VALUE AS MEMORY: ONLINE VALUE-GUIDED POST-TRAINING FOR GENERAL ROBOT POLICIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Real-world post-training of generalist robot policies faces two challenges: partial observability makes action selection ambiguous, while offline post-training methods delay feedback from deployment experience. We present Value as Memory (VaM), a framework that addresses these challenges through value caching and online real-world reinforcement learning (RL). Value caching conditions actions on recent value predictions as task-progress memory, while online updates enable timely policy improvement by combining offline demonstrations, off-policy interventions, and on-policy rollouts for sample-efficient adaptation. As a result, VaM transitions from offline RL warmup to online autonomous learning with sparse rewards. Across long-horizon tasks involving deformable objects and contact-rich manipulation, VaM yields a mean absolute success-rate gain of 29.4% with value caching, while achieving a 32.8% gain with online autonomous learning over matched offline training. The evaluated online schedule performs $8\times$ as many policy iterations under the same gradient-step budget, with more than $2\times$ lower recorded time per gradient step with concurrent data collection and training.

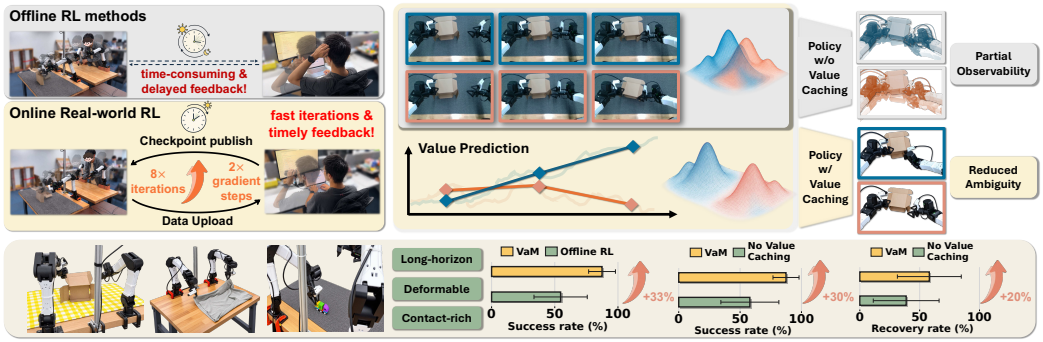


Figure 1: VaM uses online real-world RL and value caching to boost sample efficiency and mitigate partial observability in long-horizon, deformable-object, and contact-rich manipulation.

1 INTRODUCTION

Real-world reinforcement learning (RL) enables generalist robot policies to learn from physical interaction and improve on deployment tasks (Chen et al., 2025b; Physical Intelligence et al., 2025). Initialized from offline expert demonstrations, a policy aims to maximize cumulative task rewards (Luo et al., 2024; Frans et al., 2025) and optionally learns from human-intervention data during deployment (Luo et al., 2025a; Wang et al., 2026).

However, current real-world RL methods face limitations in data reuse and policy-update frequency. On-policy updates primarily use fresh rollouts, limiting the reuse of historical experience (Chen et al., 2025a). Iterated offline RL separates data collection from optimization, delaying feedback from newly encountered states (Physical Intelligence et al., 2025). Besides, partial observability poses another challenge. With limited observation histories, visually similar states may require different actions. Meanwhile, value functions provide compact task-progress estimates, but typically serve as filtering scores or optimization weights (Physical Intelligence et al., 2025; Frans et al.,

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.

2025). We therefore ask whether value estimates on observation histories can augment policy inputs to mitigate partial observability, and how an online RL system can exploit a learned value function to incorporate offline demonstrations, off-policy data, and on-policy rollouts for joint adaptation of the policy and value functions with high sample efficiency and rapid closed-loop feedback.

To this end, we introduce **Value as Memory (VaM)**, an offline-to-online RL system with two key features. **First**, we propose value caching, which extends value predictions beyond training supervision by supplying their recent history as an additional policy input. Alongside visual and proprioceptive histories, this cache indicates whether estimated progress improves, stalls, or declines, helping distinguish states requiring different actions. **Second**, VaM uses a hybrid training data scheme that combines offline demonstrations, off-policy trajectories, and current on-policy rollouts for sample-efficient joint adaptation of policy and value models. Our training regime progresses from offline RL warmup through human-assisted collection to autonomous learning with sparse rewards. Across real-world tasks involving deformable objects and contact-rich manipulations, VaM improves mean absolute success rate by 29.4% over its variant without value caching. It gains 32.8% in mean absolute success rate over its offline RL counterpart under matched training budgets. The evaluated online schedule performs $8\times$ as many policy iterations under the same gradient-step budget, with more than $2\times$ lower recorded time per gradient step with concurrent data collection and training. These results support value caching for mitigating partial observability and our online RL system with a hybrid training data scheme for improving sample efficiency and policy iteration frequency.

Our contributions are twofold:

1. We propose value caching to condition policies on recent task-progress estimates, with theoretical analysis and real-world ablations supporting its role in mitigating action ambiguity under partial observability.
2. We develop an online RL framework that reuses offline demonstrations, off-policy trajectories, and current on-policy rollouts for joint policy and value adaptation, improving sample efficiency and policy iteration frequency.

2 Preliminaries

2.1 Problem Formulation

We model iterative robot post-training as a Markov Decision Process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, $P(s_{t+1} \mid s_t, a_t)$ is the transition kernel, $r(s_t, a_t)$ is the reward function, and $\gamma \in [0, 1)$ is the discount factor. The policy $\pi_\theta(a_{t:t+H} \mid s_t)$ has parameters θ and predicts the action sequence $a_{t:t+H} = (a_t, \ldots, a_{t+H-1})$, where H is the prediction horizon. We let $\mathcal{D}_{\text{offline}} = \{\tau_{\text{offline},n}\}_{n=1}^{N}$ denote N successful expert demonstrations and $\pi_{\theta^{(0)}}$ the policy initialized from them. Post-training then runs for M rounds. At each round $m \in \{0, \ldots, M-1\}$, we deploy $\pi_{\theta^{(m)}}$ to collect rollout and human-intervention trajectories $\mathcal{D}^m_{\text{online}}$ and use the accumulated data to obtain $\pi_{\theta^{(m+1)}}$. With transition reward $r_t = r(s_t, a_t)$, our objective is to maximize the expected discounted cumulative reward $\mathbb{E}_{\pi_\theta}[\sum_{t=0}^{\infty} \gamma^t r_t]$.

2.2 Distributional Value Learning

We learn a distributional value function to estimate task progress and provide the scalar advantages used for policy improvement. Following previous literature on distributional value learning (Bellemare et al., 2017; Li et al., 2025; Physical Intelligence et al., 2025), the value network with parameters ψ predicts a categorical distribution $p_\psi(z_k \mid s)$ on $K \geq 2$ uniformly spaced atoms $\mathcal{Z} = \{z_k\}_{k=1}^{K} \subset [0, 1]$, where $z_k = (k-1)/(K-1)$ and the scalar value is $V_\psi(s) = \sum_{k=1}^{K} z_k p_\psi(z_k \mid s)$. We initialize the value expert with normalized demonstration progress, then adapt it through temporal-difference (TD) learning with sparse outcome rewards. For a successful demonstration $\tau_{\text{offline}} \in \mathcal{D}_{\text{offline}}$ with T timesteps, the initialization target q_t^{init} is a normalized Gaussian label over $\mathcal{Z}$ centered at t/T with variance $\sigma^2_{\text{init}} > 0$. The cross-entropy (CE) initialization objective is

$$\mathcal{L}_{\text{init}}(\psi) = \mathbb{E}_{\tau_{\text{offline}},t}\left[\text{CE}\big(q_t^{\text{init}}, p_\psi(\cdot \mid s_t)\big)\right], \quad \text{CE}(q, p) = -\sum_{k=1}^{K} q_k \log p_k. \tag{1}$$

For TD learning, only successful terminal transitions receive $+1$, with zero rewards otherwise. After each gradient step, the target parameters $\bar{\psi}$ follow the exponential moving average (EMA)

$\bar{\psi} \leftarrow (1-\beta)\bar{\psi} + \beta\psi$, with $\beta \in (0,1]$. We define $R_{t:t+H} = \sum_{h=0}^{H-1} \gamma^h r_{t+h}$ and a terminal indicator $d_t \in \{0,1\}$ for the H-step segment. The TD target q_t^{TD} moves each mass $p_{\bar{\psi}}(z_k \mid s_{t+H})$ to $R_{t:t+H} + \gamma^H(1-d_t)z_k$ before linear projection onto $\mathcal{Z}$ (detailed derivation in Appendix E). We train the value expert with

$$\mathcal{L}_{\mathrm{TD}}(\psi) = \mathbb{E}_t\left[\mathrm{CE}\big(q_t^{\mathrm{TD}}, p_\psi(\cdot \mid s_t)\big)\right]. \tag{2}$$

The TD-adapted value expert gives the scalar advantage estimate $\widehat{A}_t = R_{t:t+H} + \gamma^H(1-d_t)V_\psi(s_{t+H}) - V_\psi(s_t)$.

2.3 Advantage-conditioned Policy Learning

The action expert learns π_θ by flow matching, with an optional advantage condition to distinguish action quality. For a sample $(s_t, a_{t:t+H}) \sim \mathcal{D}_{\mathrm{BC}}$, where $\mathcal{D}_{\mathrm{BC}}$ is the behavior-cloning dataset and $a_{t:t+H}$ is an H-step action sequence, we draw noise $\epsilon \sim \mathcal{N}(0, I)$ with identity covariance I and flow timestep $\lambda \sim \mathcal{U}(0,1)$, and set $a_\lambda = (1-\lambda)\epsilon + \lambda a_{t:t+H}$ (Black et al., 2025). The velocity field v_θ shares the policy parameters θ and takes an optional condition c, where $c = \emptyset$ denotes behavior cloning. With the expectation taken over the sampled data, noise, and flow timestep, the flow-matching objective and its behavior-cloning instance are

$$\mathcal{L}_{\mathrm{FM}}(\theta; c) = \mathbb{E}\Big[\|v_\theta(a_\lambda, \lambda, s_t, c) - (a_{t:t+H} - \epsilon)\|_2^2\Big], \quad \mathcal{L}_{\mathrm{BC}}(\theta) = \mathcal{L}_{\mathrm{FM}}(\theta; \emptyset). \tag{3}$$

For advantage conditioning, CFGRL provides a policy-improvement guarantee when the optimality probability increases monotonically with the reference-policy advantage (Frans et al., 2025; Physical Intelligence et al., 2025). For online rollout data $\mathcal{D}_{\mathrm{online}}^m$ collected at round m, we set the threshold $\xi_m^\rho = \mathrm{Quantile}_\rho(\{\widehat{A}_i : (s_i, a_{i:i+H}) \in \mathcal{D}_{\mathrm{online}}^m\})$ with percentile parameter $\rho \in (0,1)$. The binary advantage condition is $o_t = \mathbb{I}[\widehat{A}_t \geq \xi_m^\rho] \in \{0,1\}$. During training, we randomly replace o_t with $\emptyset$, yielding $o_t \in \{\emptyset, 0, 1\}$, and minimize $\mathcal{L}_{\mathrm{FM}}(\theta; o_t)$. At inference, $o_t = 1$ requests high-advantage actions, and classifier-free guidance with weight $w \geq 1$ combines the conditional and unconditional velocity fields (Ho & Salimans, 2022).

3 Method

3.1 Method Overview

VaM uses a distributional value expert to supply recent progress predictions to the action expert through a value cache and advantage labels for policy post-training. Training proceeds through four stages: offline demonstration initialization, offline RL warmup, Online DAgger, and Learn from Experience (LfE). The value and action experts are updated in parallel during online RL. Figure 2 summarizes this training pipeline and the value caching architecture.

3.2 Value Caching and Soft Aliasing Analysis

3.2.1 Value Caching

The value cache gives the action expert explicit recent progress information without delaying its current action. At deployment round k, we fix the value snapshot $V_{\psi^{(k-1)}}$ obtained at the end of the preceding training round and use the expectation of its categorical return distribution as the scalar progress signal. At timestep t, we denote the action expert's visual-observation history by h_t^π, the value expert's visual-observation history by h_t^V, and the action expert's proprioceptive-observation history by s_t. The fixed value snapshot predicts $\widehat{V}_{\psi^{(k-1)},t} = V_{\psi^{(k-1)}}(h_t^V)$. Value prediction and action generation run asynchronously, so the value prediction for timestep t is unavailable when the action expert generates $a_{t:t+H}$. Therefore, we let $H_\pi \geq 1$ denote the action expert's observation horizon and define the value cache $c_t^{(k-1)}$ and the action expert's complete observation x_t^π as

$$\begin{aligned} c_t^{(k-1)} &= \left(\widehat{V}_{\psi^{(k-1)},t-H_\pi}, \ldots, \widehat{V}_{\psi^{(k-1)},t-1}\right), \\ x_t^\pi &= \left(h_t^\pi, s_t, c_t^{(k-1)}\right). \end{aligned} \tag{4}$$

The cache summarizes recent task progress and, when $H_\pi \geq 2$, whether progress is improving, stalled, or declining, helping distinguish visually similar states that require different optimal actions.

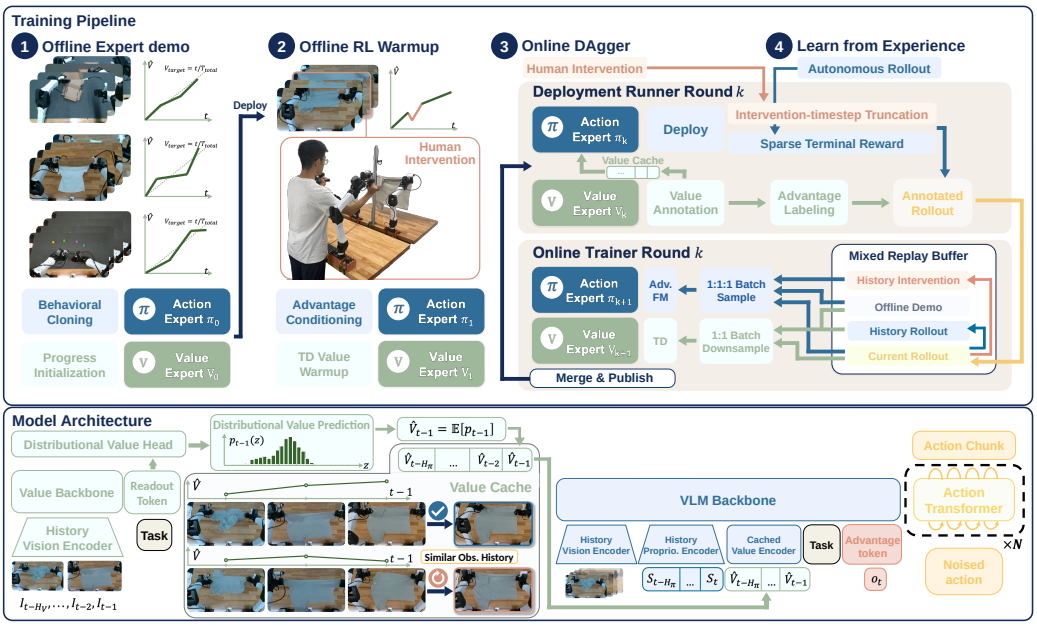


Figure 2: **VaM training pipeline and model architecture.** The top panel summarizes the four-stage offline-to-online training pipeline. The bottom panel shows how the distributional value expert turns previously computed value predictions into the value cache that conditions the action expert.

3.2.2 Why Value Caching Eases Soft Aliasing

Value caching targets soft aliasing, in which visually similar states require different high-advantage action distributions. We formalize how augmenting the policy input with the value cache can relax a representation-induced lower bound on policy prediction error. We consider two states indexed by $i \in \{1, 2\}$ within a fixed deployment round k. Equation 4 gives their value caches $c_i^{(k-1)}$ and complete action observations $x_i^\pi = (h_i^\pi, s_i, c_i^{(k-1)})$. We let z_i^{obs} denote the representation of the visual-proprioceptive components (h_i^π, s_i). For state i, μ_i denotes the distribution of training action chunks assigned the high-advantage label defined in Sec. 2.3. We measure the difference between action distributions with the Wasserstein distance W_2. We represent the complete action observation x_i^π as $z_i^\pi = (z_i^{\text{obs}}, c_i^{(k-1)})$. We define

$$\begin{aligned} \varepsilon_{\text{obs}} &= \|z_1^{\text{obs}} - z_2^{\text{obs}}\|_2, \\ \varepsilon_\pi &= \|z_1^\pi - z_2^\pi\|_2 = \sqrt{\varepsilon_{\text{obs}}^2 + \|c_1^{k-1} - c_2^{k-1}\|_2^2}, \\ \Delta_a &= W_2(\mu_1, \mu_2), \end{aligned} \tag{5}$$

Soft aliasing occurs when ε_{obs} is small but Δ_a is large: the policy sees similar inputs even though the desired actions differ. We assume that the policy is locally L-Lipschitz with respect to its input representation: for any two representations z and z' in the relevant neighborhood,

$$W_2(\pi_\theta(\cdot \mid z, o_t = 1), \pi_\theta(\cdot \mid z', o_t = 1)) \le L\|z - z'\|_2. \tag{6}$$

Theorem 1 (Local soft-aliasing bound)**.** *For representation $r \in \{\text{obs}, \pi\}$, z_i^r denotes corresponding input for state i and ε_r its pairwise distance in Eq. 5. Any locally L-Lipschitz policy must incur*

$$\max_{i \in \{1,2\}} W_2(\pi_\theta(\cdot \mid z_i^r, o_t = 1), \mu_i) \ge \frac{1}{2}(\Delta_a - L\varepsilon_r)_+, \tag{7}$$

where $(x)_+ = \max(x, 0)$. In particular, the lower bound is strictly positive whenever $\Delta_a > L\varepsilon_r$, so no locally L-Lipschitz policy can match both target action distributions exactly. If $L\varepsilon_{\text{obs}} < \Delta_a \le L\varepsilon_\pi$, an observation-only policy cannot match both target action distributions while satisfying the local Lipschitz condition, whereas exact matching at z_1^π and z_2^π does not violate this constraint.

The formal setup and proof are given in Appendices F and G. Because $\varepsilon_\pi \ge \varepsilon_{\text{obs}}$, the value cache can weaken the aliasing lower bound by making the two states easier to separate.

3.3 CONTINUAL OFFLINE-TO-ONLINE RL SYSTEM

Algorithm 1 in Appendix D summarizes the four-stage training loop, with data and objective details in Appendix E.

3.3.1 TRAINING OBJECTIVES

Offline demonstration initialization trains the value expert with the progress objective $\mathcal{L}_{\text{init}}$ in Eq. 1 and the action expert with the behavior-cloning objective $\mathcal{L}_{\text{BC}}$ in Eq. 3. From offline RL warmup onward, the value expert uses the TD objective in Eq. 2, and the action expert uses the advantage-conditioned objective $\mathcal{L}_{\text{FM}}(\theta; o_t)$ in Eq. 3. Algorithm 1 provides pseudocode for the complete offline-to-online RL system and denotes these operations by Init, BC, TD, and FM.

3.3.2 TRAINING STAGES

Offline demonstration initialization. Offline expert demonstrations $\mathcal{D}_{\text{demo}}$ initialize $V_{\psi^{(0)}}$ and $\pi_{\theta^{(0)}}$ using the initialization objectives in Section 3.3.1.

Offline RL warmup. The warmup adapts both experts to deployment-specific states before online RL begins. We deploy $(\pi_{\theta^{(0)}}, V_{\psi^{(0)}})$ to collect policy rollouts and human-intervention (HI) trajectories for training the value expert. Following the failed-prefix augmentation of GR-RL (Li et al., 2025), we construct prefixes from the rollout start to each human takeover. Each prefix ends with zero reward and $d_t = 1$, providing failure supervision for stuck or retry stages. The complete rollout retains its outcome reward and bootstraps across takeovers to preserve recovery information. Thanks to the distributional value model, we can represent a mixture of the prefix and complete-rollout TD targets for shared samples.

We temporally downsample the value-training trajectories to traverse them more times within a fixed update budget. We sample from $\mathcal{D}_{\text{demo}}$ and the processed warmup rollout data to update the value expert and obtain $V_{\psi_{\text{warm}}}$.

We then annotate warmup trajectories with advantages from $V_{\psi_{\text{warm}}}$ and train the action expert on $\mathcal{D}_{\text{demo}}$ and these annotated trajectories to obtain θ_{warm}.

Online DAgger. Online DAgger co-adapts both experts to the current deployment distribution while human corrections continue to supervise recovery behavior. With $\theta^{(k)}$ denoting round-k policy parameters, we initialize the online rounds from $(\pi_{\theta^{(0)}}, V_{\psi^{(0)}}) = (\pi_{\theta_{\text{warm}}}, V_{\psi_{\text{warm}}})$. At each round k, we deploy $\pi_{\theta^{(k-1)}}$ with human intervention enabled and annotate the current rollouts with $V_{\psi^{(k-1)}}$. We train the value expert on current rollouts and replay data comprising $\mathcal{D}_{\text{demo}}$, warmup data, and earlier online rollouts. We train the action expert on $\mathcal{D}_{\text{demo}}$, historical human-controlled segments, and the annotated current rollouts. The value and action experts are then updated synchronously, and the current trajectories are incorporated into the historical buffers for the next round.

Learn from Experience. LfE continues online improvement using sparse terminal rewards from autonomous rollouts, without collecting new human corrections. We downsample the complete rollouts for value training and keep the historical intervention buffer fixed to the interventions collected before LfE. Data mixing, annotation with $V_{\psi^{(k-1)}}$, and synchronous expert updates follow the Online DAgger procedure.

3.4 IMPLEMENTATION DETAILS

The action expert inherits the complete architecture and pretrained parameters of $\pi_{0.5}$, including its PaliGemma-2B visual-language backbone and separate 300M-parameter action-generation transformer (Black et al., 2025). The value expert repurposes the original $\pi_{0.5}$ 300M-parameter action-generation transformer as its visual-language backbone. Following Multi-Scale Embodied Memory (MEM) (Torne et al., 2026), the action and value experts use temporal-spatial attention to encode their respective image histories h_t^{π} and h_t^{V}, each ending at observation timestep t. The action expert encodes past value predictions and proprioceptive states with a gated recurrent unit (GRU) and appends its final hidden state after the vision tokens. To predict categorical task progress, we append a learnable readout token to the end of the value backbone's input sequence after temporal-spatial history processing. A linear head maps the readout token's output representation to K logits, and a softmax produces the categorical value distribution over the fixed support in Sec. 2. Specific values of key hyperparameters are provided in Appendix B.5.

4 EXPERIMENTS

Our experiments aim to answer the following questions:

RQ 1. How much does value caching improve policy performance in RL post-training?

RQ 2. Why does value caching improve policy performance?

RQ 3. How much does VaM's offline-to-online RL system enhance sample efficiency?

4.1 TASKS AND EVALUATION PROTOCOL

We consider three real-world manipulation tasks, illustrated in Fig. 3: *Box Assembly*, *Laundry Folding*, and *Magnetic Bus Assembly*. All three tasks are long-horizon manipulation workflows that cover contact-rich assembly and deformable-object manipulation. Appendix A.1 provides the detailed procedure and success criterion for each task.

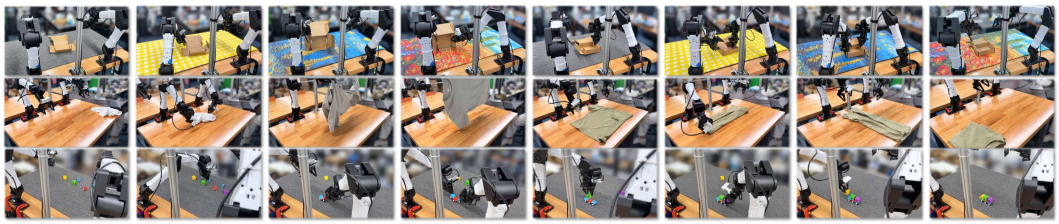


Figure 3: **Task illustrations for *Box Assembly* (top), *Laundry Folding* (middle), and *Magnetic Bus Assembly* (bottom).**

Evaluation protocol. We report success rate (SR), defined as the fraction of evaluation episodes completed successfully within the task-specific time limit. For the *Offline RL* and *Online RL* comparison, we additionally report experimental wall-clock time per gradient step. The real-robot setup and task-specific data-collection budgets are detailed in Appendix A and Table 3. Detailed hyperparameters for the evaluation protocol are provided in Appendix B.5.

4.2 HOW MUCH DOES VALUE CACHING IMPROVE POLICY PERFORMANCE IN RL POST-TRAINING?

To answer RQ 1, Fig. 4 compares VaM with ablations that remove value caching or the offline RL warmup, and with RECAP (Physical Intelligence et al., 2025). VaM and its ablation without value caching start from policies warmed up using their respective rollout and human-intervention trajectories. VaM without offline RL warmup and RECAP start from policies trained on $\mathcal{D}_{\text{demo}}$ and receive additional Online DAgger rounds to compensate for the other methods' warmup data.

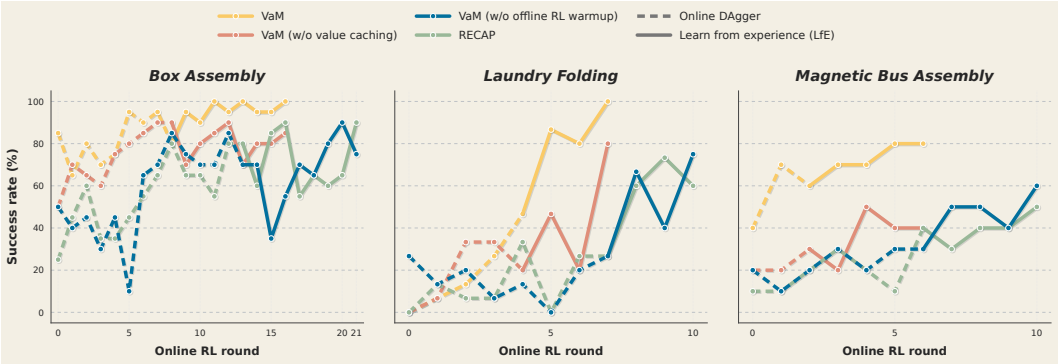


Figure 4: **Across *Box Assembly*, *Laundry Folding*, and *Magnetic Bus Assembly*, value caching yields a mean absolute success-rate gain of 29.4%.**

Across all three tasks, VaM exceeds the ablation without value caching by 29.4% in mean success rate over the final three checkpoints (Table 1).

The gap is most pronounced during LfE and persists after the policy stops receiving on-policy DAgger data. Averaging all available LfE checkpoints yields a consistent gap of 31.3% (full results in Appendix B.2). These observations support the use of cached task-progress estimates during LfE. VaM also exceeds its ablation without offline RL warmup on all three tasks, supporting the benefit of offline RL warmup for final performance.

Table 1: **Success rate over the final three online RL checkpoints.** Entries are mean ± sample standard deviation across the final three online RL checkpoints. By replacing on-policy value estimates with counterfactual estimates at inference, the bottom block demonstrates the importance of on-policy values for effective value caching.

Method	*Box Assembly* (%)	*Laundry Folding* (%)	*Magnetic Bus Assembly* (%)
VaM	**96.7** ±2.9	**88.9** ±10.2	**76.7** ±5.8
VaM (w/o value caching)	81.7 ±2.9	48.9 ±30.1	43.3 ±5.8
VaM (w/o offline RL warmup)	81.7 ±7.6	60.6 ±18.3	50.0 ±10.0
RECAP	71.7 ±16.1	64.4 ±7.7	43.3 ±5.8
VaM (demo-trained value)	80.0 ±13.2	71.1 ±20.4	40.0 ±10.0
VaM (post-warmup value)	85.0 ±13.2	68.9 ±10.2	50.0 ±10.0

Importance of on-policy value estimates. To probe the importance of accurate, on-policy estimates for value caching, we retain the final three LfE action experts and replace only their value experts with snapshots obtained after progress initialization on $\mathcal{D}_{\text{demo}}$ or offline RL warmup. These substitutions provide counterfactual value estimates while preserving the value caching mechanism across all three tasks. Both earlier value snapshots yield lower success rates than the original LfE value experts on all three tasks (Table 1), supporting the importance of value estimates adapted to the current policy for effective value caching. Appendix B.1 details the replacement protocol.

4.3 Why Does Value Caching Improve Policy Performance?

To answer RQ 2, Fig. 5 compares principal component analysis (PCA) projections of value-expert embeddings, value predictions, and robot behavior on *Box Assembly*. In the displayed cached rollouts, PCA projection trajectories are smoother and ordered by progress, with deviations at stuck states preceding corrective actions. Without caching, stages are more intermingled, with low or fluctuating values during failed execution. Appendix C extends these visualizations to *Laundry Folding* and *Magnetic Bus Assembly* and details a *Laundry Folding* comparison probing partial observability: we hold observations fixed and compare policy predictions using demo-trained and post-warmup value replacements following Sec. 4.2.

Recovery behavior during LfE. To assess corrective behavior, we conduct blind testing on all LfE rollouts from the four methods, where human annotators do not know which method produced each rollout.

They count events requiring recovery and successful recoveries in which the policy executes corrective actions to complete the affected stage. We define *Recovery Rate* as the ratio of successful recoveries to recovery-requiring events.

Table 2: **Recovery rate during LfE.**

Method	*Box Assembly* (%)	*Laundry Folding* (%)	*Magnetic Bus Assembly* (%)
VaM	**54.5**	**86.4**	**33.3**
VaM (w/o value caching)	50.0	58.8	7.4
VaM (w/o offline RL warmup)	25.3	72.7	27.3
RECAP	42.4	76.6	13.3

VaM exceeds its caching ablation in observed recovery rate on all three tasks (Table 2), with detailed experimental settings and event counts in Appendix B.3. The qualitative separation is consistent with the conditional soft-aliasing analysis in Sec. 3.2.2.

4.4 How Much Does VaM's Offline-to-Online RL System Enhance Sample Efficiency?

To answer RQ 3, we compare *Offline RL* and *Online RL*, initialized from the same VaM model after offline RL warmup, under matched gradient-step and data-collection budgets. Figures 6 and 7 report success rate over the final three checkpoints of each stage and experimental time per gradient step.

After LfE, VaM exceeds *Offline RL* on all three tasks by 32.8% on average, supporting online closed-loop policy updates under the evaluated budgets. VaM reduces mean experimental time per gradient step by 57.8% across tasks by running data collection, value training, and action training concurrently, while *Offline RL* performs them sequentially. For example, on *Box Assembly*, *Online RL* uses 16 iterations versus 2 for *Offline RL* under matched gradient-step budgets (detailed experimental settings in Appendix B.4).

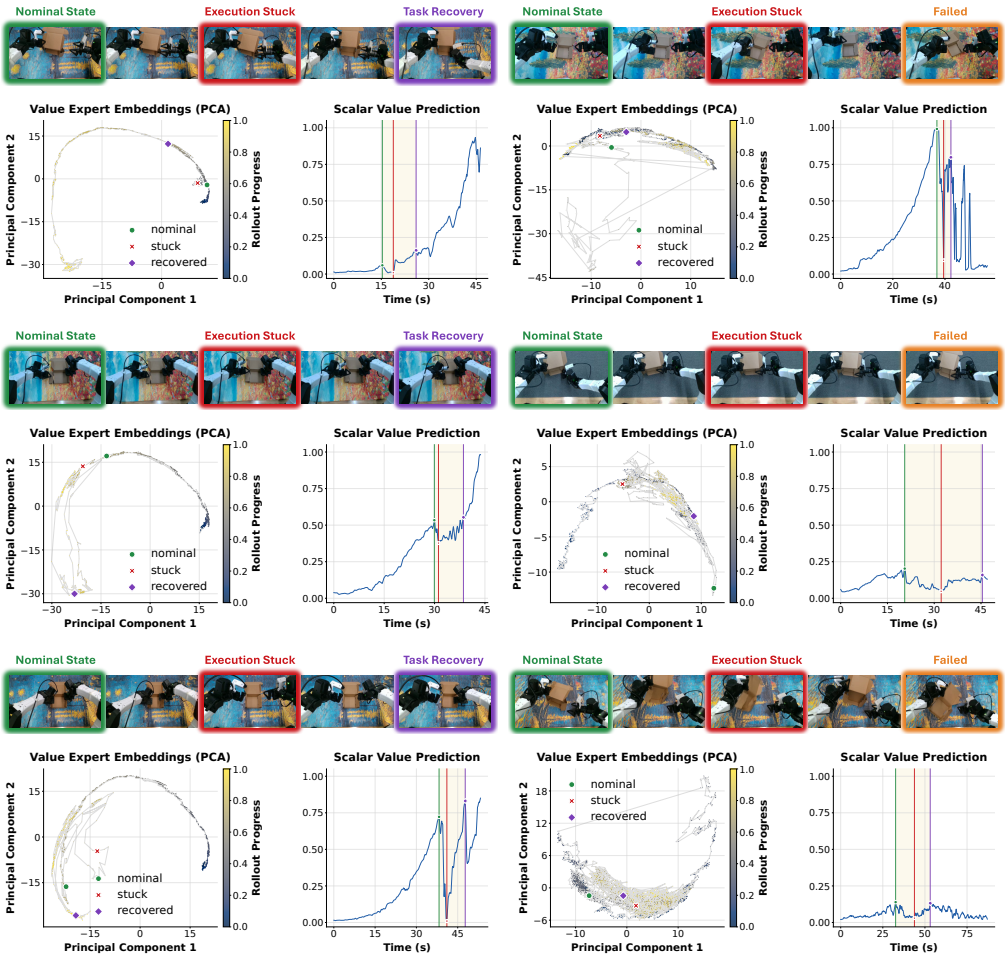


Figure 5: **Value caching supports progress-aware recovery: VaM (left) versus VaM (w/o value caching) (right) on *Box Assembly*.** We illustrate the eye-to-hand camera images, PCA embeddings of the value expert colored by rollout progress, and scalar value predictions.

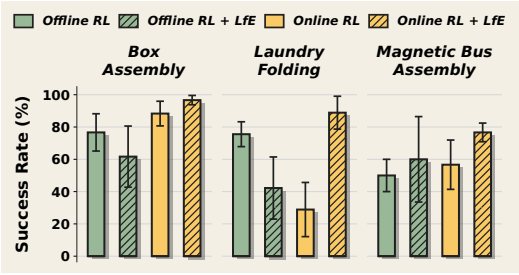


Figure 6: **Across all tasks, VaM yields an average absolute success-rate gain of** 32.8% **over matched *Offline RL* baselines after LfE.** Bars show mean $\pm$ sample standard deviation.

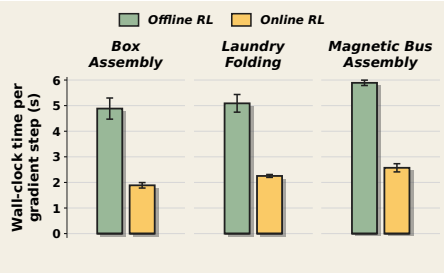


Figure 7: **Across all tasks, *Online RL* reduces experimental time per gradient step by** 57.8% **on average.** Bars show mean $\pm$ sample standard deviation.

5 Related Work

5.1 Post-training of Generalist Robot Policies

Policy-centric post-training adapts generalist robot policies through residual updates or RL objectives tailored to consistency, diffusion, and flow-based policies (Xiao et al., 2026; Chen et al., 2025b; Li et al., 2026; Chen et al., 2025a). Likelihood-based, hierarchical, and latent-space formulations

support optimization of generative policies (Ren et al., 2025; Lei et al., 2026; Wagenmaker et al., 2025; Li et al., 2025; Chen et al., 2025a), while world models reduce physical interaction by providing imagined or synthetic experience (Yang et al., 2026; Zhu et al., 2026; GigaBrain Team et al., 2026). To learn from a robot's own experience, some prior methods learn an additional value function to filter out suboptimal segments. General-purpose reward and critic models learn transferable evaluations from robot trajectories or vision-language supervision (Liang et al., 2026; Tan et al., 2026; Zhai et al., 2026), while advantage-oriented methods derive relative progress from temporal relations in demonstrations and trajectories (Liu et al., 2026; Yu et al., 2026a; Zhang et al., 2025). RECAP combines demonstrations, policy rollouts, and human interventions to estimate value and annotate data for advantage-conditioned policy extraction through CFGRL (Physical Intelligence et al., 2025; Frans et al., 2025). However, learning value estimates from fixed or periodically refreshed datasets can leave the critic stale as the deployment distribution changes between updates. In addition, when value predictions serve only as rewards, filtering scores, annotations, or optimization weights, the deployed policy does not receive them as inputs. Without this explicit task-progress information, visually similar observations can obscure different stages of execution, leaving the policy with partial observability of task progress. VaM instead continually updates value from online rollouts and human interventions and uses its predictions both as policy-learning supervision and an explicit policy input.

5.2 Human-in-the-loop Online Adaptation

Human operators intervene during deployment to correct uncertain or failing policy behavior (Luo et al., 2025b; Deng et al., 2026; Hu et al., 2026). Methods such as HG-DAgger and Sirius use interventions as corrective demonstrations for policy improvement (Kelly et al., 2019; Liu et al., 2023). Fleet-DAgger, SiriusFleet, and ARMADA scale intervention collection by allocating operators or requesting assistance only when necessary (Hoque et al., 2023; Liu et al., 2025; Yu et al., 2026b), while recent fleet-scale systems combine online robot rollouts with human corrections across multiple robots and tasks (Pan et al., 2026; Wang et al., 2026). VaM further uses intervention trajectories to update a value model that guides policy learning and conditions the policy input.

5.3 Offline-to-Online Reinforcement Learning

Offline-to-online RL stabilizes policy improvement by retaining or replaying offline data during online training (Ball et al., 2023; Peng et al., 2019; Nair et al., 2020; Li et al., 2025). RLPD mixes offline and online samples, Cal-QL calibrates an offline value function for online fine-tuning, and Warm-Start RL initializes online learning with replay data that preserves pretrained behavior (Ball et al., 2023; Nakamoto et al., 2023; Zhou et al., 2025). In real-world robotics, SERL and HIL-SERL combine behavior-cloning initialization with off-policy learning for physical manipulation (Luo et al., 2024; 2025a). VaM combines offline demonstrations, historical off-policy intervention data, and current on-policy experience for sample-efficient online adaptation.

6 Discussions and Conclusions

Though VaM demonstrates improvement over prior methods in sample efficiency and mitigates partial observability in generalist robot policies, it has several limitations that suggest directions for future work. First, value caching depends on a value function that closely tracks actual task progress, making offline RL warmup necessary for stable online training. The cached history consists of the value expert's own predictions, so its usefulness as task-progress memory depends on the quality of those predictions. Second, our real-world RL framework is currently limited to a single robot. Our experiments do not establish how the framework scales when multiple robots contribute data. Scaling VaM to robot fleets with shared experience is a promising direction for further improving sample efficiency.

In conclusion, we presented VaM, a framework that uses recent value predictions as task-progress memory and combines offline demonstrations, off-policy trajectories, and on-policy rollouts for online policy and value adaptation. Value caching addresses action ambiguity under partial observability, while the offline-to-online RL pipeline supports continued learning from deployment experience. Together, these components offer a practical approach to post-training generalist robot policies.

AI USE STATEMENT

We used generative artificial intelligence (AI) tools to polish writing, assist with research execution, and draft sections of the paper. We did not use these tools for literature retrieval or discovery, research ideation, synthetic dataset generation, or proving mathematical claims. The authors take responsibility for the final content of this paper.

REPRODUCIBILITY STATEMENT

All code will be released as open source and is currently being organized for release.

REFERENCES

Philip J. Ball, Laura Smith, Ilya Kostrikov, and Sergey Levine. Efficient online reinforcement learning with offline data. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 1577–1594. PMLR, 2023.

Marc G. Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pp. 449–458. PMLR, 2017.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael 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. *arXiv preprint arXiv:2504.16054*, 2025.

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, Xiang Li, Quanlu Zhang, Zhaofei Yu, Guoliang Fan, Tiejun Huang, Yu Wang, and Chao Yu. π_{RL}: Online RL fine-tuning for flow-based vision-language-action models. *arXiv preprint arXiv:2510.25889*, 2025a.

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. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025b. doi: 10.15607/RSS.2025.XXI.019.

Haoyuan Deng, Yudong Lin, Yuanjiang Xue, Haoyang Du, Qianzhun Wang, Boyang Zhou, Zhenyu Wu, and Ziwei Wang. E2hil: Entropy-guided sample selection for efficient real-world human-in-the-loop reinforcement learning. *IEEE Robotics and Automation Letters*, 11(7):8084–8091, 2026. doi: 10.1109/LRA.2026.3693577.

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

GigaBrain Team, Boyuan Wang, Bohan Li, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Jie Li, Jindi Lv, Jingyu Liu, Lv Feng, Mingming Yu, Peng Li, Qiuping Deng, Tianze Liu, Xinyu Zhou, Xinze Chen, Xiaofeng Wang, Yang Wang, Yifan Li, Yifei Nie, Yilong Li, Yukun Zhou, Yun Ye, Zhichao Liu, and Zheng Zhu. GigaBrain-0.5M*: a VLA that learns from world model-based reinforcement learning. *arXiv preprint arXiv:2602.12099*, 2026.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Ryan Hoque, Lawrence Yunliang Chen, Satvik Sharma, Karthik Dharmarajan, Brijen Thananjeyan, Pieter Abbeel, and Ken Goldberg. Fleet-DAgger: Interactive robot fleet learning with scalable human supervision. In *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 368–380. PMLR, 2023.

Zheyuan Hu, Robyn Wu, Naveen Enock, Jasmine Li, Riya Kadakia, Zackory Erickson, and Aviral Kumar. RaC: Robot learning for long-horizon tasks by scaling recovery and correction. *IEEE Transactions on Robotics*, 42:2843–2860, 2026.

Michael Kelly, Chelsea Sidrane, Katherine Driggs-Campbell, and Mykel J. Kochenderfer. HG-DAgger: Interactive imitation learning with human experts. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8077–8083. IEEE, 2019. doi: 10.1109/ICRA.2019.8793698.

Kun Lei, Huanyu Li, Dongjie Yu, Zhenyu Wei, Lingxiao Guo, Zhennan Jiang, Ziyu Wang, Shiyu Liang, and Huazhe Xu. Performant robotic manipulation with real-world reinforcement learning. *Science Robotics*, 11(116):eaed6267, 2026. doi: 10.1126/scirobotics.aed6267.

Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaohui Yang, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, Dehui Wang, Dingxiang Luo, Yuchen Fan, Youbang Sun, Jia Zeng, Jiangmiao Pang, Shanghang Zhang, Yu Wang, Yao Mu, Bowen Zhou, and Ning Ding. SimpleVLA-RL: Scaling VLA training via reinforcement learning. In *International Conference on Learning Representations*, 2026.

Yunfei Li, Xiao Ma, Jiafeng Xu, Yu Cui, Zhongren Cui, Zhigang Han, Liqun Huang, Tao Kong, Yuxiao Liu, Hao Niu, Wanli Peng, Jingchao Qiao, Zeyu Ren, Haixin Shi, Zhi Su, Jiawen Tian, Yuyang Xiao, Shenyu Zhang, Liwei Zheng, Hang Li, and Yonghui Wu. GR-RL: Going dexterous and precise for long-horizon robotic manipulation. *arXiv preprint arXiv:2512.01801*, 2025.

Anthony Liang, Yigit Korkmaz, Jiahui Zhang, Minyoung Hwang, Abrar Anwar, Sidhant Kaushik, Aditya Shah, Alex S. Huang, Luke Zettlemoyer, Dieter Fox, Yu Xiang, Anqi Li, Andreea Bobu, Abhishek Gupta, Stephen Tu, Erdem Biyik, and Jesse Zhang. Robometer: Scaling general-purpose robotic reward models via trajectory comparisons. In *Robotics: Science and Systems*, 2026.

Huihan Liu, Soroush Nasiriany, Lance Zhang, Zhiyao Bao, and Yuke Zhu. Robot learning on the job: Human-in-the-loop autonomy and learning during deployment. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.005.

Huihan Liu, Yu Zhang, Vaarij Betala, Evan Zhang, James Liu, Crystal Ding, and Yuke Zhu. Multi-task interactive robot fleet learning with visual world models. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 4286–4313. PMLR, 2025.

Zhihao Liu, Qiuyi Gu, Yitao Wang, Dongming Qiao, Yixian Zhang, Shuaihang Chen, Liangzhi Shi, Tianxing Zhou, Zefang Huang, Kang Chen, Zhen Guo, Quanlu Zhang, Jincheng Yu, Xiaodan Liang, Guoliang Fan, Yu Wang, Feng Gao, Xinlei Chen, and Chao Yu. STEAM: Self-supervised temporal ensemble advantage modeling for real-world robot learning. *arXiv preprint arXiv:2606.29834*, 2026.

Jianlan Luo, Zheyuan Hu, Charles Xu, You Liang Tan, Jacob Berg, Archit Sharma, Stefan Schaal, Chelsea Finn, Abhishek Gupta, and Sergey Levine. SERL: A software suite for sample-efficient robotic reinforcement learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 16961–16969. IEEE, 2024. doi: 10.1109/ICRA57147.2024.10610040.

Jianlan Luo, Charles Xu, Jeffrey Wu, and Sergey Levine. Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning. *Science Robotics*, 10(105):eads5033, 2025a. doi: 10.1126/scirobotics.ads5033.

Shengcheng Luo, Quanquan Peng, Jun Lv, Kaiwen Hong, Katherine Rose Driggs-Campbell, Cewu Lu, and Yong-Lu Li. Human-agent joint learning for efficient robot manipulation skill acquisition. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1370–1377. IEEE, 2025b. doi: 10.1109/ICRA55743.2025.11127637.

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

Mitsuhiko Nakamoto, Yuexiang Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-QL: Calibrated offline RL pre-training for efficient online fine-tuning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 62244–62269, 2023.

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

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.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, Danny Driess, Michael Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachy Groom, Hunter Hancock, Karol Hausman, Gashon Hussein, Brian Ichter, Szymon Jakubczak, Rowan Jen, Tim Jones, Ben Katz, Liyiming Ke, Chandra Kuchi, Marinda Lamb, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Yao Lu, Vishnu Mano, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Charvi Sharma, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Alex Swerdlow, James Tanner, Marcel Torne, Quan Vuong, Anna Walling, Haohuan Wang, Blake Williams, Sukwon Yoo, Lili Yu, Ury Zhilinsky, and Zhiyuan Zhou. $\pi^*_{0.6}$: a VLA that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025.

Allen Z. Ren, Justin Lidard, Lars Lien Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. In *International Conference on Learning Representations*, 2025.

Huajie Tan, Sixiang Chen, Yijie Xu, Zixiao Wang, Yuheng Ji, Cheng Chi, Yaoxu Lyu, Zhongxia Zhao, Xiansheng Chen, Peterson Co, Shaoxuan Xie, Guocai Yao, Pengwei Wang, Zhongyuan Wang, and Shanghang Zhang. Robo-dopamine: General process reward modeling for high-precision robotic manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Marcel Torne, Karl Pertsch, Homer Walke, Kyle Vedder, Suraj Nair, Brian Ichter, Allen Z. Ren, Haohuan Wang, Jiaming Tang, Kyle Stachowicz, Karan Dhabalia, Michael Equi, Quan Vuong, Jost Tobias Springenberg, Sergey Levine, Chelsea Finn, and Danny Driess. MEM: Multi-scale embodied memory for vision language action models. *arXiv preprint arXiv:2603.03596*, 2026.

Andrew Wagenmaker, Yunchu Zhang, Mitsuhiko Nakamoto, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 258–282. PMLR, 2025.

Yi Wang, Xinchen Li, Pengwei Xie, Pu Yang, Buqing Nie, Yunuo Cai, Qinglin Zhang, Chendi Qu, Jeffrey Wu, Jianheng Song, Xinlin Ren, Jingshun Huang, Mingjie Pan, Siyuan Feng, Zhi Chen, and Jianlan Luo. Learning while deploying: Fleet-scale reinforcement learning for generalist robot policies. *arXiv preprint arXiv:2605.00416*, 2026.

Wenli Xiao, Haotian Lin, Andy Peng, Haoru Xue, Tairan He, Yuqi Xie, Fengyuan Hu, Jimmy Wu, Zhengyi Luo, Linxi Jim 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.

Jiazhi Yang, Kunyang Lin, Jinwei Li, Wencong Zhang, Tianwei Lin, Longyan Wu, Zhizhong Su, Hao Zhao, Ya-Qin Zhang, Li Chen, Ping Luo, Xiangyu Yue, and Hongyang Li. RISE: Self-improving robot policy with compositional world model. In *Proceedings of Robotics: Science and Systems*, 2026.

Checheng Yu, Chonghao Sima, Gangcheng Jiang, Hai Zhang, Haoguang Mai, Hongyang Li, Huijie Wang, Jin Chen, Kaiyang Wu, Li Chen, Lirui Zhao, Modi Shi, Ping Luo, Qingwen Bu, Shijia Peng, Tianyu Li, and Yibo Yuan. χ_0: Resource-aware robust manipulation via taming distributional inconsistencies. *arXiv preprint arXiv:2602.09021*, 2026a.

Wenye Yu, Jun Lv, Zixi Ying, Yang Jin, Chuan Wen, and Cewu Lu. ARMADA: Autonomous online failure detection and human shared control empower scalable real-world deployment and adaptation. *IEEE Robotics and Automation Letters*, 2026b. doi: 10.1109/LRA.2026.3709635.

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. In *International Conference on Machine Learning*, 2026.

Jiahui Zhang, Yusen Luo, Abrar Anwar, Sumedh Anand Sontakke, Joseph J Lim, Jesse Thomason, Erdem Biyik, and Jesse Zhang. Rewind: Language-guided rewards teach robot policies without new demonstrations. In *Proceedings of the 9th Conference on Robot Learning*, volume 305, pp. 460–488, 2025.

Zhiyuan Zhou, Andy Peng, Qiyang Li, Sergey Levine, and Aviral Kumar. Efficient online reinforcement learning fine-tuning need not retain offline data. In *International Conference on Learning Representations*, volume 2025, pp. 32343–32368, 2025.

Fangqi Zhu, Zhengyang Yan, Zicong Hong, Quanxin Shou, Xiao Ma, and Song Guo. WMPO: World model-based policy optimization for vision-language-action models. In *International Conference on Learning Representations*, 2026.