LEARNING FROM CONSEQUENCES: POSTERIOR-TRANSITION REWEIGHTING FOR BEHAVIOR CLONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Robot demonstrations contain more than action labels: they also record the observations that follow each action. We investigate whether these future observations can guide how action supervision is allocated during vision-language-action (VLA) post-training. Posterior-Transition Reweighting (PTR) identifies the observed future of each demonstrated action chunk among alternatives drawn from the training data. The same identification model provides an auxiliary representation-learning objective and a bounded, stop-gradient weight on the original action loss. This separates two uses of future information: learning features and weighting supervision. PTR requires neither reward labels nor an evaluable action likelihood and applies directly to flow-matching regression. Our analysis shows that the expected score equals the identification information carried by the candidate list minus the scorer's error, and that the bounded weights redistribute supervision without discarding any example. Experiments on LIBERO, RoboCasa, and twelve real-robot tasks examine heterogeneous pooled training and controlled corruption of demonstrations. In pooled training, PTR raises real-robot success from 50.0% to 63.8% over standard fine-tuning, with gains on all twelve tasks. Controls in pooled simulation training show that reweighting contributes beyond memory and auxiliary representation learning. These results support future observations as a practical source of additional supervision for VLA adaptation.

1 INTRODUCTION

Supervised post-training adapts a pretrained vision-language-action (VLA) policy by fitting demonstrated action chunks (Brohan et al., 2022; Zitkovich et al., 2023; Kim et al., 2024; Black et al., 2024; Intelligence et al., 2025). The demonstration pool often mixes embodiments, viewpoints, control rates, and operators (Mandlekar et al., 2019; Collaboration et al., 2023; Khazatsky et al., 2024; Walke et al., 2023), and some of its action labels can be unreliable. The action objective nevertheless gives every sampled chunk the same loss coefficient, and its own error cannot tell a corrupted label from a valid action that the current policy finds hard to fit.

Demonstrations also record the observations that follow each action. A model that sees the context and the action can exploit recurring relations among context, action, and what follows to pick the recorded future out of futures taken from other demonstrations. Corrupting an action label while the camera stream still shows what the robot did can break these relations and make the identification harder. Identifying the future therefore gives evidence about each chunk that complements the regression error and requires no additional annotation.

Such future observations are usually exploited as auxiliary prediction targets; contrastive prediction, for example, learns representations by identifying a future among alternatives (Oord et al., 2018). That changes shared features but leaves the coefficients of the action loss untouched. We ask a different question: *can the observed future decide how much each demonstrated action should count?* Separating the two uses yields both a method and a direct control, which keeps the identification model and all of its auxiliary gradients but fixes the action weights to one.

Posterior-Transition Reweighting (PTR), shown in Figure 1, implements this idea. A scorer reads the policy context and the action-head features of a chunk and identifies the future recorded in the same trajectory among candidates from other samples. The log-probability of the correct candidate relative to uniform guessing becomes a bounded, positive weight on the original flow-matching action loss.

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.

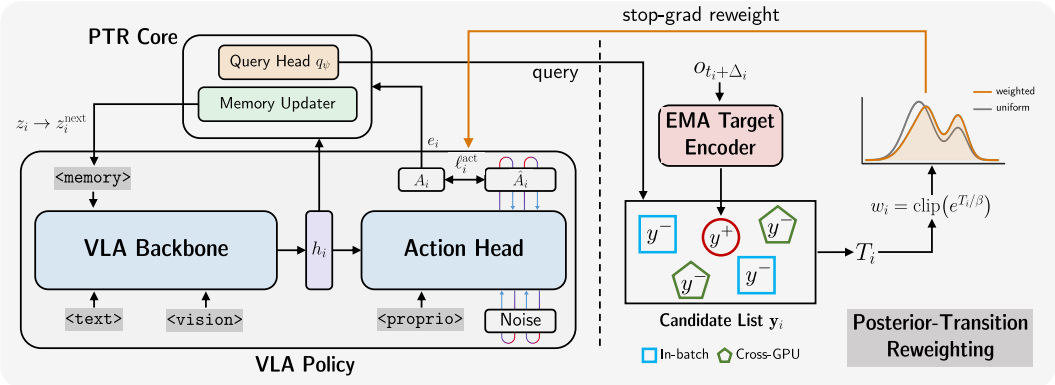


Figure 1: Overview of PTR for one example i. The VLA backbone and flow-matching action head produce the action loss ℓ_i^{act} on the recorded chunk A_i. During training, the query head q_ψ reads h_i and e_i, and the scorer identifies the EMA-encoded future observation $o_{t_i+\Delta_i}$ in the candidate list $\mathbf{y}_i$ (matched target y_i^+, alternatives y^- from other examples). The score T_i becomes a bounded weight w_i (shown for $\rho = 1$) that rescales the action loss through a stop-gradient path (orange); the density sketch shows the resulting redistribution of loss mass, and the identification losses also train the shared features. Components right of the dashed line are used only in training; the memory updater, which passes z_i^{next} to the next chunk, is kept at deployment.

The scorer is trained jointly with the policy representation, while the weights are detached during the action update, and neither a reward, new environment interaction, nor an action likelihood is required.

Three properties clarify how PTR uses this evidence. First, the expected score equals the information the candidate list carries about which candidate is the recorded future, minus the scorer's error, so the score has a precise meaning for any candidate construction. Second, the weights are positive and bounded, so they move supervision between examples without removing any, and they change continuously with the scores instead of making a keep-or-drop decision. Third, the weights only rescale the original action objective, so PTR plugs into existing supervised post-training as it is.

Our contributions are a weighting mechanism that sets the coefficients of the original VLA action objective from observed futures; an analysis of its score and of the supervised update it induces; and an evaluation that separates pooled adaptation, training-data corruption, and clean specialist training. Controls in pooled simulation show that the weights contribute beyond memory and auxiliary representation learning. On corrupted demonstrations, PTR gives lower mean raw weights to chunks that carry a corruption tag, although it never sees the tags. Across three real-robot embodiments, pooled training with PTR raises the success rate from 50.0% to 63.8% and improves all twelve tasks.

2 Related Work

VLA policies adapt pretrained vision-language representations to robot action prediction (Driess et al., 2023; Brohan et al., 2022; Zitkovich et al., 2023; Kim et al., 2024; Team et al., 2024; Black et al., 2024; Intelligence et al., 2025; Pertsch et al., 2025; Bjorck et al., 2025; Luo et al., 2025; 2026b; Feng et al., 2026; Luo et al., 2026a; Wang et al., 2026), and many predict continuous actions with diffusion or flow-matching heads (Ho et al., 2020; Chi et al., 2025; Lipman et al., 2022; Liu et al., 2022; Zhang et al., 2025b). Reinforcement-learning post-training of VLAs uses online interaction, rewards, preferences, or learned critics (Li et al., 2025; Guo et al., 2025; Huang et al., 2025; Chen et al., 2025b; Lu et al., 2025; Liu et al., 2025; Zhang et al., 2024; Zhai et al., 2025; Frans et al., 2025; Chen et al., 2025c; Zhang et al., 2025a). Offline reinforcement learning constrains value-based policies to the data (Levine et al., 2020; Fujimoto et al., 2019; Kumar et al., 2019a), and return-conditioned policies condition on a target return (Kumar et al., 2019b; Chen et al., 2021). PTR stays within supervised post-training and needs neither rewards nor a tractable action density.

Demonstration quality and data composition strongly affect imitation learning (Belkhale et al., 2023; Kuhar et al., 2023; Hejna et al., 2024). DemInf estimates state-action mutual information to curate demonstrations offline without quality labels (Hejna et al., 2025). Influence-based and datamodel-based methods instead assess data relative to a downstream performance or validation criterion (Agia

et al., 2025; Lee et al., 2026; Dass et al., 2025; Chen et al., 2025a). PTR uses a different signal: whether policy-side context and action features identify the future observation already paired with a chunk. Self-paced learning also adapts example weights to the model's current difficulty (Kumar et al., 2010); PTR instead derives its weights from an auxiliary identification loss.

PTR uses the candidate-identification construction of noise-contrastive and contrastive predictive learning (Gutmann & Hyvärinen, 2010; Oord et al., 2018), and its analysis builds on the associated density-ratio and variational-information viewpoints (Poole et al., 2019). Exponential sample weights also appear in reward-weighted and advantage-weighted regression and related policy search (Peters & Schaal, 2007; Peters et al., 2010; Peng et al., 2019; Nair et al., 2020; Abdolmaleki et al., 2018), where they follow from KL-regularized policy improvement (Levine, 2018). PTR instead exponentiates an identification score, and uses the future as a comparison target in training, not as a relabeled goal.

3 Posterior-Transition Reweighting

PTR uses the future observation of each demonstrated chunk in two ways: to train an auxiliary identification model, and to set the coefficient of that chunk in the original action loss. We first define the identification score and establish what it measures, then turn it into bounded coefficients. The index i denotes a training example, j a candidate position, and t a time step in a recorded trajectory; B always denotes the size of one group over which the action weights are normalized. The operator sg stops gradients, and $\mathrm{clip}_{[a,b]}(x) = \min\{b, \max\{a, x\}\}$.

3.1 Actions and observed futures

Example i contains an action chunk $A_i = a_{t_i:t_i+L-1}$ and deployable inputs summarized by h_i: images, instruction, proprioception, and memory from preceding chunks. The base policy supplies a scalar loss $\ell_i^{\mathrm{act}}(\theta)$ with its own action mask and reduction; for a fully active action vector, it is the flow-matching loss (Lipman et al., 2022)

$$\begin{aligned} X_i &= \tau_i A_i + (1-\tau_i)\epsilon_i, & V_i &= A_i - \epsilon_i, \\ \ell_i^{\mathrm{act}}(\theta) &= \left\| v_\theta(X_i, \tau_i, h_i) - V_i \right\|_2^2, & \epsilon_i &\sim \mathcal{N}(0, I), \quad \tau_i \sim p_\tau, \end{aligned} \tag{1}$$

with the norm taken after flattening the chunk. Standard fine-tuning averages these losses; PTR changes their coefficients, not their targets.

When available, a future observation is taken at $t_i + \Delta_i$, with $\Delta_i = k_i L$ and k_i drawn from the valid part of a predefined grid $\{1, \dots, H\}$; by default $L = 16$ and $H = 8$. A longer offset also reflects later recorded actions, so the target is not an isolated counterfactual outcome of A_i. Future images are used only in training. Let $\mathcal{I} \subseteq \{1, \dots, B\}$ be the examples for which such a future exists.

3.2 Identifying the observed future

The action summary e_i pools features from the action head's supervised forward pass; it may carry context, flow-time, and noise information, and no separate clean-action encoder is assumed. The scorer input is $c_i = (h_i, e_i)$, and the detached target is $y_i^+ = \mathrm{sg}(\phi_{\bar{\theta}}(o_{t_i+\Delta_i}))$, where $\phi_{\bar{\theta}}$ is an exponential moving average (EMA) of the policy's vision encoder. A list $\boldsymbol{y}_i = (y_{i1}, \dots, y_{iN_i})$ holds this target at position j_i and alternatives from other examples, drawn from the batch, other GPUs, a queue, and harder same-task or similar-state targets (Appendix B.3); repeated values remain distinct positions. A query head q_ψ scores each candidate with a clamped cosine logit. With $\mathrm{nrm}(x) = x / \max\{\|x\|_2, \zeta\}$ for a small $\zeta > 0$, temperature $\kappa > 0$, and clamp level $C > 0$,

$$\begin{aligned} s_\psi(c, y) &= \mathrm{clip}_{[-C,C]}\left(\frac{\langle \mathrm{nrm}(q_\psi(c)), \mathrm{nrm}(y) \rangle}{\kappa} \right), \qquad \widehat{p}_i(j) = \frac{\exp s_\psi(c_i, y_{ij})}{\sum_{r=1}^{N_i} \exp s_\psi(c_i, y_{ir})}, \\ T_i &= \log\big(N_i \widehat{p}_i(j_i)\big) = \log N_i - \ell_i^{\mathrm{id}}, \qquad \ell_i^{\mathrm{id}} = -\log \widehat{p}_i(j_i). \end{aligned} \tag{2}$$

The word posterior in the name of the method refers to $\widehat{p}_i$, a distribution over candidate positions. A score $T_i > 0$ means that the scorer puts more probability on the recorded future than uniform guessing on this list, and $T_i \le \log N_i$. It compares futures, not the recorded action with alternatives.

Algorithm 1 One PTR training step

Require: group of B examples with inputs, chunks A_i, and futures $o_{t_i+\Delta_i}$ where available; memory; target queue; controller state $(\kappa, \beta, \eta^{\mathrm{hard}})$

1: Run the policy forward pass: context h_i, action-head features e_i, and action loss ℓ_i^{act} (Eq. (1))
2: **for** each example i with a valid future **do**
3: $\quad y_i^+ \leftarrow \mathrm{sg}\big(\phi_{\bar{\theta}}(o_{t_i+\Delta_i})\big)$
4: $\quad$ Build the list $\boldsymbol{y}_i$ from in-batch, cross-GPU, queued, and harder targets
5: $\quad$ Compute $\widehat{p}_i(j_i)$ and T_i by Eq. (2), and w_i by Eq. (6)
6: **end for**
7: Set $w_i \leftarrow 1$ for every example without a valid future
8: $\alpha_i \leftarrow \mathrm{sg}(w_i)/\sum_r \mathrm{sg}(w_r)$ and $\mathcal{L}_{\mathrm{act}}^{\mathrm{PTR}} \leftarrow \sum_i \alpha_i \ell_i^{\mathrm{act}}$
9: Update the policy and scorer parameters with the gradient of $\mathcal{L}_{\mathrm{total}}$ (Eq. (8))
10: Update the EMA encoder (Eq. (B.4)), the target queue, and the controller (Eq. (B.6))
11: Pass the detached memory tokens to the next chunk of each trajectory segment (Eq. (B.1))

Before such a score can weight supervision, its meaning must be pinned down: a high value could come from information in the example or merely from a well-fitted scorer facing an easy list. To separate the two, freeze the training state, including representations, sampler, and temperatures, and fix a valid scorer input c and a list size $N \geq 2$. An independent uniform permutation of the list places the matched entry at a uniform index J without changing the matched score, since the scorer is permutation equivariant. Let $\mathbf{Y}$ be the permuted list and $p^*(\cdot \mid c, \mathbf{Y})$ its true matched-index posterior; all expectations condition on (c, N). Write $T = \log\big(N\widehat{p}_\psi(J \mid c, \mathbf{Y})\big)$ for the score of the matched entry. The scorer's approximation error is

$$\mathcal{E}_N(c) = \mathbb{E}_{\mathbf{Y}|c,N}\, D_{\mathrm{KL}}(p^*(\cdot \mid c, \mathbf{Y}) \,\|\, \widehat{p}_\psi(\cdot \mid c, \mathbf{Y}))\,. \tag{3}$$

Proposition 1 (Identification evidence)**.** *If $\mathcal{E}_N(c) < \infty$, then*

$$\mathbb{E}[T \mid c, N] = I(J; \mathbf{Y} \mid c, N) - \mathcal{E}_N(c), \qquad 0 \leq I(J; \mathbf{Y} \mid c, N) \leq \log N. \tag{4}$$

No independence or common-distribution assumption on the candidates is needed. Under the independent reference model, the positive has law P and the other $N-1$ entries are independent draws from Q, independent also of the positive. If $P \ll Q$ and $D_{\mathrm{KL}}(P \,\|\, Q) < \infty$, then

$$\mathbb{E}[T \mid c, N] = D_{\mathrm{KL}}(P \,\|\, Q) - b_N - \mathcal{E}_N(c), \qquad b_N = D_{\mathrm{KL}}\big(\mathsf{M}_N \,\big\|\, Q^{\otimes N}\big) \geq 0, \tag{5}$$

where M_N is the law of the permuted list. If additionally $\chi^2(P\|Q) < \infty$, then

$$b_N \leq \log\big(1 + \chi^2(P\|Q)/N\big).$$

The proposition splits the expected score into what the identification problem offers and what the fitted scorer misses. The first term is the information that the list carries about which candidate is the recorded future. That information may come from the context, the action-head features, or the way the list was built; the proposition does not say how much of it is due to the action. Under the independent reference model, the information becomes the divergence between the futures that follow this input and the reference futures, minus a finite-list correction b_N. For fixed P and Q, b_N does not increase with N and, when $\chi^2(P\|Q)$ is finite, vanishes as the list grows (Appendix A.1). A practical queue or hard-negative bank can change the reference law, so this limit guides candidate coverage without certifying a sampler; the derivation follows the density-ratio view of contrastive identification (Oord et al., 2018; Poole et al., 2019).

These facts shape how PTR uses the score. With representations and sampling fixed, the expected identification loss is a conditional entropy plus $\mathcal{E}_N(c)$, and the entropy does not depend on the query head, so minimizing this loss over the query head targets the approximation error. During joint training the identification problem itself changes, so a falling loss does not by itself show a falling error. The rank loss of Section 3.4 encourages sensitivity to the action-head features but cannot exclude context or sampler cues. Finally, a low score may reflect weak evidence or an inaccurate scorer, and a high score can still describe a coherent but task-incorrect action. PTR therefore introduces the weights gradually and applies the score through a bounded soft map, never as a keep-or-drop label.

3.3 FROM EVIDENCE TO BOUNDED COEFFICIENTS

For $\beta > 0$, $0 < w_{\min} \le 1 \le w_{\max}$, and $\rho \in [0, 1]$, the raw weight of a valid example $i \in \mathcal{I}$, the normalized coefficients, and the weighted action objective are

$$w_i = 1 - \rho + \rho \, \mathrm{clip}_{[w_{\min}, w_{\max}]}\left(e^{T_i/\beta}\right), \qquad \alpha_i = \frac{\mathrm{sg}(w_i)}{\sum_{r=1}^{B} \mathrm{sg}(w_r)}, \qquad \mathcal{L}_{\mathrm{act}}^{\mathrm{PTR}} = \sum_{i=1}^{B} \alpha_i \ell_i^{\mathrm{act}}(\theta), \tag{6}$$

with $w_i = 1$ for $i \notin \mathcal{I}$. The mixture coefficient ρ rises gradually from zero to one, and by default $[w_{\min}, w_{\max}] = [0.25, 4]$. The coefficients are uniform exactly when all raw weights in the group coincide, so a raw weight of one is not by itself an SFT coefficient. Three properties of this map, proved in Appendix A.2, explain why the coefficients are soft and bounded.

The map is a soft gate on identification. For valid-future examples with $\rho = 1$, identification losses at or below $\ell_{0,i} = \log N_i - \beta \log w_{\max}$ receive $w_{\max}$; above this threshold the raw weight decays exponentially until it reaches $w_{\min}$. Better identification cannot raise a weight that is already on the upper plateau, so clipping caps the preference for easily identified futures rather than removing it. With the defaults and $\rho = 1$, a chunk whose recorded future is identified no better than chance, $T_i \le 0$, receives a raw weight of at most one, whereas a clearly identified chunk, $T_i \ge \beta \log 4$, receives four. The map itself says nothing about how many training examples reach either plateau.

The coefficients are bounded. Set $a = 1 - \rho + \rho w_{\min}$, $b = 1 - \rho + \rho w_{\max}$, and $R = b/a$. For any subset $\mathcal{S}$ that occupies a fraction $\mu = |\mathcal{S}|/B$ of a normalization group,

$$\frac{\mu}{\mu + R(1-\mu)} \;\le\; \sum_{i \in \mathcal{S}} \alpha_i \;\le\; \frac{R\mu}{R\mu + 1 - \mu}. \tag{7}$$

In particular $R^{-1} \le B\alpha_i \le R$ and $D_{\mathrm{KL}}(\alpha \,\|\, u_B) \le \log R$, where u_B is uniform over the group. With the defaults and $\rho = 1$, $R = 16$: a wrong score can move supervision between examples, but no example is removed, and each keeps between $1/16$ and 16 times its uniform share.

The coefficients are stable. With the scales, the group, and the validity flags fixed, they change continuously with the scores: if every score moves by at most δ, every coefficient changes by at most a factor $e^{2\delta/\beta}$, whereas a keep-or-drop rule jumps at its threshold (compared in Section 4.4).

These coefficients change the original supervised regression objective and nothing else. Appendix A.3 derives the regression target they induce in a finite normalization group, including the effect of the loss normalizer, and states when it can be read as flow matching toward reweighted clean actions. PTR thus remains a supervised method: it reweights imitation of the demonstrations.

3.4 AUXILIARY LEARNING AND GRADIENT ROUTING

Let $\widetilde{e}_i$ be the action-head feature assigned to example i by a within-batch permutation of the valid examples. With $\mathrm{softplus}(x) = \log(1 + e^x)$, the auxiliary losses are

$$\mathcal{L}_{\mathrm{id}} = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \ell_i^{\mathrm{id}}, \qquad \mathcal{L}_{\mathrm{rank}} = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \mathrm{softplus}\big(s_\psi((h_i, \widetilde{e}_i), y_i^+) - s_\psi((h_i, e_i), y_i^+)\big), \tag{8}$$

both zero when $\mathcal{I}$ is empty, and the training objective is $\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{act}}^{\mathrm{PTR}} + \lambda_{\mathrm{id}}\mathcal{L}_{\mathrm{id}} + \lambda_{\mathrm{rank}}\mathcal{L}_{\mathrm{rank}} + \mathcal{L}_{\mathrm{mem}}$. The rank term rewards a higher score for the chunk's own action-head features than for exchanged ones, which may themselves contain context and noise. Algorithm 1 summarizes one update, a single forward and backward pass with no environment interaction, relabeling, or second training stage. Auxiliary gradients train the query head and the shared policy features, whereas future targets and action-loss weights are detached, so the two uses of the future can be separated: a control that keeps the entire auxiliary path but fixes $w_i = 1$ isolates the weighting intervention at the level of complete training runs (Section 4.3), although it does not hold the learned representation fixed. At deployment the policy keeps its memory path (Appendix B.2) and drops the scorer, target encoder, candidate bank, and controller, so its inference cost matches that of SFT+Memory.

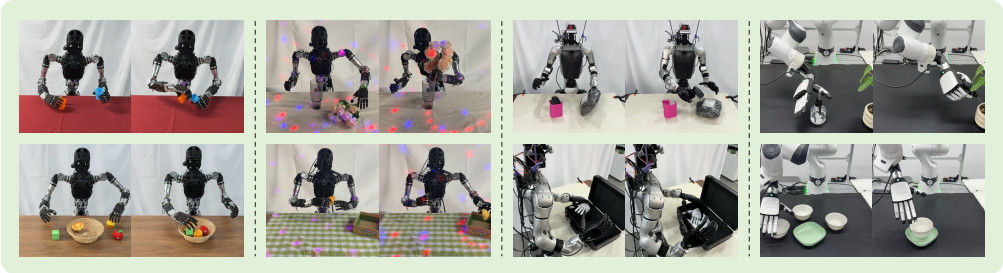


Figure 2: Representative real-robot tasks on G1 with LinkerHand O6 hands (26 DoF, 20 Hz), Adam-U (31 DoF, 20 Hz), and FR3 with an Inspire hand (13 DoF, 50 Hz). Figure 10 in Appendix C.7 shows all twelve tasks.

Table 1: Success rates (%) by training regime. Simulation entries are means and sample standard deviations over three training seeds. Real-robot entries pool twelve tasks of twenty trials, one trained policy per method. The corrupted specialist uses the Combined corruption. The better entry of each pair is in bold.

	LIBERO		RoboCasa		Real robots	
Training regime	**SFT**	**PTR**	**SFT**	**PTR**	**SFT**	**PTR**
Pooled generalist	96.0±0.3	**97.2**±0.3	50.7±0.7	**54.9**±0.6	50.0	**63.8**
Corrupted specialist	85.0±0.4	**90.9**±0.4	36.2±0.9	**45.6**±0.7	–	–
Clean specialist	**98.2**±0.2	97.7±0.3	54.1±0.9	**55.6**±0.5	60.8	**67.1**

4 Experiments

Following Section 3, the evaluation asks where the gains appear across training regimes, whether the weights add to the auxiliary path, and how the soft map compares with a threshold.

4.1 Experimental setup

The base policy is Being-H0.5 (Luo et al., 2026b). LIBERO (Liu et al., 2023) contains four ten-task suites, evaluated with fifty episodes per task, while the RoboCasa study (Nasiriany et al., 2024) contains twenty-four kitchen tasks evaluated with fifty trials per task over five layouts. The hardware study contains twelve tasks on G1, Adam-U, and FR3, each with twenty evaluation trials (Figure 2). Specialist training uses the corresponding target data. Pooled generalist training combines the two simulation and three robot sources, with per-benchmark step budgets in Appendix B.5.

The corruption study modifies only specialist training demonstrations. Action Noise Injection (ANI) adds Gaussian noise with standard deviation 0.1 to the recorded actions of 30% of the trajectories; Trajectory Truncation (TT) retains 40–70% of the length of 25% of the trajectories; Label Noise (LN) reassigns the instructions of 20% of the trajectories. Combined applies all three, and evaluation environments are unchanged. Simulation results report the mean and sample standard deviation over three training seeds, except for the curation and corruption-type studies, which are separate experiments. Hardware results use one trained policy per method in each regime.

4.2 Results across training regimes

Table 1 summarizes the three regimes. In pooled generalist training, PTR is above SFT on all three benchmarks, by 1.2 percentage points on LIBERO, 4.2 on RoboCasa, and 13.8 on the real robots. These gains hold on all four LIBERO suites (Table 8) and all three RoboCasa categories (Table 2). Under the Combined corruption, the differences reach 5.9 points on LIBERO and 9.4 on RoboCasa. On clean specialist data, PTR shows no benefit on near-ceiling LIBERO (97.7% versus 98.2%) but improves RoboCasa from 54.1% to 55.6% and the hardware specialists from 146 to 161 successes, so the gap is larger in pooled and corrupted training.

Table 2: Pooled-simulation controls (success rate, %, three training seeds). Each row adds one component to the row above: the memory path; the auxiliary path, that is, the scorer objectives and their gradients into the shared features, with every action weight fixed to one; and the learned action weights. The last two rows differ only in the action weights. RoboCasa is broken down into its three task categories.

		RoboCasa			
Configuration	**LIBERO**	**Pick & place**	**Doors & drawers**	**Others**	**Average**
SFT	96.0±0.3	32.8±0.9	67.5±0.8	51.7±0.7	50.7±0.7
SFT+Memory	96.3±0.3	33.6±0.8	68.2±0.7	52.0±0.7	51.3±0.7
Full scorer, $w = 1$	96.6±0.3	34.5±0.8	69.0±0.7	52.6±0.7	52.0±0.7
PTR	**97.2**±0.3	**37.3**±0.7	**72.3**±0.6	**55.1**±0.6	**54.9**±0.6

Table 3: Comparisons on the Combined corruption (success rate, %). (a) Study with matching SFT and PTR runs; MI-based curation, validation-loss influence (Influence), and datamodel selection adapt Hejna et al. (2025), Lee et al. (2026), and Dass et al. (2025). (b) Three training seeds, a separate experiment not ranked jointly with (a); the last two columns give the mean raw weights of PTR before normalization for untagged and tagged chunks, and training never uses the tags.

(a) Adapted curation methods

	SFT	**Random**	**Loss-based**	**MI-based**	**Influence**	**Datamodel**	**PTR**
LIBERO	85.4	85.1	87.3	88.6	89.2	89.6	**91.0**
RoboCasa	36.4	36.0	39.2	42.1	42.8	43.5	**45.8**

	(b) Weighting rules and a tag reference, three seeds				**Mean raw weight**	
	SFT	**Hard-PTR**	**PTR**	**Oracle-Tag**	**Untagged**	**Tagged**
LIBERO	85.0±0.4	89.7±0.5	90.9±0.4	91.6±0.4	1.48	0.72
RoboCasa	36.2±0.9	43.8±0.7	45.6±0.7	46.7±0.6	1.56	0.61

4.3 What the weighting path adds

Table 2 contains the key control: the full identification model, its auxiliary gradients, and memory are present, while the action weights are fixed to one. Applying learned weights raises the pooled success rate from 52.0% to 54.9% on RoboCasa, in all three task categories, and from 96.6% to 97.2% on near-ceiling LIBERO. Since the last two configurations differ only in the action weights, the weighting path adds to the auxiliary path, clearly on RoboCasa and by a smaller margin on LIBERO.

4.4 Robustness to corrupted supervision

Table 3 compares PTR with other weighting and curation rules on the Combined corruption. Hard-PTR zeros negative-score action coefficients and renormalizes the retained losses, with its own scorer; it is the keep-or-drop rule that Section 3.3 contrasts with the soft map. PTR outperforms it on both simulation benchmarks, and also in pooled real-robot training without injected corruption, 63.8% against 58.3% for Hard-PTR and 50.0% for SFT. PTR also reaches about 89% of the improvement over SFT of Oracle-Tag, which weights untagged trajectories by one and tagged ones by 0.25; this reference is not an upper bound, since trajectory-level tags can cover useful prefix chunks. The curation block also places PTR above every adapted curation method.

The raw weights also track the corruption, as Section 1 anticipated. Without seeing the tags, PTR gives tagged chunks a mean raw weight of 0.72 on LIBERO and 0.61 on RoboCasa, against 1.48 and 1.56 for untagged chunks (Table 3b), and untagged chunks make up 87.5% and 91.3% of the top-weight quartile. These values are raw weights before normalization.

A further study applies each corruption type separately (Figure 3). For every type on both benchmarks, the drop of PTR from its own clean result is 46–57% of the corresponding SFT drop (Appendix C.4).

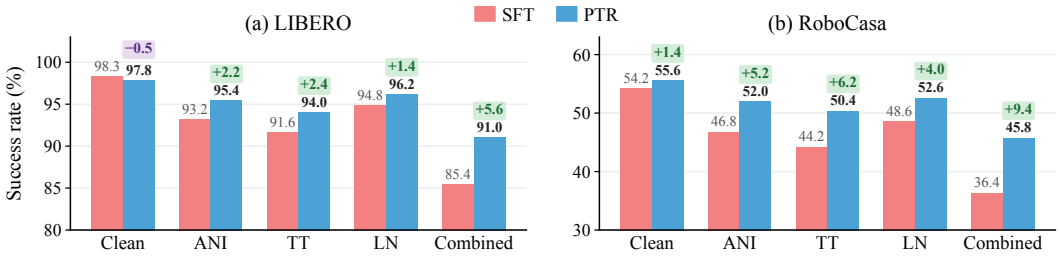


Figure 3: Study of each corruption type (success rate, %). Clean is uncorrupted training data, and badges show PTR minus SFT. These runs are separate from the three-seed results of Table 1.

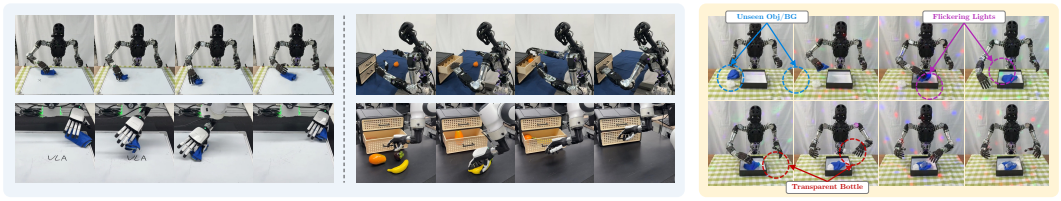


Figure 4: Left: Wipe Board and Drawer Organization on Adam-U (top) and FR3 (bottom). Right: a qualitative PTR rollout with unseen objects and background, flickering lights, and a transparent bottle.

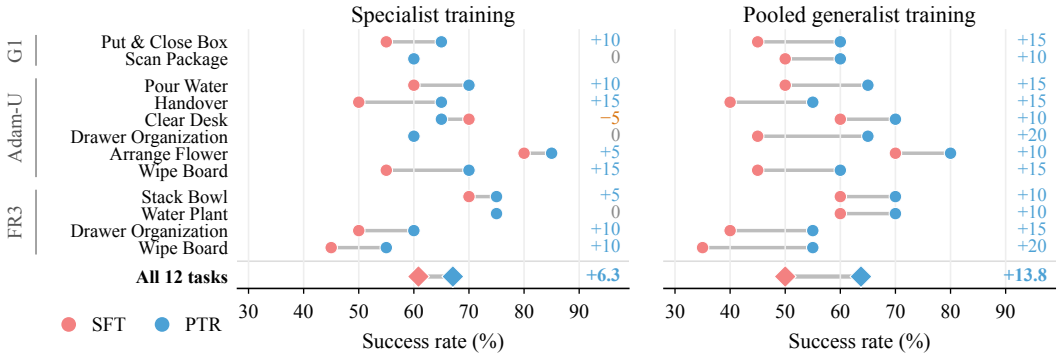


Figure 5: Per-task real-robot success rates, twenty trials per task and one trained policy per method and regime. Numbers on the right are PTR minus SFT in points; the bottom row pools all 240 trials (Table 7).

4.5 Real-robot coverage

Pooled training on the real robots mixes three action layouts, with 26, 31, and 13 active dimensions of the shared action space, and two tasks run on two embodiments (Figure 4, left). The pooled result rises from 120 to 153 successes out of 240 trials, or 50.0% to 63.75% before rounding, and all twelve task differences are positive, with a median of 15 percentage points and a range of 10–20 (Figure 5). By platform the counts rise from 19/40 to 24/40 on G1, 62/120 to 79/120 on Adam-U, and 39/80 to 50/80 on FR3, so the gain is not concentrated in one embodiment. The two tasks run on both Adam-U and FR3 improve on both embodiments: Wipe Board by 15 and 20 points and Drawer Organization by 20 and 15 points.

4.6 Training dynamics

Figure 6 follows the scorer in pooled and in specialist training. Accuracy, mean score, and logit margin describe different aspects of identification and need not saturate together: accuracy stops rising once the recorded future usually ranks first, while the margin to the hardest alternative can keep growing. As the representation, candidates, and temperatures change, the levels are not compared across settings. In pooled real-robot training the raw weights start near one and later spread across the clipped range (panel d); Appendix C.6 adds further curves.

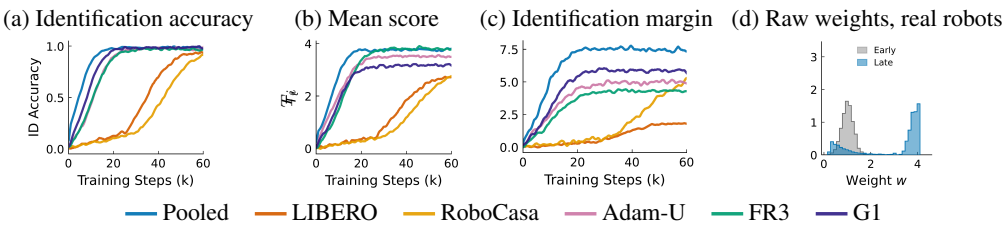


Figure 6: Training dynamics: (a) identification accuracy, (b) mean score T_i, and (c) margin between the matched logit and the hardest alternative in six settings; (d) raw weights in pooled real-robot training, early and late.

Table 4: Design analyses (success rate, %). (a) Clean specialist training, three seeds; each column removes one component, and w/o cross-GPU removes candidates gathered from other GPUs. (b) Pooled training; RoboCasa uses three seeds, the real robots one policy per setting. (c) RoboCasa, Combined corruption, three seeds (SFT: 36.2 ± 0.9).

(a) Components, clean specialist training

	SFT	PTR	w/o memory	w/o cross-GPU	w/o controller	w/o clipping
LIBERO	98.2±0.2	97.7±0.3	97.3±0.3	96.9±0.4	97.0±0.4	95.3±0.5
RoboCasa	54.1±0.9	55.6±0.5	53.8±0.7	54.1±0.7	51.7±0.7	48.9±0.9

(b) Future-offset grid, pooled training

	$H = 1$	$H = 2$	$H = 4$	$H = 8$
RoboCasa	53.6±0.7	54.1±0.6	54.5±0.6	54.9±0.6
Real robots	59.6	60.8	62.5	63.8

(c) Rank loss, Combined

Variant	RoboCasa
w/o rank loss	44.0±0.7
PTR	45.6±0.7

4.7 Design analyses

In the clean-specialist ablations of Table 4a, removing the clip, which provides the bounds of Eq. (7), causes the largest drop on both benchmarks, and removing the scale controller costs 3.9 points on RoboCasa. Removing the memory tokens or the cross-GPU candidates also lowers success on both benchmarks, to 53.8% and 54.1% on RoboCasa from 55.6%, in line with the role of temporal context and of longer candidate lists in Section 3.2. Without the rank loss, success on RoboCasa Combined falls from 45.6% to 44.0% and stays above SFT at 36.2% (Table 4c). A larger future-offset grid, reaching 6.4 s ahead at 20 Hz and 2.56 s at 50 Hz for $H = 8$, raises success gradually on RoboCasa and the real robots (Table 4b). Every tested component thus contributes, and bounding the weights matters most on both benchmarks, as the analysis of Section 3.3 suggests.

5 Conclusion and Limitations

PTR weights the original action objective by how well the recorded future of each demonstrated chunk is identified. Its expected score is identification information minus scorer error, and its positive, bounded coefficients limit how far each update redistributes supervision. The weights improve on a pooled-simulation control that already has memory and auxiliary representation learning, make training more robust to corrupted demonstrations, and improve all twelve real-robot tasks in pooled training. Since the weights come from observations that every demonstration already records, PTR adds this supervision signal without new annotation, at about 12% training overhead (Appendix B.6) and with the inference cost of SFT+Memory. Because it only changes per-example loss coefficients, the same mechanism applies to any behavior-cloning objective with a scalar per-example loss.

PTR relies on future observations stored with the demonstrations, so chunks whose future lies beyond the end of a trajectory keep a raw weight of one. The future offset is drawn from a fixed grid, so selecting it adaptively for each task is a further extension. Its score is predictive evidence relative to the candidate list rather than task value, which makes combining it with task-level signals such as success labels, and evaluating it with other behavior-cloning architectures, natural next steps.

AI USE STATEMENT

Generative AI tools assisted with discovery of related work and polishing the writing. The authors take responsibility for the final content, including its claims, citations, figures, and results.

ETHICS STATEMENT

The reported experiments concern teleoperated object manipulation in controlled laboratory settings. The dataset descriptions state that the trajectories contain no personally identifiable human-subject data. PTR provides no safety guarantee: an easily identified consequence can still be undesirable. Deployment requires task-level validation, monitoring, and hardware safety measures.

REPRODUCIBILITY STATEMENT

Section 3 specifies the weighting and auxiliary objectives, and Appendix A states the assumptions and gives complete derivations. Appendix B gives the training procedure, the memory and controller equations, the shared configuration, the training budgets, and the definitions of all controls and baselines. Appendix C gives the per-task hardware results and aggregation formulas.

REFERENCES

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. *arXiv preprint arXiv:1806.06920*, 2018.

Christopher Agia, Rohan Sinha, Jingyun Yang, Rika Antonova, Marco Pavone, Haruki Nishimura, Masha Itkina, and Jeannette Bohg. Cupid: Curating data your robot loves with influence functions. *arXiv preprint arXiv:2506.19121*, 2025.

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

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

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

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Annie S Chen, Alec M Lessing, Yuejiang Liu, and Chelsea Finn. Curating demonstrations using online experience. *arXiv preprint arXiv:2503.03707*, 2025a.

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, Quanlu Zhang, Zhaofei Yu, Guoliang Fan, et al. πrl: Online rl fine-tuning for flow-based vision-language-action models. *arXiv preprint arXiv:2510.25889*, 2025b.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

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

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, 2025.

OX-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 1(2), 2023.

Shivin Dass, Alaa Khaddaj, Logan Engstrom, Aleksander Madry, Andrew Ilyas, and Roberto Martín-Martín. Datamil: Selecting data for robot imitation learning with datamodels. *arXiv preprint arXiv:2505.09603*, 2025.

Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

Yicheng Feng, Wanpeng Zhang, Ye Wang, Hao Luo, Haoqi Yuan, Sipeng Zheng, and Zongqing Lu. Spatial-aware vla pretraining through visual-physical alignment from human videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Kevin Frans, Seohong Park, Pieter Abbeel, and Sergey Levine. Diffusion guidance is a controllable policy improvement operator. *arXiv preprint arXiv:2505.23458*, 2025.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pp. 2052–2062. PMLR, 2019.

Yanjiang Guo, Jianke Zhang, Xiaoyu Chen, Xiang Ji, Yen-Jen Wang, Yucheng Hu, and Jianyu Chen. Improving vision-language-action model with online reinforcement learning. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 15665–15672. IEEE, 2025.

Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 297–304. JMLR Workshop and Conference Proceedings, 2010.

Joey Hejna, Chethan Bhateja, Yichen Jiang, Karl Pertsch, and Dorsa Sadigh. Re-mix: Optimizing data mixtures for large scale imitation learning. *arXiv preprint arXiv:2408.14037*, 2024.

Joey Hejna, Suvir Mirchandani, Ashwin Balakrishna, Annie Xie, Ayzaan Wahid, Jonathan Tompson, Pannag Sanketi, Dhruv Shah, Coline Devin, and Dorsa Sadigh. Robot data curation with mutual information estimators. *arXiv preprint arXiv:2502.08623*, 2025.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Dongchi Huang, Zhirui Fang, Tianle Zhang, Yihang Li, Lin Zhao, and Chunhe Xia. Co-rft: Efficient fine-tuning of vision-language-action models through chunked offline reinforcement learning. *arXiv preprint arXiv:2508.02219*, 2025.

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

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

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

Sachit Kuhar, Shuo Cheng, Shivang Chopra, Matthew Bronars, and Danfei Xu. Learning to discern: Imitating heterogeneous human demonstrations with preference and representation learning. In *Conference on Robot Learning*, pp. 1437–1449. PMLR, 2023.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in neural information processing systems*, 32, 2019a.

Aviral Kumar, Xue Bin Peng, and Sergey Levine. Reward-conditioned policies. *arXiv preprint arXiv:1912.13465*, 2019b.

M Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. *Advances in neural information processing systems*, 23, 2010.

Haeone Lee, Taywon Min, Junsu Kim, Sinjae Kang, Fangchen Liu, Lerrel Pinto, and Kimin Lee. Quality over quantity: Demonstration curation via influence functions for data-centric robot learning. *arXiv preprint arXiv:2603.09056*, 2026.

Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

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. *arXiv preprint arXiv:2509.09674*, 2025.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

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

Jijia Liu, Feng Gao, Bingwen Wei, Xinlei Chen, Qingmin Liao, Yi Wu, Chao Yu, and Yu Wang. What can rl bring to vla generalization? an empirical study. *arXiv preprint arXiv:2505.19789*, 2025.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Guanxing Lu, Wenkai Guo, Chubin Zhang, Yuheng Zhou, Haonan Jiang, Zifeng Gao, Yansong Tang, and Ziwei Wang. Vla-rl: Towards masterful and general robotic manipulation with scalable reinforcement learning. *arXiv preprint arXiv:2505.18719*, 2025.

Hao Luo, Yicheng Feng, Wanpeng Zhang, Sipeng Zheng, Ye Wang, Haoqi Yuan, Jiazheng Liu, Chaoyi Xu, Qin Jin, and Zongqing Lu. Being-h0: vision-language-action pretraining from large-scale human videos. *arXiv preprint arXiv:2507.15597*, 2025.

Hao Luo, Ye Wang, Wanpeng Zhang, Haoqi Yuan, Yicheng Feng, Haiweng Xu, Sipeng Zheng, and Zongqing Lu. Joint-aligned latent action: Towards scalable vla pretraining in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026a.

Hao Luo, Ye Wang, Wanpeng Zhang, Sipeng Zheng, Ziheng Xi, Chaoyi Xu, Haiweng Xu, Haoqi Yuan, Chi Zhang, Yiqing Wang, et al. Being-h0.5: Scaling human-centric robot learning for cross-embodiment generalization. *arXiv preprint arXiv:2601.12993*, 2026b.

Ajay Mandlekar, Jonathan Booher, Max Spero, Albert Tung, Anchit Gupta, Yuke Zhu, Animesh Garg, Silvio Savarese, and Li Fei-Fei. Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1048–1055. IEEE, 2019.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. *arXiv preprint arXiv:2406.02523*, 2024.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

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

Jan Peters and Stefan Schaal. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the 24th international conference on Machine learning*, pp. 745–750, 2007.

Jan Peters, Katharina Mulling, and Yasemin Altun. Relative entropy policy search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24, pp. 1607–1612, 2010.

Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In *International conference on machine learning*, pp. 5171–5180. PMLR, 2019.

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

Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pp. 1723–1736. PMLR, 2023.

Ye Wang, Sipeng Zheng, Hao Luo, Wanpeng Zhang, Haoqi Yuan, Chaoyi Xu, Haiweng Xu, Yicheng Feng, Mingyang Yu, Zhiyu Kang, et al. Rethinking visual-language-action model scaling: Alignment, mixture, and regularization. *arXiv preprint arXiv:2602.09722*, 2026.

Shaopeng Zhai, Qi Zhang, Tianyi Zhang, Fuxian Huang, Haoran Zhang, Ming Zhou, Shengzhe Zhang, Litao Liu, Sixu Lin, and Jiangmiao Pang. A vision-language-action-critic model for robotic real-world reinforcement learning. *arXiv preprint arXiv:2509.15937*, 2025.

Hongyin Zhang, Shuo Zhang, Junxi Jin, Qixin Zeng, Runze Li, and Donglin Wang. Robustvla: Robustness-aware reinforcement post-training for vision-language-action models. *arXiv preprint arXiv:2511.01331*, 2025a.

Wanpeng Zhang, Ye Wang, Hao Luo, Haoqi Yuan, Yicheng Feng, Sipeng Zheng, Qin Jin, and Zongqing Lu. Dig-flow: Discrepancy-guided flow matching for robust vla models. *arXiv preprint arXiv:2512.01715*, 2025b.

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

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023.

A Mathematical Analysis and Complete Derivations

The argument follows the two objects defined in the method: a distribution over candidate positions, and a distribution of action-loss coefficients over training examples. The former determines the evidence; the latter determines the supervised update. All logarithms are natural. Conditional laws are taken on standard Borel spaces. Every calculation fixes the relevant training state; none assumes that representations, sampling, or weights remain fixed throughout joint training. Here D_{KL} is relative entropy, H is the discrete entropy of the candidate index, and I is mutual information; their conditional versions always refer to the stated fixed condition.

The three subsections follow the argument of Section 3. Appendix A.1 proves Proposition 1, which explains what the score measures, and examines what enlarging the candidate list does. Appendix A.2 proves the three properties of the weight map used in Section 3.3: bounded coefficients, the soft-gate form, and continuity in the scores. Appendix A.3 derives the regression target induced by the weighted objective, which is why PTR is described as reweighted imitation.

A.1 Identification evidence

Section 3.2 reads the expected score as identification information minus the scorer's approximation error. This subsection proves that reading, with $\mathcal{E}_N(c)$ as in Eq. (3); the proposition is restated below.

Proposition 1 (Identification evidence). *If $\mathcal{E}_N(c) < \infty$, then*

$$\mathbb{E}[T \mid c, N] = I(J; \mathbf{Y} \mid c, N) - \mathcal{E}_N(c), \qquad 0 \le I(J; \mathbf{Y} \mid c, N) \le \log N. \tag{4}$$

No independence or common-distribution assumption on the candidates is needed. Under the independent reference model, the positive has law P and the other $N-1$ entries are independent draws from Q, independent also of the positive. If $P \ll Q$ and $D_{\mathrm{KL}}(P \,\|\, Q) < \infty$, then

$$\mathbb{E}[T \mid c, N] = D_{\mathrm{KL}}(P \,\|\, Q) - b_N - \mathcal{E}_N(c), \qquad b_N = D_{\mathrm{KL}}\big(\mathsf{M}_N \,\big\|\, Q^{\otimes N}\big) \ge 0, \tag{5}$$

where M_N is the law of the permuted list. If additionally $\chi^2(P\|Q) < \infty$, then

$$b_N \le \log\big(1 + \chi^2(P\|Q)/N\big).$$

Proof of Proposition 1. Fix (c, N) and suppress it within this proof. The random permutation makes J uniform, and $\widehat{p}_\psi(j \mid \mathbf{Y}) > 0$ for every position. The value $p^*(J \mid \mathbf{Y})$ is positive almost surely under the joint law of the matched index and the list. Conditional cross-entropy therefore splits as

$$\begin{aligned} \mathbb{E}[-\log \widehat{p}_\psi(J \mid \mathbf{Y})] &= \mathbb{E}[-\log p^*(J \mid \mathbf{Y})] + \mathbb{E}\log \frac{p^*(J \mid \mathbf{Y})}{\widehat{p}_\psi(J \mid \mathbf{Y})} \\ &= H(J \mid \mathbf{Y}) + \mathcal{E}_N(c). \end{aligned} \tag{A.1}$$

The entropy is finite because J takes N values, and the approximation error is finite by assumption. Using $T = \log N + \log \widehat{p}_\psi(J \mid \mathbf{Y})$ and $H(J) = \log N$ proves Eq. (4). Nonnegativity of conditional entropy and mutual information gives its stated range. No factorization of the list law is used.

Now impose the independent reference model. Let $r = dP/dQ$ and define the average density ratio on a list,

$$\overline{r}_N(\boldsymbol{y}) = \frac{1}{N}\sum_{j=1}^{N} r(y_j), \qquad \frac{d\mathsf{M}_N}{dQ^{\otimes N}}(\boldsymbol{y}) = \overline{r}_N(\boldsymbol{y}). \tag{A.2}$$

Indeed, conditional on $J = j$, only the jth coordinate differs from Q, so its list density relative to $Q^{\otimes N}$ is $r(y_j)$; averaging the N possible positions gives Eq. (A.2). Bayes' rule then yields

$$\begin{aligned} p^*(j \mid \boldsymbol{y}) &= \frac{r(y_j)}{\sum_{k=1}^{N} r(y_k)}, \\ T^* &= \log\big(N p^*(J \mid \mathbf{Y})\big) = \log r(Y_J) - \log \overline{r}_N(\mathbf{Y}). \end{aligned} \tag{A.3}$$

The denominator is positive M_N-almost surely, since $\mathsf{M}_N(\overline{r}_N = 0) = 0$. Unrestricted logits $\log r(y)$ reproduce this posterior, with $\log 0 = -\infty$ where necessary. This is a population reference, not an assertion that the bounded cosine model attains it.

For completeness, the expectations in this step are well defined. The convex function $\Phi(x) = x \log x$, with $\Phi(0) = 0$, satisfies $\Phi(x) \geq -1/e$. By convexity and $D_{\mathrm{KL}}(P \,\|\, Q) < \infty$,

$$\mathbb{E}_{Q^{\otimes N}} \Phi(\overline{r}_N) \leq \frac{1}{N} \sum_{j=1}^{N} \mathbb{E}_Q \Phi(r) = D_{\mathrm{KL}}(P \,\|\, Q) < \infty.$$

Together with the lower bound on Φ, this makes the corresponding log ratio integrable under M_N. The same argument makes $\log r$ integrable under P. Since Y_J has law P, taking expectations in Eq. (A.3) gives

$$\mathbb{E}[T^*] = D_{\mathrm{KL}}(P \,\|\, Q) - D_{\mathrm{KL}}\big(\mathsf{M}_N \,\big\|\, Q^{\otimes N}\big) = D_{\mathrm{KL}}(P \,\|\, Q) - b_N. \tag{A.4}$$

Directly, $\mathbb{E}[T^*] = \log N - H(J \mid \mathbf{Y}) = I(J; \mathbf{Y})$. Subtracting $\mathcal{E}_N(c)$ gives Eq. (5).

Finally, if $\chi^2(P\|Q) = \mathbb{E}_Q[(r-1)^2] < \infty$, independence under $Q^{\otimes N}$ and $\mathbb{E}_Q r = 1$ give

$$\mathbb{E}_{Q^{\otimes N}} \overline{r}_N^2 = \frac{N\mathbb{E}_Q r^2 + N(N-1)}{N^2} = 1 + \frac{\chi^2(P\|Q)}{N}.$$

Using the change of measure in Eq. (A.2), followed by Jensen's inequality for the logarithm, proves the remaining bound:

$$\begin{aligned} b_N &= \mathbb{E}_{\mathsf{M}_N} \log \overline{r}_N \leq \log \mathbb{E}_{\mathsf{M}_N} \overline{r}_N \\ &= \log \mathbb{E}_{Q^{\otimes N}} \overline{r}_N^2 = \log\left(1 + \frac{\chi^2(P\|Q)}{N}\right). \end{aligned} \tag{A.5}$$

□

The proof also shows what enlarging the list does. Within the independent reference model, for fixed P, Q, the Bayes expected score satisfies $0 \leq \mathbb{E}[T^*] \leq \min\{\log N, D_{\mathrm{KL}}(P \,\|\, Q)\}$. In fact b_N is nonincreasing in N, not merely bounded by a decreasing function. To see this, draw $N+1$ independent targets from Q and let $\overline{r}_N^{(-k)}$ be the mean of their density ratios with the kth entry omitted. Then

$$\overline{r}_{N+1} = \frac{1}{N+1} \sum_{k=1}^{N+1} \overline{r}_N^{(-k)},$$

$$b_{N+1} = \mathbb{E}\Phi(\overline{r}_{N+1}) \leq \frac{1}{N+1} \sum_{k=1}^{N+1} \mathbb{E}\Phi(\overline{r}_N^{(-k)}) = b_N.$$

The inequality is convexity; each omitted-entry mean has the same law as $\overline{r}_N$. Under finite $\chi^2(P\|Q)$, the bound in Eq. (A.5) also gives $b_N \to 0$ and $\mathbb{E}[T^*] \to D_{\mathrm{KL}}(P \,\|\, Q)$. These comparisons keep P, Q fixed. Enlarging a practical queue or changing hard negatives may change the reference law or its dependence structure, so this calculation alone certifies no practical sampler improvement.

For the fitted scorer, Eq. (A.1) explains the identification objective: at fixed representations and sampling, its excess over conditional entropy is exactly $\mathcal{E}_N(c)$. In joint training both terms may change. Likewise, $-\log \overline{r}_N(\mathbf{Y})$ is a per-list correction whose sign can vary; only its expectation is $-b_N \leq 0$. The ceiling $\log N$ does not imply that observed scores approach it. Candidates that share a context, camera, source, or encoding age can supply identification cues without isolating action information. The score is neither task return nor causal credit. The references for its contrastive construction are Gutmann & Hyvärinen (2010); Oord et al. (2018); the variational-information viewpoint is developed by Poole et al. (2019).

A.2 Supervision coefficients

Section 3.3 relies on three properties of the weight map: bounded coefficients, a soft-gate form, and continuity in the scores. They are proved in this order.

Fix one group of B examples and the parameters of the weight map. The bounds $0 < w_{\min} \leq 1 \leq w_{\max}$ ensure that every raw weight, including the fallback value one, lies in $[a, b] = [1 - \rho + \rho w_{\min},\ 1 - \rho + \rho w_{\max}]$. For a subset $\mathcal{S}$ with $\mu = |\mathcal{S}|/B$, write $x = \sum_{i \in \mathcal{S}} w_i$ and $y = \sum_{i \notin \mathcal{S}} w_i$.

Then $x \in [aB\mu, bB\mu]$ and $y \in [aB(1-\mu), bB(1-\mu)]$. The fraction $x/(x+y)$ increases with x and decreases with y, so its two extreme assignments give Eq. (7). The cases $\mu = 0, 1$ follow directly.

The coefficient relative to uniform weighting is

$$\begin{aligned} \Gamma_i = B\alpha_i = \frac{w_i}{B^{-1}\sum_{r=1}^{B} w_r}, \qquad & R^{-1} \le \Gamma_i \le R, \\ D_{\mathrm{KL}}(\alpha \,\|\, u_B) = \sum_i \alpha_i \log \Gamma_i \le \log R. & \end{aligned} \tag{A.6}$$

The inequalities follow from the same range $[a, b]$ for each weight and its mean. Thus every example remains present in the weighted empirical objective. This statement does not control its gradient norm or the modes of a trained network. Uniform coefficients are recovered exactly when all raw weights coincide. In particular, $\rho = 0$ removes the weighting path but keeps enabled auxiliary losses.

The same map is a soft gate on identification. For $i \in \mathcal{I}$, $\rho = 1$, and common mapping parameters, put $\ell_{0,i} = \log N_i - \beta \log w_{\max}$. Dividing the exponential in Eq. (6) by $w_{\max}$ gives $e^{T_i/\beta}/w_{\max} = \exp[-(\ell_i^{\mathrm{id}} - \ell_{0,i})/\beta]$. Clipping this ratio to $[w_{\min}/w_{\max}, 1]$ and using $\min\{1, e^{-x}\} = e^{-[x]_+}$ therefore yields

$$\frac{w_i}{w_{\max}} = \max\left\{ \frac{w_{\min}}{w_{\max}}, \exp\left(-\frac{[\ell_i^{\mathrm{id}} - \ell_{0,i}]_+}{\beta} \right) \right\}. \tag{A.7}$$

Dividing the entire weight vector by the same $w_{\max}$ leaves α unchanged; in that rescaled vector an invalid-future example has value $1/w_{\max}$, although its stored raw weight remains one. Eq. (A.7) itself applies only to valid-future examples. For $\rho < 1$, the affine mixture is applied after clipping. Further improvements in identification cannot increase a raw weight already on the upper plateau, but upper- and lower-plateau examples still have different weights. No frequency of plateau occupancy follows from this identity.

The coefficients are also continuous in the scores. Write $f(T) = 1 - \rho + \rho \operatorname{clip}_{[w_{\min}, w_{\max}]}(e^{T/\beta})$. On each clipped plateau, the derivative of $\log f$ vanishes. In the interior,

$$\frac{d}{dT} \log f(T) = \frac{\rho e^{T/\beta}}{\beta(1 - \rho + \rho e^{T/\beta})} \le L_f := \frac{\rho w_{\max}}{\beta(1 - \rho + \rho w_{\max})} \le \frac{1}{\beta}.$$

Continuity across the clipping points makes $\log f$ globally L_f-Lipschitz. If scores on the same valid examples change by at most δ while the group, validity flags, and mapping parameters stay fixed, then every raw-weight ratio lies in $[e^{-L_f\delta}, e^{L_f\delta}]$. Fallback ratios are one and satisfy the same bound. The ratio of the sums of weights lies in this interval as well, so

$$e^{-2L_f\delta} \le \frac{\alpha_i}{\alpha_i'} \le e^{2L_f\delta} \qquad \text{for every } i. \tag{A.8}$$

This is a fixed-problem sensitivity property, not a bound on training dynamics. A hard keep-or-drop rule lacks continuity at its threshold. For the unmixed map ($\rho = 1$), keeping $w_{\min} > 0$ while taking $\beta \to 0^+$ produces positive levels $w_{\min}$ and $w_{\max}$ on negative and positive scores, respectively; it does not produce hard deletion. The value at $T = 0$ remains one.

A.3 The induced regression objective

Section 3.3 describes PTR as reweighted imitation. The calculation below makes this precise: it gives the objective of one normalization group, its minimizer, the effect of the loss normalizer across devices, and a sufficient condition for a flow-matching reading with reweighted action chunks.

The preceding coefficients, rather than the raw weights alone, determine the supervised update. Let $\mathcal{F}$ contain all random variables used to build one normalization group, including its examples, candidate lists, offsets, and flow noise. Draw ι uniformly from its B indices, independently of $\mathcal{F}$, and set $\Gamma_\iota = B\alpha_\iota$. For any fixed regressor v,

$$\mathbb{E}_{\mathcal{F}}\left[\sum_{i=1}^{B} \alpha_i \ell_i(v) \right] = \mathbb{E}_{\mathcal{F},\iota}[\Gamma_\iota \ell_\iota(v)]. \tag{A.9}$$

Indeed, conditional on $\mathcal{F}$, the right side is $B^{-1}\sum_i B\alpha_i\ell_i(v)$, which is the left integrand. The identity requires no independence between the examples. When $B = 1$, the raw weight cancels exactly, so a distribution proportional to raw weights cannot generally represent the finite-group objective.

For the sampled example suppress ι, write $Z = (X_\tau, \tau, h)$, $X_\tau = \tau A + (1-\tau)\epsilon$, and $V = A - \epsilon$. Freeze the coefficients and representations. Assuming $\mathbb{E}[\Gamma\|V\|_2^2 \mid Z] < \infty$, positivity of Γ gives $\mathbb{E}[\Gamma \mid Z] > 0$. Completing the square proves

$$\begin{aligned} v_\Gamma^*(Z) &= \frac{\mathbb{E}[\Gamma V \mid Z]}{\mathbb{E}[\Gamma \mid Z]}, \\ \mathbb{E}[\Gamma\|v - V\|_2^2 \mid Z] &= \mathbb{E}[\Gamma\|v_\Gamma^* - V\|_2^2 \mid Z] + \mathbb{E}[\Gamma \mid Z]\|v - v_\Gamma^*\|_2^2. \end{aligned} \tag{A.10}$$

The cross term is zero because $\mathbb{E}[\Gamma(V - v_\Gamma^*) \mid Z] = 0$. Thus v_Γ^* is the unique pointwise minimizer for an unrestricted regressor on supervised coordinates. For a fixed embodiment mask and a positive scalar loss reduction determined by the conditioning variables, this calculation applies to the active coordinates; inactive coordinates need not have a unique minimizer. This qualification does not choose or change the base policy's masking convention.

The effective coefficients depend on how the loss is normalized across devices. Consider G devices with n examples each, $B_{\mathrm{g}} = Gn$, and device losses averaged. With an index uniform over all B_{g} examples, the two normalization choices give

$$\begin{aligned} \Gamma_i^{\mathrm{global}} &= \frac{B_{\mathrm{g}} w_i}{\sum_{i'=1}^{B_{\mathrm{g}}} w_{i'}}, \\ \Gamma_i^{\mathrm{local}} &= \frac{n w_i}{\sum_{i' \in \mathcal{G}_g} w_{i'}}, \quad i \in \mathcal{G}_g. \end{aligned} \tag{A.11}$$

For the local rule, averaging $\Gamma_i^{\mathrm{local}}\ell_i$ over the Gn indices gives $G^{-1}\sum_g \sum_{i\in\mathcal{G}_g} w_i\ell_i / \sum_{i'\in\mathcal{G}_g} w_{i'}$, the averaged device loss. The factor n, not B_{g}, is therefore required. More generally, when group losses have averaging probabilities π_g, first sample a group with law π_g and then an index uniformly inside that group; its coefficient is the group size times its within-group normalized weight. This also covers accumulation and unequal group sizes. Target gathering does not determine the loss normalizer, and the two cases in Eq. (A.11) are alternatives; neither is asserted for the reported runs.

Under an additional condition, the objective can be read as flow matching toward reweighted clean actions. Suppose the sampled context and clean action have law $p_D(h, A)$, and the flow time and Gaussian noise follow their standard independent sampling law given (h, A). A sufficient condition for a clean-endpoint interpretation is

$$\mathbb{E}[\Gamma \mid h, A, \tau, \epsilon] = m(h, A) \quad \text{for some positive function } m. \tag{A.12}$$

Because (Z, V) is measurable with respect to (h, A, τ, ϵ), the tower property gives $\mathbb{E}[\Gamma V \mid Z] = \mathbb{E}[mV \mid Z]$ and $\mathbb{E}[\Gamma \mid Z] = \mathbb{E}[m \mid Z]$. Tilting the joint law by $m(h, A)$ leaves the conditional time/noise law unchanged and replaces the action law at fixed h by

$$p_{\mathrm{w}}(A \mid h) = \frac{p_D(A \mid h)\, m(h, A)}{\mathbb{E}_{A\sim p_D(\cdot\mid h)} m(h, A)}. \tag{A.13}$$

Consequently Eq. (A.10) is precisely the conditional flow-matching regression target for this law with the same noise path (Lipman et al., 2022). The context marginal is also tilted by its conditional mean weight. This is a sufficient-condition interpretation of the frozen risk, not an assertion that reused action-head features satisfy Eq. (A.12) in the reported runs.

In general only Eq. (A.10) is asserted. If instead Γ is measurable with respect to Z alone, it cancels in the conditional ratio and leaves $v_\Gamma^*(Z) = \mathbb{E}[V \mid Z]$ unchanged. Noise dependence alone therefore does not decide whether the target changes. Likewise, a raw-weight population approximation needs concentration of the actual normalizer around the appropriate conditional mean; independent data samples are insufficient when they share a random candidate bank. These regression calculations describe the frozen action-loss branch. Finite networks, auxiliary gradients, and joint training do not inherit an exact endpoint-distribution or task-success guarantee from them.

B Implementation Specification

This appendix specifies one training step, the memory, target, and controller components, and the configuration used in the experiments. The equations give the mathematical role of each component without settling run-level choices that the source descriptions leave open.

B.1 One training step

Algorithm 1 in Section 3.4 summarizes one update; the subsections below specify its components.

B.2 Temporal context and gradient paths

For chunk i, let $U_i^{\mathrm{ctx}}, U_i^{\mathrm{act}} \in \mathbb{R}^{L\times d}$ be its context and action-channel features. The memory updater forms M slots by fusing features and normalizing each slot over the L positions:

$$\begin{aligned} u_{ik} &= \tanh\big(W_f[U_{ik}^{\mathrm{ctx}}; U_{ik}^{\mathrm{act}}]\big)\,, \\ \pi_{ikm} &= \frac{\exp((W_a u_{ik})_m/\vartheta)}{\sum_{r=1}^{L}\exp((W_a u_{ir})_m/\vartheta)}, \\ z_{im}^{\mathrm{next}} &= \mathrm{sg}\left(\sum_{k=1}^{L}\pi_{ikm}u_{ik}\right). \end{aligned} \tag{B.1}$$

Here k is a position within a chunk and m a memory slot, with $W_f \in \mathbb{R}^{d\times 2d}$ and $W_a \in \mathbb{R}^{M\times d}$. The columns of $\Pi_i = (\pi_{ikm}) \in \mathbb{R}^{L\times M}$ sum to one. The superscript "next" refers to the next processed chunk of the same trajectory segment, not example $i+1$ in a shuffled group. The initial memory is learned. With $\mathcal{H}_i$ the assignment entropy and $\mathcal{D}_i$ the Gram penalty,

$$\begin{aligned} \mathcal{H}_i &= -\frac{1}{M}\sum_{m=1}^{M}\sum_{k=1}^{L}\pi_{ikm}\log\pi_{ikm}, \quad \mathcal{D}_i = \|\Pi_i^\top\Pi_i - I_M\|_F^2, \\ \mathcal{L}_{\mathrm{mem}} &= \lambda_{\mathrm{ent}}\overline{\mathcal{H}} + \lambda_{\mathrm{div}}\overline{\mathcal{D}}, \end{aligned} \tag{B.2}$$

where bars average over processed chunks. The reported defaults are $M = 4$, $\vartheta = 1$, and $\lambda_{\mathrm{ent}} = \lambda_{\mathrm{div}} = 10^{-3}$. Minimizing the positive entropy term favors concentrated assignments; the Gram penalty discourages identical slot columns. Neither term is a Bayesian filtering objective.

Detaching the returned state blocks later losses through this recurrence. Updater-only parameters therefore receive no later task gradient through z_i^{next}; their explicit local losses are Eq. (B.2). The underlying policy features still receive ordinary action and scorer gradients. No reconstruction objective or alternate backpropagation-through-time rule is added here.

For any policy-side parameter block θ, detaching w_i gives the following exact gradient decomposition at one update:

$$\nabla_\theta\mathcal{L}_{\mathrm{total}} = \sum_i \alpha_i\nabla_\theta\ell_i^{\mathrm{act}} + \lambda_{\mathrm{id}}\nabla_\theta\mathcal{L}_{\mathrm{id}} + \lambda_{\mathrm{rank}}\nabla_\theta\mathcal{L}_{\mathrm{rank}} + \nabla_\theta\mathcal{L}_{\mathrm{mem}}. \tag{B.3}$$

The action-loss branch has zero gradient to query-head-only parameters through the weights. Detachment does not stop the shared representation, scorer, or future weights from changing across updates. At inference, the memory depends on past policy-side information, never on recorded future images.

B.3 Targets, offset selection, and candidate lists

The EMA target encoder is a copy of the vision parameters. At update ν,

$$\bar\theta_{\nu+1} = \mu_{\mathrm{ema}}\bar\theta_\nu + (1-\mu_{\mathrm{ema}})\theta_{\nu+1}^{\mathrm{vision}}, \qquad \mu_{\mathrm{ema}} = 0.999. \tag{B.4}$$

The reported target uses vision-layer-12 features. Targets are normalized, detached, and gathered without gradient; the queue holds 1024 targets with at most 64 queued alternatives per example. Excluding the current example's positive from the negative pool does not exclude identical values generated by other examples.

The valid offset indices for example i are

$$\mathcal{K}_i = \{k \in \{1, \dots, H\} : o_{t_i+kL} \text{ is recorded}\}.$$

The reported selection is a preset distribution supported on this set, dependent on target availability rather than observation/action content. No uniform probabilities are asserted when the source does not specify them. Candidates may combine within-batch, cross-GPU, queued, and harder same-task or similar-state targets. Proposition 1 applies to their actual joint identification problem; its independent-reference specialization applies only under the additional stated sampling assumptions.

In the ranking loss, permuting valid action-head features keeps h_i and y_i^+ fixed. A self-pair has zero logit difference, contributes the constant $\log 2$, and gives no net gradient if the same effective score is used in both terms. A nonself pair may also exchange context, noise, or embodiment information, so the loss rewards feature-channel sensitivity without excluding context cues.

B.4 The bounded controller

The controller adjusts the temperatures κ, β and the proportion η^{hard} of harder same-task or similar-state candidates in each list. At a controller event ν, let acc_ν be the measured valid-example identification accuracy and mrg_ν the measured matched-versus-largest-alternative logit margin. For example, with a fixed tie rule, these are the valid-example averages of $\mathbf{1}\{j_i = \arg\max_j s_\psi(c_i, y_{ij})\}$ and $s_\psi(c_i, y_i^+) - \max_{j \neq j_i} s_\psi(c_i, y_{ij})$. For either statistic x_ν, use

$$\bar{x}_\nu = \mu_{\text{ctrl}} \bar{x}_{\nu-1} + (1 - \mu_{\text{ctrl}}) x_\nu, \qquad \mu_{\text{ctrl}} = 0.98.$$

With their averages denoted by $\overline{\text{acc}}_\nu, \overline{\text{mrg}}_\nu$, the documented multiplier is

$$\gamma_\nu = \begin{cases} 1.01, & \overline{\text{acc}}_\nu < 0.05, \\ 0.995, & \overline{\text{acc}}_\nu > 0.35 \text{ and } \overline{\text{mrg}}_\nu > 0.10, \\ 1, & \text{otherwise}, \end{cases} \tag{B.5}$$

followed by

$$\begin{aligned} \kappa_{\nu+1} &= \text{clip}_{[0.03,0.20]}(\gamma_\nu \kappa_\nu), \\ \beta_{\nu+1} &= \text{clip}_{[0.5,3.0]}(\gamma_\nu \beta_\nu), \\ \eta^{\text{hard}}_{\nu+1} &= \tfrac{1}{2}\ \text{clip}_{[0,1]}\left(\frac{\overline{\text{acc}}_\nu - 0.10}{0.40}\right). \end{aligned} \tag{B.6}$$

The initial scales are $(\kappa, \beta) = (0.12, 1.5)$, and the logit clamp is $C = 20$. The reported identification warmup is 3000 steps. The weighting path is introduced through ρ increasing from zero to one. No exact interpolation schedule or controller-event frequency is inferred here. The controller changes sharpness and candidate difficulty heuristically; it is not a proof of calibration or nonsaturation. The mean score and valid-future fraction can also be monitored, but are not additional branches of Eq. (B.5).

B.5 Shared configuration and normalization

Table 5: Shared training configuration.

Setting	Value
Base policy / action dimension	Being-H0.5 / 200
Optimizer / learning rate / weight decay	AdamW / 10^{-4} / 0.01
Learning-rate schedule	cosine, 2000-step warmup
Global batch / devices	128 / 8 A100 GPUs
Chunk length / future-grid maximum	$L = 16$ / $H = 8$
Flow-time distribution / inference steps	Beta$(1.5, 1.0)$ / 4
$\lambda_{\text{id}}, \lambda_{\text{rank}}$	0.05, 0.25
Initial κ, β / logit clamp	0.12, 1.5 / 20
$w_{\min}, w_{\max}$	0.25, 4
Memory tokens / assignment temperature	4 / 1
$\lambda_{\text{ent}}, \lambda_{\text{div}}$	$10^{-3}, 10^{-3}$

The step budgets are 30k for specialist and 60k for pooled generalist training in the LIBERO results, 60k and 100k in the RoboCasa results, and 10k and 20k in the real-robot results. The policy uses embodiment-specific active slots in the shared 200-dimensional action space: 26 for G1, 31 for Adam-U, and 13 for FR3, with the remaining slots masked.

The reported flow-time law is $p_\tau = \mathrm{Beta}(1.5, 1)$ and $(\lambda_{\mathrm{id}}, \lambda_{\mathrm{rank}}) = (0.05, 0.25)$. The regression loss retains the base policy's active-coordinate mask and scalar reduction. The normalized group in Eq. (6) must be the one used by the loss implementation. Eq. (A.11) covers the global and local alternatives without asserting which one the reported runs used. These choices leave the identification result and the finite-group identity intact but matter for exact reproduction.

B.6 Controls and baselines

The pooled SFT+Memory control retains the memory path and uniform action weights while disabling the scorer. The full-scorer control retains the EMA targets, candidate bank, identification and ranking losses, memory path, and shared-feature gradients, fixing the action-loss weights to one.

Hard-PTR removes negative-score examples from its action objective and renormalizes the retained losses. Each run learns its own scorer. Oracle-Tag uses raw coefficients one for untagged trajectories and 0.25 for tagged trajectories, followed by self-normalization. Random reweighting assigns weights independently of the content of each chunk, and loss-based reweighting derives weights from the per-chunk action-regression loss. The MI-based, influence, and datamodel baselines are adaptations of the cited methods to the same VLA training setup rather than reproductions of the originals.

Relative to SFT, PTR training takes 11.8% more wall-clock time and 7.6% more peak memory. At inference only the memory path of SFT+Memory remains.

C Supplementary Results and Result Aggregation

This appendix describes the real-robot platforms, lists the per-task and per-suite results, and gives the aggregation formulas. Derived quantities are computed from the displayed counts or aggregate values and are not independent experimental observations.

C.1 Real-robot platforms and tasks

Table 6 lists the three platforms and Figure 7 shows them. All three share the 200-dimensional action space of the base policy, and each activates only the slots of its own motor groups (Appendix B.5). The twelve tasks are Pour Water, Handover, Clear Desk, Drawer Organization, Arrange Flower, and Wipe Board on Adam-U; Put & Close Box and Scan Package on G1; and Stack Bowl, Water Plant, Drawer Organization, and Wipe Board on FR3. Wipe Board and Drawer Organization appear on two embodiments with the same semantics but different kinematics and viewpoints (Figure 4, left).

Table 6: Real-robot platforms.

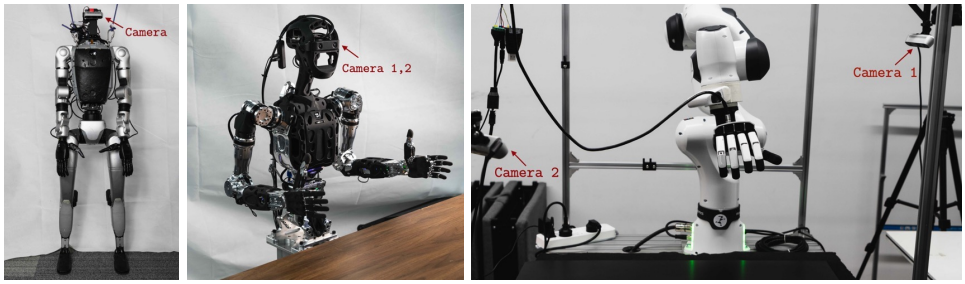
Platform	DoF	Morphology	Cameras	Rate
G1 + LinkerHand O6	26	bimanual, two hands	fixed egocentric D435	20 Hz
Adam-U	31	bimanual, head, waist	head-mounted ZED Mini	20 Hz
FR3 + Inspire Hand	13	one arm, one hand	two third-person D435	50 Hz

C.2 Per-task results and aggregation

The following definitions fix the meaning of the reported summaries; they create no new observations. The notation here is local to result aggregation: m indexes a method and k a task, n^+_{mk} counts successes, and n_k counts trials with the same trial count used for each method on a task.

The task-mean and pooled-trial success rates, in percent, are

$$\mathrm{SR}^{\mathrm{task}}_m = \frac{100}{K}\sum_{k=1}^{K}\frac{n^+_{mk}}{n_k}, \qquad \mathrm{SR}^{\mathrm{trial}}_m = 100\frac{\sum_{k=1}^{K} n^+_{mk}}{\sum_{k=1}^{K} n_k}. \tag{C.1}$$



(a) Unitree G1 & LinkerBot O6 (b) PND Adam-U (c) FR3 & Inspire Hand

Figure 7: The three real-robot platforms: (a) Unitree G1 with LinkerHand O6 hands, (b) PND Adam-U, and (c) FR3 with an Inspire hand.

Table 7: Per-task real-robot success rates (%), twenty trials per task and one trained policy per method in each regime. The last row gives the total successes out of 240 trials, and the better entry of each pair is in bold.

		Specialist		Pooled generalist	
Platform	**Task**	**SFT**	**PTR**	**SFT**	**PTR**
G1	Put & Close Box	55	**65**	45	**60**
G1	Scan Package	60	60	50	**60**
Adam-U	Pour Water	60	**70**	50	**65**
Adam-U	Handover	50	**65**	40	**55**
Adam-U	Clear Desk	**70**	65	60	**70**
Adam-U	Drawer Organization	60	60	45	**65**
Adam-U	Arrange Flower	80	**85**	70	**80**
Adam-U	Wipe Board	55	**70**	45	**60**
FR3	Stack Bowl	70	**75**	60	**70**
FR3	Water Plant	75	75	60	**70**
FR3	Drawer Organization	50	**60**	40	**55**
FR3	Wipe Board	45	**55**	35	**55**
Total successes / 240		146	**161**	120	**153**

They coincide when all n_k are equal, as in the reported twelve-task hardware study with twenty trials per task. For a fixed experimental configuration and n_{seed} training seeds, the mean and sample standard deviation of aggregate rates S_{mq} are

$$\overline{S}_m = \frac{1}{n_{\text{seed}}} \sum_{q=1}^{n_{\text{seed}}} S_{mq}, \qquad s_m = \sqrt{\frac{1}{n_{\text{seed}} - 1} \sum_{q=1}^{n_{\text{seed}}} (S_{mq} - \overline{S}_m)^2}. \tag{C.2}$$

The multi-seed simulation blocks use $n_{\text{seed}} = 3$; the separate curation and corruption-type blocks do not. Marginal standard deviations do not determine the standard deviation of a paired difference. Differences of displayed rounded means can differ in the last digit.

For the specific corruption-tag reference, the descriptive fraction of its improvement over SFT achieved by PTR is

$$\mathcal{R}_{\text{tag}} = \frac{\text{SR}_{\text{PTR}} - \text{SR}_{\text{SFT}}}{\text{SR}_{\text{OracleTag}} - \text{SR}_{\text{SFT}}}, \tag{C.3}$$

using the same experimental block and a positive denominator. From the rounded three-seed means it is approximately 0.894 on LIBERO and 0.895 on RoboCasa. This is not an optimality bound. No ratio uncertainty follows from marginal means and standard deviations alone.

The twelve positive hardware task differences give the descriptive count $12/12$. Under the additional null assumption of independent and equally likely positive and negative task differences, with no ties, the two-sided probability of all K signs agreeing is $2(1/2)^K$; for $K = 12$ it is 0.00048828125. This conditional calculation does not quantify variation across trained policies.

For a corruption type, the displayed performance drop is the clean score minus the corrupted score of the *same* method and experimental block. Comparing the PTR drop with the SFT drop is a descriptive

ratio only when the SFT drop is positive. Corruption types are not levels of a shared intensity variable. Similarly, raw clipped-weight/tag means describe an association before normalization; they do not by themselves measure the action-loss mass of tagged examples. Any such loss-mass claim must use the actual coefficients α_i or Γ_i under the implemented normalization.

C.3 Simulation breakdowns

Table 8 breaks the pooled LIBERO results down by suite; Table 2 gives the RoboCasa task categories. PTR improves on SFT in every LIBERO suite.

Table 8: Pooled generalist results on LIBERO by suite (success rate, %, three training seeds). The best entry of each column is in bold.

Method	Spatial	Object	Goal	Long	Average
SFT	97.1±0.3	96.7±0.4	95.9±0.4	94.4±0.6	96.0±0.3
SFT+Memory	97.3±0.3	96.9±0.4	96.1±0.4	94.9±0.5	96.3±0.3
PTR	**97.9**±0.3	**98.0**±0.3	**97.0**±0.3	**96.0**±0.5	**97.2**±0.3

C.4 Corruption types

Table 9 lists the study of Figure 3, which applies each corruption type separately. The protocols alter different aspects of supervision and are not points on a common corruption-intensity axis. In particular, truncation removes later coverage but leaves valid action-future pairs in a retained prefix. No universal transition-mismatch mechanism is inferred for all corruptions.

Table 9: Results by corruption type (success rate, %). Parentheses show the drop from the same method's clean result, in percentage points, computed from the displayed values; the better success rate of each pair is in bold.

Benchmark	Method	Clean	ANI	TT	LN	Combined
LIBERO	SFT	**98.3**	93.2 (5.1)	91.6 (6.7)	94.8 (3.5)	85.4 (12.9)
	PTR	97.8	**95.4** (2.4)	**94.0** (3.8)	**96.2** (1.6)	**91.0** (6.8)
RoboCasa	SFT	54.2	46.8 (7.4)	44.2 (10.0)	48.6 (5.6)	36.4 (17.8)
	PTR	**55.6**	**52.0** (3.6)	**50.4** (5.2)	**52.6** (3.0)	**45.8** (9.8)

Table 3b gives the mean raw clipped weights of untagged and tagged chunks on the Combined split, over the same three training seeds. In addition, untagged chunks make up 87.5% of the top-weight quartile on LIBERO and 91.3% on RoboCasa. These numbers describe the corruption tags and the raw weights before normalization. They are not used to infer a calibrated quality score, enrichment relative to a clean base rate, or the normalized action-loss mass, and they do not show that every lower-weight example is undesirable.

C.5 Design analyses

Table 4c gives the rank-loss comparison on RoboCasa Combined over three training seeds. The future-offset grid of Table 4b covers LH/f seconds at sampling frequency f: with $L = 16$ and $H = 8$, the maximum is 6.4 seconds at 20 Hz and 2.56 seconds at 50 Hz. These summaries do not identify a universal optimal horizon or establish that the rank loss removes context shortcuts.

C.6 Training dynamics

Figure 6 in Section 4.6 shows identification accuracy, mean score, and logit margin. Figures 8 and 9 add further logged quantities from the same runs: pooled training and specialist training on LIBERO, RoboCasa, Adam-U, FR3, and G1. These curves are descriptive observations of the coupled training system, not additional evaluations and not a decomposition of its learning progress.

The action loss decreases steadily in every setting (Figure 8a). The entropy of the memory-slot assignments decreases toward a low plateau, as the entropy term of Eq. (B.2) encourages (Figure 8b). As in Section 4.6, the representation, the candidate composition, and the temperatures all change

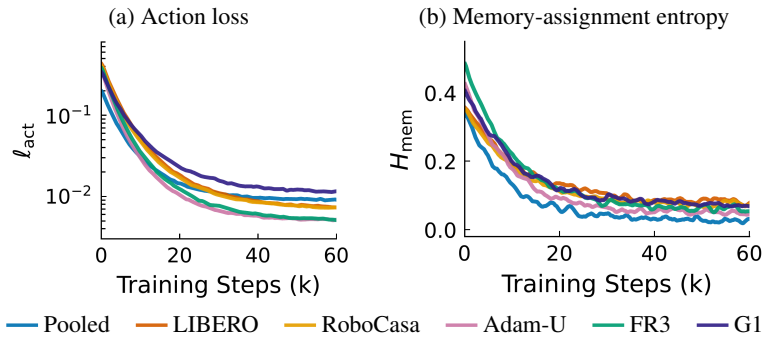


Figure 8: (a) Action loss ℓ^{act} and (b) entropy of the memory-slot assignments during training.

during training, so scores and margins from different settings are not measurements on a common identification problem, and a lower temperature alone can enlarge a margin.

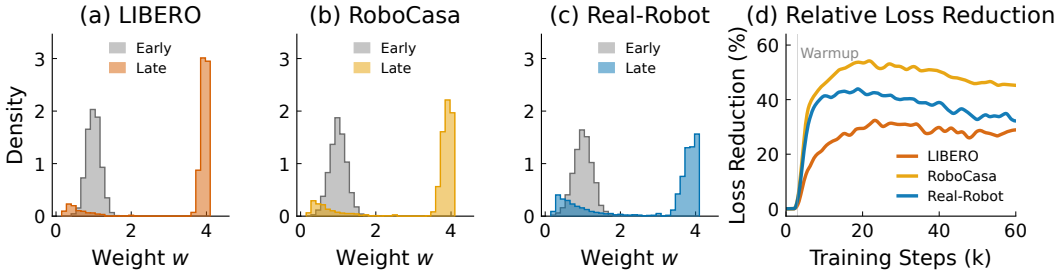


Figure 9: (a)–(c) Raw weights early and late in training for LIBERO, RoboCasa, and pooled real-robot training; (d) relative difference $(\ell_{\mathrm{SFT}} - \ell_{\mathrm{PTR}})/\ell_{\mathrm{SFT}}$ of the unweighted mean action loss of the two runs.

In the three settings of Figure 9, the raw weights stay close to one right after the warmup and spread across the clipped range later in training. These are raw weights before normalization; the coefficient bounds of Eq. (7) follow from the weight map regardless of the observed distribution. A lower action loss for PTR in panel (d) does not by itself mean better action prediction or higher success.

The controller changes both temperatures and the hard-candidate proportion according to Eqs. (B.5) and (B.6). These changes alter score magnitudes and logit margins even without any change in the underlying representation, which is another reason to read the curves above as descriptions rather than as measures of learning progress.

C.7 Qualitative rollouts

Figure 10 shows one successful rollout of the pooled PTR policy for each real-robot task, and Figure 4 (right) shows a rollout in a scene with unseen objects and background, flickering lights, and a transparent bottle. These examples illustrate behavior; they are not an additional evaluation and carry no claim about generalization.

D Assets and Licenses

LIBERO reports an MIT license for its code and CC BY 4.0 for its datasets. RoboCasa reports an MIT license for its code and CC BY 4.0 for its assets and datasets. The base policy follows the public Being-H0.5 release under the Apache-2.0 license. The real-robot demonstrations are internal teleoperation data without personally identifiable information.

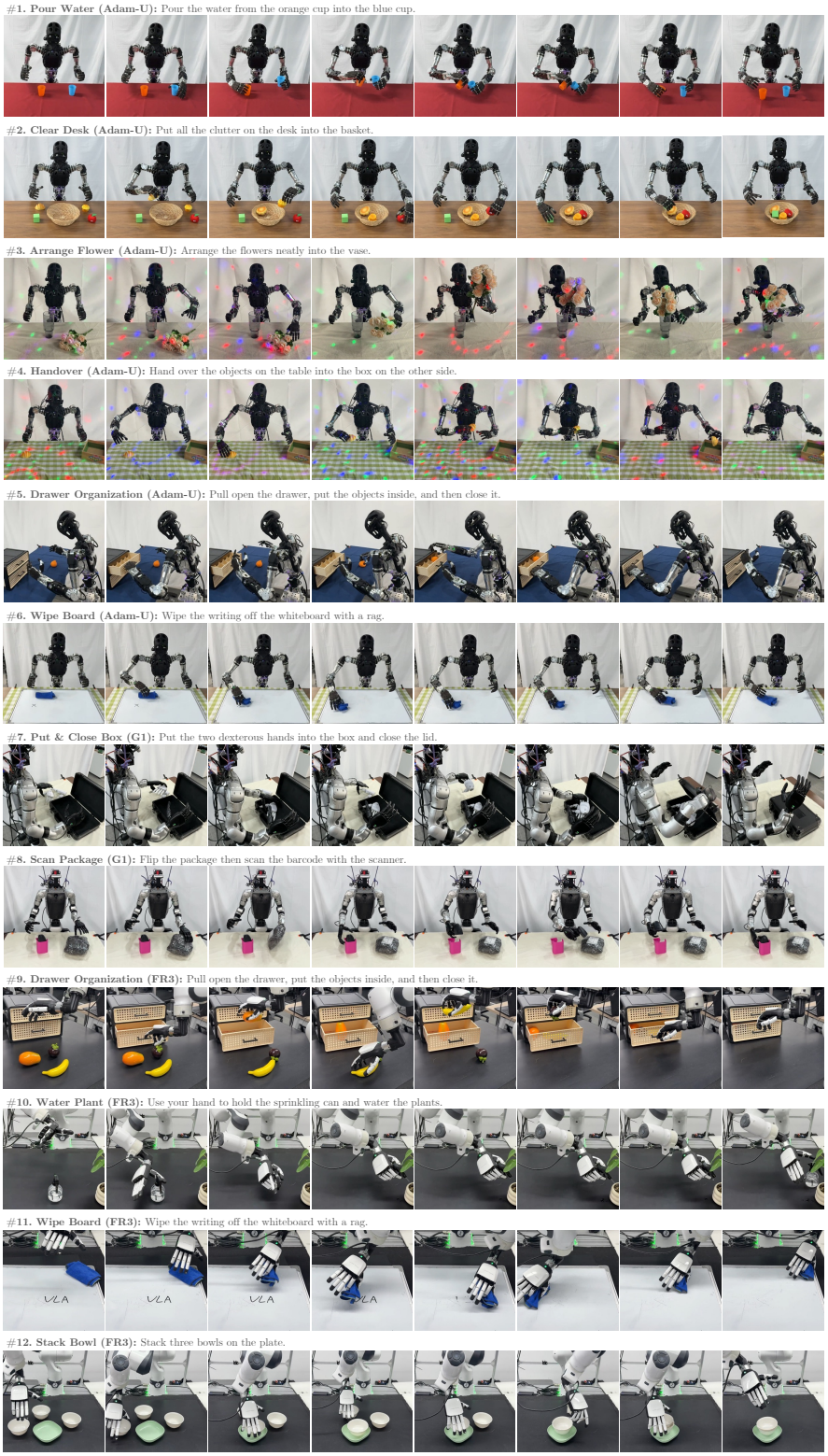


Figure 10: Rollouts of the pooled PTR policy on all twelve real-robot tasks, with eight keyframes from one successful trial per task.