DVCA: DENOISING-VARIANCE CREDIT ASSIGNMENT FOR FLOW-BASED ROBOT POLICY OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent Vision-Language-Action (VLA) models and World-Action Models (WAMs) use flow matching to model complex continuous action distributions. Online post-training can further improve these policies, but trajectory-level feedback offers limited guidance for prioritizing actions during training, limiting sample efficiency and performance gains. Process reward models and critics provide finer-grained feedback at the cost of additional training. To address this challenge, we introduce Denoising-Variance Credit Assignment (DVCA), which exploits the policy's denoising process for action-level credit assignment. Denoising variance, defined as the variance of clean-action estimates across denoising steps, serves as an intrinsic proxy for action uncertainty. Our analysis reveals that high denoising variance is concentrated in a small subset of actions associated with key phases of task execution. This observation motivates reweighting advantages or supervised losses to prioritize high-uncertainty actions. DVCA reuses intermediate predictions generated during online rollouts to estimate action uncertainty without additional training or inference. DVCA is a plug-and-play addition to existing post-training methods (e.g., GRPO and RL Token) across flow-based policies (e.g., $\pi_{0.5}$ and Fast-WAM). Simulation and real-world experiments demonstrate consistent gains in sample efficiency or task success rate over baselines.

1 INTRODUCTION

Vision-Language-Action (VLA) models (Black et al., 2025b;a; Kim et al., 2025) and World-Action Models (WAMs) (Bi et al., 2026; Kim et al., 2026; Yuan et al., 2026; Chen et al., 2026a) have been shown to be effective for a wide range of robotic manipulation tasks. Representative models in both families use flow matching for expressive modeling of complex continuous action distributions and efficient action-chunk generation (Black et al., 2025b;a; Yuan et al., 2026). This success is largely driven by behavior cloning, a simple and scalable approach that learns diverse manipulation skills through supervised fine-tuning (SFT) on large offline datasets of expert demonstrations. However, behavior cloning requires expert demonstrations, which are costly to collect, and cannot directly exploit experience collected during deployment. SFT policies may still perform suboptimally on some tasks, such as those requiring precise control, motivating further improvement using deployment experience.

Online post-training methods such as reinforcement learning (RL) offer a natural way to improve these policies. Some recent methods emphasize stable policy improvement in parallel simulation (Li et al., 2026a; Chen et al., 2026b), while others focus on sample efficiency in real world (Wagenmaker et al., 2025; Luo et al., 2025; Xu et al., 2026; Haarnoja et al., 2018). While effective, most of these methods rely on trajectory-level rewards, distributing the learning signal across all actions through shared advantages or temporal-difference (TD) backup. However, actions within a trajectory differ in their impact on task outcomes and their contribution to policy improvement. For example, action predictions around contact or precise alignment can be less stable than those during predictable motion. Failing to identify and prioritize these actions limits sample efficiency and performance gains.

Recent work on this problem has primarily focused on training auxiliary models, including process reward models (Tan et al., 2026; Chen et al., 2026c; Liu et al., 2026; Huang et al., 2026; Fang et al., 2026b) and critics (Fang et al., 2026a; Nair et al., 2021). Despite their benefits, these approaches

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.

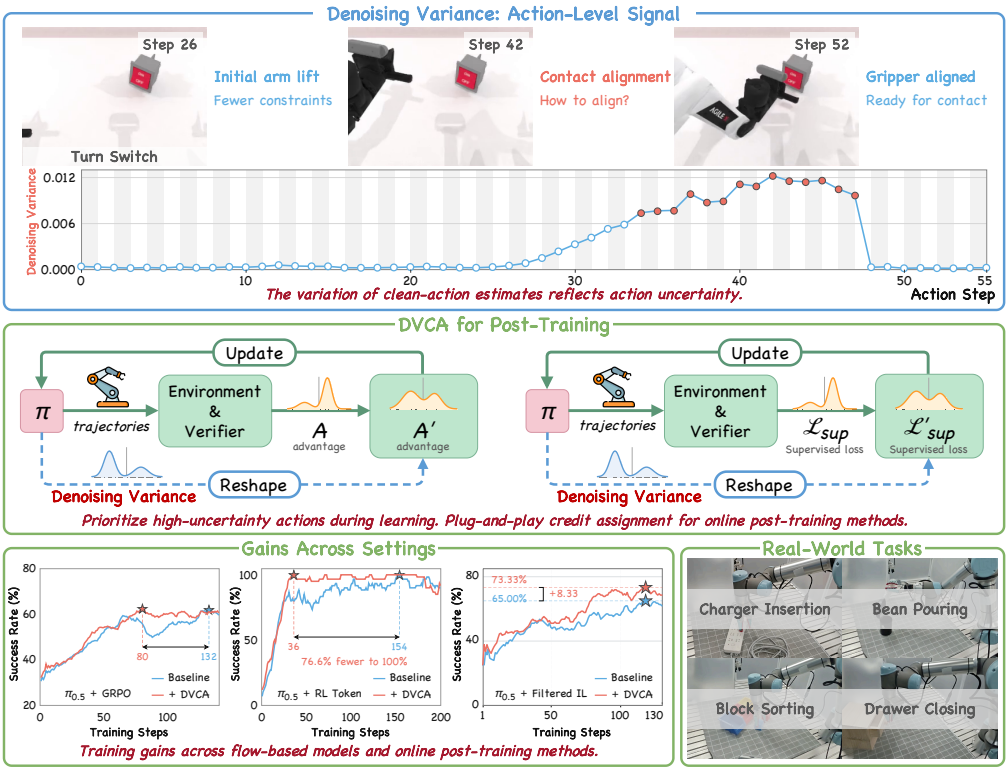

Figure 1: Overview of DVCA. Denoising variance captures uncertainty in action generation, and DVCA upweights more uncertain actions during training. It is a plug-and-play addition to different flow-based robot policies and online post-training methods, improving performance and sample efficiency.

incur additional training and inference overhead and may even require complex, carefully designed mechanisms. *Can we obtain a simple action-level signal for credit assignment within trajectories without additional training or inference, while enabling plug-and-play integration across existing online post-training methods for flow-based robot policies?*

To this end, we propose **D**enoising-**V**ariance **C**redit **A**ssignment (DVCA), which links action-level credit assignment to the denoising dynamics of flow-matching inference. As shown in Figure 1, for each action, we measure the dispersion of its clean-action estimates across denoising steps, termed denoising variance, as a simple indicator of the policy's uncertainty during inference. DVCA uses this intrinsic signal to focus training on actions with higher uncertainty. Specifically, DVCA reweights either advantages or supervised losses, depending on the underlying online post-training method. Following the same principle, DVCA can be extended to other post-training methods. Our approach follows a simple intuition: the policy should focus learning on more uncertain actions.

To sum up, our contributions are:

- We use denoising variance as a simple and intrinsic action-level uncertainty signal to identify and upweight potentially important actions for policy improvement.
- We instantiate DVCA within multiple online post-training methods for flow-based robot policies (e.g., GRPO and RL Token).
- Extensive experiments across simulated and real-world environments, flow-based robot policies, and online post-training methods demonstrate that DVCA consistently improves sample efficiency or task success rate.

2 Related Work

2.1 Generative Robot Policies

Generative robot policies learn conditional action distributions through iterative denoising (Chi et al., 2025; Ze et al., 2024). Vision-language pretraining supports language-conditioned control and cross-task generalization, forming the basis of flow-based Vision-Language-Action models (Black et al., 2025b;a; Zheng et al., 2026). World-Action Models further combine video prediction with action learning, incorporating environment dynamics into action generation (Bi et al., 2026; Yuan et al., 2026; Chen et al., 2026a; Ye et al., 2026; Cai et al., 2026; Li et al., 2026b).

2.2 Online Post-Training of Robot Policies

Online post-training improves SFT-initialized policies using interaction experience. One line of work optimizes task returns in parallel simulation through policy gradients and group-relative optimization (Li et al., 2026a; Chen et al., 2026b). Another reuses experience through off-policy actor–critic learning and builds on pretrained policies for sample-efficient adaptation (Luo et al., 2025; Wagenmaker et al., 2025; Xu et al., 2026). Online methods based on filtered imitation learning continually collect trajectories, retain successful data, and update the policy (Dong et al., 2026; Lu et al., 2025). These methods provide different optimization interfaces, within which DVCA adjusts the relative training weights of actions.

2.3 Fine-Grained Credit Assignment and Reweighting

Process reward and value models support fine-grained credit assignment by converting task progress or estimated advantages into local learning signals (Chen et al., 2026c; Huang et al., 2026; Nair et al., 2021; Fang et al., 2026a). Heuristic reweighting has also been used to prioritize critical robot actions (Peng et al., 2026). In autoregressive language models, token entropy measures uncertainty in the predicted token distribution and is used to shape token-level advantages (Wang et al., 2025; Le et al., 2026). In flow-based robot policies, denoising dynamics provide uncertainty signals for adaptive action execution and failure detection (Feng et al., 2026; Rao et al., 2026). DVCA connects denoising uncertainty to online post-training by converting action-level denoising variance into weights for advantages and supervised losses.

3 Method

As illustrated in Figures 2 and 4, DVCA uses denoising variance from flow-matching inference to focus training on more uncertain actions.

3.1 Preliminaries

Flow-Based Robot Policies. Let s_c denote the policy input at query c, comprising visual observations, proprioceptive state, and a language instruction. The policy generates an action chunk $\mathbf{a}_c \in \mathbb{R}^{H\times D}$. Indices c and h denote policy queries and within-chunk actions, respectively. We use $t = 1$ for noise and $t = 0$ for actions. Along $x_c^t = (1-t)\mathbf{a}_c + t\xi_c$, with $\xi_c \sim \mathcal{N}(0, I)$, the per-action flow-matching loss is

$$\ell_{c,h}^{\mathrm{FM}}(\theta) = \left\| v_\theta(x_c^t, t, s_c)_h - (\xi_{c,h} - a_{c,h}) \right\|_2^2 . \tag{1}$$

Averaging over actions, training data, time, and noise gives the flow-matching objective. At inference, numerical integration of the velocity field transforms Gaussian noise into an action chunk.

Group Relative Policy Optimization. GRPO estimates advantages by normalizing returns within each rollout group (Shao et al., 2024):

$$\widehat{A}_i = \frac{R_i - \overline{R}_g}{\sigma_g + \delta}. \tag{2}$$

Here, $\overline{R}_g$ and σ_g are the group return mean and standard deviation; $\delta > 0$ ensures numerical stability. Actions in the same trajectory share this advantage. Let $i(c)$ index the trajectory containing chunk

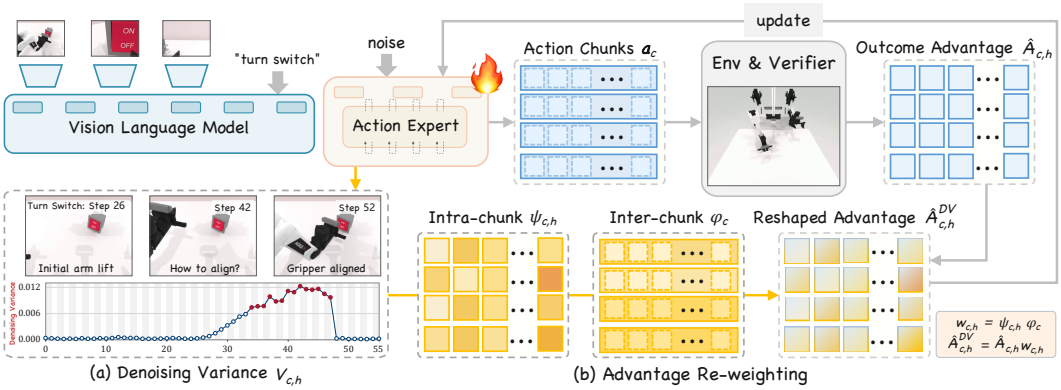

Figure 2: DVCA for GRPO consists of two steps: computing denoising variance during policy inference and using this signal to reweight advantages, assigning greater weight to more uncertain actions. The same reweighting is applied to each trajectory in the GRPO group.

c. The clipped objective is

$$\mathcal{J}_{\text{GRPO}} = \mathbb{E}_c\left[\min\left(\rho_c \widehat{A}_{i(c)}, \text{clip}(\rho_c, 1-\epsilon, 1+\epsilon)\widehat{A}_{i(c)}\right)\right]. \tag{3}$$

The likelihood ratio ρ_c is computed using stochastic denoising transitions (Chen et al., 2026b); ϵ is the clipping threshold. The expectation retains the baseline's reduction weights over valid chunks.

3.2 Denoising Variance

Each denoising step provides a clean-action estimate, whose variation reflects prediction stability. Following common practice, we measure action uncertainty by the variance of clean-action estimates over the final L denoising steps (Feng et al., 2026), indexed by $\mathcal{I}_L$. For noisy chunk $x_{c,j}$ at time t_j,

$$z_{c,j} = x_{c,j} - t_j v_\theta(x_{c,j}, t_j, s_c), \qquad V_{c,h} = \frac{1}{L}\sum_{j\in\mathcal{I}_L} \|z_{c,j,h} - \overline{z}_{c,h}\|_2^2, \tag{4}$$

where $\overline{z}_{c,h} = L^{-1}\sum_{j\in\mathcal{I}_L} z_{c,j,h}$ is the mean clean-action estimate. Larger $V_{c,h}$ indicates greater variation in the late-stage predictions of an action. This intrinsic signal reuses velocity predictions from the same inference pass, providing action-level uncertainty without additional training or inference.

3.3 Uncertainty-Guided Reweighting

DVCA assigns greater training weight to more uncertain actions, both within and across chunks. Following prior practice (Wang et al., 2026), we combine within-chunk and across-chunk weights. We use log-variance scores $q_{c,h} = \log(V_{c,h}+\delta)$ and chunk scores $\overline{q}_c = H^{-1}\sum_h q_{c,h}$. Tildes denote min–max normalization within each chunk for $q_{c,h}$ and across chunks for $\overline{q}_c$, with δ added to each range for numerical stability. The two weight levels are

$$\psi_{c,h} = \frac{\exp(\widetilde{q}_{c,h}/\tau_{\text{loc}})}{H^{-1}\sum_k \exp(\widetilde{q}_{c,k}/\tau_{\text{loc}})}, \qquad \varphi_c = \frac{\exp(\widetilde{\overline{q}}_c/\tau_{\text{grp}})}{\sum_{k\in\mathcal{C}} \nu_k \exp(\widetilde{\overline{q}}_k/\tau_{\text{grp}})}. \tag{5}$$

Within-chunk weighting emphasizes relatively uncertain actions, while across-chunk weighting prioritizes chunks with greater overall uncertainty. The temperatures $\tau_{\text{loc}}, \tau_{\text{grp}} > 0$ control the strength of within- and across-chunk weighting. Across chunks $\mathcal{C}$, coefficients $\nu_k \geq 0$, $\sum_k \nu_k = 1$, follow the baseline's contribution weights for GRPO and are uniform for supervised losses. Mean-one normalization makes the weights express relative training emphasis while preserving their average scale. The final weight is $w_{c,h} = \psi_{c,h}\varphi_c$. We gradually reduce the reweighting strength during training. Reweighting dropout removes reweighting from randomly selected action chunks by restoring their original unit weights.

3.3.1 Advantage Reweighting

As illustrated in Figure 2, DVCA redistributes the trajectory-level advantage across actions according to denoising uncertainty. We compute weights over the complete GRPO sampling group and set

$$\widehat{A}^{\mathrm{DV}}_{c,h} = w_{c,h}\widehat{A}_{i(c)}. \tag{6}$$

The superscript DV denotes DVCA reweighting. The trajectory-level advantage determines the update direction, while denoising variance determines how the update is distributed across actions. Positive weights preserve advantage signs and remain fixed during the corresponding policy updates. The weighted advantages redistribute action-level policy gradients within the clipped objective. Appendix A.4.2 gives the exact gradient formulation. Figure 3 summarizes the rollout and update procedure: denoising variance is recorded during inference and used to reweight advantages.

```
def rollout(policy, dataset, number_sample, max_steps):
    rollout_dataset = []
    for batch in dataset:
        batch = batch.repeat(number_sample)
        # Initialize parallel environments
        envs = env_process_pool.submit(batch.initialize)
        states = env_process_pool.submit(envs.setup)
        for t in range(max_steps):
            actions = policy.generate(states)
            # Reuse final action estimates from denoising
+           action_estimates = policy.denoising_predictions
+           variance = denoising_variance(action_estimates)
-           rollout_dataset.append((envs, states, actions))
+           rollout_dataset.append((envs, states, actions, variance))
            states, dones = env_process_pool.submit(envs.step, actions)
            # Remove completed tasks
            active = [(e,s) for e,s,d in zip(envs,states,dones) if not d]
            if not active:
                break
            envs, states = zip(*active)
    return rollout_dataset
# Collect rollouts and update the policy
rollout_dataset = rollout(policy, dataset, number_sample, max_steps)
rewards = outcome_rewards(rollout_dataset)
advantages = group_advantages(rewards)
+ weights = two_level_weights(rollout_dataset)
+ advantages = reweight(advantages, weights)
policy.update(rollout_dataset, advantages)
```

Figure 3: DVCA pseudocode for GRPO.

3.3.2 Supervised Loss Reweighting

For post-training methods with a behavior-cloning objective, DVCA weights per-action errors before aggregation:

$$\mathcal{L}^{\mathrm{DV}}_{\mathrm{sup}} = \mathbb{E}_c\left[\frac{1}{H}\sum_{h=1}^{H} w_{c,h}\ell^{\mathrm{sup}}_{c,h}\right]. \tag{7}$$

All averages use valid action positions. The same formulation applies to flow matching and action regression.

Flow-Matching Loss. As shown in Figure 4, DVCA reweights the per-action flow-matching loss in online post-training with filtered imitation learning. This simple procedure stores successful online rollouts in a replay buffer and updates the policy on sampled minibatches (Dong et al., 2026; Lu et al., 2025). We adopt this standard procedure to construct our Filter IL baseline. Executed actions provide supervision. We set $\ell^{\mathrm{sup}} = \ell^{\mathrm{FM}}$ and compute weights over valid chunks in each training batch.

Action Regression. In RL Token, a frozen VLA provides representations and reference action chunks, while a lightweight actor learns through Q-value maximization and BC regularization. With $\ell^{\mathrm{sup}}_{c,h} = \|a^{\theta_a}_{c,h} - a^{\mathrm{ref}}_{c,h}\|_2^2$, its objective is

$$\mathcal{L}^{\mathrm{DV}}_{\mathrm{actor}}(\theta_a) = -\lambda_Q\mathbb{E}_c\left[Q_{\theta_q}(s_c, \mathbf{a}^{\theta_a}_c)\right] + \lambda_{\mathrm{BC}}\mathcal{L}^{\mathrm{DV}}_{\mathrm{sup}}(\theta_a). \tag{8}$$

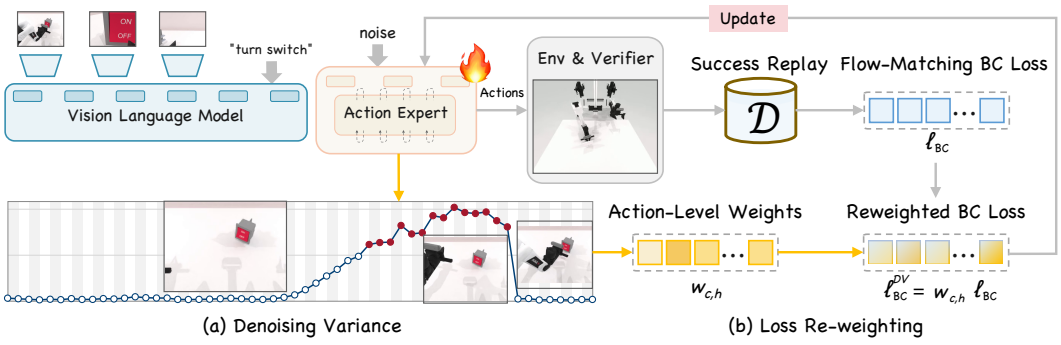

Figure 4: DVCA for methods incorporating behavior cloning (BC) consists of two steps: computing denoising variance during policy inference and using this signal to reweight the flow-matching loss, assigning greater weight to more uncertain actions.

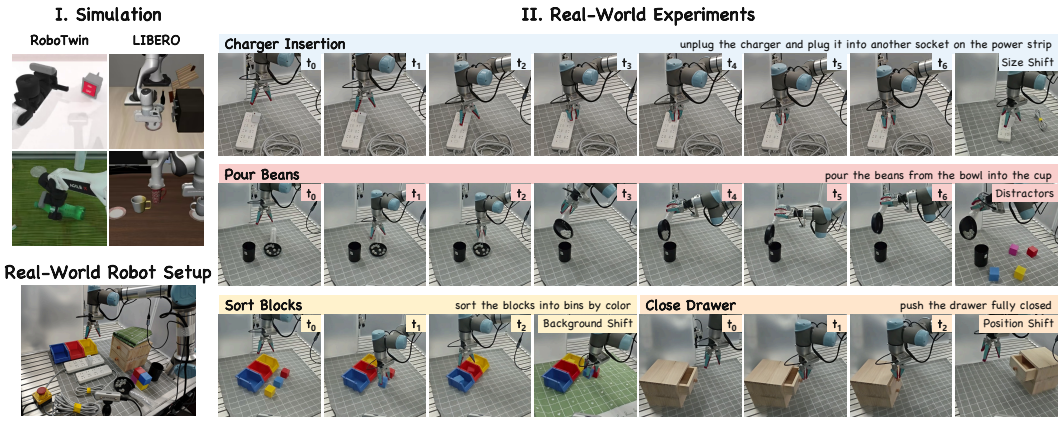

Figure 5: Experimental setup with RoboTwin, LIBERO, and four representative real-world tasks on UR5e. The numbered frames show successive stages of task execution. The final image for each task shows an out-of-distribution variation.

Both weight levels are computed over chunks from successful trajectories in the training batch; other chunks use $w_{c,h} = 1$. Denoising variance is obtained from the frozen VLA's reference-action inference and stored with each transition. Early in training, DVCA helps the actor imitate the frozen VLA more effectively.

4 Experiments

We evaluate DVCA in multiple simulated and real-world settings, cover a range of flow-based robot policies and online post-training methods.

4.1 Experimental Setup

Environments. We use the randomized setting (Rand.) of RoboTwin 2.0 (Chen et al., 2026d), selecting ten tasks with relatively low success rates across most pretrained policies. Full results are provided in Appendix A.1. Real-world experiments use a UR5e arm with a 1-DoF gripper and two Intel RealSense D435i cameras mounted at the head and wrist, covering four representative tasks. Figure 5 shows the task settings.

Baselines. We use pretrained π_0 (Black et al., 2025b), $\pi_{0.5}$ (Black et al., 2025a), and Fast-WAM (Yuan et al., 2026) as base policies, and instantiate DVCA within GRPO (Chen et al., 2026b), RL Token (Xu et al., 2026), and filtered imitation learning (Filter IL). We construct Filter IL following a standard online training procedure from prior work: the policy collects trajectories, adds successful rollouts to a buffer, and updates on samples from this buffer (Dong et al., 2026; Lu et al., 2025). Within the same Filter IL framework, we compare the heuristic reweighting of AtenA+ (Peng et al., 2026) and the progress-based reweighting of RA-BC from SARM (Chen et al., 2026c). For RA-BC, we use a pretrained RynnValue model (Huang et al., 2026) trained on data that include RoboTwin to estimate progress, and denote this implementation by RA-BC(R).

Evaluation. We report task success rates and measure training efficiency by the number of online rollouts required to reach peak success rates. RL Token training begins with reference-policy data collection and actor–critic warm-up before student-policy rollouts. For clarity, we count its rollouts from the point at which the student policy starts participating in data collection.

4.2 Simulation Results

We examine whether action-level reweighting improves learning across policy architectures and online post-training objectives. Tables 1–3 report results for GRPO, RL Token, and Filter IL, respectively.

Table 1: Task success rates (%) for GRPO on the randomized setting of RoboTwin 2.0. Results for all 10 tasks are provided in appendix.

Method	Turn Switch	Place Mouse Pad	Move Stapler Pad	Move Can Pot	···	**Avg.** (10)
X-VLA	66.2	71.7	73.4	87.6		67.7
Motus	77.9	68.0	84.4	72.5		83.3
π_0	42.7	20.4	24.6	48.3		37.1
+ GRPO	85.5	71.0	59.9	84.7		81.0
w/ DVCA	93.2	72.6	65.0	87.8		85.2
Δ	+7.7	+1.6	+5.1	+3.1		+4.2
$\pi_{0.5}$	54.9	39.0	42.8	55.7		64.7
+ GRPO	78.4	61.8	86.5	91.0		86.6
w/ DVCA	82.5	65.5	89.1	94.1		90.5
Δ	+4.1	+3.7	+2.6	+3.1		+3.9
Fast-WAM	59.1	89.7	64.2	88.4		86.7
+ GRPO	82.3	97.6	89.6	94.7		94.6
w/ DVCA	87.1	98.2	92.8	98.9		96.9
Δ	+4.8	+0.6	+3.2	+4.2		+2.3

GRPO. As shown in Table 1, DVCA consistently increases average success rates across all three policies. Figure 8 further shows GRPO learning curves with and without DVCA, demonstrating improved sample efficiency.

RL Token. RL Token eventually achieves high task success rates, and DVCA primarily improves sample efficiency. Table 2 and Figure 8 show that DVCA substantially reduces the number of samples required to reach the respective peak success rates. This improvement is obtained by reweighting the actor's behavior-cloning term, helping the actor more effectively match the teacher policy early in training. As training progresses, annealing gradually restores unit weights.

Filter IL. In Table 3 and Figure 8, DVCA consistently improves average success rates within the Filter IL framework and outperforms AttenA+ and RA-BC(R). AttenA+ uses end-effector velocity as a reweighting signal that reflects motion characteristics, and its effectiveness depends on how well the heuristic identifies actions that benefit from greater training emphasis. Progress models provide

Table 2: Results of RL Token training across flow-based robot policies on RoboTwin 2.0. SR reports peak success rate (%); Samples reports the number of rollouts required to reach the peak.

Method	Turn Switch		Place Mouse Pad		Move Stapler Pad		Move Can Pot		···	**Avg.** (10)	
	SR	Samples	SR	Samples	SR	Samples	SR	Samples		SR	Samples
π_0	42.7	–	20.4	–	24.6	–	48.3	–		37.1	–
+ RL Token	99.2	636	97.3	696	98.6	528	97.1	792		94.5	771
w/ DVCA	99.3	248	98.1	164	99.7	380	98.5	468		95.6	471
Δ	+0.1	-388	+0.8	-532	+1.1	-148	+1.4	-324		+1.1	-300
$\pi_{0.5}$	54.9	–	39.0	–	42.8	–	55.7	–		64.7	–
+ RL Token	95.2	536	97.8	616	94.6	624	98.2	448		94.2	674
w/ DVCA	97.4	208	98.5	144	96.0	268	99.1	208		95.7	439
Δ	+2.2	-328	+0.7	-472	+1.4	-356	+0.9	-240		+1.5	-236
Fast-WAM	59.1	–	89.7	–	64.2	–	88.4	–		86.7	–
+ RL Token	96.1	848	98.8	584	98.3	712	98.6	776		98.3	680
w/ DVCA	96.1	344	99.0	388	99.7	452	99.4	296		98.9	387
Δ	+0.0	-504	+0.2	-196	+1.4	-260	+0.8	-480		+0.6	-293

Table 3: Results of Filter IL training across flow-based robot policies on RoboTwin 2.0.

Method	Turn Switch	Place Mouse Pad	Move Stapler Pad	Move Can Pot	···	**Avg.** (10)
π_0	42.7	20.4	24.6	48.3		37.1
+ Filter IL	67.1	86.0	53.6	79.2		75.7
w/ RA-BC(R)	68.4	88.1	52.9	81.0		77.3
w/ AttenA+	68.4	85.7	56.9	80.2		78.0
w/ DVCA	**76.4**	**89.5**	**58.4**	**81.8**		**80.6**
$\pi_{0.5}$	54.9	39.0	42.8	55.7		64.7
+ Filter IL	87.4	65.0	81.5	69.6		85.2
w/ RA-BC(R)	87.9	60.5	82.0	69.9		85.7
w/ AttenA+	88.0	67.7	81.9	71.6		86.3
w/ DVCA	**89.3**	**73.3**	**82.9**	**72.4**		**89.7**
Fast-WAM	59.1	89.7	64.2	88.4		86.7
+ Filter IL	77.4	95.3	91.8	94.7		94.4
w/ RA-BC(R)	78.6	96.0	91.4	96.9		94.8
w/ AttenA+	80.0	95.2	92.3	95.1		95.0
w/ DVCA	**80.1**	**96.5**	**92.8**	**97.9**		**96.2**

another chunk-level signal, but are not always effective. We observe cases where predicted progress disagrees with actual task progress, as illustrated in Appendix A.5.2. Additional experimental results are reported in Appendix A.1.

4.3 Real-World Results

We next examine whether these gains extend to real-world manipulation under limited interaction budgets. Figure 6 reports final success rates on four real-world tasks. DVCA consistently improves performance across all combinations of $\pi_{0.5}$ and Fast-WAM with RL Token and Filter IL.

Figure 7 further shows online training dynamics. On most tasks, DVCA reaches high success rates with fewer interactions while also improving final performance.

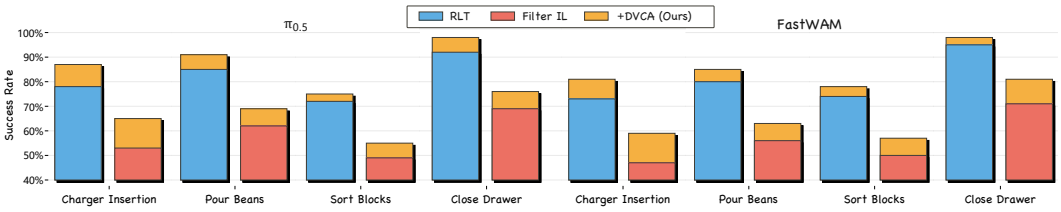

Figure 6: Real-world results on UR5e across different models and online post-training methods, with 100 evaluation trials per task.

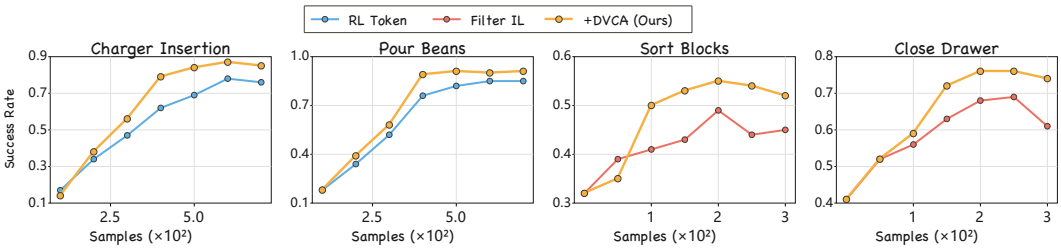

Figure 7: Success rates during training with $\pi_{0.5}$ in real-world settings. DVCA improves sample efficiency and task success rates.

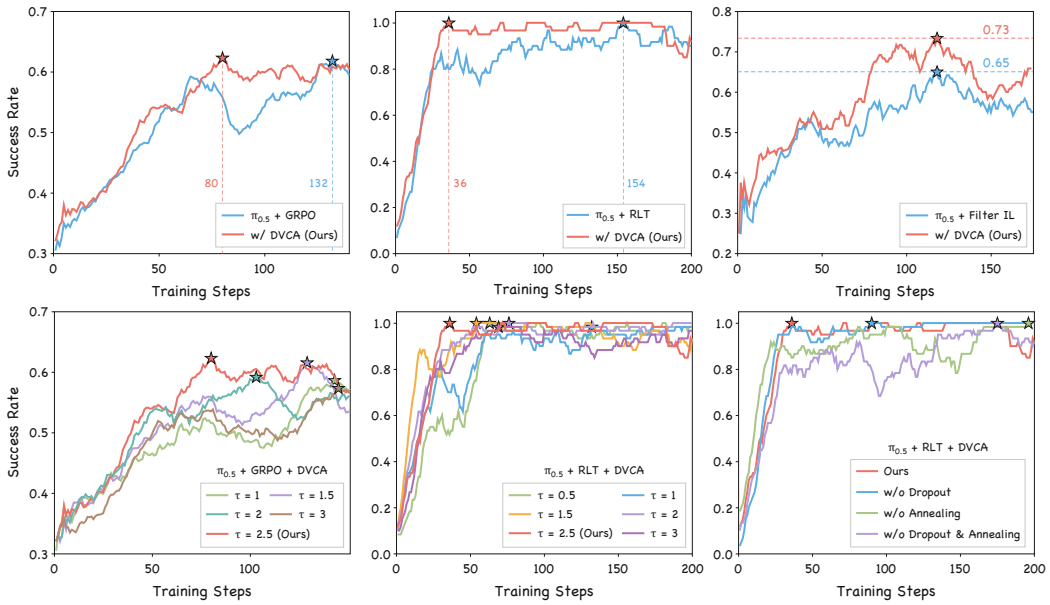

Figure 8: Effectiveness of DVCA across online post-training methods and ablations of its intervention strength and training strategies on the RoboTwin 2.0 task Place Mouse Pad.

Table 4: Ablation of reweighting temperature τ, dropout, and annealing for DVCA with $\pi_{0.5}$ and RL Token. SR and Samples report mean peak success rate (%) and mean number of rollouts required to reach the peak, respectively.

Variant	Weighting temperature τ									
	1		1.5		2		2.5		3	
	SR	Samples	SR	Samples	SR	Samples	SR	Samples	SR	Samples
w/ Drop & Anneal	94.6	736	95.1	528	95.5	452	**95.7**	**439**	95.4	470
w/o Drop	94.2	752	95.2	700	95.1	516	95.4	456	95.0	534
w/o Anneal	94.1	728	94.6	496	95.0	524	95.3	484	95.0	494
w/o Drop & Anneal	94.4	796	94.9	652	95.2	548	95.5	496	95.2	578

4.4 Ablation Studies

We study the reweighting temperature and training strategies with $\pi_{0.5}$ + RL Token (Table 4). Temperature τ controls weight differences across actions. Reweighting dropout restores unit weights for randomly selected action chunks, while annealing gradually reduces reweighting strength. Among the tested values, $\tau = 2.5$ achieves the highest peak success rate with the fewest samples, favoring a moderate reweighting strength in this setting. At $\tau = 2.5$, removing dropout, annealing, or both requires more samples, with corresponding differences in learning speed shown in Figure 8. Both strategies improve online training dynamics. Further ablations are provided in Appendix A.1.2.

5 Conclusion

We propose DVCA, which uses action-level denoising variance from flow-matching inference to improve credit assignment in online post-training. DVCA focuses learning on more uncertain actions without additional training or inference. DVCA supports plug-and-play integration across flow-based robot policies and online post-training methods. Experiments across simulated and real-world environments, robot policies, and post-training methods demonstrate that DVCA consistently improves sample efficiency or task success rate.

AI USE STATEMENT

We used generative AI sparingly for code implementation under human supervision and thoroughly reviewed all AI-generated code. We did not use generative AI for constructing theoretical models or conceptual frameworks, formulating or refining research hypotheses, designing or providing feedback on methodology and experiments, translation, qualitative or thematic analysis, or interpreting results. Synthetic dataset generation, proposing mathematical claims, providing essential elements of proofs, proof writing, and dataset cleaning or reformatting were not applicable to this study. We take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Our method and implementation details are provided in Section 3 and Appendix A.4. Experimental settings are described in Section 4.1 and Appendix A.2.

REFERENCES

Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, Hongyan Zhao, Hanyu Liu, Zhizhong Su, Lei Ma, Hang Su, and Jun Zhu. Motus: A unified latent action world model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 35101–35113, June 2026.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, brian ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 27–30 Sep 2025a. URL https://proceedings.mlr.press/v305/black25a.html.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A Vision-Language-Action Flow Model for General Robot Control. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025b. doi: 10.15607/RSS.2025.XXI.010.

Jisong Cai, Long Ling, Shiwei Chu, Zhongshan Liu, Jiayue Kang, Zhixuan Liang, Wenjie Xu, Yinan Mao, Weinan Zhang, Xiaokang Yang, Ru Ying, Ran Zheng, and Yao Mu. Aha-wam:asynchronous horizon-adaptive world-action modeling with observation-guided context routing, 2026. URL https://arxiv.org/abs/2606.09811.

Jialei Chen, Kai Wang, Kang Chen, Shuaihang Chen, Feng Gao, Wenhao Tang, Zhiyuan Li, Weilin Liu, Zhuyu Yao, Boxun Li, Yuanbo Xu, and Chao Yu. Lawam: Latent world action models for efficient dynamics-aware robot policies, 2026a. URL https://arxiv.org/abs/2606.15768.

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, 2026b. URL https://arxiv.org/abs/2510.25889.

Qianzhong Chen, Justin Yu, Mac Schwager, Pieter Abbeel, Fred Shentu, and Philipp Wu. Sarm: Stage-aware reward modeling for long horizon robot manipulation. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 17091–17116,

2026c. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/1c92edb990a05f2269f0cc3afbb4c952-Paper-Conference.pdf.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, Kailun Su, Tianling Xu, Guodong Liu, Mengkang Hu, Huan ang Gao, Kaixuan Wang, Zhixuan Liang, Yusen Qin, Xiaokang Yang, Ping Luo, and Yao Mu. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. In *Forty-third International Conference on Machine Learning*, 2026d. URL https://openreview.net/forum?id=itonej9GIV.

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. doi: 10.1177/02783649241273668. URL https://doi.org/10.1177/02783649241273668.

Perry Dong, Suvir Mirchandani, Dorsa Sadigh, and Chelsea Finn. What matters for batch online reinforcement learning in robotics? In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 112294–112308, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/b650f2838de7e1e817b5408b48a19a63-Paper-Conference.pdf.

Tongyan Fang, Siyuan Huang, Naiyu Fang, Ganlong Zhao, Zhongjin Luo, Jianbo Liu, Xiaogang Wang, Ying Dong, and Hongsheng Li. Hierarchical advantage weighting for online rl fine-tuning of vlas from sparse episode outcomes, 2026a. URL https://arxiv.org/abs/2606.17043.

Yu Fang, Wanxi Dong, Jiaqi Liu, Yue Yang, Mingxiao Huo, Yao Mu, Huaxiu Yao, Li Erran Li, Daniel Szafir, and Mingyu Ding. Densereward: Dense reward learning via failure synthesis for robotic manipulation, 2026b. URL https://arxiv.org/abs/2607.13033.

Xiangdong Feng, Yuxuan Cheng, Chen Shi, Boyao Han, Yuxuan Yan, Yitong Hong, Zhuotao Tian, and Li Jiang. Denoising tells when to replan: Denoising-variance adaptive chunking for flow-based robot policies, 2026. URL https://arxiv.org/abs/2606.03847.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/haarnoja18b.html.

Dongchi Huang, Hongyin Zhang, Bohan Hou, Siteng Huang, Zhian Su, Hang Guo, Tong Lu, Zhaofeng Xu, Jiahao Tang, Jianfei Yang, Donglin Wang, Peixi Peng, Mingxiu Chen, Deli Zhao, and Xin Li. Rynnvalue: Scaling robotic value foundation models with temporal distance, 2026. URL https://arxiv.org/abs/2608.09853.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, and Jinwei Gu. Cosmos policy: Fine-tuning video models for visuomotor control and planning. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 71531–71552, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/748becc400a57c0e31cfe6a2e7951467-Paper-Conference.pdf.

Thanh-Long V. Le, Myeongho Jeon, Kim Vu, Viet Lai, and Eunho Yang. No prompt left behind: Exploiting zero-variance prompts in llm reinforcement learning via entropy-guided advantage shaping. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 121956–121982, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/c64dfa06f20f746486da743d64f4c624-Paper-Conference.pdf.

Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Yang Zhaohui, 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 C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 125846–125869, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/cbfbcb4da14235bd69b134070898ae9d-Paper-Conference.pdf.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Luyao Zhang, Mingrui Yu, Zelin Gao, Nan Xue, Boyu Zhou, Xing Zhu, Mingyu Ding, Yujun Shen, and Yinghao Xu. Causal World Modeling for Robot Control. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026b. doi: 10.15607/RSS.2026.XXII.016.

Yuyang Liu, Chuan Wen, Yihang Hu, Dinesh Jayaraman, and Yang Gao. Timerewarder: Learning dense reward from passive videos via frame-wise temporal distance. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=XztRm216YS.

Guanxing Lu, Rui Zhao, Haitao Lin, He Zhang, and Yansong Tang. Human-in-the-loop online rejection sampling for robotic manipulation, 2025. URL https://arxiv.org/abs/2510.26406.

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, 2025. doi: 10.1126/scirobotics.ads5033. URL https://www.science.org/doi/abs/10.1126/scirobotics.ads5033.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets, 2021. URL https://arxiv.org/abs/2006.09359.

Daojie Peng, Fulong Ma, Jiahang Cao, Qiang Zhang, Xupeng Xie, Jian Guo, Ping Luo, Andrew F. Luo, Boyu Zhou, and Jun Ma. Attena+: Rectifying action inequality in robotic foundation models, 2026. URL https://arxiv.org/abs/2605.13548.

Ziyang Rao, Yiren Zhao, Weiyu Guo, Ben Fei, Yandong Guo, and Hui Xiong. The geometry of flow-matching uncertainty: A cost-free uncertainty proxy and its application in flow-based vla failure detection, 2026. URL https://arxiv.org/abs/2607.27933.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Huajie Tan, Sixiang Chen, Yijie Xu, Zixiao Wang, Cheng Chi, Yuheng Ji, Yaoxu Lyu, Zhongxia Zhao, Xiansheng Chen, Peterson Co, Shaoxuan Xie, Guocai Yao, Pengwei Wang, Zhongyuan Wang, and Shanghang Zhang. General process reward modeling for robotic reinforcement learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22412–22422, June 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 Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine*

Learning Research, pp. 258–282. PMLR, 27–30 Sep 2025. URL https://proceedings.mlr.press/v305/wagenmaker25a.html.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 115452–115486. Curran Associates, Inc., 2025. doi: 10.52202/085713-3850. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/a797c2d2e0c1fdabf4d1ab8cd0b465c6-Paper-Conference.pdf.

Zengbin Wang, Feng Xiong, Liang Lin, Xuecai Hu, Yong Wang, Yanlin Wang, Man Zhang, and Xiangxiang Chu. Visually-guided policy optimization for multimodal reasoning. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6646–6664, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.301. URL https://aclanthology.org/2026.acl-long.301/.

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

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, Ayaan Malik, Kyungmin Lee, William Liang, Nadun Ranawaka, Jiasheng Gu, Yinzhen Xu, Guanzhi Wang, Fengyuan Hu, Avnish Narayan, Johan Bjorck, Jing Wang, Gwanghyun Kim, Dantong Niu, Ruijie Zheng, Yuqi Xie, Jimmy Wu, Qi Wang, Ryan Julian, Danfei Xu, Yilun Du, Yevgen Chebotar, Scott Reed, Jan Kautz, Yuke Zhu, Linxi "Jim" Fan, and Joel Jang. World action models are zero-shot policies, 2026. URL https://arxiv.org/abs/2602.15922.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-wam: Do world action models need test-time future imagination?, 2026. URL https://arxiv.org/abs/2603.16666.

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3D Diffusion Policy: Generalizable Visuomotor Policy Learning via Simple 3D Representations. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.067.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Tai Wang, Ya-Qin Zhang, Jingjing Liu, and Xianyuan Zhan. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 60580–60606, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/630d293833e09e1ecd892a898a20b074-Paper-Conference.pdf.

A Appendix

A.1 Complete Results

A.1.1 Per-Task Results

We evaluate ten RoboTwin tasks under the randomized setting: Turn Switch, Place Mouse Pad, Move Stapler Pad, Move Can Pot, Pick Diverse Bottles, Stamp Seal, Press Stapler, Place A2B Left, Place A2B Right, and Place Object Scale.

GRPO. Table 5 complements Table 1 with results on the remaining six RoboTwin tasks, using the same method groups and evaluation metrics. DVCA improves success rates across all six tasks and all three policies, with gains of 0.4–9.7 percentage points.

Table 5: Task success rates (%) for GRPO on the remaining six tasks in the randomized setting of RoboTwin 2.0. The other four tasks and the average over all ten tasks are reported in Table 1.

Method ···	Pick Diverse Bottles	Stamp Seal	Press Stapler	Place A2B Left	Place A2B Right	Place Object Scale
X-VLA	36.5	81.3	98.1	49.8	37.5	74.7
Motus	91.8	90.9	99.3	78.6	85.3	84.1
π_0	31.2	32.8	37.4	46.7	34.2	52.3
+ GRPO	80.3	92.7	88.8	82.0	83.5	81.8
w/ DVCA	84.8	94.7	93.6	87.2	89.9	83.6
Δ	+4.5	+2.0	+4.8	+5.2	+6.4	+1.8
$\pi_{0.5}$	71.1	54.8	83.0	81.5	83.8	80.3
+ GRPO	91.2	78.6	95.6	93.8	95.1	94.2
w/ DVCA	95.9	88.3	97.5	97.9	97.1	97.3
Δ	+4.7	+9.7	+1.9	+4.1	+2.0	+3.1
Fast-WAM	85.4	94.5	97.2	92.5	98.8	96.7
+ GRPO	93.8	96.7	98.0	96.2	99.1	98.1
w/ DVCA	97.9	98.4	98.5	98.8	99.8	98.5
Δ	+4.1	+1.7	+0.5	+2.6	+0.7	+0.4

RL Token. Table 6 complements Table 2 with results on the remaining six RoboTwin tasks, using the same method groups and evaluation metrics.

Filtered Imitation Learning. Table 7 complements Table 3 with results on the remaining six RoboTwin tasks, using the same method groups and evaluation metrics. DVCA improves Filter IL across all policy–task pairs and outperforms both reweighting baselines.

A.1.2 Additional Ablations of DVCA

Signal Decomposition. We examine the contributions of chunk position and task-dependent variation using $\pi_{0.5}$ with GRPO (Table 8). For each task and chunk length, we estimate a fixed position profile from calibration rollouts:

$$b_h = \frac{1}{N}\sum_{n=1}^{N} q_{n,h}^{\text{cal}}, \qquad r_{c,h} = q_{c,h} - b_h, \qquad q_{c,h} = b_h + r_{c,h}, \tag{9}$$

where $q_{c,h} = \log(V_{c,h} + \delta)$ and N is the number of calibration queries. Position-only and Position-adjusted replace $q_{c,h}$ with b_h and $r_{c,h}$, respectively, in the existing reweighting rule. Both components yield modest gains, indicating that DV captures useful uncertainty associated with both prediction horizon and task phase.

Table 6: Results of RL Token training on the remaining six tasks in the randomized setting of RoboTwin 2.0. SR and Samples follow the definitions in Table 2, which reports the other four tasks and averages over all ten tasks.

	...	Pick Diverse Bottles		Stamp Seal		Press Stapler		Place A2B Left		Place A2B Right		Place Object Scale	
Method		SR	Samples	SR	Samples	SR	Samples	SR	Samples	SR	Samples	SR	Samples
π_0		31.2	–	32.8	–	37.4	–	46.7	–	34.2	–	52.3	–
+ RL Token		90.2	692	98.6	700	97.8	744	93.5	1056	90.4	608	82.2	1256
w/ DVCA		91.7	580	99.5	592	97.1	612	94.2	492	92.8	380	85.5	792
Δ		+1.5	-112	+0.9	-108	-0.7	-132	+0.7	-564	+2.4	-228	+3.3	-464
$\pi_{0.5}$		71.1	–	54.8	–	83.0	–	81.5	–	83.8	–	80.3	–
+ RL Token		91.4	636	87.3	580	98.2	472	95.6	1264	96.5	1076	87.4	492
w/ DVCA		93.5	308	89.3	384	99.0	376	97.7	1184	96.9	848	89.1	460
Δ		+2.1	-328	+2.0	-196	+0.8	-96	+2.1	-80	+0.4	-228	+1.7	-32
Fast-WAM		85.4	–	94.5	–	97.2	–	92.5	–	98.8	–	96.7	–
+ RL Token		98.7	708	97.3	676	98.9	328	99.2	896	99.2	508	98.1	764
w/ DVCA		99.5	560	98.1	380	98.9	264	99.5	648	99.6	304	99.4	232
Δ		+0.8	-148	+0.8	-296	+0.0	-64	+0.3	-248	+0.4	-204	+1.3	-532

Table 7: Task success rates (%) for Filter IL on the remaining six tasks in the randomized setting of RoboTwin 2.0. The other four tasks and the average over all ten tasks are reported in Table 3.

Method	...	Pick Diverse Bottles	Stamp Seal	Press Stapler	Place A2B Left	Place A2B Right	Place Object Scale
π_0		31.2	32.8	37.4	46.7	34.2	52.3
+ Filter IL		63.4	78.4	83.2	72.1	85.5	88.6
w/ RA-BC(R)		66.5	80.0	86.3	72.7	85.3	91.3
w/ AttenA+		66.1	79.6	87.1	74.5	88.0	93.5
w/ DVCA		**68.8**	**80.5**	**90.1**	**77.9**	**89.4**	**93.6**
$\pi_{0.5}$		71.1	54.8	83.0	81.5	83.8	80.3
+ Filter IL		87.2	89.1	93.7	92.6	93.2	93.1
w/ RA-BC(R)		92.8	90.6	93.6	91.4	95.3	92.8
w/ AttenA+		91.4	89.5	94.9	94.4	94.6	89.2
w/ DVCA		**95.8**	**94.4**	**97.5**	**96.7**	**97.6**	**96.6**
Fast-WAM		85.4	94.5	97.2	92.5	98.8	96.7
+ Filter IL		93.3	97.5	98.5	97.4	99.0	99.2
w/ RA-BC(R)		94.8	97.1	98.5	97.7	97.8	98.7
w/ AttenA+		93.2	98.9	97.6	98.6	99.3	99.4
w/ DVCA		**99.0**	**99.0**	**98.9**	**99.1**	**99.6**	**99.5**

Advantage Signs and Chunk Length. We also restrict reweighting to positive or negative advantages, assigning unit weights to the other sign. Reweighting both signs performs better than either one-sided variant. Finally, we compare action-chunk lengths $H \in \{50, 20, 10\}$. DVCA provides larger gains with longer chunks, consistent with the stronger contribution of position-dependent uncertainty for actions predicted farther into the future.

Table 8: Additional ablations of DVCA with $\pi_{0.5}$ and GRPO on four RoboTwin tasks. We vary the uncertainty component, the advantage signs receiving reweighting, and action-chunk length H. Entries are task success rates (%); Avg. (4) is their four-task mean.

Method	Turn Switch	Place Mouse Pad	Move Stapler Pad	Move Can Pot	**Avg.** (4)
Action-chunk length $H = 50$					
$\pi_{0.5}$	54.9	39.0	42.8	55.7	48.1
+ GRPO	78.4	61.8	86.5	91.0	79.4
w/ DVCA	**82.5**	**65.5**	**89.1**	**94.1**	**82.8**
w/ Position-only	80.2	62.8	86.7	92.6	80.6
w/ Position-adjusted	81.7	65.0	88.4	93.1	82.1
w/ Positive-only	79.5	61.1	88.8	91.4	80.2
w/ Negative-only	79.3	62.0	85.7	90.5	79.4
Action-chunk length $H = 20$					
$\pi_{0.5}$	60.2	46.5	47.9	63.0	54.4
+ GRPO	82.6	64.3	90.1	93.7	82.7
w/ DVCA	85.0	66.7	92.5	96.1	85.1
Action-chunk length $H = 10$					
$\pi_{0.5}$	57.5	41.2	44.1	57.6	50.1
+ GRPO	80.9	62.9	88.6	92.3	81.2
w/ DVCA	83.0	64.3	90.1	94.0	82.9

A.1.3 LIBERO Results

Table 9 organizes the comparison by the four LIBERO suites, with separate rows for SFT initialization, GRPO, and GRPO with DVCA. DVCA improves success rates across all four suites for both policies, with the largest gains on LIBERO-Long.

Table 9: Task success rates (%) on LIBERO. Avg. denotes the mean across Spatial, Object, Goal, and Long; Δ denotes the improvement over GRPO.

Method	Spatial	Object	Goal	Long	**Avg.**
π_0 (SFT)	65.4	64.9	50.7	52.0	58.3
+ GRPO	98.4	98.2	83.9	81.5	90.5
w/ DVCA	99.8	98.6	88.0	87.3	93.4
Δ	+1.4	+0.4	+4.1	+5.8	+2.9
$\pi_{0.5}$ (SFT)	84.7	95.4	85.2	45.6	77.7
+ GRPO	97.2	99.5	91.1	78.9	91.7
w/ DVCA	98.5	99.7	94.2	86.1	94.6
Δ	+1.3	+0.2	+3.1	+7.2	+3.0

A.1.4 PPO Results

A.1.5 Additional Training Comparisons

Figure 9 compares the concentration of action-level DV during GRPO training and collection success under an alternative Filter IL configuration. For GRPO, the top 20% of actions account for

Table 10: Task success rates (%) for PPO on four tasks in the randomized setting of RoboTwin 2.0. Avg. denotes the mean over these four tasks; Δ denotes the improvement over PPO.

Method	Turn Switch	Place Mouse Pad	Move Stapler Pad	Move Can Pot	**Avg.**
π_0	42.7	20.4	24.6	48.3	34.0
+ PPO	89.6	88.2	78.5	91.1	86.9
w/ DVCA	91.8	93.6	82.5	94.7	90.7
Δ	+2.2	+5.4	+4.0	+3.6	+3.8
$\pi_{0.5}$	54.9	39.0	42.8	55.7	48.1
+ PPO	89.2	78.6	92.8	93.4	88.5
w/ DVCA	92.8	79.9	95.7	94.4	90.7
Δ	+3.6	+1.3	+2.9	+1.0	+2.2
Fast-WAM	59.1	89.7	64.2	88.4	75.4
+ PPO	86.8	98.3	94.5	95.1	93.7
w/ DVCA	90.0	99.1	96.8	97.9	96.0
Δ	+3.2	+0.8	+2.3	+2.8	+2.3

31.71% of total DV in the baseline and 28.49% with DVCA, averaged over iterations 75–93. The lower share indicates that denoising variance is distributed more broadly across actions. For Filter IL, each round collects eight trajectories and performs five updates. DVCA reaches a higher peak collection success rate under this configuration.

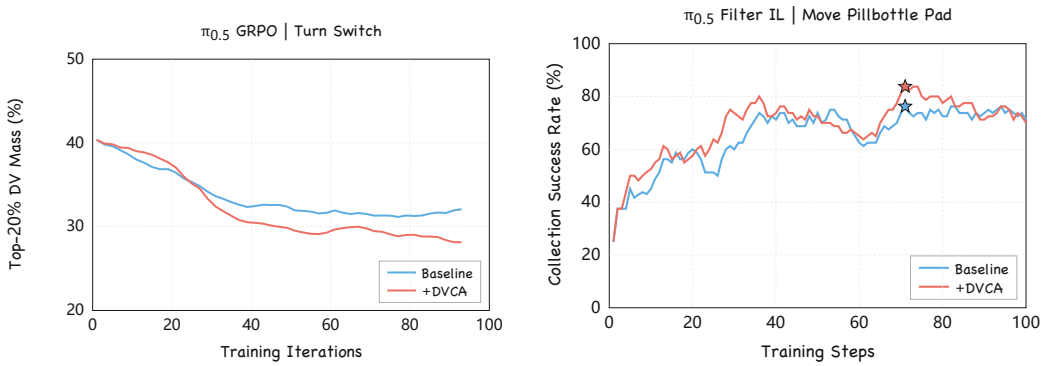

Figure 9: Additional training comparisons with $\pi_{0.5}$. Left: the fraction of total denoising variance contributed by the top 20% of actions in each trajectory, averaged across trajectories during GRPO training. Right: collection success rates for Filter IL with eight trajectories and five updates per round; stars mark the peaks of the smoothed curves.

A.2 Experimental Protocols and Implementation

A.2.1 Base Policies

We use π_0, $\pi_{0.5}$, and Fast-WAM as flow-based robot policies. π_0 combines pretrained vision-language representations with a flow-matching action expert that generates continuous action chunks (Black et al., 2025b). $\pi_{0.5}$ extends this approach with broader training data and hierarchical task information to support generalization across environments and tasks (Black et al., 2025a). Both policies condition action generation on visual observations, proprioception, and language.

Fast-WAM uses a pretrained video-model backbone to encode current observations and an action expert to generate action chunks (Yuan et al., 2026). Its action generation also produces a sequence of velocity predictions. Consequently, all three policies expose the same interface for extracting clean-action estimates and action-level denoising variance during inference.

GRPO and filtered imitation learning update the action-generation components of the base policy. RL Token instead retains a frozen reference policy and trains a lightweight actor and critics. DVCA is applied to the advantage interface in GRPO, the flow-matching objective in filtered imitation learning, and the actor's BC term in RL Token.

A.2.2 Simulation Protocols

We use the randomized setting of RoboTwin 2.0 and compare each base policy and post-training method with and without DVCA across ten tasks. We report task success rates and measure online interaction by the number of collected trajectories.

Hyperparameters. Tables 11–13 summarize the policy, optimization, and DVCA settings. Prediction horizon and action dimension specify the action chunk, while denoising steps specify the integration budget. Trajectories per round, batch size, and learning rate control data collection and policy updates. The trainable component identifies the parameters being optimized. DV tail length L specifies the final denoising steps used to compute variance. Reweighting dropout is the probability of restoring unit chunk weights, and the annealing interval specifies the rounds over which weights return to one. The weighting interface identifies the loss or advantage being reweighted.

GRPO Settings. Rollout group size specifies the number of trajectories sharing an advantage-normalization group. Update epochs specify passes over each rollout batch. Likelihood clipping sets the ratio-clipping range, and the Flow-SDE noise level controls stochasticity during denoising.

Table 11: Policy, optimization, and DVCA hyperparameters for GRPO experiments with π_0, $\pi_{0.5}$, and Fast-WAM.

Parameter	π_0	$\pi_{0.5}$	Fast-WAM
Policy configuration			
Action prediction horizon	50	50	50
Action dimension	14	14	14
Denoising steps	10	10	10
Trainable component	Action expert	Action expert	Action expert
Online post-training			
Trajectories per round	128	128	256
Rollout group size	8	8	8
Update epochs	2	2	2
Global batch size	512	512	512
Learning rate	5×10^{-6}	5×10^{-6}	5×10^{-6}
Likelihood clipping	0.2	0.2	0.2
Flow-SDE noise level	0.5	0.5	0.5
DVCA			
DV tail length L	5	5	5
Reweighting dropout	0.2	0.2	0.2
Annealing interval (rounds)	1–200	1–200	1–200
Weighting interface	Advantages	Advantages	Advantages

RL Token Settings. Representation dimension specifies the frozen reference policy's feature size. Separate learning rates update the actor and critics. The reward discount controls the weighting of future rewards, and the target update coefficient controls target-network averaging. Warm-up updates precede student-policy rollouts; the critic-to-actor update ratio specifies their relative update frequency. BC and Q coefficients balance reference-action fitting and Q-value maximization over training. DVCA reweights BC losses on successful trajectories.

Table 12: Policy, optimization, and DVCA hyperparameters for RL Token experiments with π_0, $\pi_{0.5}$, and Fast-WAM.

Parameter	π_0	$\pi_{0.5}$	Fast-WAM
Policy configuration			
Reference policy	Frozen	Frozen	Frozen
Action dimension	14	14	14
Representation dimension	2048	2048	2048
Reference denoising steps	10	10	10
Online post-training			
Trajectories per round	4	4	8
Global batch size	512	512	512
Actor learning rate	10^{-4}	10^{-4}	10^{-4}
Critic learning rate	10^{-4}	10^{-4}	10^{-4}
Reward discount	0.99	0.99	0.99
Target update coefficient	0.005	0.005	0.005
Warm-up updates	15,000	15,000	15,000
Critic-to-actor update ratio	2	2	2
BC coefficient, early $\rightarrow$ late	$7 \rightarrow 2.5$	$7 \rightarrow 2.5$	$7 \rightarrow 2.5$
Q coefficient, early $\rightarrow$ late	$0.05 \rightarrow 0.45$	$0.05 \rightarrow 0.45$	$0.05 \rightarrow 0.45$
DVCA			
DV tail length L	5	5	5
Reweighting dropout	0.2	0.2	0.2
Annealing interval (rounds)	1–500	1–500	1–500
Weighting interface	BC loss	BC loss	BC loss
Reweighted data	Successful	Successful	Successful

Filtered Imitation Learning Settings. Updates per round specify the number of optimizer updates after data collection. Training uses successful online trajectories with a flow-matching objective, and DVCA reweights the corresponding per-action losses.

Table 13: Policy, optimization, and DVCA hyperparameters for filtered imitation learning experiments with π_0, $\pi_{0.5}$, and Fast-WAM.

Parameter	π_0	$\pi_{0.5}$	Fast-WAM
Policy configuration			
Action prediction horizon	50	50	32
Action dimension	14	14	14
Denoising steps	10	10	10
Trainable component	Action expert	Action expert	Action expert
Online post-training			
Trajectories per round	4	4	8
Updates per round	5	5	5
Global batch size	1024	1024	1024
Learning rate	2.5×10^{-5}	2.5×10^{-5}	2.5×10^{-5}
Training data	Successful	Successful	Successful
Training objective	Flow matching	Flow matching	Flow matching
DVCA			
DV tail length L	5	5	5
Reweighting dropout	0.2	0.2	0.2
Annealing interval (rounds)	1–200	1–200	1–200
Weighting interface	FM loss	FM loss	FM loss

A.2.3 Online Post-Training Baselines

GRPO. We compute likelihood ratios from stochastic denoising transitions following πRL (Chen et al., 2026b). Returns within each rollout group define normalized advantages, and the clipped objective updates the policy. DVCA computes weights over the complete sampling group and redistributes action-level gradients.

RL Token. We implement RL Token following the original method (Xu et al., 2026) and adapt its observations, action chunks, and task rewards to RoboTwin. A frozen reference policy provides representations and reference actions, while the actor learns through Q-value maximization and BC regularization. Reference-policy data collection and actor–critic warm-up precede student-policy rollouts. Each online round collects four trajectories for π_0 and $\pi_{0.5}$, and eight for Fast-WAM. Sample-efficiency comparisons start when student-policy rollouts begin. DVCA weights BC errors on chunks from successful trajectories; the remaining chunks use unit weights.

Filtered Imitation Learning. The policy collects trajectories online, adds successful rollouts to a buffer, and learns from sampled action chunks through flow matching (Dong et al., 2026; Lu et al., 2025). DVCA follows the same collection and update procedure, with weights applied to per-action flow-matching losses.

A.2.4 Real-World Protocol

Hardware and Inputs. Our platform consists of a UR5e arm, a 1-DoF gripper, and two Intel RealSense D435i cameras. The policy receives external-view and wrist-view images, proprioceptive state, and a task instruction. Demonstrations are collected at 10 Hz.

Action Execution. Real-world deployment uses the same number of denoising steps as the corresponding simulation experiments. We halve the executed action-prefix length at each policy query to improve execution stability.

Demonstrations and Evaluation. For each task, we collect 100 demonstrations for supervised fine-tuning, with limited randomization of object positions, object instances, and backgrounds. We hold out instances along each randomization axis. Each task is evaluated over 100 trials: 50 include a held-out instance along one randomization axis, and 50 use the standard evaluation conditions. Table 14 summarizes the tasks and distribution shifts.

Table 14: Real-world tasks, success criteria, and out-of-distribution (OOD) conditions.

Task	Task Description	Success Criterion	OOD Condition
Charger Insertion	Align and insert a charger into a socket.	The charger is inserted into the target socket.	Size shift: a smaller charger.
Close Drawer	Approach and push an open drawer closed.	The target drawer is fully closed.	Position shift: a relocated cabinet.
Pour Beans	Grasp a bean container and pour into a target cup.	The beans are poured into the target cup.	Distractors: colored blocks in the workspace.
Sort Blocks	Place colored blocks in matching containers.	All blocks are in the corresponding containers.	Background: a different tablecloth.

A.2.5 Computational Resources

Most experiments run on three servers, each equipped with eight NVIDIA H100 GPUs and 2 TB of RAM. A subset of experiments, primarily RL Token simulations, uses four NVIDIA A800 GPUs and 480 GB of RAM. Real-world deployment uses two NVIDIA RTX A6000 GPUs.

A.3 Additional Method Details

A.3.1 DVCA Implementation

During policy sampling, DVCA reuses the velocity predictions from the flow-matching solver to compute clean-action estimates $z_{c,j} = x_{c,j} - t_j v_\theta(x_{c,j}, t_j, s_c)$. For each action position, we retain the final $L = 5$ estimates and compute their variance across denoising steps, summed over action dimensions. This produces one scalar $V_{c,h}$ per action. The variance vector is stored with the corresponding action chunk and reused during training, connecting the inference-time signal to the actions being optimized.

Weights are computed over the complete comparison domain before splitting the data into micro-batches. The log-variance scores are normalized within and across action chunks and mapped to mean-one weights. Within-chunk weights emphasize relatively uncertain actions, while across-chunk weights adjust the relative contribution of chunks.

Annealing interpolates the weighting factors toward one, while reweighting dropout restores unit weights for a randomly selected chunk. The resulting weights remain fixed during the corresponding policy updates and are applied to action advantages or supervised losses. The trainable policy components are updated with the baseline optimizer. Appendix A.4 specifies the weighting domains and gradient formulation.

A.3.2 Integration with Online Post-Training

For GRPO, stochastic denoising transitions provide action log-likelihoods, following πRL (Chen et al., 2026b). DVCA collects the variance vectors and trajectory-group assignments across rollout workers, then constructs weights within each complete sampling group. The

same weights are reused across optimization epochs for that group. The implementation redistributes action-level gradient contributions while preserving the baseline's chunk-level likelihood ratio and clipping rule. Thus, trajectory returns determine the update direction, and denoising variance determines the relative contribution of individual actions.

We implement filtered imitation learning following standard practice: successful rollouts are added to a replay buffer and provide action targets for flow matching (Dong et al., 2026; Lu et al., 2025). DVCA stores the variance vector alongside each retained chunk, constructs weights on the sampled training batch, and applies them to per-action flow-matching errors before loss reduction. Weight computation precedes microbatch splitting, so the comparison domain is the full training batch.

For RL Token (Xu et al., 2026), the frozen reference policy supplies representations, reference actions, and their denoising variances. The lightweight actor is trained with a Q-value objective and a reference-action BC term. DVCA weights the per-action BC errors for chunks from successful trajectories; other chunks retain unit weights. The Q-value term follows the baseline objective. The reference policy remains frozen throughout this stage.

A.4 Supplementary Formalization

A.4.1 Action-Level Denoising Variance

Equation 4 measures the variation of clean-action estimates at a fixed action position. Prior analysis relates this variance to the finite-step endpoint error of the learned flow, $E_{c,h} \le C_{\mathrm{tail}}\sqrt{V_{c,h}}$, under local regularity and integration conditions (Feng et al., 2026). Smaller variance gives a tighter bound and indicates more stable denoising predictions. DVCA uses this signal to allocate training emphasis.

A.4.2 Weight Construction and Training Interfaces

Weight allocation. Exponential weighting has a relative-entropy interpretation, as in advantage-weighted policy improvement (Nair et al., 2021). For normalized uncertainty scores u_j, baseline coefficients $\mu_j > 0$ with $\sum_j \mu_j = 1$, and temperature $\tau > 0$,

$$\begin{aligned} \max_{w_j>0,\ \sum_j \mu_j w_j = 1} \quad & \sum_j \mu_j w_j u_j - \tau \sum_j \mu_j w_j \log w_j, \\ & w_j^\star = \frac{\exp(u_j/\tau)}{\sum_k \mu_k \exp(u_k/\tau)}. \end{aligned} \tag{10}$$

The first term favors uncertain actions; the second controls weight concentration. The stationarity condition $u_j - \tau(\log w_j + 1) - \eta = 0$, together with normalization, gives the unique solution. Applying it within and across chunks gives the two factors in Equation 5, following the two-level organization of prior reweighting methods (Wang et al., 2026). Within-chunk coefficients are proportional to valid action dimensions. Across-chunk coefficients follow the baseline reduction for GRPO and are uniform over eligible supervised chunks.

Effect on updates. Let $g_{c,h}$ be an action's baseline gradient contribution and $\mu_{c,h}$ its reduction coefficient. Holding weights fixed yields

$$g^{\mathrm{DV}} = \sum_{c,h} \mu_{c,h} w_{c,h} g_{c,h} = g^{\mathrm{base}} + \sum_{c,h} \mu_{c,h}(w_{c,h} - 1) g_{c,h}. \tag{11}$$

Positive weights preserve each contribution's direction and change its relative influence. Unit weights recover the baseline update.

Annealing and reweighting dropout. At iteration u, each strength $\alpha_r(u)$, $r \in \{\mathrm{loc}, \mathrm{grp}\}$, decreases linearly between its configured endpoints and remains constant outside the annealing interval. The factors are

$$\psi_{c,h}^{(u)} = 1 + \alpha_{\mathrm{loc}}(u)(\psi_{c,h} - 1), \qquad \varphi_c^{(u)} = 1 + \alpha_{\mathrm{grp}}(u)(\varphi_c - 1). \tag{12}$$

Zero strength restores unit weighting. With a shared chunk mask $m_c \sim \text{Bernoulli}(1 - p_{\text{drop}})$ and eligibility indicator χ_c, the effective weight is

$$w_{c,h} = \text{sg}\Big[1 + \chi_c m_c \big(\psi_{c,h}^{(u)} \varphi_c^{(u)} - 1\big)\Big]. \tag{13}$$

All GRPO chunks and retained Filter IL chunks are eligible; RL Token reweights successful-trajectory chunks. Dropped or ineligible chunks receive unit weights. The mask and weights remain fixed during the corresponding updates; sg denotes stop-gradient.

GRPO gradients. Let $\ell_{c,h}^{\log p}(\theta)$ denote action h's contribution to the stochastic denoising log-likelihood (Chen et al., 2026b), so that $\log \rho_c = \sum_h [\ell_{c,h}^{\log p}(\theta) - \ell_{c,h}^{\log p}(\theta_{\text{old}})]$. Define

$$\widetilde{\rho}_{c,h} = \exp\Big\{\text{sg}(\log \rho_c) + H\big[\ell_{c,h}^{\log p} - \text{sg}(\ell_{c,h}^{\log p})\big]\Big\}. \tag{14}$$

Its forward value equals ρ_c, while its gradient is assigned through action h. Using $\widehat{A}_{c,h}^{\text{DV}} = w_{c,h}\widehat{A}_{i(c)}$, we optimize

$$\mathcal{J}_{\text{DVCA}} = \mathbb{E}_c\left[\frac{1}{H}\sum_h \min\Big(\widetilde{\rho}_{c,h}\widehat{A}_{c,h}^{\text{DV}}, \text{clip}(\widetilde{\rho}_{c,h}, 1-\epsilon, 1+\epsilon)\widehat{A}_{c,h}^{\text{DV}}\Big)\right]. \tag{15}$$

The factor H offsets action averaging. Positive weights retain the baseline clipping decision, and unit weights recover its gradient.

Supervised losses. Equation 7 weights flow-matching errors in Filter IL and reference-action regression errors in RL Token. The latter retains its Q-guided actor term (Xu et al., 2026). Implementation averages squared errors over valid action dimensions and then over chunks. For full $H \times D$ chunks, this adds a factor $1/D$ to the main-text squared-norm convention; setting $\lambda_{\text{BC,paper}} = \lambda_{\text{BC,impl}}/D$ preserves the relative Q and BC weighting.

A.5 Qualitative Examples

A.5.1 Denoising Variance Across Action Positions

We average action-level denoising variance at each position over policy queries from 32 rollouts per task using $\pi_{0.5}$. Figure 10 shows that mean DV rises near the end of the chunk, while profiles at earlier positions vary across tasks. Together with the task-phase examples in Figures 12–18, these profiles suggest that DV reflects both the manipulation phase and the prediction horizon: later actions are predicted farther from the current observation. DVAC similarly uses denoising variance to adapt the action execution horizon (Feng et al., 2026).

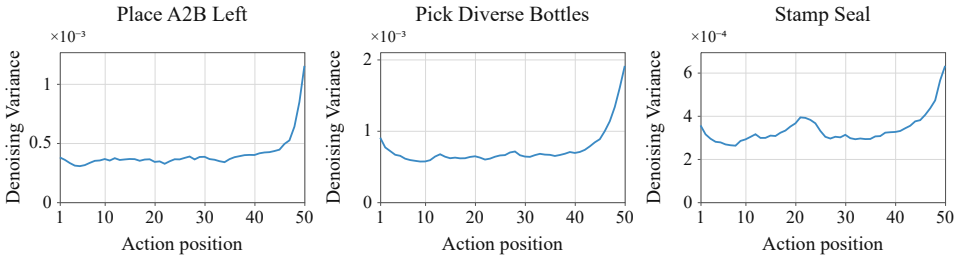

Figure 10: Mean action-level denoising variance across the 50 positions of an action chunk. Each curve averages policy queries from 32 rollouts with $\pi_{0.5}$; DV uses the final three denoising steps.

A.5.2 Progress-Prediction Example

Pretrained VLM-based process reward models learn to estimate task progress from large-scale robotic data. Prediction errors can affect downstream training. Figure 11 shows a successful rollout in which the initial remaining-time prediction is underestimated. Although the robot makes progress by approaching the object, the subsequent increase in predicted remaining time is interpreted as a regression, giving the first action chunk zero raw RA-BC(R) weight in Filter IL.

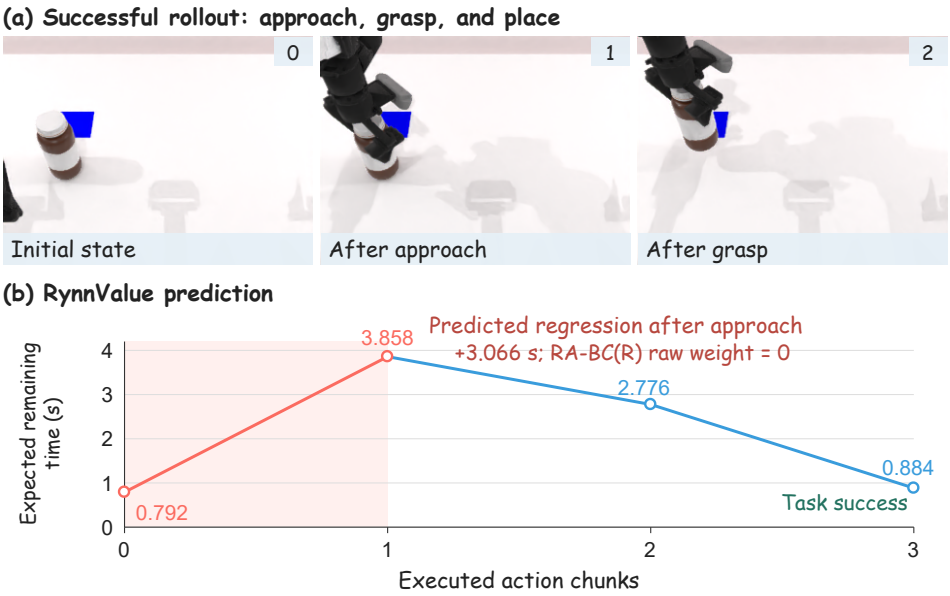

Figure 11: RynnValue predictions along a successful Move Pillbottle Pad rollout. After the robot approaches the bottle, the expected remaining time increases from 0.792 to 3.858 s, assigning the first action chunk a zero RA-BC(R) raw weight. Images show three recorded query-boundary observations; the final point reports the prediction at task success.

A.5.3 Action-Level Denoising Variance

We visualize action-level denoising variance along task execution. For supplemental training analysis, mean DV describes overall prediction variation, while concentration describes how this variation is distributed across actions. Together, they characterize changes in the signal distribution during training with DVCA.

Figures 12–18 show fourteen RoboTwin rollouts. Across these examples, higher DV often occurs near end-effector alignment and grasping, providing a qualitative view of action uncertainty during manipulation. Numbered images illustrate the task stages marked on each action-level curve; Step denotes the action index.

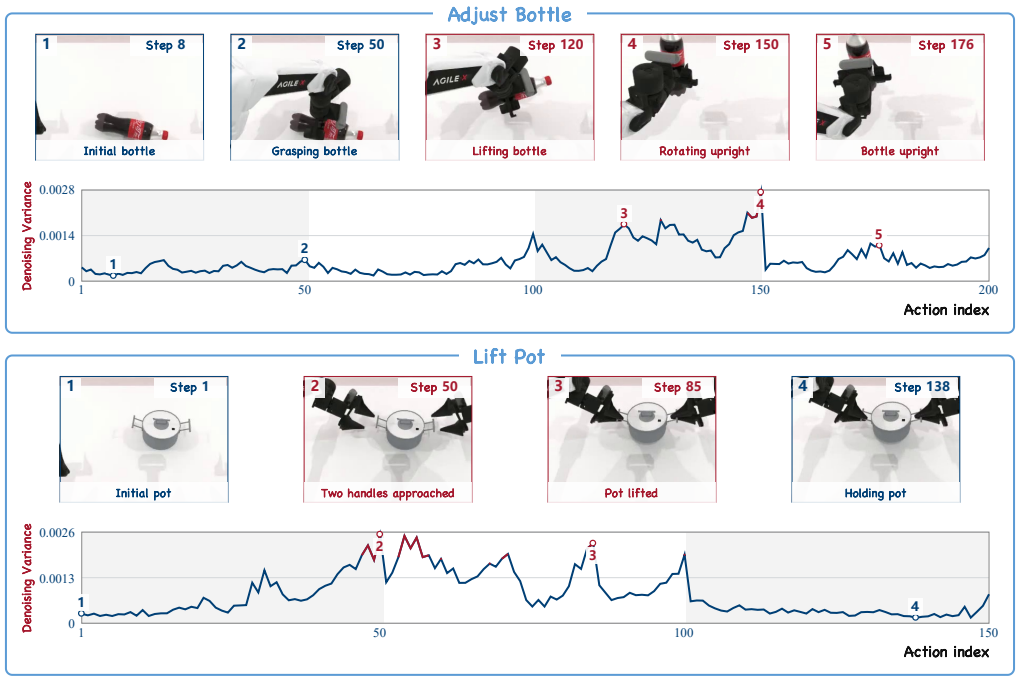

Figure 12: Action-level denoising variance during Adjust Bottle (top) and Lift Pot (bottom).

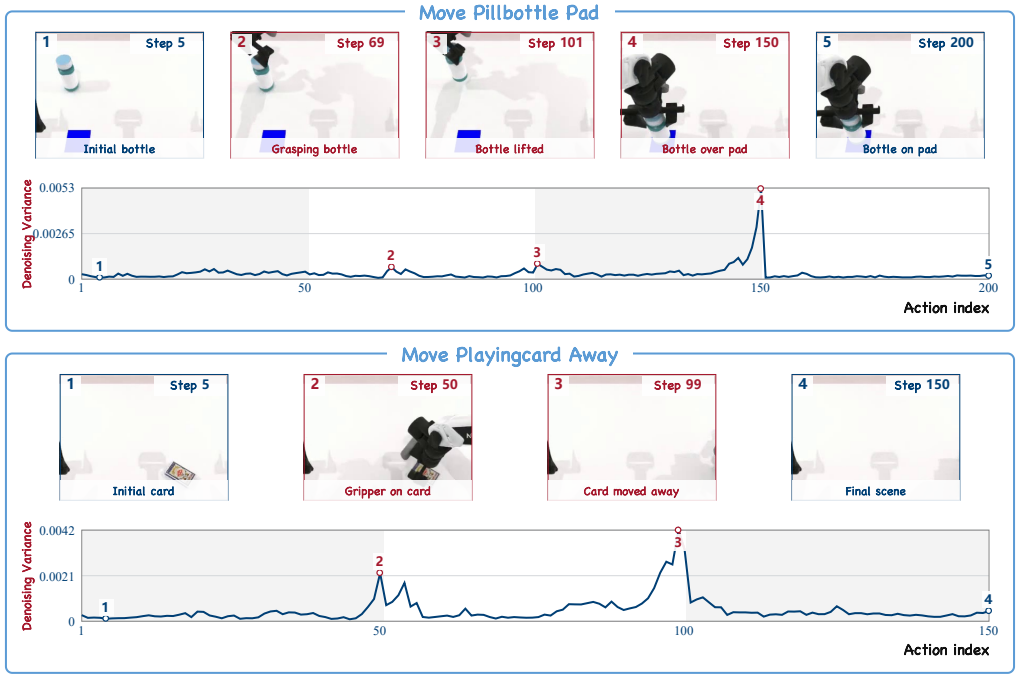

Figure 13: Action-level denoising variance during Move Pillbottle Pad (top) and Move Playingcard Away (bottom).

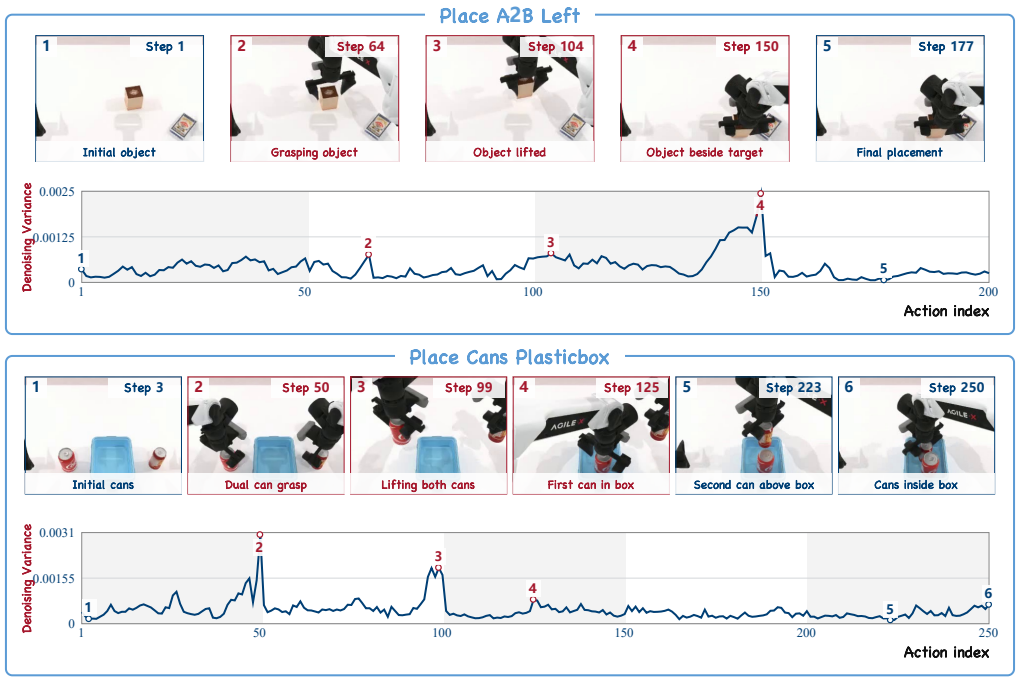

Figure 14: Action-level denoising variance during Place A2B Left (top) and Place Cans Plasticbox (bottom).

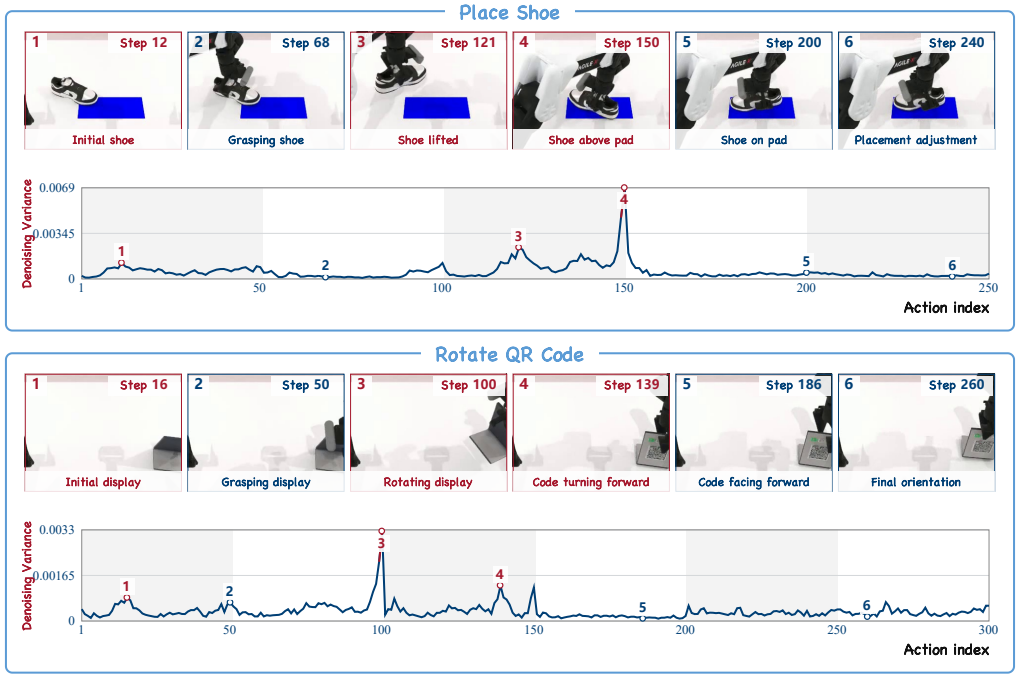

Figure 15: Action-level denoising variance during Place Shoe (top) and Rotate QR Code (bottom).

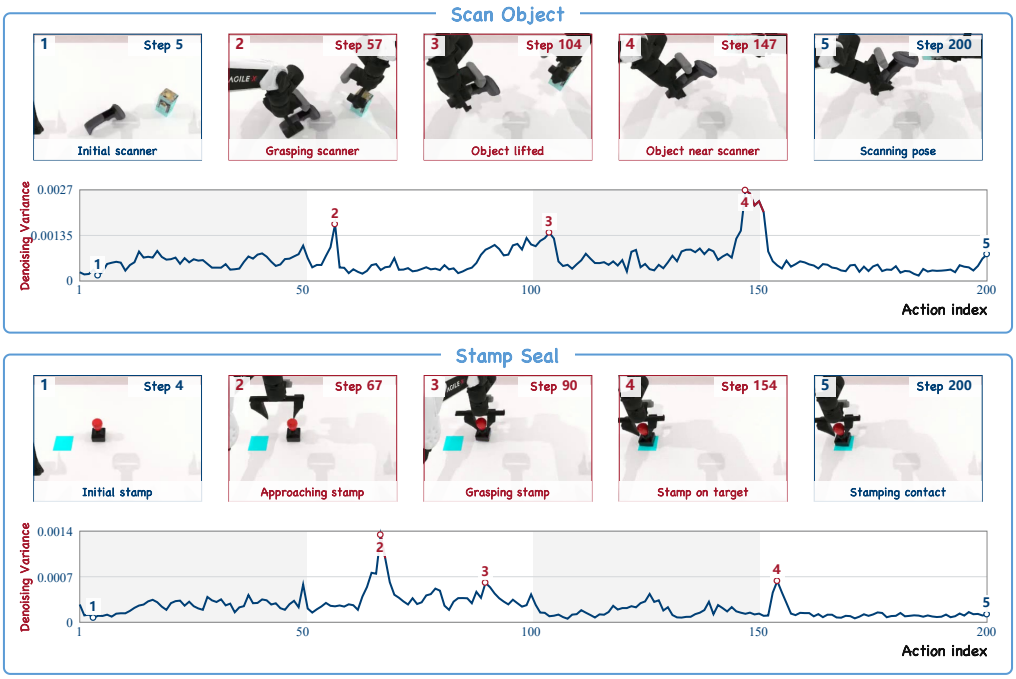

Figure 16: Action-level denoising variance during Scan Object (top) and Stamp Seal (bottom).

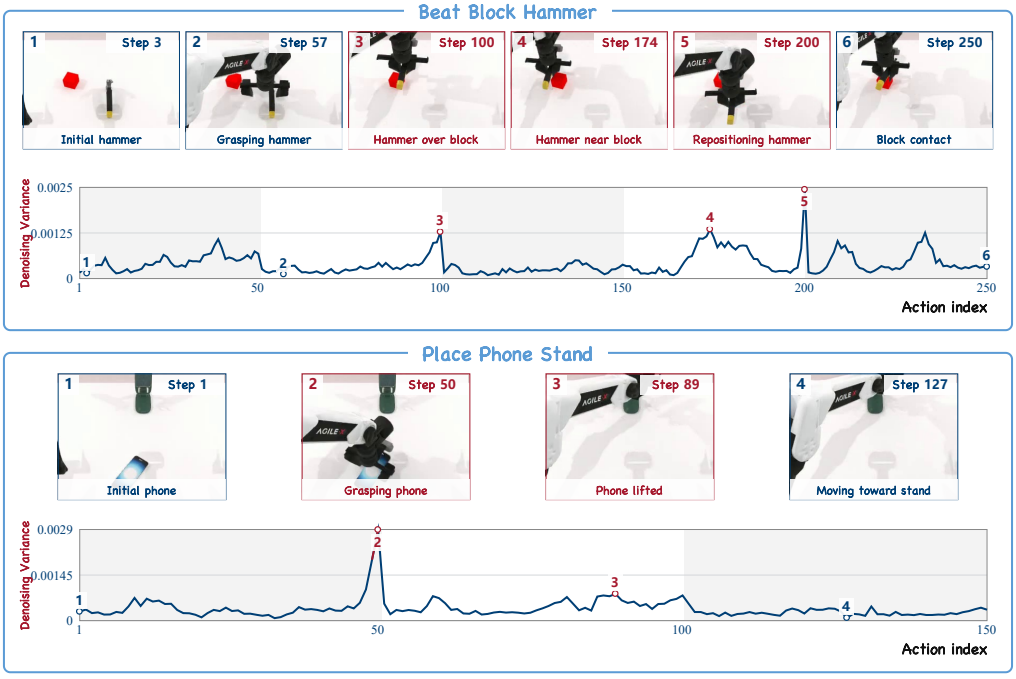

Figure 17: Action-level denoising variance during Beat Block Hammer (top) and Place Phone Stand (bottom).

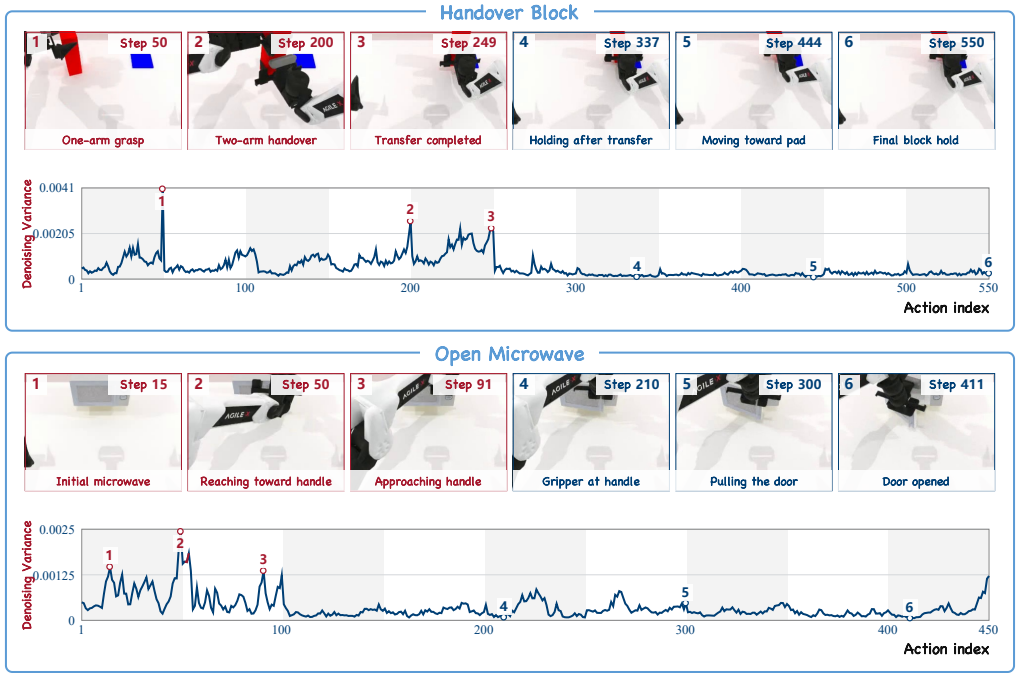

Figure 18: Action-level denoising variance during Handover Block (top) and Open Microwave (bottom).