GENERALIZABLE ACTION EDITOR: SHARING CORRECTIONS ACROSS TASKS FOR ONLINE POST-TRAINING OF PRETRAINED ROBOT POLICIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Different manipulation tasks can share local correction structure even when their goals differ. We introduce the *Generalizable Action Editor* (GAE), a shared policy that learns through online reinforcement learning to edit the action proposals of a frozen pretrained robot policy. Conditioning on local observations and the base proposal allows one editor to express corrections across tasks. To allocate limited training resources, we score each executed correction by its critic-estimated value gain over the same-state base proposal. This score guides CISA replay sampling and score-aware retention within ReSOP, our asynchronous multi-task learning system. A correction-support analysis characterizes how source-context coverage and compatibility of local edits shape transfer. On RoboTwin 2.0 Clean, we train one editor on 46 source tasks and evaluate it across all 50 tasks, including four held-out tasks without editor-parameter updates. Shared editing raises macro success from 90.00% to 90.84%. Four-task comparisons further favor intervention-value-based sampling over uniform and TD-error prioritization under matched learner-update budgets.

1 INTRODUCTION

Pretrained robot policies can now span broad manipulation repertoires across tasks and embodiments, from action-chunked imitation and diffusion policies to vision-language-action (VLA) models (Zhao et al., 2023; Chi et al., 2024; Ghosh et al., 2024; Kim et al., 2025; Black et al., 2025b;a; Physical Intelligence et al., 2026). Yet a policy that supplies useful task behavior can still fail during local alignment or contact. Online post-training can refine that behavior. We ask whether the corrections learned in doing so can become a reusable resource across tasks, rather than remaining specific to each task's training process.

Online RL already provides effective interfaces for refining pretrained robot policies, including external action corrections, latent steering, and direct policy updates (Johannink et al., 2019; Yuan et al., 2025; Wagenmaker et al., 2025; Xu et al., 2026; Dong et al., 2026). These interfaces make online correction learning possible. Our focus is how to pool that experience across tasks and reuse the resulting editor on tasks held out from its training.

Grasping, placing, and insertion can involve similar local alignment corrections despite their different task goals. The appropriate adjustment depends on both local geometry and the motion the base proposes. We introduce the **Generalizable Action Editor (GAE)**, one shared editor conditioned on visual observations, proprioception, and the current base proposal. It learns from online rewards while the base remains frozen, pooling correction experience across tasks.

Our correction-support account asks when this pooled experience can help a held-out task. The analysis considers source tasks that provide nearby correction contexts with locally compatible edits: additional tasks need not help if their contexts are irrelevant or their required corrections conflict. Figure 1 illustrates this distinction. Appendix A.6 separates nearby source evidence and edit compatibility from the implemented editor's approximation and learning gap; Section 4.5 evaluates held-out behavior across source breadths.

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.

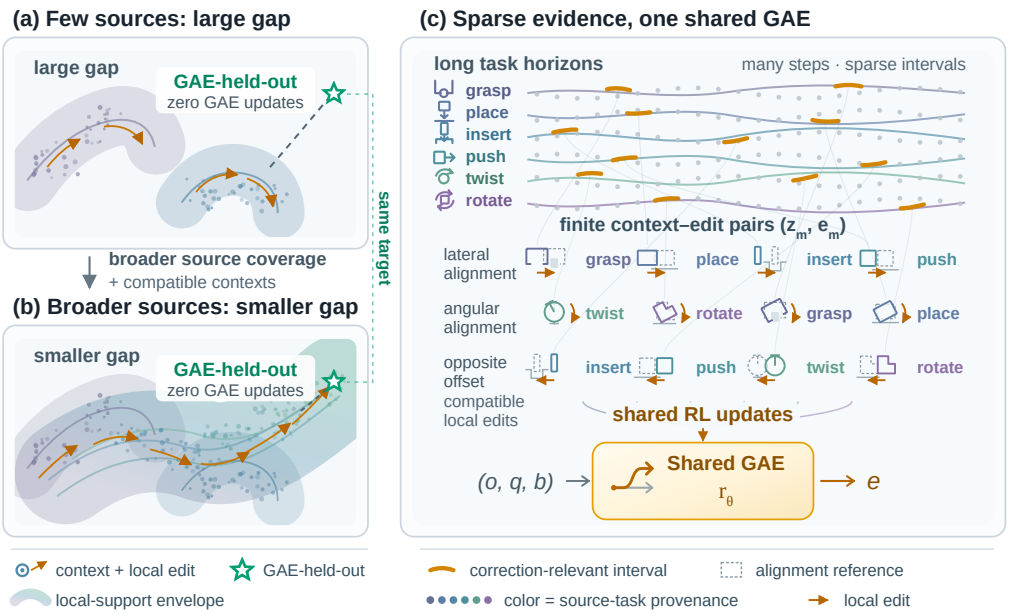


Figure 1: **From sparse task-local corrections to shared empirical support.** (a–b) Few sources leave a large support gap; additional locally compatible contexts can reduce the gap to the same GAE-held-out target. (c) Task-colored context–edit pairs illustrate recurring alignment relations: similar offsets can admit compatible edits, while opposite offsets can require opposite edits. Online RL pools this experience into one shared editor; its horizontal interface maps sensory context and the base proposal to a local edit. Dashed outlines and support envelopes schematically depict local alignment and context coverage. GAE-held-out tasks are excluded from editor training.

Sharing also requires choosing which correction experiences to revisit. Figure 2 shows a motivating diagnostic from a RoboTwin `hanging_mug` rollout. The estimated value added by an edit varies along the trajectory, suggesting that correction experience need not receive equal training weight. Appendix C.1 describes this exploratory trace and its critic configuration. We define *Critical Intervention Value* as the critic-estimated value contrast between an executed correction and the same-state base proposal. CISA uses this signal to allocate within-task learner updates. Score-aware retention uses it separately to determine which transitions survive finite replay turnover.

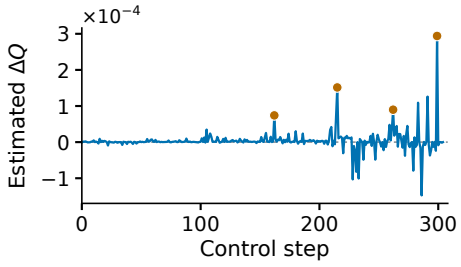


Figure 2: **Correction value varies over time.** Signed ΔQ during `hanging_mug`; dots mark separated positive peaks.

ReSOP implements this learning loop across asynchronous task streams on RLinf. Shared replay pools experience labeled by task. Critic scoring responds to requests for new or refreshed scores, independently of optimizer updates, and supplies the values used for sampling and retention. Figure 3 presents the collection, learning, and scoring paths.

We evaluate one shared GAE on 46 source tasks and four GAE-held-out tasks in RoboTwin 2.0 Clean, compare CISA with uniform and TD-error-prioritized sampling on four source tasks under fixed score-aware retention and matched learner-update budgets, and study held-out behavior across nested source-task sets. A separate two-task sequence tests CIV-guided selection of historical replay memory. Across all 50 tasks, the S46-trained editor raises macro success from 90.00% to 90.84%, with 88.75% on the four GAE-held-out tasks. The selected-checkpoint success is higher with CISA than either sampling comparator on all four S4 tasks.

Our contributions are:

- We introduce one shared, proposal-conditioned correction field for online post-training across robot tasks, together with a correction-support account of how context coverage and local edit consistency shape reuse.
- We define a base-referenced Critical Intervention Value and use it for two distinct finite-resource decisions: CISA allocates task-local learner updates, while score-aware retention allocates replay-row lifetime.
- We realize these mechanisms in ReSOP, an asynchronous multi-task RL system built on RLinf that couples shared correction learning with demand-driven scoring and replay allocation.

2 Related Work

Modular action interfaces. Hierarchical robot policies can separate broad reasoning from precise execution through an intermediate action interface. Coarse end-effector paths and sparse geometric plans, for example, can condition reusable low-level action modules (Li et al., 2025; Liu et al., 2026b). GAE places online correction at this boundary: the editor receives the current executable proposal together with local sensory context and learns how that proposal should change.

Online post-training and residual refinement. Online post-training improves a competent base through several adaptation interfaces. External action-space corrections preserve a fixed controller or imitation policy while learning a separate residual policy (Silver et al., 2018; Johannink et al., 2019; Ankile et al., 2025b; Yuan et al., 2025; Ankile et al., 2025a; Muttaqien et al., 2026). Other interfaces steer latent noise or expose compact actor–critic heads (Wagenmaker et al., 2025; Xu et al., 2026), reshape a generative policy's action distribution (Sun & Song, 2026), or directly fine-tune VLA parameters (Dong et al., 2026). BEE (Zhao et al., 2026) combines VLA latent features with human corrections that inform dimension-wise policy constraints, linking representation reuse with adaptive intervention constraints. Force-aware proposals and contact-focused refinement extend this line to precision assembly (Deng et al., 2026). Uncertainty-aware residual RL gates exploration using base-policy confidence (Dodeja et al., 2026); its analysis also identifies settings in which confidence and correctness become weakly related. In demonstration-supervised ACT refinement, UGR learns a lightweight uncertainty head to select positions within an action chunk (Wang et al., 2026). These methods select where to apply corrections. CIV instead scores realized corrections for subsequent replay allocation; the signals serve different decisions. Residual interactions can also yield recovery data for later distillation (Xiao et al., 2026), while general post-training systems scale VLA collection and optimization (Pan et al., 2026; Li et al., 2026). We take these refinement interfaces as the starting point for ReSOP's resource-sharing problem: learning one proposal-conditioned correction field across task streams while allocating shared online data by estimated intervention value.

Critical states and replay allocation. Prior methods allocate learning effort unevenly in either time or replay. Rollout diagnostics in DICE-RL associate value gains and entropy reductions with contact-sensitive phases (Sun & Song, 2026), motivating local rather than uniformly distributed refinement. Task-Level Iterative Learning Control optimizes error at a designated critical point (Suresh & Atkeson, 2026), while replay prioritization conventionally allocates updates using temporal-difference error (Schaul et al., 2016). Analyses of prioritized replay identify stale priorities and insufficient coverage as important failure modes (Pan et al., 2022), while value-of-experience analyses connect replay utility to value learning (Li et al., 2021). Residual RL also poses a distinct value-estimation problem around a strong base policy (Ma et al., 2026). CISA operationalizes criticality as the critic-estimated value contrast of the realized edit against the same-state base proposal. TD-error priority measures Bellman inconsistency; CIV instead compares two candidate actions at a common observation. These signals can rank transitions differently. This distinction motivates our matched comparison of CIV, absolute TD-error priority, and Uniform under fixed retention (Section 4.3). ReSOP also uses CIV to preserve valuable intervention records as finite replay fills and replaces old data, separating retention time from sampling probability. Adaptive task allocation provides a separate level of interaction control in multi-task learning (Sharma et al., 2017; Kalashnikov et al., 2022). ReSOP separates task-level collection and learning shares from row-level critic scores. Horizon-weighted collection and balanced learning are specified in Section 3.3 and Appendix A.4.

Parameter sharing and context-dependent routing. Mixtures of experts provide conditional capacity within a shared policy. DyGRO-VLA routes residual experts using an observation-derived task embedding learned with task-supervised contrastive learning (Lin et al., 2026). Task-indexed prototypes provide training references; inference uses the predicted embedding. GAE routes directly on visual–proprioceptive features and the base proposal, with an expert-load balancing objective. We study the reuse of proposal-conditioned corrections and the allocation of online correction data. Continued target training and retention of prior skills extend this setting to continual robot learning (Lesort et al., 2020; Liu et al., 2023b; 2026a; Lu et al., 2026).

3 METHOD

GAE learns a shared proposal-conditioned correction field from online experience across tasks. ReSOP allocates that experience at three scopes: task allocation sets collection and learning shares, score-aware retention controls replay lifetime, and CISA selects learner updates within a task. We define the editor, its critic-based allocation signal, and the asynchronous learning loop shown in Figure 3.

3.1 A SHARED PROPOSAL-CONDITIONED CORRECTION FIELD

Consider a task family $\mathcal{T}$ and a pretrained base policy π^b. For task $m \in \mathcal{T}$, the base receives its usual task-conditioned inputs and emits an action chunk $A_k^b = (a_{k,1}^b, \ldots, a_{k,H}^b)$. ReSOP caches the chunk and exposes its current proposal a_t^b to the same GAE at every physical control step. Let o_t be the multi-view RGB observation, q_t the robot proprioception, and $b_t = \mathcal{A}(a_t^b)$ the proposal rescaled using fixed action statistics from the base policy (the affine action coordinate). GAE produces a bounded edit $e_t = r_\theta(o_t, q_t, b_t)$ and the controller executes

$$\bar{a}_t = b_t + \lambda_m e_t, \qquad a_t^{\text{exec}} = \mathcal{A}^{-1}(\bar{a}_t), \tag{1}$$

where $\lambda_m = 5 \times 10^{-4}$ for every task throughout training and evaluation (Appendix A.2). The editor acts at control rate while the base remains at chunk rate. Appendix A.1 specifies chunk execution and the affine coordinate. GAE learns from online RL rewards.

All tasks use the same editor parameters and input tuple (o_t, q_t, b_t). Task labels organize collection, replay indexing, and critic selection. For held-out evaluation, editor parameters and the action scale remain fixed. Sharing is useful when the proposal and sensory context distinguish the required local corrections. Appendix A.6 separates remaining context ambiguity from source–target mismatch and actor approximation.

Context-conditioned experts. We parameterize the editor with a sparse mixture of E experts. A router takes fused visual features, proprioception, and the base proposal and selects two experts per input. Their weighted Gaussian means and log standard deviations define a single Gaussian before the policy's tanh transform. All tasks share the router and expert pool, providing capacity for distinct context-dependent correction regimes. Appendix A.5 specifies routing, initialization, and expert-load balancing. Fixed-state diagnostics in Appendix C.2 show overlapping expert use across tasks and changes in their combinations during training.

3.2 CRITICAL INTERVENTION VALUE AND DATA ALLOCATION

Brief corrections can matter disproportionately along a trajectory that otherwise follows a competent base, motivating nonuniform update allocation. Policy uncertainty can guide exploration, but does not directly quantify the benefit of a particular edit: several different actions may be equally effective, whereas a concentrated policy may consistently miss an alignment. ReSOP therefore estimates the value of the *realized intervention* relative to the base proposal.

For stored transition (replay row) i, let $x_i = (o_i, q_i)$, let b_i be the corresponding base proposal, and let $\bar{a}_i$ be its recorded executed affine action. The tuple $(x_i, b_i, \bar{a}_i)$ is an *intervention record*. We use a task-specific K-head critic ensemble for each source task m, with a shared visual encoder. Define $Q_{\phi,m}^{\min}(x, a) = \min_{h \in \{1,\ldots,K\}} Q_{\phi,m,h}(x, a)$ and

$$c_i = \left[Q_{\phi,m_i}^{\min}(x_i, \bar{a}_i) - Q_{\phi,m_i}^{\min}(x_i, b_i) \right]_+, \qquad [v]_+ = \max\{0, v\}. \tag{2}$$

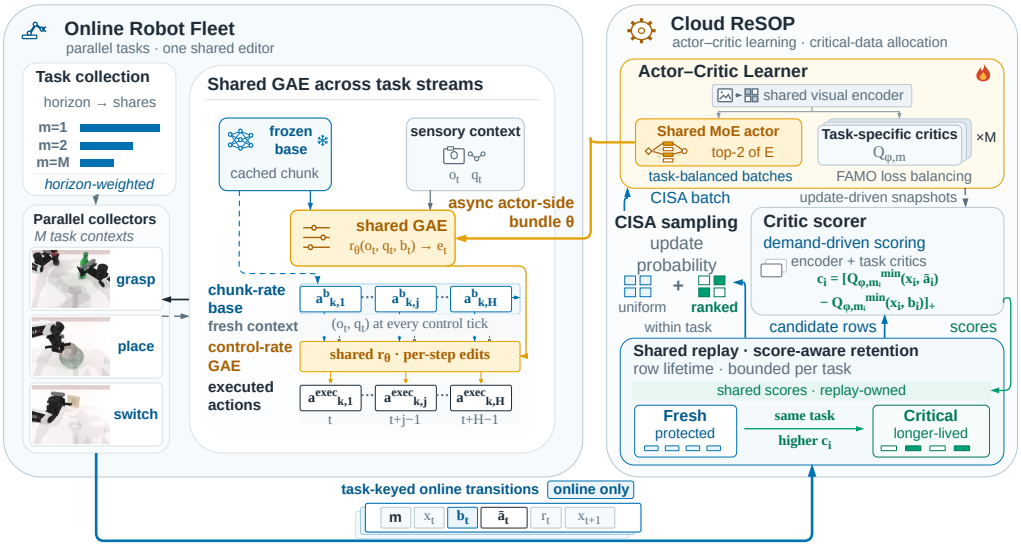


Figure 3: **Cross-task correction learning and critical-data allocation in ReSOP.** Collectors execute a shared GAE over frozen-base proposals and return transitions labeled by task. Collection is horizon-weighted and learner tasks are balanced. One visual encoder serves the actor's shared top-2 expert pool and M task-specific critic ensembles, each with K heads. FAMO balances critic training losses. A separate scorer evaluates transitions using read-only copies of the learner's encoder and critics. Replay stores these scores for CISA sampling and longer-term retention after the protected Fresh stage. CISA batches train the learner, which sends updated actor and encoder parameters to collectors. Bars and expert blocks are schematic; the inset shows chunk-rate proposals and control-rate editing. Task thumbnails are RoboTwin 2.0 demonstrations (Chen et al., 2025).

We denote the signed contrast inside the brackets by ΔQ_i, so that $c_i = [\Delta Q_i]_+$. Here $Q^{\min}$ is the minimum across the online critic ensemble, evaluated using the scorer's read-only model copy. Task identity m_i selects the ensemble, whose input tuple is (o, q, a). CIV compares the stored action pair through two critic queries at the same observation. It estimates intervention value; the unexecuted base-action query remains subject to critic extrapolation error (Appendix A.3; Section 5).

Scoring reuses the learned critic and stored action pair; it requires neither a separate uncertainty predictor nor additional base-policy samples. It is performed after collection to allocate training data. Critical-Intervention-State-Aware (CISA) Sampling turns this score into a task-local update distribution:

$$P_{\mathrm{CISA}}(i \mid m) = (1-\rho)U_m(i) + \rho P_{\mathrm{crit}}(i \mid m), \tag{3}$$

where U_m covers retained rows uniformly and P_{crit} emphasizes high-ranked c_i values. This distribution defines the reweighted actor and critic objectives; the losses use no inverse-probability importance weights. The uniform component maintains nonzero probability for every retained row. Appendix A.3 gives the rank and tie rules.

The same score governs a separate storage decision. New transitions receive an initial period of protected retention in a Fresh region; after scoring, stronger task-local candidates can replace lower-value rows in a longer-lived Critical region. Task-local capacity control prevents one task from monopolizing replay, and scores expire so an obsolete critic cannot determine lifetime indefinitely. CISA thus allocates update probability among retained rows, whereas score-aware retention allocates row lifetime under finite capacity. Appendix A.3 specifies both rules and their refresh semantics. Section 4.6 also tests CIV at a task boundary, where it selects a fixed historical memory for learning the next task.

3.3 Asynchronous collection, scoring, and learning

Built on RLinf's distributed execution substrate (Yu et al., 2026), ReSOP separates collection, actor–critic optimization, critic scoring, and replay storage for shared correction learning.

Parallel collectors receive the latest published GAE actor and visual encoder, compose the editor with their task's frozen base, and append task-labeled online transitions to shared replay. Collection probabilities are proportional to configured task horizons, while learner batches balance currently sampleable tasks. CISA selects transitions within each task (Appendix A.4). Task shares, transition retention time, and within-task update probability remain separate allocation levels. Table 6 specifies the S4 comparison controls.

Cloud ReSOP separates the actor–critic learner, critic scorer, and shared replay. The learner jointly updates GAE with Soft Actor-Critic (Haarnoja et al., 2018a) and task-specific REDQ-style ensembles (Chen et al., 2021). Critic losses train the shared visual encoder; actor updates use detached visual features. FAMO (Liu et al., 2023a) balances per-task critic losses, while the actor's RL objective averages per-task losses equally and adds a separate expert-load balancing term. Collectors receive only the actor, encoder, and input-processing state needed to compute edits, so held-out execution needs no target-task critic. Scoring uses read-only copies of the visual encoder, task-specific critics, and input transforms; only the learner updates their trainable parameters. The task-sharded implementation synchronizes each logical learner update across worker processes, distributing complete task-specific critic ensembles while synchronizing the shared actor and encoder. Collection and publication remain asynchronous; update budgets count these synchronized global updates.

Increasing collector parallelism creates score demand independently of optimizer progress. ReSOP therefore uses demand-driven scoring: new candidates and requests to refresh old scores trigger scoring work. Separately, update-driven publication supplies scorers with newer read-only model copies. Returned values update a replay-owned score table used by both CISA and retention. This split permits scoring to run asynchronously. Appendix A.5 specifies where scoring runs, how outdated results are handled, and the available execution options.

4 EXPERIMENTS

We first evaluate a shared GAE on ten- and 46-task source sets and its zero editor-parameter-update use on four GAE-held-out tasks in RoboTwin 2.0 Clean. We then compare CISA with uniform within-task and TD-error-prioritized sampling under fixed score-aware retention and matched learner-update budgets, compare visual encoders with their regularization, examine held-out behavior across source-task breadths, and test historical-memory selection in a two-task sequence.

4.1 EXPERIMENTAL SETUP

We evaluate 50 RoboTwin 2.0 tasks (Chen et al., 2025) with a LingBot-VLA checkpoint (Wu et al., 2026) fine-tuned for this benchmark suite. All source and held-out tasks use the same fixed action scale $\lambda_m = 5 \times 10^{-4}$ (Appendix A.2). The base remains frozen during GAE training and evaluation, and every GAE run trains exclusively from its own online interaction. The shared-training split contains 46 source tasks and four *GAE-held-out* targets that supply no GAE update or replay row during source training. This terminology isolates editor transfer: the benchmark-tuned base already has task-specific competence on every target. All studies use Clean for training and evaluation. Throughout, "zero-update" denotes zero target-side editor-parameter updates.

We use one training run per arm and measure terminal success over 100 episodes per task, macro-averaged across tasks: each task's success rate receives equal weight. The S4 ablations match the training budget at 500,000 learner updates and use a common evaluation protocol. Each arm selects one late-training checkpoint using success on the source anchor `hanging_mug`, the task used for model selection; the same checkpoint is evaluated on all tasks. The anchor evaluation is reused in the macro average, which is therefore subject to checkpoint-selection bias. Appendix B.1 specifies the candidate grid and tie rule. Runs were performed on different hardware; environment interactions, wall-clock time, and compute are not matched. Appendix B specifies the split, evaluation cadence, stochastic controls, and budget accounting. Tables 6 and 7 summarize the comparison controls.

4.2 ONE SHARED EDITOR ACROSS SOURCE AND GAE-HELD-OUT TASKS

This study tests whether one editor can improve a broad source family and be composed with held-out tasks before target-side updates. On the ten-task source set S_{10}, shared editing raises macro

success from 91.30% to 92.30% (+1.00 percentage points; Table 1), with 86.75% on the four GAE-held-out tasks. Appendix B.2 gives the task-level comparison.

With 46 sources and a 3M-update training budget, source-task success rises from 90.13% to 91.02% (+0.89 percentage points). The same editor reaches 88.75% on H4. Across all 50 tasks, macro success increases from 90.00% to 90.84% (+0.84 percentage points).

Table 1: Shared GAE evaluation in Clean. Each editor is trained on its source set and evaluated on source and GAE-held-out tasks. Entries are macro success over 100 episodes per task; Δ is relative to the frozen base in percentage points.

Source set	Evaluation split	Frozen base	Base+GAE	Δ
S_{10}	Source (10)	91.30%	92.30%	+1.00
	GAE-held-out (4)	88.50%	86.75%	-1.75
S_{46}	Source (46)	90.13%	91.02%	+0.89
	GAE-held-out (4)	88.50%	88.75%	+0.25
	All (50)	90.00%	90.84%	+0.84

Figure 4 shows S46 task-level changes alongside the averages: 21 tasks improve, 16 tie, and 13 decline. The largest gains are on `click_bell` (+21 pp), `stack_bowls_three` (+13 pp), and `blocks_ranking_size` (+10 pp). The largest decline is -4.00 pp, shared by `place_bread_basket`, `stack_blocks_three`, and the held-out `place_fan`.

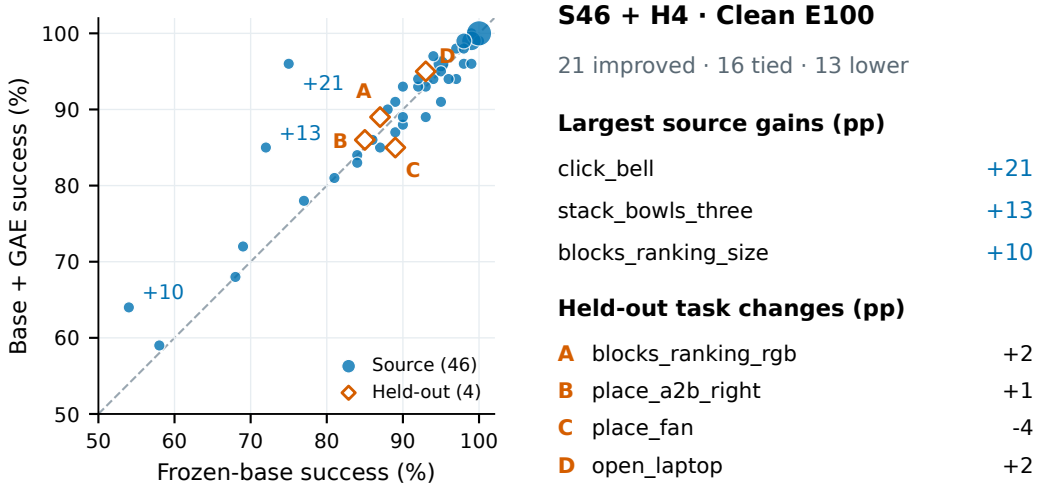


Figure 4: Task-level shared-GAE outcomes in Clean. The diagonal denotes no change; marker shape distinguishes source and GAE-held-out tasks. Coincident source outcomes are grouped by marker area; no jitter is applied. Complete success counts and episode lengths appear in Appendix B.1.

4.3 CISA SAMPLING UNDER MATCHED TRAINING BUDGETS

Under a training budget of 500,000 learner updates and common anchor-based checkpoint selection, CISA has higher macro success than uniform within-task sampling on the four Clean source tasks (S_4), from 82.75% to 86.00% (+3.25 percentage points; Table 2). A matched TD-error-priority arm reaches 81.00%, below CISA by +5.00 percentage points. This arm replaces CIV with absolute TD error in the same sampling framework; retention continues to use CIV. Task allocation, replay capacity, architecture, and action scales are fixed. The training budget and checkpoint-selection procedure are matched; the selected update index can differ across arms.

CISA has higher observed success than either comparator on all four tasks. Relative to Uniform, the largest change is on `hanging_mug` (65% to 71%). The frozen-base reference averages 84.75%:

both comparators are below this reference, whereas CISA is above it. These outcomes favor CIV as the sampling signal within the matched S4 framework. Table 12 gives every task-level comparison.

Table 2: CISA sampling ablation on four Clean source tasks (S_4). All trained arms share CIV-based retention, a 500k-update training budget, and source-anchor checkpoint selection (Appendix B.1). Success is macro-averaged over 100 episodes per task, including the reused anchor evaluation. Δ is relative to the frozen base in percentage points.

Policy / sampling	Success ↑	Δ vs. base (pp)
Frozen base (reference)	84.75%	—
GAE / Uniform	82.75%	-2.00
GAE / TD-error priority	81.00%	-3.75
GAE / CISA	**86.00%**	+1.25

4.4 Visual encoder ablation on S4

The shared visual encoder supplies features to both the editor and the critics. We compare two visual packages under the same 500,000-learner-update budget on S_4: a lightweight CNN with A-LIX regularization (Cetin et al., 2022) and random shift, and an ImageNet-pretrained ResNet-10 with random shift. Both use CISA and CIV-based score-aware retention, with the same anchor-based checkpoint-selection rule. The CNN+A-LIX arm reuses the full-configuration CISA result from Section 4.3.

CNN+A-LIX achieves 86.00% macro success, compared with 81.00% for ResNet-10 (+5.00 percentage points; Table 3). Success is higher on all four tasks. The largest difference is on `turn_switch`, from 67% to 78%. Mean episode length decreases from 158.73 to 154.56 (4.17 fewer steps), with reductions on three of the four tasks. This comparison favors the CNN+A-LIX visual package for the evaluated online learning setting, motivating its use in the shared editor.

Table 3: Visual encoder ablation on S_4 Clean under a 500k-update training budget, with CISA and score-aware retention fixed. Both arms follow the source-anchor selection protocol; CNN+A-LIX reuses the CISA result in Table 2. Each task uses 100 evaluation episodes; steps average all episodes, including failures. Saved steps are ResNet-10 minus CNN+A-LIX.

	Success ↑		Mean episode steps		
Task	ResNet-10	CNN+A-LIX	ResNet-10	CNN+A-LIX	Saved ↑
`turn_switch`	67%	78%	97.08	84.96	12.12
`move_playingcard_away`	98%	100%	108.81	106.27	2.54
`pick_diverse_bottles`	91%	95%	107.30	107.33	-0.03
`hanging_mug`	68%	71%	321.72	319.69	2.03
Macro average	81.00%	**86.00%**	158.73	154.56	4.17

4.5 Source-breadth evaluation

Table 4: GAE-held-out success across source sets in Clean, with 100 episodes per target. Updates are total source-training budgets; Δ is relative to the frozen base in percentage points.

Source set	Training updates	Success ↑	Δ vs. base (pp)
Frozen base	—	88.50%	—
S_1	100k	85.25%	-3.25
S_4	500k	86.50%	-2.00
S_{10}	1M	86.75%	-1.75
S_{46}	3M	88.75%	+0.25

Motivated by the correction-support analysis, we evaluate nested source sets on the same H4 targets without editor updates (Table 4). Macro success rises from 85.25% with S_1 through 86.50% with

S_4 and 86.75% with S_{10} to 88.75% with S_{46}. At S_{46}, three targets exceed their frozen-base success rates, while `place_fan` remains below its reference. Training budgets grow from 100k to 500k, 1M, and 3M learner updates; task-specific critic capacity also grows with source count. These results therefore describe a joint source-breadth/training-budget trajectory. Appendix B.2 gives the source sets and task-level SR/steps.

4.6 TWO-TASK CONTINUAL LEARNING

We test historical-memory selection by training on `pick_dual_bottles` for 300k learner updates (stage A), then on `hanging_mug` for an additional 500k (stage B). At the boundary, critical-state memory retains the top 10% of stage A's final replay by CIV; uniform memory selects 10% uniformly. Stage B allocates half its samples to the current task and half to this fixed memory, with no fresh old-task interaction. Both branches sample uniformly within each pool, isolating memory selection from CISA sampling. Each stage selects a checkpoint by its current task's success (Appendix B.5).

Table 5: Two-task continual learning (Clean E100). Rows report selected-checkpoint SR and mean steps, including failures. Branches differ only in historical-memory selection.

		Uniform memory		Critical-state memory	
Stage	Task	SR	Steps	SR	Steps
A	`pick_dual_bottles`	99.00%	103.70	100.00%	103.57
B	`pick_dual_bottles`	98.00%	104.03	100.00%	103.85
B	`hanging_mug`	66.00%	325.06	73.00%	323.35

Critical-state memory maintains 100.00% on the old task, while uniform memory changes from 99.00% to 98.00%. New-task success is 73.00% versus 66.00%, giving final two-task averages of 86.50% and 82.00% (+4.50 percentage points). These outcomes favor CIV-guided memory selection on this task transition.

5 LIMITATIONS AND OUTLOOK

Our evaluation covers one benchmark-tuned base policy in RoboTwin 2.0 Clean. Broader validation would require additional bases and tests under domain randomization, physical sensor delay, calibration error, and contact mismatch. One training run per arm leaves training-seed variability unmeasured; the 100 episodes per task characterize the selected checkpoint. Reusing checkpoint-selection evaluations introduces optimism into the corresponding success rates and macro averages (Appendices B.1 and B.5).

CIV-based allocation depends on critic accuracy and score freshness. Its base-action query can extrapolate from the executed-action distribution, and errors can affect both sampling and retention. Cross-task editing also requires sufficiently informative context: indistinguishable (o, q, b) inputs with conflicting required corrections remain ambiguous. Force/tactile inputs and richer goal context are natural extensions for such cases.

Average gains over the strong frozen base remain modest and uneven across tasks. A key limitation of the current editor is its compact CNN and lightweight expert pool, which learn visual features and diverse corrections from online experience without reusing pretrained representations. Our next step is to incorporate pretrained visual or VLA features, expand expert capacity, and improve routing, reducing the representation-learning burden on online RL and supporting a richer range of corrections.

All tasks share a fixed action scale of 5×10^{-4}, which may exclude corrections needed to recover from some errors. State- or dimension-dependent bounds offer a direction for balancing stronger interventions with preservation of competent base behavior, using autonomously collected online experience. In the discussion-stage revision, we will add targeted ablations and diagnostics to further evaluate shared correction and critical-data allocation.

REPRODUCIBILITY STATEMENT

Appendix A specifies the action coordinate, fixed action scale, CISA sampling, score-aware replay retention, learner runtime, and complete correction-support analysis. Appendix B records task splits, interaction and update budgets, checkpoint selection, matched controls, and budget accounting. Appendix C specifies the recorded action-value traces and fixed-state expert-routing probes. Reported quantities are derived from the corresponding experiment artifacts. Appendix B.1 discloses the loss of the S46 training records and checkpoints and the retained evaluation summaries.

AI USE STATEMENT

Generative AI tools assisted with hypothesis and experiment refinement, formal-analysis and proof drafts, literature search and synthesis, software and LaTeX editing, scientific-figure construction, manuscript organization, drafting, and readability edits. They also assisted with organizing experimental results, checking numerical consistency, and interpreting results. The authors reviewed the resulting claims, derivations, citations, code changes, figures, and compiled manuscript. All reported empirical values are computed from experiment artifacts rather than generated by AI. The authors take responsibility for the complete submission, including AI-assisted text and artifacts.

REFERENCES

Lars Ankile, Zhenyu Jiang, Rocky Duan, Guanya Shi, Pieter Abbeel, and Anusha Nagabandi. Residual off-policy rl for finetuning behavior cloning policies. *arXiv preprint arXiv:2509.19301*, 2025a. URL https://arxiv.org/abs/2509.19301.

Lars Ankile, Anthony Simeonov, Idan Shenfeld, Marcel Torne, and Pulkit Agrawal. From imitation to refinement: Residual RL for precise assembly. In *IEEE International Conference on Robotics and Automation*, 2025b.

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 *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 2025a.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, 2025b. doi: 10.15607/RSS.2025.XXI.010.

Edoardo Cetin, Philip J. Ball, Stephen Roberts, and Oya Celiktutan. Stabilizing off-policy deep reinforcement learning from pixels. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 2784–2810. PMLR, 2022.

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

Xinyue Chen, Che Wang, Zijian Zhou, and Keith Ross. Randomized ensembled double q-learning: Learning fast without a model. In *International Conference on Learning Representations*, 2021.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10–11):1684–1704, 2024.

Haoyuan Deng, Haichao Liu, Wenkai Guo, Yuan Ling, Zaijia Yang, Yuanjiang Xue, Haosheng Sun, Liangzi Wang, and Ziwei Wang. Facet-0: A robotic foundation model for contact-rich precise manipulation. *arXiv preprint arXiv:2609.01596*, 2026.

Lakshita Dodeja, Karl Schmeckpeper, Shivam Vats, Thomas Weng, Mingxi Jia, George Konidaris, and Stefanie Tellex. Accelerating residual reinforcement learning with uncertainty estimation. *IEEE Robotics and Automation Letters*, 11(1):970–977, 2026. doi: 10.1109/LRA.2025.3636808. URL https://arxiv.org/abs/2506.17564.

Perry Dong, Kuo-Han Hung, Tian Gao, Dorsa Sadigh, and Chelsea Finn. EXPO-FT: Sample-efficient reinforcement learning finetuning for vision-language-action models. *arXiv preprint arXiv:2605.25477*, 2026. URL https://arxiv.org/abs/2605.25477.

William Fedus, Barret Zoph, and Noam Shazeer. Switch Transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022. URL https://www.jmlr.org/papers/v23/21-0998.html.

Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Quan Vuong, Ted Xiao, Pannag R. Sanketi, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems*, 2024. doi: 10.15607/RSS.2024.XX.090.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1861–1870. PMLR, 2018a.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b. URL https://arxiv.org/abs/1812.05905.

Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Residual reinforcement learning for robot control. In *IEEE International Conference on Robotics and Automation*, pp. 6023–6029. IEEE, 2019.

Dmitry Kalashnikov, Jake Varley, Yevgen Chebotar, Benjamin Swanson, Rico Jonschkowski, Chelsea Finn, Sergey Levine, and Karol Hausman. Scaling up multi-task robotic reinforcement learning. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 557–575. PMLR, 2022.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P. Foster, Pannag R. Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025.

Samory Kpotufe. k-NN regression adapts to local intrinsic dimension. In *Advances in Neural Information Processing Systems*, volume 24, pp. 729–737, 2011.

Timothée Lesort, Vincenzo Lomonaco, Andrei Stoian, Davide Maltoni, David Filliat, and Natalia Díaz-Rodríguez. Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges. *Information Fusion*, 58:52–68, 2020. doi: 10.1016/j.inffus.2019.12.004. URL https://doi.org/10.1016/j.inffus.2019.12.004.

Ang A. Li, Zongqing Lu, and Chenglin Miao. Revisiting prioritized experience replay: A value perspective. *arXiv preprint arXiv:2102.03261*, 2021. doi: 10.48550/arXiv.2102.03261. URL https://arxiv.org/abs/2102.03261.

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*, 2026.

Yi Li, Yuquan Deng, Jesse Zhang, Joel Jang, Marius Memmel, Raymond Yu, Caelan Reed Garrett, Fabio Ramos, Dieter Fox, Anqi Li, Abhishek Gupta, and Ankit Goyal. HAMSTER: Hierarchical action models for open-world robot manipulation. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2502.05485.

Sixu Lin, Yunpeng Qing, Litao Liu, Ming Zhou, Ruixing Jin, Xiaoyi Fan, and Guiliang Liu. DyGRO-VLA: Cross-task scaling of vision-language-action models via dynamic grouped residual optimization. *arXiv preprint arXiv:2605.17486*, 2026. doi: 10.48550/arXiv.2605.17486. URL https://arxiv.org/abs/2605.17486.

Bo Liu, Yihao Feng, Peter Stone, and Qiang Liu. FAMO: Fast adaptive multitask optimization. In *Advances in Neural Information Processing Systems*, 2023a.

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

Huihan Liu, Changyeon Kim, Bo Liu, Minghuan Liu, and Yuke Zhu. Pretrained vision-language-action models are surprisingly resistant to forgetting in continual learning. *arXiv preprint arXiv:2603.03818*, 2026a. doi: 10.48550/arXiv.2603.03818. URL https://arxiv.org/abs/2603.03818.

Mingyu Liu, Zheng Huang, Xiaoyi Lin, Muzhi Zhu, Canyu Zhao, Yating Wang, Haoyi Zhu, Hao Chen, and Chunhua Shen. GAE: Unleashing physical potential of VLM with generalizable action expert. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026b. URL https://arxiv.org/abs/2510.03896.

Haodong Lu, Chongyang Zhao, Minhui Xue, Lina Yao, Kristen Moore, and Dong Gong. Little by little: Continual learning via incremental mixture of rank-1 associative memory experts. *arXiv preprint arXiv:2506.21035*, 2026. URL https://arxiv.org/abs/2506.21035. Accepted at ICML 2026.

Guozheng Ma, Lu Li, Haoyu Wang, Zixuan Liu, Pierre-Luc Bacon, and Dacheng Tao. What makes value learning efficient in residual reinforcement learning? *arXiv preprint arXiv:2602.10539*, 2026. doi: 10.48550/arXiv.2602.10539. URL https://arxiv.org/abs/2602.10539.

Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. In *Proceedings of the 22nd Annual Conference on Learning Theory*, 2009. URL https://www.learningtheory.org/colt2009/papers/003.pdf.

Muhammad A. Muttaqien, Tomohiro Motoda, Ryo Hanai, and Yukiyasu Domae. ORPA: Online residual policy adaptation for robot manipulation control with human feedback. *arXiv preprint arXiv:2608.17323*, 2026. doi: 10.48550/arXiv.2608.17323. URL https://arxiv.org/abs/2608.17323.

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.

Yangchen Pan, Jincheng Mei, Amir-massoud Farahmand, Martha White, Hengshuai Yao, Mohsen Rohani, and Jun Luo. Understanding and mitigating the limitations of prioritized experience replay. In *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pp. 1561–1571. PMLR, 2022.

Physical Intelligence, Bo Ai, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, et al. $\pi_{0.7}$: a steerable generalist robotic foundation model with emergent capabilities. *arXiv preprint arXiv:2604.15483*, 2026. doi: 10.48550/arXiv.2604.15483. URL https://arxiv.org/abs/2604.15483.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *International Conference on Learning Representations*, 2016. URL https://arxiv.org/abs/1511.05952.

Sahil Sharma, Ashutosh Jha, Parikshit Hegde, and Balaraman Ravindran. Learning to multi-task by active sampling. *arXiv preprint arXiv:1702.06053*, 2017. doi: 10.48550/arXiv.1702.06053. URL https://arxiv.org/abs/1702.06053.

Tom Silver, Kelsey Allen, Josh Tenenbaum, and Leslie Kaelbling. Residual policy learning. *arXiv preprint arXiv:1812.06298*, 2018.

Zhanyi Sun and Shuran Song. From prior to pro: Efficient skill mastery via distribution contractive rl finetuning. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*. PMLR, 2026. URL https://arxiv.org/abs/2603.10263.

Krishna Suresh and Christopher G. Atkeson. Learning dynamic rope manipulation using task-level iterative learning control. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, 2026. URL https://arxiv.org/abs/2602.21302.

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

Chenyang Wang, Yuntian Wang, Xiaoxiong Yang, Dingde Jiang, Siao Liu, and Yang Liu. Uncertainty-guided sparse refinement for action chunking transformer policies. *arXiv preprint arXiv:2609.15840*, 2026. URL https://arxiv.org/abs/2609.15840.

Wei Wu, Fan Lu, Yunnan Wang, Shuai Yang, Shi Liu, Fangjing Wang, Qian Zhu, He Sun, Yong Wang, Shuailei Ma, et al. A pragmatic VLA foundation model. *arXiv preprint arXiv:2601.18692*, 2026. URL https://arxiv.org/abs/2601.18692.

Wenli Xiao, Haotian Lin, Andy Peng, Haoru Xue, Tairan He, Yuqi Xie, Fengyuan Hu, Jimmy Wu, Zhengyi Luo, Linxi Fan, Guanya Shi, and Yuke Zhu. Self-improving vision-language-action models with data generation via residual RL. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2511.00091.

Charles Xu, Jost Tobias Springenberg, Michael Equi, Ali Amin, Adnan Esmail, Sergey Levine, and Liyiming Ke. RL token: Bootstrapping online RL with vision-language-action models. *arXiv preprint arXiv:2604.23073*, 2026. doi: 10.48550/arXiv.2604.23073. URL https://arxiv.org/abs/2604.23073.

Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021. URL https://arxiv.org/abs/2107.09645.

Chao Yu, Yuanqing Wang, Zhen Guo, Hao Lin, Si Xu, Hongzhi Zang, Quanlu Zhang, Yongji Wu, Chunyang Zhu, Junhao Hu, Zixiao Huang, Mingjie Wei, Yuqing Xie, Ke Yang, Bo Dai, Zhexuan Xu, Jiakun Du, Xiangyuan Wang, Xu Fu, Letong Shi, Zhihao Liu, Kang Chen, Weilin Liu, Gang Liu, Boxun Li, Jianlei Yang, Zhi Yang, Guohao Dai, and Yu Wang. RLinf: Flexible and efficient large-scale reinforcement learning via macro-to-micro flow transformation. In *20th USENIX Symposium on Operating Systems Design and Implementation (OSDI 26)*, pp. 829–846. USENIX Association, 2026. URL https://www.usenix.org/conference/osdi26/presentation/yu-chao.

Xiu Yuan, Tongzhou Mu, Stone Tao, Yunhao Fang, Mengke Zhang, and Hao Su. Policy decorator: Model-agnostic online refinement for large policy model. In *International Conference on Learning Representations*, 2025.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.016.

Weihui Zhao, Xiaohan Yan, Zunian Wan, Xuan Du, Zhaozhan Chi, Jianbo Mao, Ruipu Wu, Rushuai Yang, Houlin Li, Shukai Yang, Jing Wu, Yuxiang Yan, Yongcheng Liu, Chuankang Li, Guanghui Ren, Wei Shan, and Maoqing Yao. BEE: Intervention-adaptive real-world reinforcement learning with vision-language-action models. *arXiv preprint arXiv:2609.27450*, 2026. doi: 10.48550/arXiv.2609.27450. URL https://arxiv.org/abs/2609.27450.

A Method and Analysis Details

A.1 Action interface and chunk execution

We use the fixed action-normalization statistics supplied with the frozen base policy. For each action dimension j, ℓ_j and u_j are the 1st and 99th percentiles of the dataset actions. The affine map below sends these two values to -1 and 1, respectively, without clipping values outside this interval. These statistics define a numerical scale, not physical actuator limits. All operations are applied elementwise:[1]

$$\mathcal{A}(a) = 2\frac{a-\ell}{u-\ell} - 1, \qquad \mathcal{A}^{-1}(b) = \ell + \frac{b+1}{2}(u-\ell). \tag{4}$$

An action chunk is a sequence of H actions produced by one base-policy query. ReSOP caches this sequence. At each control step, GAE edits its current action in the affine coordinate, and $\mathcal{A}^{-1}$ converts the result back to the environment's action units. The base is queried again after all H actions have been used. The LingBot-VLA experiments use $H = 50$.

Each replay transition stores the base proposal and the executed action in the same affine coordinate, preserving the pair compared by CISA. Nonterminal transitions also retain the next-step base proposal and references that identify both action chunks; this supplies the successor information needed to compute critic training targets when transitions are replayed later. With zero edit, the inverse map recovers the original base action up to floating-point tolerance. The edit remains bounded, but the composed action $\bar{a}_t$ can lie outside $[-1, 1]$. Downstream controller saturation and workspace restrictions are properties of the environment interface.

A.2 Fixed action scale

We use $\lambda_m = 5 \times 10^{-4}$ for every source and held-out task, unchanged throughout training and evaluation. This common hyperparameter is fixed in advance, including for held-out deployment.

The actor learns the edit's direction and state-dependent normalized magnitude. With the bounded edit $e_t \in [-1, 1]^{d_a}$ in d_a action dimensions, each coordinate of the correction satisfies $|\bar{a}_{t,j} - b_{t,j}| \leq 5 \times 10^{-4}$ in the affine action coordinate. The corresponding physical-unit displacement depends on the inverse affine map in Eq. 4. The correction's Euclidean magnitude remains $\lambda_m \|r_\theta(o_t, q_t, b_t)\|_2$.

A.3 CISA sampling and score-aware replay retention

A shared, finite-capacity replay buffer stores transitions collected during the current run, indexed by task. CISA scores stored transitions using a read-only copy of the critic, ranks valid scores within each task, and samples the priority component of Eq. 3 as

$$P_{\text{crit}}(i \mid m) = \begin{cases} \dfrac{\sum_{j=a_i}^{b_i} (\varepsilon + j/N_m)^\kappa}{(b_i - a_i + 1)\sum_{j=1}^{N_m} (\varepsilon + j/N_m)^\kappa}, & i \in \mathcal{S}_m,\ \mathcal{S}_m \neq \varnothing, \\ U_m(i), & \mathcal{S}_m = \varnothing, \\ 0, & \text{otherwise}, \end{cases} \tag{5}$$

where $\mathcal{S}_m$ is the set of currently scored rows for task m. For $N_m = |\mathcal{S}_m| > 0$, order scores in ascending order. Rank j has normalized position $j/N_m \in \{1/N_m, \ldots, 1\}$. The indices a_i, b_i bound the contiguous equal-score block containing row i. The sampler draws a rank according to its mass and then draws uniformly inside that rank's tie block. Thus equal scores receive equal probability,

[1] The base configuration names this transform `bounds_99_woclip`; the stored percentile fields are `q01` and `q99`. The suffix `woclip` means "without clipping."

including uniform sampling over scored rows when all scores are zero. Each row receives the average rank mass of its tie block. The positive ε is a small offset in the rank weights. The uniform component covers every retained replay row; the priority component equals uniform coverage when $\mathcal{S}_m$ is empty. Generic defaults are $\rho = 0.9$, $\kappa = 6$, and $\varepsilon = 10^{-6}$; the S4 configuration overrides the first two to 0.99 and 30. The learner is configured to publish a new read-only model copy for scoring every eight updates. Individual scores are refreshed on demand, with their age also affected by transfer delays and queued work.

The same score separately controls how long transitions remain in replay. Once a transition record is complete, it enters a protected *Fresh* region. The *Critical* region holds transitions selected for longer retention. When a transition leaves Fresh, a bounded set of candidates and low-score Critical transitions from the same task are scored with the same critic copy. Within a task, a candidate replaces a Critical transition if its score is higher; equal scores favor the more recently finalized transition. Per-task capacity quotas prevent one task from occupying the entire buffer. A task below its quota may reclaim space from a task above its quota without comparing raw scores across tasks. Expired scores are refreshed so old critic estimates do not determine retention indefinitely.

The specified replay configuration uses 100,000 slots with 10,000 Fresh and 90,000 Critical slots. Scores expire after 10,000 learner updates. Each bounded replacement decision considers up to four currently low-score Critical rows and four additional Critical transitions from the same task, rescored to check the cached ranking.

Intervention value and policy uncertainty. CIV evaluates the benefit of a specific recorded edit. Policy entropy instead describes action dispersion; the editor's conditional entropy given (o, q, b) can be estimated from its log density. CIV scoring uses stored action pairs and incurs critic-inference and refresh costs. A consequential state receives zero CIV if the realized edit has a nonpositive estimated value contrast.

Sensitivity to critic error. The two-query structure makes score sensitivity explicit. For a fixed task and continuation policy, let $Q^{\mathrm{ref}}(x, a)$ be the reference conditional action value in the affine coordinate, and let $\widehat{Q}$ denote the scorer's ensemble minimum. For one stored pair, suppose $|\widehat{Q}(x, \bar{a}) - Q^{\mathrm{ref}}(x, \bar{a})| \le \delta_a$ and $|\widehat{Q}(x, b) - Q^{\mathrm{ref}}(x, b)| \le \delta_b$. Define $\widehat{c} = [\widehat{Q}(x, \bar{a}) - \widehat{Q}(x, b)]_+$ and $c^{\mathrm{ref}} = [Q^{\mathrm{ref}}(x, \bar{a}) - Q^{\mathrm{ref}}(x, b)]_+$. Since the positive-part map is 1-Lipschitz, the triangle inequality gives

$$|\widehat{c} - c^{\mathrm{ref}}| \le \big|[\widehat{Q}(x, \bar{a}) - \widehat{Q}(x, b)] - [Q^{\mathrm{ref}}(x, \bar{a}) - Q^{\mathrm{ref}}(x, b)]\big| \le \delta_a + \delta_b. \tag{6}$$

If all four queries for two records from the same task have error at most δ, a reference-score separation greater than 4δ preserves their ordering. This is a conditional sensitivity statement: ensemble minimization alone does not establish the required query-error bounds.

Rollout diagnostics. Appendix C.1 examines recorded signed contrasts along `hanging_mug` rollouts and specifies the diagnostic critic and example-selection procedure used in Figure 2.

A.4 Collection and learner task allocation

Horizon-weighted collection and balanced learning. Let $\mathcal{T}_{\mathrm{act}}$ contain the tasks active in collection and H_m^{env} be task m's configured maximum episode length in environment steps. The requested collection share and learner task share are

$$p_{\mathrm{collect}}(m) = \frac{H_m^{\mathrm{env}}}{\sum_{j \in \mathcal{T}_{\mathrm{act}}} H_j^{\mathrm{env}}}, \qquad p_{\mathrm{learn}}(m) = \frac{1}{|\mathcal{T}_{\mathrm{sample}}|}, \tag{7}$$

where $\mathcal{T}_{\mathrm{sample}}$ contains tasks with transitions available for replay sampling. Here H_m^{env} measures maximum episode length in environment steps. The learner forms balanced task batches and applies CISA within each task. Realized collection shares also depend on throughput and episode lengths. FAMO subsequently weights the per-task critic losses.

A.5 Online learner

The actor uses multi-view RGB observations, proprioception, and the current affine base proposal. Each source task has a separate critic ensemble, selected by the replay row's task identity. Its in-

puts are shared visual features, proprioception, and a candidate action, defining $Q_{\phi,m}(o, q, a)$ for the selected task m. The visual encoders are optimized with the critic loss, while actor updates use detached visual features, as in established visual-RL practice (Yarats et al., 2021). The learner uses SAC-style entropy-regularized actor updates with REDQ-style target estimation (Haarnoja et al., 2018a; Chen et al., 2021). ReSOP separates actor–critic learning, critic scoring, and replay storage; CISA selects batches from replay. Updates sent to collectors contain the residual actor, visual encoders, and learned camera-augmentation parameters needed for inference. Only the learner updates critics, target networks, and optimizer states. A scorer receives a read-only model copy, called a *scoring view*, containing the visual encoder, task-specific critic ensembles, camera-augmentation parameters, and observation-normalization statistics. These copies let scoring proceed without modifying the learner. Communication among collectors, replay and the learner uses RLinf (Yu et al., 2026); ReSOP adds the scoring and data-allocation rules described above.

Context-conditioned sparse actor. Let $u = [v(o), q, b] \in \mathbb{R}^{284}$ concatenate the shared actor-side visual compression (256 dimensions), proprioception (14), and base proposal (14). The compression is shared by the experts. A linear map with bias computes routing scores from $\mathrm{LN}(u)$, where LN is layer normalization without learned scale or bias. Each expert instead receives u without this additional layer normalization. For E experts, define

$$g = W\,\mathrm{LN}(u) + d + \xi, \qquad S(u) = \mathrm{Top2}(g), \qquad w_j = \frac{\exp(g_j)}{\sum_{k\in S(u)} \exp(g_k)}, \quad j \in S(u). \tag{8}$$

Here W, d are learned router parameters, ξ is training-time routing noise, and Top2 returns the indices of the two largest scores. The weights w_j sum to one over those two experts. Each expert contains a three-layer, 512-wide actor trunk and separate mean and log-standard-deviation heads. Only selected experts are executed for an input. The policy combines their Gaussian parameters:

$$\mu(u) = \sum_{j\in S(u)} w_j \mu_j(u), \qquad \ell(u) = \sum_{j\in S(u)} w_j \ell_j(u), \qquad \sigma(u) = \exp\big(\mathrm{clamp}(\ell(u), \ell_{\min}, \ell_{\max})\big). \tag{9}$$

The clamp operation bounds each log standard deviation between $\ell_{\min}$ and $\ell_{\max}$. The combined parameters define one diagonal Gaussian. In the normalized edit coordinates of Eq. (1), stochastic training uses $e = \tanh(\mu + \sigma \odot \epsilon)$ with $\epsilon \sim \mathcal{N}(0, I)$; mean-action execution uses $e = \tanh(\mu)$. The external gain is applied once.

Router weights and biases start at zero. Experts copy the baseline actor, preserving output-head initialization; hidden weights receive independent perturbations with standard deviation 10^{-3} times each layer's weight standard deviation. During stochastic training, router noise is isotropic Gaussian with standard deviation $0.05 \max(0, 1 - n_a/100000)$, where n_a counts successful actor optimizer updates. Routing noise is disabled for deterministic inference.

The actor adds an expert-load balancing term, computed over the combined batch from all learner workers, to its task-averaged SAC objective, adapting the probability–load product of Fedus et al. (2022) to top-2 routing. With full-router probabilities $p_{ij} = \mathrm{softmax}(g_i)_j$ and batch size B, let $\bar{p}_j$ be expert j's mean routing probability and $\bar{f}_j$ its fraction of the $2B$ selected expert assignments. The stop-gradient operator sg leaves a value unchanged but blocks its gradient. The balancing term is

$$\bar{p}_j = \frac{1}{B}\sum_i p_{ij}, \qquad \bar{f}_j = \frac{1}{2B}\sum_i \mathbf{1}\{j \in S(u_i)\},$$
$$L_{\mathrm{bal}} = 10^{-3}\left(E \sum_j \bar{p}_j\, \mathrm{sg}(\bar{f}_j) - 1\right). \tag{10}$$

Statistics are aggregated across learner workers, balancing expert use over the global batch. The constant offset leaves gradients unchanged.

Arm-wise entropy tuning. We apply automatic temperature tuning (Haarnoja et al., 2018b) separately to the left- and right-arm action groups, each containing seven coordinates. Let $z = (o, q, b)$ and let $\ell_g = \log \pi_{\theta,g}(e_g \mid z)$ sum the tanh-corrected log densities over group $g \in \{L, R\}$. With

$Q^{\text{avg}}_{\phi,m} = K^{-1}\sum_{h=1}^{K} Q_{\phi,m,h}$, the actor loss is

$$L_\theta = \mathbb{E}_{\text{task-avg},\, e\sim\pi_\theta}\left[\alpha_L \ell_L + \alpha_R \ell_R - Q^{\text{avg}}_{\phi,m}(x, b + \lambda_m e)\right] + L_{\text{bal}}. \tag{11}$$

The expectation averages replay-state losses within each active task, then equally across active tasks. Both temperatures are held fixed during the actor step. Writing $\alpha_g = \exp(\nu_g)$, each has its own Adam optimizer minimizing the log-temperature surrogate

$$L_{\nu_g} = -\nu_g \,\text{sg}\left(\mathbb{E}_{\text{online}}[\ell_g] + \mathcal{H}^*_g\right). \tag{12}$$

Here the expectation is a row-wise mean over sampled online replay states, aggregated across learner workers, and $\mathcal{H}^*_g$ is the arm's target entropy. The pair (α_L, α_R) is shared across tasks and experts. Separate temperature losses prevent opposing arm-wise entropy deviations from canceling in a single aggregate adaptation signal. The entropy is defined in normalized edit coordinates, before applying λ_m.

Task-specific ensembles and loss balancing. The critic ensembles use $K = 6$ heads per task. CIV takes the minimum over all online heads of the selected task; Bellman targets instead use a random two-head minimum from its target ensemble and the target visual encoder. Actor updates use the online ensemble mean in Eq. (11). For bootstrapping, the current stochastic actor samples an edit at the next context and composes it with the stored next base proposal. The configured TD backup omits the entropy bonus: its bootstrap term is the discounted target-critic value. After each critic update, both the target critic and target visual encoder receive an exponential moving average of their online counterparts, $\psi^- \leftarrow (1-\tau)\psi^- + \tau\psi$. For task batch $\mathcal{B}_m$ and TD target y_i, the configured critic loss is

$$L_m = \frac{1}{|\mathcal{B}_m|}\sum_{i\in\mathcal{B}_m}\left(\frac{1}{K}\sum_{h=1}^{K}|Q_{\phi,m,h}(x_i, \bar{a}_i) - y_i|\right)^2. \tag{13}$$

The loss squares the mean absolute head error. Fast Adaptive Multitask Optimization (FAMO) (Liu et al., 2023a) adapts task weights according to their loss progress. With learned task weighting logits v and a positive numerical offset ϵ_F, it combines task losses through

$$\begin{aligned} z &= \text{softmax}(v), \qquad D_m = L_m + \epsilon_F, \\ C &= \text{sg}\left(\sum_m z_m / D_m\right), \qquad L_F = C^{-1}\sum_m \text{sg}(z_m)\log D_m, \end{aligned} \tag{14}$$

over active tasks. Here z gives normalized task weights and C is a normalizing factor held constant during differentiation. The effective loss-gradient weights are proportional to z_m/D_m. The logits are updated from the change in log loss before and after an optimizer step, using the same batch, training targets, and augmentation randomness. The configured FAMO learning rate is 0.025, regularization coefficient 0.001, and $\epsilon_F = 10^{-8}$. FAMO weights the critic and encoder gradients; the actor uses the task-averaged objective in Eq. (11).

S46 implementation configuration. The specified 46-source training configuration uses eight experts with top-2 routing, a four-layer CNN with 32 channels per layer, A-LIX and per-camera adaptive 3×3 average pooling, without GroupNorm. Plain critic weights and biases retain their default uniform initialization. It uses the deterministic actor output $\tanh(\mu)$ during collection, no offline replay, no additional network residual shortcuts, and eight critic updates per actor update. Actor and temperature updates occur together. The runtime configuration initializes both temperatures to 0.01, uses separate Adam optimizers at 3×10^{-4}, and sets $\mathcal{H}^*_L = \mathcal{H}^*_R = -3.5$. Actor and critic/encoder optimizers use AdamW with learning rate 3×10^{-4} and weight decay 10^{-4}; target averaging uses $\tau = 0.005$. The actor's log standard deviation starts at -10 and is bounded to $[-20, 2]$. Gaussian action samples and router noise are used in stochastic learner calls; collection uses deterministic routing and actions. Table 7 summarizes the controls and interpretation of the reported S4 comparisons.

One logical, task-sharded learner. Each learner worker (distributed rank) holds a disjoint set of complete task-specific critic ensembles, including targets and optimizer states. Actor and encoder

replicas are synchronized across workers. Rank zero selects a global replay batch and sends each task's transitions to the worker holding its critics. Shared gradients and task-loss statistics are combined for one logical update. A complete CPU scoring copy combines the critics from all workers and the shared modules from the same update. The reported S46 training budget is 3M learner updates. This logical-update count is separate from the number of learner workers or collection environments.

Scoring requests and model updates. Candidate availability and score-refresh demand trigger scoring work, rather than requiring one scoring pass per optimizer update. Completed optimizer updates independently trigger publication of a newer scoring view. The remote option transfers a fixed model copy, while requests already being processed can finish with an older copy. Each result identifies its transition, the critic version, and a counter that changes when its storage slot is reused. Replay rejects scores for overwritten transitions or model versions older than its allowed threshold before updating its score table. Accepted online scores feed both the task-local priority distribution used by CISA and Critical retention. This separation targets score-supply pressure as collection grows without changing the CIV definition or the two allocation rules.

Score age and execution options. Stored scores carry learner-update indices. Model-version lag compares the critic copy's update index with the learner's when a score is accepted; copying, transfer, activation, and queued work contribute to this lag. Version checks and refresh/expiry rules limit stale-score reuse.

The default scoring path runs in the learner process. An optional remote path transfers complete read-only scoring views to separate workers. Collector publication transfers actor inference state, while scorer publication transfers the encoder and task-specific critic ensembles with their input-processing state.

A.6 Proposal-conditioned correction-support analysis

We analyze three ingredients of correction transfer: information in the proposal-conditioned context, nearby source evidence, and compatibility of the required edits. A finite-sample bound for a local estimator separates these ingredients from the approximation and learning gap of the implemented editor. The construction draws on local nonparametric estimation and domain-adaptation analyses (Kpotufe, 2011; Mansour et al., 2009) and formalizes the transfer perspective in Section 1.

Ideal local edit. Let s be the latent execution state, $X = (O, Q)$ the observable visual and proprioceptive state, B the affine base proposal, and $Z = \phi(X, B) \in (\mathcal{Z}, d)$ the analysis context. The representation ϕ, metric d, and task/context distribution are fixed throughout this analysis. We consider edit maps measurable in Z; taking ϕ to be the identity includes editors using the full input tuple (X, B). For each task τ, fix a continuation policy π_τ^{ref} and let Q_τ^{ref} be its environment action value, rather than a learned critic estimate. With the common scale $\lambda = 5 \times 10^{-4}$, define

$$q_\tau(z, e) = \mathbb{E}\big[Q_\tau^{\mathrm{ref}}\big(s, \mathcal{A}^{-1}(B + \lambda e)\big) \mid Z = z, T = \tau\big]. \tag{15}$$

Assume q_τ is measurable and continuous in e on the compact edit set $\mathcal{E} = [-1, 1]^{d_a}$. Its maximizer set is nonempty and compact. Choose a minimum-norm maximizer,

$$e_\tau^\star(z) = \operatorname*{arg\,min}_{e \in \arg\max_{u \in \mathcal{E}} q_\tau(z,u)} \|e\|_2, \tag{16}$$

assuming the final selection admits a fixed measurable tie break. Across the task mixture, let $E^\star = e_T^\star(\phi(X, B))$.

Proposal information. For squared correction error, let $\mathcal{R}_X^\star$ and $\mathcal{R}_{X,B}^\star$ denote the Bayes risks over measurable square-integrable predictors observing X and (X, B), respectively.

Proposition 1 (Proposal information)**.** *For square-integrable $E^\star$,*

$$\mathcal{R}_X^\star - \mathcal{R}_{X,B}^\star = \mathbb{E}\Big[\|\mathbb{E}[E^\star \mid X, B] - \mathbb{E}[E^\star \mid X]\|_2^2\Big] \geq 0. \tag{17}$$

The inequality is strict exactly when the proposal changes the conditional mean edit on a set of positive probability.

Proof of Proposition 1. The square-loss Bayes predictors are $m_X = \mathbb{E}[E^\star \mid X]$ and $m_{X,B} = \mathbb{E}[E^\star \mid X, B]$. Conditional-expectation orthogonality gives

$$\mathbb{E}\|E^\star - m_X\|_2^2 = \mathbb{E}\|E^\star - m_{X,B}\|_2^2 + \mathbb{E}\|m_{X,B} - m_X\|_2^2, \tag{18}$$

which proves Eq. 17. Equality holds precisely when $m_{X,B} = m_X$ almost surely. If B is already a deterministic function of X, the information gain is zero. The identity concerns population information; finite-network learnability is a separate question.

Compatibility at a fixed context. For the same task mixture, write $\mu_{\mathrm{mix}}(Z) = \mathbb{E}[E^\star \mid Z]$. Conditional-expectation orthogonality also gives, for any square-integrable edit map g,

$$\mathbb{E}\|g(Z) - E^\star\|_2^2 = \mathbb{E}\|g(Z) - \mu_{\mathrm{mix}}(Z)\|_2^2 + \mathbb{E}\|E^\star - \mu_{\mathrm{mix}}(Z)\|_2^2. \tag{19}$$

The cross term vanishes after conditioning on Z. The first term concerns fitting a shared correction map; the second measures ambiguity among the selected ideal edits after observing Z. Context-conditioned experts can represent distinct correction regimes when their inputs distinguish them. Additional experts alone do not resolve conflicting edits at identical inputs. This conditional variance is a lower bound on mixed-task squared edit error, distinct from the source–target mismatch below.

Source field and target mismatch. Let P_τ^Z be the distribution induced by a fixed critical-context selection rule on task τ. The result is conditional on this analysis distribution. For source set S, use fixed positive task weights summing to one, and let P_S^Z be the corresponding mixture, $\mathcal{M}_S = \mathrm{supp}(P_S^Z)$, and

$$\mu_S(z) = \mathbb{E}[e_T^\star(Z) \mid Z = z, T \in S], \qquad \mu_t(z) = e_t^\star(z). \tag{20}$$

The conditional expectation uses the same source task weights. The conditional mean μ_S is defined only P_S^Z-almost everywhere. Assume there exists a bounded Hölder extension μ on $\mathcal{M}_S \cup \mathrm{supp}(P_t^Z)$ that agrees with μ_S P_S^Z-almost everywhere. Fix one such extension throughout the analysis; the extension specifies its values outside source support. Assume a measurable nearest-support map $\Pi_S(z) \in \arg\min_{u \in \mathcal{M}_S} d(z,u)$ on target support. The chosen extension is bounded by M and is α-Hölder on $\mathcal{M}_S \cup \mathrm{supp}(P_t^Z)$:

$$\|\mu(z) - \mu(z')\|_2 \le L d(z,z')^\alpha, \qquad \|\mu(z)\|_2 \le M, \qquad \alpha \in (0,1]. \tag{21}$$

For target task t, define mismatch relative to this fixed extension by

$$\eta_t = \left(\mathbb{E}_{Z \sim P_t^Z} \|\mu_t(Z) - \mu(Z)\|_2^2\right)^{1/2}. \tag{22}$$

and define the target support gap

$$\rho_\alpha(t \mid S) = \left[\mathbb{E}_{Z \sim P_t^Z} d(Z, \mathcal{M}_S)^{2\alpha}\right]^{1/2}. \tag{23}$$

The gap ρ_α measures distance to source support, whereas η_t measures disagreement with the chosen extension. Assume ρ_α is finite. The target mismatch is an upper-bound term, not an irreducible risk lower bound.

Local evidence and sampling assumptions. Let $\mathbb{B}(u,h) = \{z : d(z,u) \le h\}$ be the radius-h neighborhood of u. For some $h > 0$ and $\beta_S \in (0,1]$, assume

$$P_S^Z\big(\mathbb{B}(\Pi_S(z), h)\big) \ge \beta_S, \qquad z \in \mathrm{supp}(P_t^Z). \tag{24}$$

Thus β_S lower-bounds source probability mass near each target's nearest point on source support. The finite-source analysis uses n independent pairs $\mathcal{D}_S = \{(Z_i, Y_i)\}_{i=1}^n$, where $Z_i \sim P_S^Z$ and $Y_i = \mu(Z_i) + \xi_i$. Conditional on all contexts, the noises are independent, have zero mean, and have per-coordinate variance at most σ^2. This is a local-regression surrogate for correction learning from online RL. For $u \in \mathcal{M}_S$, let

$$N_h(u) = \sum_{i=1}^n \mathbf{1}\{d(Z_i,u) \le h\}, \qquad \widehat{\mu}_h(u) = \begin{cases} \dfrac{1}{N_h(u)} \displaystyle\sum_{i : d(Z_i,u) \le h} Y_i, & N_h(u) > 0, \\ 0, & N_h(u) = 0. \end{cases} \tag{25}$$

Define the projected local estimator $\widehat{g}_S(z) = \widehat{\mu}_h(\Pi_S(z))$. For any square-integrable edit map g, define its target correction risk by

$$\mathcal{R}_t(g) = \mathbb{E}_{Z \sim P_t^Z} \|g(Z) - \mu_t(Z)\|_2^2. \tag{26}$$

Theorem 2 (Local correction estimation across supports)**.** *Suppose n independent source correction samples have per-coordinate noise variance at most σ^2, the fixed bounded extension μ is α-Hölder with constant L, and every radius-h neighborhood around projected target support has source mass at least β_S. Under the construction above, the projected local estimator satisfies*

$$\Big(\mathbb{E}_{\mathcal{D}_S}\mathcal{R}_t(\widehat{g}_S)\Big)^{1/2} \le \varepsilon_n(h,\beta_S) + L\rho_\alpha(t \mid S) + \eta_t, \tag{27}$$

where, for edit dimension d_a and field bound M,

$$\varepsilon_n(h,\beta_S) = Lh^\alpha + \sigma\sqrt{\frac{2d_a}{(n+1)\beta_S}} + Me^{-n\beta_S/2}. \tag{28}$$

Proof of Theorem 2. Fix any $z \in \mathrm{supp}(P_t^Z)$, set $u = \Pi_S(z)$, and write $N = N_h(u)$. Its success probability satisfies $p_u = P_S^Z(\mathbb{B}(u,h)) \ge \beta_S$, so $N \sim \mathrm{Binomial}(n, p_u)$. On $N > 0$, Hölder continuity bounds the conditional bias of the local average by Lh^α. Conditional independence bounds its noise second moment by $d_a\sigma^2/N$. Moreover,

$$\mathbb{E}\left[\frac{\mathbf{1}\{N>0\}}{N}\right] \le 2\mathbb{E}\left[\frac{1}{N+1}\right] \le \frac{2}{(n+1)\beta_S}, \qquad \Pr(N=0) \le e^{-n\beta_S}. \tag{29}$$

Minkowski's inequality, including the empty-neighborhood branch, therefore gives uniformly over projected target contexts

$$\big(\mathbb{E}_{\mathcal{D}_S}\|\widehat{\mu}_h(u) - \mu(u)\|_2^2\big)^{1/2} \le Lh^\alpha + \sigma\sqrt{\frac{2d_a}{(n+1)\beta_S}} + Me^{-n\beta_S/2} = \varepsilon_n(h,\beta_S). \tag{30}$$

For an independent target context, insert $\mu(\Pi_S(z))$ and $\mu(z)$ between $\widehat{g}_S(z)$ and $\mu_t(z)$. Taking the joint $L_2(\mathcal{D}_S \times P_t^Z)$ norm and applying Minkowski's inequality bounds the three resulting terms by $\varepsilon_n(h,\beta_S)$, $L\rho_\alpha(t \mid S)$, and η_t. This proves Eq. 27.

Connection to the implemented editor. Let $\mathcal{F}$ be a class of deterministic edit maps from $\mathcal{Z}$ to $\mathcal{E}$, and let $g_\theta \in \mathcal{F}$ denote the trained editor's deterministic evaluation map. Write Ω for training randomness. For each source sample, assume a measurably selected $L_2(P_t^Z)$ best-in-class approximation $g_{\mathcal{F}}^\star$ to $\widehat{g}_S$ exists. Define

$$\begin{aligned}\epsilon_{\mathrm{app}} &= \Big(\mathbb{E}_{\mathcal{D}_S, Z\sim P_t^Z}\|g_{\mathcal{F}}^\star(Z) - \widehat{g}_S(Z)\|_2^2\Big)^{1/2},\\ \epsilon_{\mathrm{bridge}} &= \Big(\mathbb{E}_{\mathcal{D}_S,\Omega, Z\sim P_t^Z}\|g_\theta(Z) - g_{\mathcal{F}}^\star(Z)\|_2^2\Big)^{1/2}.\end{aligned} \tag{31}$$

The target draw is independent of the source sample and training randomness. Two applications of the triangle inequality give

$$\Big(\mathbb{E}_{\mathcal{D}_S,\Omega}\mathcal{R}_t(g_\theta)\Big)^{1/2} \le \varepsilon_n(h,\beta_S) + L\rho_\alpha(t \mid S) + \eta_t + \epsilon_{\mathrm{app}} + \epsilon_{\mathrm{bridge}}. \tag{32}$$

Here ϵ_{app} measures the chosen actor class's approximation gap, while $\epsilon_{\mathrm{bridge}}$ includes optimization and source-to-target generalization. These are target-distribution bridge quantities, not controlled by Theorem 2. They locate where representation, expert capacity, and learning affect the implemented editor beyond the local evidence bound.

Corollary 3 (Local value consequence)**.** *Suppose $|q_t(z,e) - q_t(z,e')| \le G_t\lambda\|e - e'\|_2$ for all $e, e' \in \mathcal{E}$ and target contexts z. Then the expected local value regret of g_θ relative to the selected ideal edit is at most $G_t\lambda\big(\mathbb{E}_{\mathcal{D}_S,\Omega}\mathcal{R}_t(g_\theta)\big)^{1/2}$.*

Proof of Corollary 3. Because $\mu_t(z) = e_t^\star(z)$ maximizes the conditional local value and $g_\theta(z) \in \mathcal{E}$, the assumed Lipschitz condition gives

$$0 \le q_t(Z,\mu_t(Z)) - q_t(Z,g_\theta(Z)) \le G_t\lambda\|\mu_t(Z) - g_\theta(Z)\|_2. \tag{33}$$

Taking expectation over $\mathcal{D}_S, \Omega, Z$ and applying Cauchy–Schwarz gives the stated result. Different editors can induce different future state distributions, so this local inequality does not imply a full-return or terminal-success bound.

Reading the bound. The finite-source term separates local resolution Lh^{α}, sampling noise of order $(n\beta_S)^{-1/2}$, and the risk of an empty neighborhood. For fixed h, the resolution term remains in the upper bound as n grows; it is not a proved lower bound on estimation error. In a shrinking-bandwidth regime, $h_n \to 0$ and $n\beta_S(h_n) \to \infty$ make the finite-source term vanish for fixed L, σ, M, d_a.

Coverage and local mass play different roles. Adding sources can reduce ρ_α while diluting relevant neighborhoods under a fixed sample budget; changing the task mixture can also change μ_S and η_t. The useful regime combines nearby contexts, sufficient local evidence $n\beta_S$, and compatible edits. These conditions motivate the source-breadth study in Section 4.5.

Connection to data allocation. CISA reallocates optimization exposure within retained experience, while score-aware retention changes which evidence remains available in finite replay. The former can affect the learned-actor bridge; the latter can change local coverage and mass. Their effect depends on which contexts are emphasized or preserved. In this analysis n counts independent source examples, so replaying a transition more often does not add independent evidence. The score sensitivity in Appendix A.3 separately describes how critic errors can perturb the allocation signal.

B Experimental Details

This appendix specifies the protocols and comparison controls for Section 4. All experiments use Clean, including the 46-source main study, its four held-out targets, and every source-breadth point in the scaling study.

B.1 Shared-training split and evaluation

The frozen base is a LingBot-VLA checkpoint fine-tuned for the 50-task RoboTwin 2.0 suite. GAE training uses 46 source tasks. The four GAE-held-out targets are `blocks_ranking_rgb`, `open_laptop`, `place_a2b_right`, and `place_fan`; none contributes an editor training transition or task-allocation share during source training.

Collection is horizon-weighted and learner tasks are balanced (Appendix A.4). Training uses online interaction with the frozen base and direct affine action composition without numerical clipping at the GAE coordinate boundary. Within each S4 ablation, the allocation configuration and learner-update budget are fixed while the declared ablation factor changes. Realized task exposure can differ across independent asynchronous runs.

Table 6: Allocation and replay controls for the S4 comparisons. Both studies use a 500k-learner-update training budget.

Study	Fixed controls	Changed factor
Sampling	Horizon-weighted collection; balanced learner tasks; CIV-based retention	Uniform, absolute TD-error priority, or CIV priority within each task.
Visual package	Same collection and learner-task allocation; CISA sampling; CIV-based retention	ResNet-10 versus CNN+A-LIX.

Table 7: Architecture controls for the S4 ablations.

Comparison	Architecture controls	Compared factor
Sampling	Same shared actor, visual encoder, task-specific critics, and critic-side FAMO	Within-task sampling distribution under fixed retention.
Visual package	Same actor, task-specific critics, FAMO, and sampling	ResNet-10 versus CNN+A-LIX as complete visual packages.

The CNN+A-LIX/CISA arm is the common reference for both studies.

The checkpoint selected on the source anchor is fixed before held-out evaluation. Every task uses 100 evaluation episodes. The base-only control uses the same frozen checkpoint and Clean setting. For split S, macro success is

$$\mathrm{SR}(S) = \frac{1}{|S|} \sum_{m \in S} \frac{n_m^{\mathrm{succ}}}{n_m^{\mathrm{eval}}}. \tag{34}$$

Task-level improve, tie, and degrade counts compare integer success counts under the common 100-episode denominator. The static paired scatter labels the four held-out targets and the largest source-task gains; the result artifact retains all 50 task names and counts.

Source-anchor checkpoint selection. For each S4 arm (Uniform, TD-error priority, CISA, and ResNet-10), training runs for 500,000 learner updates. We evaluate `hanging_mug`, a source task, over 100 episodes at each candidate checkpoint and select the one with the highest anchor success rate. Ties favor the later checkpoint:

$$\mathcal{C}_4 = \{300{,}000 + 25{,}000j : j = 0, \ldots, 8\}, \qquad u^\star = \max\left(\arg\max_{u \in \mathcal{C}_4} \widehat{s}_{\mathrm{mug},100}(u) \right).$$

The same selected checkpoint is evaluated on all tasks. The 500k training budget includes all updates through the end of the candidate grid. The nine candidate evaluations comprise 900 anchor episodes per arm, separate from online training interaction. The selected anchor's 100-episode measurement is reused in the final table. This selected maximum can introduce optimism into the anchor success and the macro averages containing it.

The same source-anchor rule, including the later-checkpoint tie break, is the default for S10 and S46. Their candidate update ranges follow their respective training budgets rather than inheriting S4's numerical grid. The four GAE-held-out tasks are excluded from checkpoint selection.

S46 record availability. Due to an accidental data-loss incident, the training-process records and saved checkpoints for the S46 run were lost. The reported source-task and held-out results are based on retained task-level evaluation summaries; the lost checkpoints are unavailable for re-evaluation. We will supplement this appendix with further S46 experimental details and supporting evidence in a subsequent revision.

Table 8 reports all 50 task outcomes for the S46-trained editor. Mean episode length across all tasks is 204.14 for the base and 204.03 for GAE. Source-task means are 202.59 and 202.57, respectively; H4 means are 222.01 and 220.83. These averages include failures.

B.2 Nested source-breadth evaluation

Table 9 defines the nested sequence $S_1 \subset S_4 \subset S_{10} \subset S_{46}$. The sets specify training membership. All four GAE-held-out targets remain outside every source set. The specified S10 and S46 MoE configurations use different expert counts, and task-specific critic capacity increases with source count. This study therefore follows joint data/model scaling across source breadths. Total training budgets are 100k, 500k, 1M, and 3M learner updates for S1, S4, S10, and S46. Environment-transition counts are unreported. These are run-level budgets, distinct from the selected checkpoint indices. S4, S10, and S46 use the source-anchor selection protocol.

Each source-trained checkpoint is evaluated for 100 episodes on each of the four held-out targets, with no target-side editor-parameter updates. Table 10 gives every success rate and mean episode length; Table 4 summarizes macro success and training budgets. The reported S1, S4, S10 and S46 outcomes correspond to 341, 346, 347 and 355 successes out of 400 episodes, respectively.

Source-task performance for S_{10}. Table 11 reports the ten source tasks alongside the frozen-base reference. The task-level successes sum to 913/1000 for the base and 923/1000 with GAE. Mean episode length is 148.89 for the base and 148.01 with GAE, averaged across all 100 evaluation episodes per task, including failures.

B.3 Matched ablation protocols

The sampling and visual-package comparisons use S_4 from Table 9 in Clean. The sampling study compares uniform within-task sampling, CISA, and a TD-error-priority comparator. The TD-error

Table 8: Task-level Clean evaluation of the S46-trained editor. Success columns give successes/evaluation episodes; mean steps include failures. S denotes a source task and H a GAE-held-out target.

		Success		Mean episode steps	
Task	Split	Base	+GAE	Base	+GAE
adjust_bottle	S	97/100	98/100	118.62	115.19
beat_block_hammer	S	89/100	91/100	122.26	116.45
blocks_ranking_rgb	H	87/100	89/100	466.27	461.89
blocks_ranking_size	S	54/100	64/100	466.74	459.39
click_alarmclock	S	84/100	84/100	84.09	106.04
click_bell	S	75/100	96/100	70.15	48.32
dump_bin_bigbin	S	95/100	96/100	170.85	179.81
grab_roller	S	99/100	100/100	85.89	82.78
handover_block	S	97/100	94/100	270.03	273.16
handover_mic	S	100/100	99/100	205.93	208.14
hanging_mug	S	68/100	68/100	323.39	329.46
lift_pot	S	99/100	99/100	101.78	101.27
move_can_pot	S	89/100	87/100	147.53	147.29
move_pillbottle_pad	S	100/100	100/100	137.64	137.44
move_playingcard_away	S	100/100	100/100	107.36	106.39
move_stapler_pad	S	69/100	72/100	184.89	184.82
open_laptop	H	93/100	95/100	134.38	133.24
open_microwave	S	58/100	59/100	639.66	664.09
pick_diverse_bottles	S	94/100	94/100	106.81	105.75
pick_dual_bottles	S	98/100	98/100	104.52	104.25
place_a2b_left	S	87/100	85/100	145.92	146.41
place_a2b_right	H	85/100	86/100	144.49	143.71
place_bread_basket	S	93/100	89/100	218.49	211.91
place_bread_skillet	S	90/100	93/100	139.34	141.25
place_burger_fries	S	98/100	96/100	216.62	214.76
place_can_basket	S	86/100	86/100	249.35	246.06
place_cans_plasticbox	S	100/100	100/100	226.99	227.04
place_container_plate	S	98/100	99/100	133.98	132.80
place_dual_shoes	S	81/100	81/100	251.99	252.33
place_empty_cup	S	100/100	100/100	144.58	145.38
place_fan	H	89/100	85/100	142.91	144.48
place_mouse_pad	S	90/100	88/100	150.25	154.43
place_object_basket	S	93/100	93/100	243.89	244.17
place_object_scale	S	92/100	93/100	144.16	137.05
place_object_stand	S	94/100	97/100	131.75	128.46
place_phone_stand	S	99/100	96/100	115.24	117.47
place_shoe	S	99/100	99/100	148.46	148.45
press_stapler	S	88/100	90/100	86.19	89.03
put_bottles_dustbin	S	95/100	95/100	611.50	602.94
put_object_cabinet	S	96/100	94/100	258.80	261.59
rotate_qrcode	S	90/100	89/100	162.02	163.39
scan_object	S	92/100	94/100	176.92	167.75
shake_bottle	S	99/100	99/100	97.93	97.39
shake_bottle_horizontally	S	100/100	100/100	98.27	96.88
stack_blocks_three	S	95/100	91/100	451.16	452.74
stack_blocks_two	S	98/100	99/100	291.96	291.31
stack_bowls_three	S	72/100	85/100	460.64	452.50
stack_bowls_two	S	95/100	96/100	287.44	286.50
stamp_seal	S	84/100	83/100	140.48	143.85
turn_switch	S	77/100	78/100	86.71	94.15

Table 9: Nested source-task memberships. Each row adds tasks to the preceding set. The S_4 sampling ablation uses the same four source tasks.

Set	Membership / additions
S_1	`hanging_mug`
S_4	S_1 + `turn_switch`, `move_playingcard_away`, `pick_diverse_bottles`
S_{10}	S_4 + `adjust_bottle`, `grab_roller`, `beat_block_hammer`, `lift_pot`, `place_object_scale`, `stack_blocks_two`
S_{46}	All benchmark tasks except `blocks_ranking_rgb`, `open_laptop`, `place_a2b_right`, and `place_fan`

Table 10: Per-task GAE-held-out evaluation in Clean. The two panels report success over 100 episodes and mean episode steps over all episodes, including failures. The frozen base is the common reference.

Task	Frozen base	S_1	S_4	S_{10}	S_{46}
Success rate					
`blocks_ranking_rgb`	87%	81%	82%	84%	89%
`place_a2b_right`	85%	83%	82%	84%	86%
`place_fan`	89%	84%	89%	86%	85%
`open_laptop`	93%	93%	93%	93%	95%
Macro average	88.50%	85.25%	86.50%	86.75%	88.75%
Mean episode steps (including failures)					
`blocks_ranking_rgb`	466.27	469.06	468.69	468.78	461.89
`place_a2b_right`	144.49	145.52	145.24	144.70	143.71
`place_fan`	142.91	147.35	141.57	146.30	144.48
`open_laptop`	134.38	136.55	135.90	134.13	133.24
Macro average	222.01	224.62	222.85	223.48	220.83

Table 11: S10 source-task evaluation in Clean under a 1M-learner-update training budget. Each task uses 100 episodes; Δ is the change in percentage points. Mean steps include failures. Source-anchor checkpoint selection follows Appendix B.1.

	Success			Mean episode steps	
Task	Base	+GAE	Δ (pp)	Base	+GAE
`turn_switch`	77%	79%	+2.00	86.71	85.39
`move_playingcard_away`	100%	100%	0.00	107.36	107.33
`pick_diverse_bottles`	94%	95%	+1.00	106.81	106.01
`hanging_mug`	68%	70%	+2.00	323.39	322.52
`adjust_bottle`	97%	97%	0.00	118.62	118.59
`grab_roller`	99%	99%	0.00	85.89	85.95
`lift_pot`	99%	99%	0.00	101.78	101.82
`stack_blocks_two`	98%	99%	+1.00	291.96	291.02
`place_object_scale`	92%	90%	-2.00	144.16	146.12
`beat_block_hammer`	89%	95%	+6.00	122.26	115.38
Macro average	91.30%	92.30%	+1.00	148.89	148.01

comparator uses the same sampling framework as CISA, replacing only the priority signal with absolute TD error. All three arms retain the same CIV-based score-aware retention policy from the start of training. The controls include the task-allocation rule and its parameters, replay and Fresh/Critical capacities, score-refresh configuration, architecture, initialization protocol, and action scales. Uniform sampling removes the priority component while retaining the scoring needed by retention. The collectors remain independent: changing the sampler can change the learned policy, visited states, and realized task exposure. The training budget is 500,000 learner updates for each arm; checkpoint selection follows Appendix B.1. Evaluation uses 100 episodes per task with matched seeds and instructions across the sampling arms.

Tables 12 and 13 show selected-checkpoint outcomes alongside the frozen-base reference. The TD-error arm retains CISA's rank-based framework and CIV-based retention, using absolute TD error as its sampling score. Its losses also use no inverse-probability importance weights.

Table 12: S4 success under a 500k-update training budget in Clean. Each arm uses its anchor-selected checkpoint; the `hanging_mug` selection evaluation is reused in the table and macro average. Each trained arm uses 100 evaluation episodes per task; the existing frozen-base evaluation is shown as a reference. Δ is CISA minus Uniform in percentage points.

Task	Frozen base	Uniform	TD-error	CISA	Δ (pp)
`turn_switch`	77%	75%	70%	78%	+3
`move_playingcard_away`	100%	98%	99%	100%	+2
`pick_diverse_bottles`	94%	93%	88%	95%	+2
`hanging_mug`	68%	65%	67%	71%	+6
Macro average	84.75%	82.75%	81.00%	86.00%	+3.25

Table 13: Mean evaluation episode length at the selected S4 checkpoints. Steps are averaged over all evaluation episodes, including failures.

Task	Frozen base	Uniform	TD-error	CISA
`turn_switch`	86.71	89.16	94.14	84.96
`move_playingcard_away`	107.36	109.15	108.14	106.27
`pick_diverse_bottles`	106.81	105.61	109.20	107.33
`hanging_mug`	323.39	323.38	322.25	319.69

The TD-error arm has a mean episode length of 158.43 steps, macro-averaged across tasks. Relative to Uniform, mean episode length is lower with CISA on three tasks and higher on `pick_diverse_bottles`, where success also improves.

The visual-package comparison and its task-level results appear in Section 4.4 and Table 3; the fixed controls are summarized in Table 7.

B.4 Learner-update budgets and stochasticity

Table 14 summarizes the available S4 training and evaluation budgets. We match learner-update budgets and checkpoint selection. Runs use different hardware; environment-transition counts, update-to-data ratios, wall-clock times, and compute costs are unreported. The comparisons therefore measure selected-checkpoint success at a fixed learner-update budget. All settings use one training run per arm, with 100 evaluation episodes per task and task-level outcomes reported alongside macro averages.

Table 14: S4 Clean training budgets and evaluation denominators. Anchor selection uses nine 100-episode evaluations per arm; the selected anchor evaluation is reused in the reported results.

Comparison	Training updates / arm	Reported episodes / task
Uniform / TD-error / CISA	500,000	100
ResNet-10 / CNN+A-LIX	500,000	100

B.5 Two-task historical-memory protocol

The sequence is `pick_dual_bottles` followed by `hanging_mug`, with stage budgets of 300k and 500k additional learner updates, respectively (800k in total per branch). Both branches use Clean, the frozen base, and the common action scale $\lambda_m = 5\times10^{-4}$. At the end of stage A, historical memory is formed from the transitions then resident in its replay: the CIV-guided branch retains the highest-scoring 10%, while the comparator selects 10% uniformly. The percentage refers to this boundary replay, not all historical interactions. The selected memory stays fixed throughout B.

Only hanging_mug supplies new interactions in B. Current-task replay and old-task memory each receive half of the learner sampling allocation; within-pool sampling is uniform in both branches. This boundary memory selection is distinct from ongoing replay-row lifetime management and from CISA's within-task prioritized sampling.

Stage A selects its best checkpoint by pick_dual_bottles success; stage B selects by hanging_mug success. The selected current-task E100 evaluations are reused in Table 5, and the stage-B checkpoint is also evaluated on the old task. Stage labels identify completed training phases; evaluation uses their selected checkpoints. Mean episode steps include successful and failed episodes. The old-task changes between the reported checkpoints are 0.00 pp for critical-state memory and a 1.00 pp decrease for uniform memory. The final old-task gap also includes their stage-A performance difference. This study evaluates memory selection with rehearsal in both branches on one task transition.

C Action-value and expert-routing diagnostics

C.1 Temporal concentration of estimated correction value

Recorded configuration and selection. Figures 2 and 5 use archived S4 hanging_mug evaluations with an earlier shared ten-head critic. The same-state contrast is $\Delta Q_t = \min_h Q_h(x_t, \bar{a}_t) - \min_h Q_h(x_t, b_t)$. We select a successful episode (05, seed 322) and a failed episode (09, seed 368) for qualitative inspection; the former supplies the Introduction inset. Four positive local maxima per episode are selected by descending value with a 30-step minimum separation; labels follow temporal order. Curves retain every signed value without smoothing or amplitude normalization; frames precede the corresponding actions.

Concentration across the archived panel. For each of the first ten archived episodes (3,159 control steps in total), let T be its length, $c_t = [\Delta Q_t]_+$, and I_T index its largest $\lceil 0.1T \rceil$ values. The fraction $\sum_{t \in I_T} c_t / \sum_{t=1}^{T} c_t$ has median 68.36% and range 59.59%–74.91% in this panel. A small fraction of steps thus carries most positive estimated contrast. Positive peaks also occur in failure: local estimated benefit need not imply episode success. This exploratory evidence motivates selective allocation; the endpoint comparisons use the current task-specific critics.

C.2 Shared-expert use on fixed source states

Probe protocol. We examine an independent S4 training run with four experts and top-2 routing. The probe bank contains 32 fixed states per source task, sampled by Algorithm R reservoir sampling (seed 0) from the first 256 accepted candidates per task during early Clean collection. Selection uses no evaluation outcome. The same 128 states are evaluated every 10,000 critic updates from 10,000 to 600,000, giving 60 snapshots. Actor updates occur once per eight critic updates; the final snapshot has 75,000 actor updates. Diagnostic inference disables augmentation, dropout, router noise, and action sampling. The external action scale is 5×10^{-4}. These are longitudinal observations of one fixed probe bank, not repeated training runs or held-out evaluations.

Overlapping expert combinations. Figure 6 averages normalized top-2 weights over the 32 states of each task at the final snapshot. Expert use overlaps across tasks: E1 and E2 both contribute substantially to hanging_mug and move_playingcard_away, while E3 receives substantial mass on turn_switch and pick_diverse_bottles. Figure 7 connects selected combinations to their visual inputs. For each task, we choose the stored episode containing the most probe states and display four chronological quantiles; expert identity and success do not enter this choice. The original 84×84 network inputs are shown without enhancement. Together, the figures illustrate reuse through overlapping combinations rather than a one-expert-per-task assignment.

Expert predictions and training dynamics. Figure 8 displays each expert's deterministic scaled correction for the second selected probe of each task in Figure 7. The different coordinate-wise predictions show the alternatives available to the router; only selected experts contribute to the editor's combined output. Figure 9 follows the same probe bank through training. Full-router entropy approaches its maximum while top-2 expert sets continue to change and expert-output differences

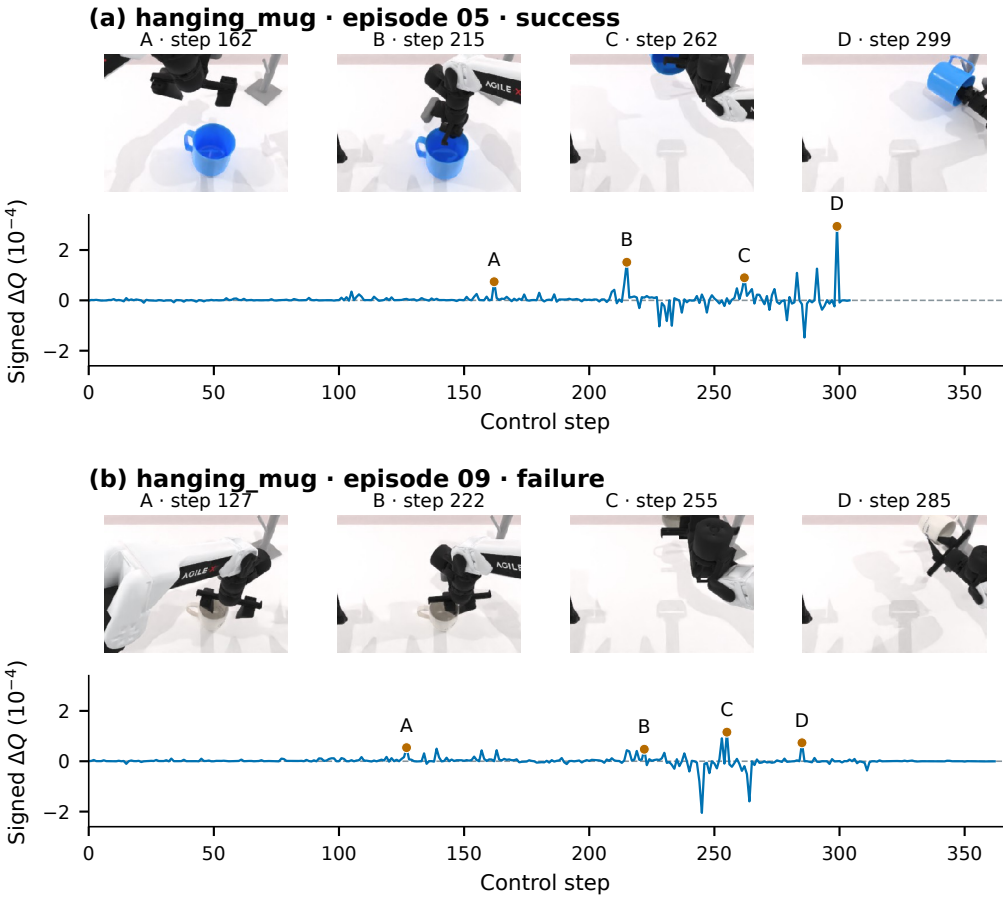


Figure 5: **Signed correction-value contrasts in successful and failed rollouts.** Both panels show complete `hanging_mug` episodes on shared axis scales. Letters match positive peaks to pre-action camera frames; negative contrasts remain visible. The upper trace also appears in Figure 2.

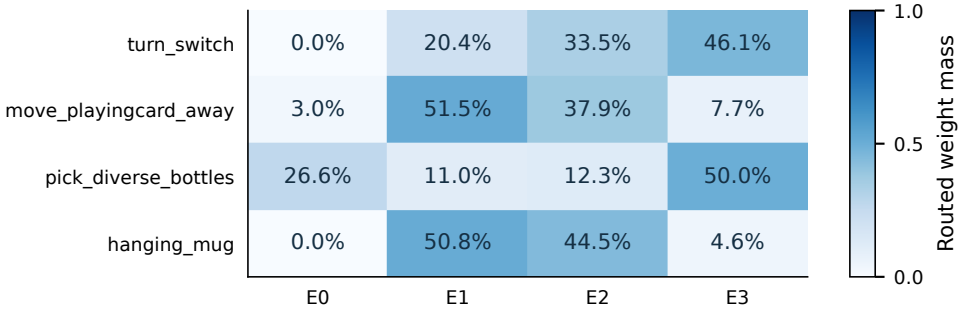


Figure 6: **Expert use overlaps across source tasks.** Mean normalized top-2 routing weights over 32 fixed states per task at 600k critic updates. Each row sums to one; expert indices identify shared parameter sets.

increase. These observations can coexist: small differences among near-uniform full-router probabilities can change which experts enter the top-2 set. The diagnostics describe the learned combinations and predictions; measuring their benefit over a dense actor requires a matched architectural ablation.



Figure 7: **Visual inputs and their routed expert combinations.** Four chronological probes from one stored episode per task at the final snapshot. Bars show normalized weights on the selected experts. The images are fixed source inputs, evaluated by the same saved actor.

For reproducibility, normalized router entropy is $\mathbb{E}_z[H(p(\cdot \mid z))]/\log 4$, where p is the full-router softmax. For task m, let $\bar{w}_{m,e}$ be expert e's mean normalized routed weight. The effective expert count is $\exp(H(\bar{w}_m))$, averaged across tasks, not the number of experts activated on a single input. Top-2 change rate compares unordered expert sets against the preceding snapshot. Output difference is the mean pairwise RMS between unscaled expert corrections, averaged over probe states and expert pairs. The first change-rate value is undefined; lines join the recorded captures without smoothing.

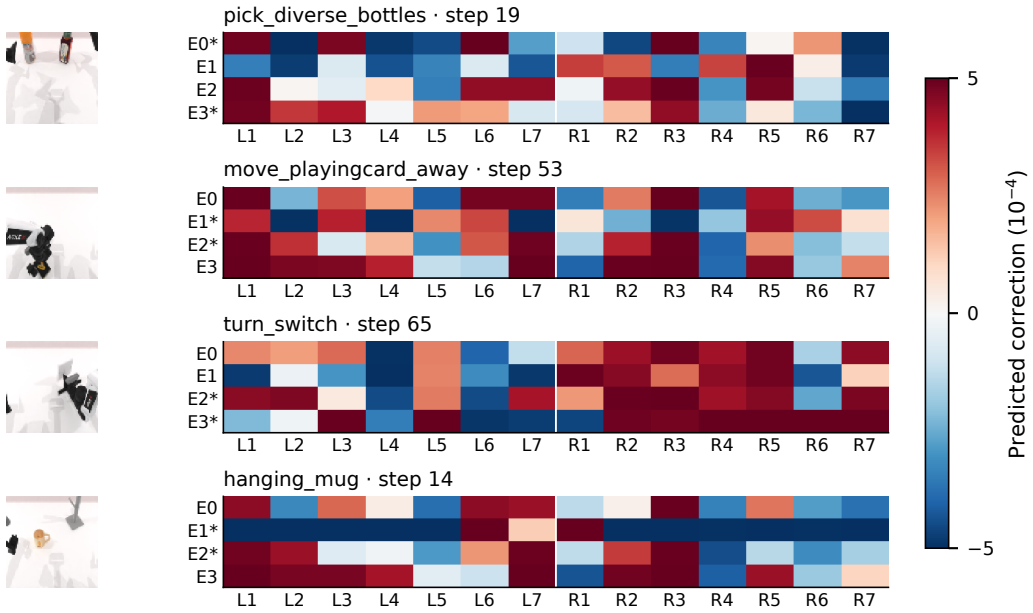


Figure 8: **Different expert predictions for the same input.** Each task panel uses its second displayed probe. Rows show all four experts; asterisks mark those selected by the router. L1–L7 and R1–R7 are left- and right-arm action coordinates. The common color scale is the external correction range $[-5 \times 10^{-4}, 5 \times 10^{-4}]$ in normalized coordinates.

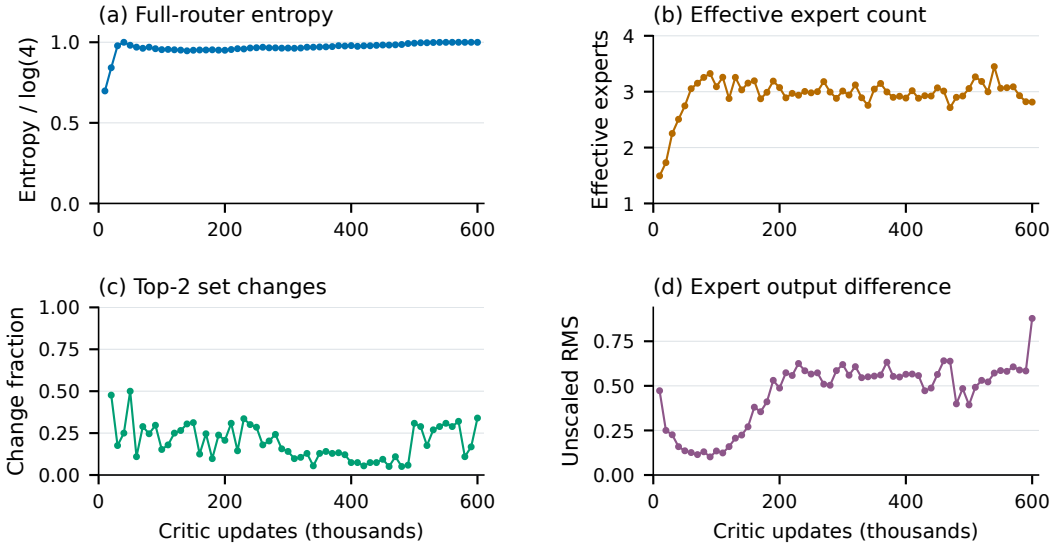


Figure 9: **Routing and expert outputs evolve on unchanged inputs.** Sixty snapshots of the same 128 source-state probes track full-router entropy, effective expert count, top-2 set changes, and expert-output differences. Metrics are defined in Appendix C.2.