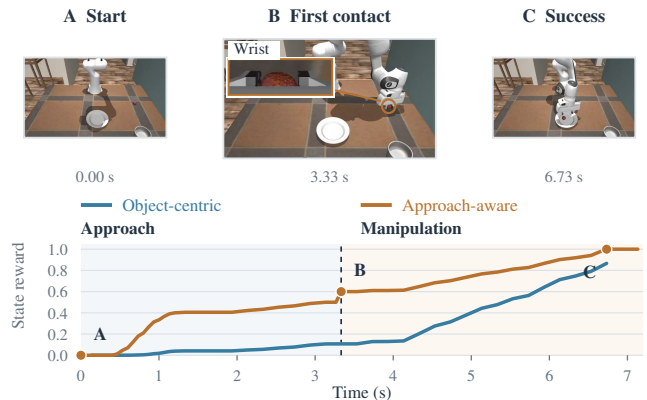


Fig. 3. **Two state-derived reward definitions.** A, B, and C denote the start, first contact, and task success, respectively. B includes the corresponding wrist-camera view. First sustained contact marks the transition from approach to manipulation for the approach-aware reward.

C. Workspace alignment and user-defined target tasks

For real-world validation (Fig. 2), researchers can align the simulated tabletop, background, and camera poses with their target workspace. Re-rendering preserves the recorded actions and object states while producing pretraining observations aligned with the target workspace and camera configuration. Researchers then collect real-world demonstrations for target-task FT and evaluate policies under fixed task definitions and success criteria. Our reference implementation uses robosuite [29], built on MuJoCo [30], to re-render trajectories for a Franka FR3 workspace.

IV. STATE-DERIVED QUALITY AND PRETRAINING

We use progress toward the task goal as a measure of demonstration quality for learning from naturally suboptimal human data. As illustrated in Fig. 2, we compute progress-based rewards from recorded states, derive quality labels at the chunk or episode level, and incorporate them into pretraining through weighting, filtering, or conditioning.

A. State-derived progress rewards

We derive progress signals from simulator trajectories to quantify how each demonstration advances toward the task goal (Fig. 3). Task-specific object progress can remain unchanged during both purposeful approach motion and hesitation. We compare a reward that combines object progress with end-effector proximity and an approach-aware reward that treats approach and manipulation separately. Let d_t denote the distance between the end effector and the manipulated object, and $p_t \in [0, 1]$ the task-specific object progress at time step t . The *object-centric reward* is

$$r_t = 0.8p_t + 0.2c_t, \quad (1)$$

where $c_t = 1 - \tanh(5d_t)$ captures end-effector proximity. For the approach-aware reward, we isolate the task-progress component as

$$p_t = \text{clip}((r_t - 0.2c_t)/0.8, 0, 1). \quad (2)$$

The approach-aware signal uses the first sustained finger-object contact as a task-agnostic boundary between approach

and manipulation. Before contact, progress is measured by the fraction of the required approach completed. After contact, it is measured by the additional object-centric progress achieved relative to the contact state. Let t_c denote the first frame in a sequence of at least three consecutive finger-object contact frames. We define the *approach-aware reward* as

$$R_t = \begin{cases} \lambda \text{clip}\left(\frac{d_0 - d_t}{d_0 - d_{t_c}}, 0, 1\right), & t < t_c, \\ \lambda + b + (1 - \lambda - b) \text{clip}(p_t - p_{t_c}, 0, 1), & t \geq t_c, \end{cases} \quad (3)$$

with $\lambda = 0.5$ and $b = 0.1$. This construction scores approach and manipulation separately. Episodes with contact at the start or no measurable approach use the manipulation branch from the start; episodes with no contact or contact only at the tail use approach progress normalized by the closest approach reached. Episodes without a valid object assignment remain unscored.

B. From reward changes to quality labels

We convert each state-derived reward into chunk-level and episode-level quality scores. For an episode of T time steps and either reward $z_t \in \{r_t, R_t\}$, we first compute the raw reward change over a fixed $H = 10$ frame window:

$$\Delta_t^{\text{raw}} = z_{\min(t+H-1, T-1)} - z_t, \quad H = 10. \quad (4)$$

We obtain the final chunk quality score Δ_t by applying validity and phase-boundary rules to this raw change. For the approach-aware reward, windows that cross the contact boundary are truncated so that the contact offset is not counted as local progress. If h_t intervals remain after truncation, we rescale the within-phase change by $(H - 1)/h_t$; windows with $h_t < 3$ or invalid reward values receive a score of zero. Positive, zero, and negative values of Δ_t indicate progress, no measured progress, and regression, respectively.

The *episode quality score* is the fraction of retained time steps whose corresponding action chunks make positive progress:

$$P_e = \frac{1}{|\mathcal{T}_e|} \sum_{t \in \mathcal{T}_e} \mathbf{1}[\Delta_t > 0], \quad (5)$$

where $\mathcal{T}_e$ contains the retained time steps from episode e . Thus, Δ_t captures the magnitude of local progress, whereas P_e captures how consistently an execution makes progress over the episode.

For conditioning, we rank valid quality scores across the pretraining dataset and partition them into five equal-frequency bins, yielding labels $q \in \{1, \dots, 5\}$ from low to high quality. Chunk-level labels are derived from Δ_t , whereas episode-level labels are derived from P_e and shared across all retained time steps within an episode.

We further assess whether the state-derived episode scores align with human judgments. Three annotators mark unproductive intervals in 50 simulation demonstrations, and we define human-assessed episode quality as $H_e = 1 - \text{median}_j U_{e,j}$, where $U_{e,j}$ is the fraction of judgeable retained time steps marked unproductive by annotator j . Ambiguous annotations are excluded, leaving at least two

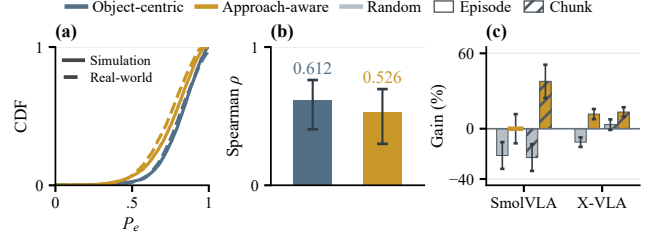


Fig. 4. **Quality distributions, alignment with human judgments, and utility for data selection.** (a) CDFs of episode-level quality scores for all 5,000 simulation episodes and 17,758 scorable simulation episodes from the 19,250 used for real-world experiments; object-centric scores for the latter are shown for reference. (b) Spearman correlations between state-derived episode scores and human judgments, with 95% episode-bootstrap confidence intervals (10,000 resamples). (c) Random selection versus approach-aware quality filtering at the same half-data budget for SmolVLA and X-VLA; solid and hatched bars denote episode- and chunk-level selection, respectively. Gains are relative to each VLA’s full-data plain-BC baseline; whiskers show rescaled marginal ± 1 SE.

annotators per episode. *The state-derived episode scores correlate positively with human judgments*, with Spearman $\rho(P_e, H_e) = 0.612$ for the object-centric reward and 0.526 for the approach-aware reward (Fig. 4(b)). To assess their utility for policy learning, we compare quality filtering with random selection at each granularity, matching retained time steps and training updates (Fig. 4(c)). For SmolVLA and X-VLA, approach-aware filtering improves over random selection by 1.08 and 5.75 percentage points (pp) at episode level, and by 3.08 and 2.58 pp at chunk level, respectively. The benefit depends on reward choice: object-centric chunk filtering exceeds random selection by 3.17 pp for SmolVLA but falls below it by 3.00 pp for X-VLA.

C. Weighting, filtering, and conditioning

We compare three ways of incorporating the same state-derived quality scores into pretraining: loss weighting, sample filtering, and quality-label conditioning. Chunk-level methods use Δ_t , whereas episode-level methods use P_e .

Plain BC and weighting. Plain BC samples retained demonstration time steps uniformly and optimizes the standard behavioral cloning loss. For quality weighting, following advantage-weighted regression (AWR) [11] and WVM [10], we assign each time step the chunk-level weight $w_t = \min(\exp(10\Delta_t), 2)$ and optimize the weighted loss $\sum_t w_t \ell_t / \sum_t w_t$, where ℓ_t is the flow-matching loss for the action chunk starting at time step t . For episode-level weighting, all time steps within an episode share a weight determined by its P_e . To maintain a comparable weight distribution across granularities, we map the time step-weighted percentile rank of P_e to the empirical distribution of chunk-level weights.

Filtering. Top-quality-50% filtering retains half of the preprocessed training time steps. Chunk-level filtering ranks time steps by Δ_t , whereas episode-level filtering ranks episodes by P_e and retains the highest-quality episodes until the same time step budget is reached. When necessary, the boundary episode is partially retained to match the budget exactly. We repeat the selected data to match the number of training updates used by plain BC.

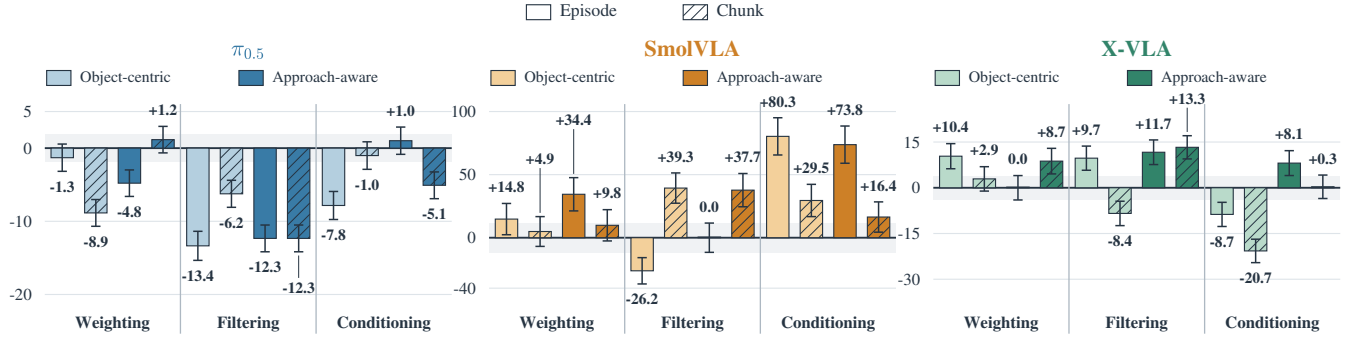


Fig. 5. **Quality-aware learning on 24 tasks:** AWR-style weighting, top-50% filtering, and conditioning. Light/dark shades indicate object-centric/approach-aware rewards; solid/hatched bars indicate episode/chunk labels. Values are relative improvements (%) over plain BC, whose success rates are 57.42%, 5.08%, and 25.75% for $\pi_{0.5}$, SmolVLA, and X-VLA, respectively. Matched FT-only controls reach 8.83%, 2.50%, and 6.17%. Whiskers/bands show rescaled marginal ± 1 SE.

Conditioning. Quality conditioning retains all training data with uniform sampling and appends a `Quality: q` tag to each language instruction following $\pi_{0.7}$ [6]. In simulation, tagged target-task FT assigns tag 5 to every demonstration before tag dropout, and conditioned inference also uses tag 5. We apply quality-tag dropout during pre-training and tagged FT; the dropout settings and real-world conditioning protocol appear in the Appendix.

V. SIMULATION EXPERIMENTS

We ask three questions: which quality-aware pretraining choices improve performance across VLAs, how label granularity interacts with the learning method, and whether quality tags should be retained during target-task FT. We also examine how performance changes with dataset size.

A. Experimental overview

We compare plain BC, weighting, filtering, and conditioning on 5,000 pretraining demonstrations, followed by FT on 354 demonstrations covering 24 target tasks. Within each VLA, comparisons share the initialization, data, training budget, and evaluation layouts. We initialize $\pi_{0.5}$ [4] from the released $\pi_{0.5}$ -DROID checkpoint and X-VLA [13] from its released base policy; SmolVLA [12] uses a frozen pretrained VLM and a randomly initialized action expert. Plain BC, weighting, and filtering use plain FT and untagged inference; conditioning uses constant-tag FT and tag-5 inference, with stage-specific controls below. Matched FT-only controls omit AXIS pretraining. For each of the 24 tasks, we evaluate all configurations on the same set of 50 initial scene layouts (1,200 trials per configuration) and report mean task success. Error bars in Figs. 5–7 denote ± 1 evaluation SE. Results are plotted relative to each VLA’s plain-BC reference, with absolute success rates given in the figure captions. Detailed training, evaluation, and plotting protocols are in appendix.

B. Cross-VLA gains from episode conditioning

Finding 1: Approach-aware episode conditioning improves mean success across all three VLAs. Fig. 5 compares all 36 combinations of VLA, reward definition, label granularity, and learning method. With tagged FT and tag-5 inference, approach-aware episode conditioning improves

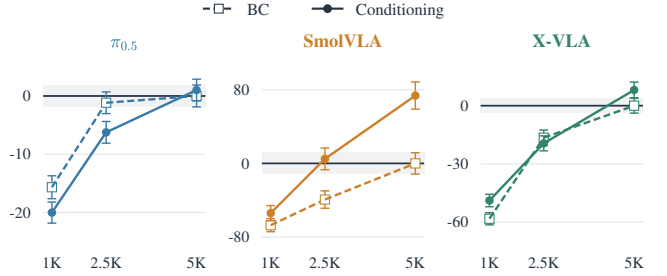


Fig. 6. **Scaling with pretraining data size.** Values are relative improvements (%) over each VLA’s fixed 5K plain BC. Dashed squares denote plain BC with plain FT/untagged inference; solid circles denote conditioning with tagged FT/tag 5. Whiskers show rescaled marginal SE; shaded bands show plain BC uncertainty.

over plain BC by 1.02%, 73.77%, and 8.09% relative for $\pi_{0.5}$, SmolVLA, and X-VLA, respectively. Method rankings vary across models. Across the four reward-granularity settings, $\pi_{0.5}$ conditioning and weighting have similar means above filtering; SmolVLA favors conditioning, whereas X-VLA favors filtering and weighting. X-VLA filtering exceeds conditioning in all four matched settings.

C. Coupling between label granularity and learning method

Finding 2: Conditioning favors episode labels and filtering favors chunk labels on average, with VLA- and reward-dependent exceptions. In Fig. 5, episode conditioning exceeds chunk conditioning in five of six matched VLA–reward settings. Chunk filtering exceeds episode filtering in four settings, ties in one, and falls below it in one. Under approach-aware reward, episode conditioning exceeds chunk conditioning by 3.50, 2.92, and 2.00 pp for $\pi_{0.5}$, SmolVLA, and X-VLA, respectively, while chunk filtering matches or exceeds episode filtering for each VLA. The two reversals occur with object-centric reward: $\pi_{0.5}$ conditioning favors chunk labels by 3.92 pp, and X-VLA filtering favors episode labels by 4.67 pp. An episode label may help conditioning distinguish overall execution patterns through a consistent quality context. Chunk filtering instead selects locally productive training start points, including those within episodes containing unproductive behavior. For X-VLA, random chunk selection also exceeds random episode selection

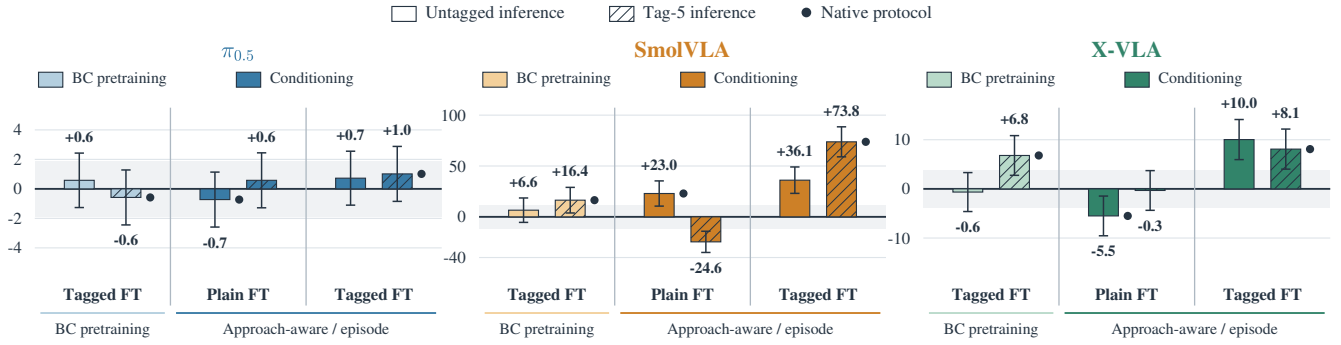


Fig. 7. **Conditioning stage and inference context.** Light/dark shades indicate plain BC/conditioned pretraining; solid/hatched bars indicate untagged/tag-5 inference. • marks native protocols: untagged after plain FT, tag 5 after tagged FT. Values are relative improvements (%) over plain BC/plain-FT/untagged inference. Whiskers/bands show rescaled marginal SE.

by 3.58 pp (Fig. 4(c)), suggesting that data composition may contribute to its granularity preference.

Weighting has reward-dependent preferences: object-centric weighting favors episode labels across all three models; approach-aware weighting favors chunk labels for $\pi_{0.5}$ and X-VLA, but episode labels for SmolVLA. Approach-aware chunk weighting also improves mean success over BC across models, with small gains for $\pi_{0.5}$ and SmolVLA (+0.67 and +0.50 pp).

D. Retaining quality tags during fine-tuning

Finding 3: After conditioned pretraining, tagged FT improves mean success under both inference modes. Fig. 7 compares plain and constant-tag FT after approach-aware episode conditioning under both inference modes for all three VLAs. We compare plain and constant-tag FT within each VLA, keeping the inference protocol fixed. Inference comparisons evaluate each FT checkpoint with and without tag 5. With tag-5 inference, tagged FT exceeds plain FT by 0.25, 5.00, and 2.17 pp for $\pi_{0.5}$, SmolVLA, and X-VLA, respectively. Under untagged inference, the corresponding differences are 0.83, 0.67, and 4.00 pp. The $\pi_{0.5}$ differences are small; SmolVLA gains more with tag-5 inference, and X-VLA without it. Adding tag 5 at inference changes SmolVLA success by -2.42 pp after plain FT and $+1.92$ pp after tagged FT. X-VLA shows the opposite pattern ($+1.33$ and -0.50 pp, respectively), illustrating dependence on the VLA and FT protocol. After plain-BC pretraining, switching from plain FT/untagged inference to tagged FT/tag-5 inference has mixed effects: -0.33 , $+0.83$, and $+1.75$ pp for $\pi_{0.5}$, SmolVLA, and X-VLA. Adding the tag only at inference also has opposite effects for $\pi_{0.5}$ and SmolVLA ($+0.92$ and -0.67 pp). The constant FT tag supplies no quality ranking among target-task demonstrations. Retaining the pretraining input format may instead help preserve the policy’s response to conditioning during adaptation. Studies of metadata conditioning [27], prompt templates [31], and implicit inference after FT [32] motivate this hypothesis. These opposing inference effects show that tag 5 alone is not always beneficial.

Overall Finding: Taken together, these results highlight the need to jointly consider reward definition, label granularity, learning method, and target-task adaptation. No single

TABLE I

Training-seed replication. Entries are mean $\pm$ sample SD of BC-normalized success-rate differences (%) over three seeds per VLA, using approach-aware reward. For each row $A - B$, we compute $100(\text{SR}_A - \text{SR}_B)/\text{SR}_{\text{BC}}$ within each seed before averaging. Episode conditioning uses tagged FT/tag-5 inference; BC uses plain FT/untagged inference. The last two rows compare tagged and plain FT after episode-conditioned pretraining, with inference fixed as indicated.

Comparison	$\pi_{0.5}$	SmolVLA	X-VLA
Episode cond. – BC	$+3.86 \pm 5.83$	$+23.42 \pm 45.42$	$+5.62 \pm 5.39$
Cond.: episode – chunk	-0.90 ± 8.52	$+24.71 \pm 32.64$	$+11.36 \pm 15.16$
Filter: chunk – episode	-1.90 ± 3.56	$+6.63 \pm 28.34$	$+5.59 \pm 8.05$
Tagged – plain: untagged	$+0.36 \pm 4.30$	$+9.58 \pm 12.31$	$+7.86 \pm 6.81$
Tagged – plain: tag 5	$+0.93 \pm 4.28$	$+62.29 \pm 31.29$	$+8.93 \pm 1.95$

pretraining configuration achieves the highest success rate across all evaluated VLAs.

E. Scaling with dataset size

Fig. 6 compares conditioning with plain BC at matched budgets of 1K, 2.5K, and 5K demonstrations, plotted relative to each VLA’s fixed 5K BC reference. For $\pi_{0.5}$, conditioning falls below BC at the smaller budgets and slightly exceeds it at 5K (58.00% versus 57.42%). SmolVLA conditioning exceeds BC at all three sizes, reaching 8.83% versus 5.08% at 5K. X-VLA conditioning exceeds BC at 1K and 5K, but falls below it at 2.5K (20.75% versus 21.50%). **Larger pretraining datasets improve success in these sweeps, while the conditioning advantage depends on the VLA and dataset size.** The sweep scales training updates with data size, so these results reflect the combined change in data and computation.

F. Training-seed replication

To assess the consistency of the findings across training runs, we evaluate the approach-aware comparisons with three training seeds per VLA: 42/43/44 for $\pi_{0.5}$ and 0/1/2 for SmolVLA and X-VLA. The method, scaling, and FT/inference comparisons in Figs. 5–7 use seed 42 for $\pi_{0.5}$ and seed 0 for SmolVLA and X-VLA. We use the 24-task evaluation protocol above, with the same 50 layouts for a given task across all configurations. Table I summarizes the within-seed success-rate differences normalized by the corresponding plain-BC success rate. **The three-seed results**

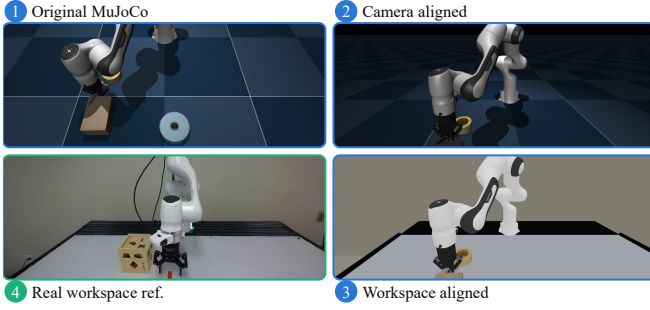


Fig. 8. **Workspace alignment for real-world pretraining.** The original MuJoCo scene (1) is re-rendered with an aligned camera view (2), followed by tabletop and background alignment (3). The real workspace (4) provides the visual reference. Recorded actions and object states are preserved during re-rendering.

retain positive mean gains from episode conditioning and tagged FT. Averaged across VLAs and seeds, conditioning favors episode labels and filtering favors chunk labels, with model- and seed-dependent reversals. Conditioning exceeds plain BC in six of nine seed-level comparisons, with one tie and two negative differences. On this BC-normalized scale, equal averaging over all nine VLA-seed combinations favors episode labels for conditioning (+11.72%) and chunk labels for filtering (+3.44%). Both directions hold in the SmolVLA and X-VLA means but reverse for $\pi_{0.5}$.

All six VLA-inference comparisons favor tagged FT on average, with 15 of 18 seed-level differences positive. Inference-time tags show a different pattern: after tagged FT, removing the inference tag improves X-VLA success at all three seeds, while adding it improves SmolVLA success in one run and reduces it in two.

VI. REAL-WORLD EXPERIMENTS

A. Experimental overview

We evaluate simulation pretraining on ten real-world manipulation tasks in a Franka FR3 workspace, with 25 real-world demonstrations collected per task for target-task fine-tuning. Each policy is evaluated with 24 trials per task (240 trials in total). All three configurations start from $\pi_{0.5}$ -base and use the same fine-tuning dataset. We compare an FT-only control with plain BC and approach-aware episode conditioning pretrained on 19,250 simulation demonstrations. The FT-only and plain-BC configurations use plain FT and untagged inference; conditioning uses tagged FT and tag-5 inference. Hardware, training, and evaluation details appear in the appendix.

Fig. 8 illustrates workspace alignment: we first adjust the simulated camera view, then configure the tabletop and background to match the real workspace. Re-rendering changes the observations while preserving recorded actions and object states. Configuring our reference workspace took approximately 30 minutes, after which recorded trajectories could be re-rendered for pretraining.

B. A simulation-pretraining baseline for real-world tasks

The plain-BC and no-pretraining arms differ only in whether AXIS pretraining precedes the same plain FT and

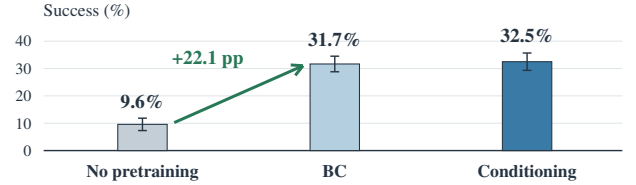


Fig. 9. **Real-world improvement from simulation pretraining.** All policies start from $\pi_{0.5}$ -base. The FT-only reference and plain BC use plain FT/untagged inference; conditioning uses tagged FT/tag 5. Whiskers show ± 1 evaluation SE over layout groups.

untagged inference. Success rises from 9.6% to 31.7%, improving on nine of ten tasks. **Simulation pretraining improves real-world target-task success**, establishing a baseline for quality-aware methods. Conditioning achieves 32.5% success, a gain of 0.83 pp over plain BC, with five tasks improving, four decreasing, and one tied. This real-world evaluation uses a different initialization, data size, and training recipe from the simulation benchmark.

VII. DISCUSSION AND CONCLUSION

In the evaluated configurations, approach-aware episode conditioning improves mean success over plain BC, and constant-tag FT improves mean success after conditioned pretraining under both tested inference protocols. Conditioning favors episode labels and filtering favors chunk labels on average, with model-dependent reversals. The study covers a shared tabletop simulation setting, with real-world validation using one VLA in a reference workspace. Testing larger pre-training budgets and additional workspaces and embodiments would clarify how broadly the findings transfer. Initialization and trainable modules vary across VLAs, so the comparisons do not isolate the causes of their different method rankings. Matched controls could test whether retaining data coverage explains conditioning’s benefits, and which aspects of initialization and adaptation explain the observed differences.

Our progress-based labels capture only part of demonstration quality and may undervalue necessary preparation or recovery with little immediate progress. Preference-based reward learning [33] and personalized policy conditioning [34] suggest ways to incorporate user-dependent objectives beyond task progress. Demonstration decomposition for planner alignment [35] also motivates examining quality over meaningful subtasks. Whether such signals improve quality-aware pretraining remains an open question.

Quality supervision could also complement task-relevant visual attention [36] and semantic guidance for pretrained policies [37]. Beyond nominal task success, deployment requires evaluating robustness to sensory and instruction perturbations [38] and scene-task inconsistencies [39]. Safety alignment through constrained learning [40] addresses a further objective that progress-based quality labels do not explicitly encode. These directions would extend the present evaluation beyond task completion to the reliability and safety of learned behavior. AXIS-Bench supports further comparisons by enabling new quality labels to be computed and trajectories to be re-rendered for user-defined workspaces and tasks.

A. Simulation experiment details

For $\pi_{0.5}$, pretraining updates all parameters for 20,605 updates (batch 64, learning rate 5×10^{-5}). FT starts from the EMA checkpoint (decay 0.999) and runs for 13,580 updates (batch 32, learning rate 2.5×10^{-5}), using LoRA for language/action modules and a trainable vision encoder. Pretraining and FT use 2,060 and 500 warmup updates, respectively. Target demonstrations follow a fixed within-task ranking by approach-aware progress established during dataset curation. SmolVLA and X-VLA use 20,621 pretraining and 6,790 FT updates, with batch 64 and learning rate 10^{-4} . SmolVLA starts from a pretrained SmolVLM2 with a randomly initialized action expert and updates only the action expert and state projection, keeping the VLM and vision encoder frozen; X-VLA updates all parameters, with one-tenth the learning rate for its VLM. Simulation normalization statistics are fitted separately to each stage’s data and shared across methods. All models receive five FT epochs, and we evaluate the final checkpoint. Images use 224×224 center crops for $\pi_{0.5}$ and padding to 512×512 for SmolVLA or 224×224 for X-VLA. SmolVLA uses AdamW with cosine scheduling; X-VLA uses domain-prompt index 20.

Random-selection controls use a fixed sampling seed and match the retained time steps and training updates of quality filtering. Quality-tag dropout in simulation uses two independent dropout operations with probabilities 0.15 and 0.05 during both pretraining and tagged FT. Pretraining masks are sampled once offline and shared across corresponding experiments; tagged-FT masks are resampled each epoch. For evaluation, the 24 tasks comprise six containment, six support-placement, seven spatial-relation, two separation, and three short-sequence tasks. The native prediction horizons are 10, 50, and 30 actions for $\pi_{0.5}$, SmolVLA, and X-VLA, respectively. Policies return the first ten predicted actions, and the client executes eight before replanning. The $H = 10$ labeling window is independent of the native prediction horizon. Matched FT-only controls omit AXIS pretraining while retaining the corresponding initialization.

Relative improvements use $100(\text{SR}/\text{SR}_{\text{BC}} - 1)$ with unrounded success rates; marginal SEs are scaled by $100/\text{SR}_{\text{BC}}$. The reference is each VLA’s 5K plain BC with plain FT and untagged inference. Table I uses the same BC reference for each VLA and training seed: we compute $100(\text{SR}_A - \text{SR}_B)/\text{SR}_{\text{BC}}$ within each seed, then report the mean and sample SD. Thus, row differences are expressed relative to BC, including comparisons between two non-BC configurations. The relative gains in Fig. 1 use the same calculation for episode conditioning versus BC. Training-seed SDs describe variation across three runs, not confidence intervals or significance tests.

B. Real-world experiment details

The Franka FR3 setup uses a Robotiq gripper, ZED 2i external camera, and ZED Mini wrist camera. Pretraining

and FT use 20K and 16K updates, respectively, with batch 64 and AdamW cosine learning rates from 2.5×10^{-5} to 2.5×10^{-6} . Pretraining uses LoRA for language/action modules and a frozen vision encoder; FT makes the vision encoder trainable. The learning-rate schedules use 1,000 and 1,600 warmup updates for pretraining and FT, respectively. Both stages use fixed DROID normalization statistics without re-estimation, 224×224 center crops, and a 15-action prediction horizon. Simulation observations are rendered at 640×360 with workspace-aligned backgrounds and camera views. Every checkpoint includes the same normalization statistics for training and inference. Pretraining applies color jitter, blur, posterization, sharpness, and grayscale augmentation following MolmoB0T [19].

Conditioning assigns episode-score quintiles to 17,758 demonstrations and neutral tag 3 to 1,492 unscorable episodes, without pretraining tag dropout. The ten real-world tasks cover nesting, stacking, alignment, rotation, pressing, closing, insertion, and extraction. Evaluation uses 12 layouts per task and two trials per layout; equal trial counts make task-averaged and pooled success rates identical.

REFERENCES

- [1] A. O’Neill, A. Rehman, A. Maddukuri *et al.*, “Open X-Embodiment: Robotic Learning Datasets and RT-X Models,” in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 6892–6903.
- [2] D. Ghosh, H. R. Walke, K. Pertsch *et al.*, “Octo: An open-source generalist robot policy,” in *Proceedings of Robotics: Science and Systems*, 2024.
- [3] M. J. Kim, K. Pertsch, S. Karamcheti *et al.*, “OpenVLA: An Open-Source Vision-Language-Action Model,” in *Proceedings of The 8th Conference on Robot Learning*, 2025, pp. 2679–2713.
- [4] K. Black, N. Brown, J. Darpinian *et al.*, “ $\pi_{0.5}$: a Vision-Language-Action Model with Open-World Generalization,” in *Conference on Robot Learning*, 2025, pp. 17–40.
- [5] J. Hejna, C. A. Bhateja, Y. Jiang, K. Pertsch, and D. Sadigh, “ReMix: Optimizing Data Mixtures for Large Scale Imitation Learning,” in *Conference on Robot Learning*, 2025, pp. 145–164.
- [6] Physical Intelligence, B. Ai, A. Amin *et al.*, “ $\pi_{0.7}$: a Steerable Generalist Robotic Foundation Model with Emergent Capabilities,” *arXiv preprint arXiv:2604.15483*, 2026.
- [7] J. Hejna, S. Mirchandani, A. Balakrishna *et al.*, “Robot data curation with mutual information estimators,” in *Proceedings of Robotics: Science and Systems*, 2025.
- [8] Y. Zhang, Y. Xie, H. Liu *et al.*, “SCIZOR: A Self-Supervised Approach to Data Curation for Large-Scale Imitation Learning,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2026.
- [9] Physical Intelligence, A. Amin, R. Aniceto *et al.*, “ $\pi_{0.6}^*$: a VLA That Learns From Experience,” *arXiv preprint arXiv:2511.14759*, 2025.
- [10] Z. Wang, J. Li, Y. Cui *et al.*, “World Value Models for Robotic Manipulation,” *arXiv preprint arXiv:2606.24742*, 2026.
- [11] X. B. Peng, A. Kumar, G. Zhang, and S. Levine, “Advantage-weighted regression: Simple and scalable off-policy reinforcement learning,” *arXiv preprint arXiv:1910.00177*, 2019.
- [12] M. Shukor, D. Aubakirova, F. Capuano *et al.*, “SmolVLA: A vision-language-action model for affordable and efficient robotics,” *arXiv preprint arXiv:2506.01844*, 2025.
- [13] J. Zheng, J. Li, Z. Wang *et al.*, “X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model,” in *International Conference on Learning Representations*, 2026.
- [14] M. Zhao, D. Huang, Y. Tang *et al.*, “AXIS: A Growable Community-Driven Data Engine for Scalable Robot Manipulation,” *arXiv preprint arXiv:2607.21588*, 2026.
- [15] H. R. Walke, K. Black, T. Z. Zhao *et al.*, “BridgeData V2: A Dataset for Robot Learning at Scale,” in *Conference on Robot Learning*, 2023, pp. 1723–1736.

- [16] A. Khazatsky, K. Pertsch, S. Nair *et al.*, “DROID: A large-scale in-the-wild robot manipulation dataset,” in *Proceedings of Robotics: Science and Systems*, ser. Robotics: Science and Systems - Online Proceedings. Robotics: Science and Systems, Jul. 2024, pp. 1–13.
- [17] A. Mandlekar, Y. Zhu, A. Garg *et al.*, “RoboTurk: A Crowdsourcing Platform for Robotic Skill Learning through Imitation,” in *Conference on Robot Learning*, 2018, pp. 879–893.
- [18] A. Mandlekar, S. Nasiriany, B. Wen *et al.*, “MimicGen: A Data Generation System for Scalable Robot Learning using Human Demonstrations,” in *Conference on Robot Learning*, 2023, pp. 1820–1864.
- [19] A. Deshpande, M. Guru, R. Hendrix *et al.*, “MolmoB0T: Large-scale simulation enables zero-shot manipulation,” in *Proceedings of the 10th Conference on Robot Learning*, 2026.
- [20] A. Mandlekar, D. Xu, J. Wong *et al.*, “What Matters in Learning from Offline Human Demonstrations for Robot Manipulation,” in *Conference on Robot Learning*, 2022, pp. 1678–1690.
- [21] S. Dass, A. Khaddaj, L. Engstrom, A. Madry, A. Ilyas, and R. Martín-Martín, “DataMIL: Selecting data for robot imitation learning with datamodels,” in *International Conference on Learning Representations*, 2026.
- [22] C. Agia, R. Sinha, J. Yang *et al.*, “CUPID: Curating Data your Robot Loves with Influence Functions,” in *Conference on Robot Learning*, 2025, pp. 2907–2932.
- [23] U. Ghosh, D. S. Raychaudhuri, J. Li, K. Karydis, and A. K. Roy-Chowdhury, “Robust Offline Imitation Learning from Diverse Auxiliary Data,” *Transactions on Machine Learning Research*, 2025.
- [24] L. Chen, K. Lu, A. Rajeswaran *et al.*, “Decision Transformer: Reinforcement Learning via Sequence Modeling,” in *Advances in Neural Information Processing Systems*, vol. 34, 2021, pp. 15 084–15 097.
- [25] S. Emmons, B. Eysenbach, I. Kostrikov, and S. Levine, “RvS: What is Essential for Offline RL via Supervised Learning?” in *International Conference on Learning Representations*, 2022.
- [26] T. Korbak, K. Shi, A. Chen *et al.*, “Pretraining language models with human preferences,” in *International conference on machine learning*. PMLR, 2023, pp. 17 506–17 533.
- [27] T. Gao, A. Wettig, L. He, Y. Dong, S. Malladi, and D. Chen, “Metadata conditioning accelerates language model pre-training,” in *International Conference on Machine Learning*, 2025.
- [28] W. Kang, J. Mun, S. Lee, and B. Roh, “Noise-aware learning from web-crawled image-text data for image captioning,” in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, 2023, pp. 2930–2940.
- [29] Y. Zhu, J. Wong, A. Mandlekar *et al.*, “robosuite: A Modular Simulation Framework and Benchmark for Robot Learning,” *arXiv preprint arXiv:2009.12293*, 2020.
- [30] E. Todorov, T. Erez, and Y. Tassa, “MuJoCo: A Physics Engine for Model-Based Control,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.
- [31] K. Lyu, H. Zhao, X. Gu, D. Yu, A. Goyal, and S. Arora, “Keeping LLMs aligned after fine-tuning: The crucial role of prompt templates,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 118 603–118 631, 2024.
- [32] S. Kotha, J. Springer, and A. Raghunathan, “Understanding catastrophic forgetting in language models via implicit inference,” in *International Conference on Learning Representations*, 2024.
- [33] U. Ghosh, D. S. Raychaudhuri, J. Li, K. Karydis, and A. Roy-Chowdhury, “Reducing Oracle Feedback with Vision-Language Embeddings for Preference-Based RL,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2026.
- [34] Z. Wang, H. Jiang, S. Dong, Y. Wang, H. Qiu, and J. Li, “Drive My Way: Preference Alignment of Vision-Language-Action Model for Personalized Driving,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.
- [35] M. Yan, Y. Wang, Z. Liu, and J. Li, “RDD: Retrieval-based demonstration decomposer for planner alignment in long-horizon tasks,” in *Advances in Neural Information Processing Systems*, vol. 38, 2025, pp. 47 336–47 361.
- [36] L. Gong, F. Bahrani, Y. Zhou, A. Banayeezade, J. Li, and E. Bıyık, “AutoFocus-IL: VLM-based Saliency Maps for Data-Efficient Visual Imitation Learning without Extra Human Annotations,” in *International Conference on Robotics and Automation (ICRA)*, 2026.
- [37] M. Gao, X. Li, Q. Yan *et al.*, “GUIDES: Guidance Using Instructor-Distilled Embeddings for Pre-trained Robot Policy Enhancement,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2026.
- [38] H. Jiang, Y. Chaudhary, Y. Wang *et al.*, “NavTrust: Benchmarking Trustworthiness for Embodied Navigation,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2026.
- [39] T. Chakraborty, U. Ghosh, X. Zhang *et al.*, “HEAL: An empirical study on hallucinations in embodied agents driven by large language models,” in *Findings of the Association for Computational Linguistics: EMNLP 2025*. Association for Computational Linguistics, 2025, pp. 21 226–21 243.
- [40] B. Zhang, Y. Zhang, J. Ji *et al.*, “SafeVLA: Towards safety alignment of vision-language-action model via constrained learning,” in *Advances in Neural Information Processing Systems*, vol. 38, 2025, pp. 153 335–153 373.