VERIFY, ACT, DISTILL: TURNING TEST-TIME COMPUTE INTO TEST-TIME ADAPTATION FOR VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Test-time compute (TTC), which scores multiple candidate actions at inference time, is a powerful way to improve the execution performance of vision-language-action (VLA) models, but existing methods keep the policy frozen, so the gain must be recomputed at every execution. We propose *Verify, Act, Distill*, a framework that *consolidates the gains of TTC into the VLA itself*. A fixed verifier scores candidates generated by a flow-matching VLA, the weighted aggregate is executed, and the observations and aggregated chunks from successful adaptation episodes are distilled directly into LoRA parameters with the standard flow-matching loss. This reuses supervision that TTC has already produced during execution, without additional reinforcement learning or rollouts dedicated to teacher generation. On the 40 LIBERO tasks with a SmolVLA 5k checkpoint, distilling from only 10 episodes per task raises the mean success rate of verifier-free single-candidate generation from 54.1% to 70.6%, retaining about 74% of the improvement obtained by aggregating eight candidates. Applying the same aggregation to the distilled policy further raises the success rate from 76.4% to 82.3%, with a particularly large gain on LIBERO-Long (44.0% to 57.8%), and the single-candidate policy also improves on LIBERO-plus, which contains distribution shifts. These results show that using verification as a learning signal, rather than only as an execution-time selector, both retains the improvements of TTC and re-amplifies the effect of subsequent TTC.

1 INTRODUCTION

Generative vision-language-action (VLA) policies can produce diverse action chunks from the same observation, and test-time compute (TTC) that scores such candidates with a learned evaluator exploits this diversity to elicit better actions (Nakamoto et al., 2025; Kwok et al., 2025). Because existing methods keep the base policy frozen, however, the good decisions made at inference time never reach the policy's parameters: even when the same task is executed repeatedly, the gain must be recomputed every time by generating and evaluating candidates from scratch.

This inefficiency hides an unexploited learning opportunity. An execution process that evaluates candidates does not merely consume compute; it produces, by itself, high-quality supervision about which action to take for the current observation. *If this supervision were fed back into the VLA, the TTC gain could be carried over to single-candidate generation that needs no additional compute.* Moreover, if the updated VLA then proposes better candidates, distillation would strengthen TTC itself rather than merely substitute for it. This work examines this **retention and re-amplification of improvement**. We target VLAs that still have room to adapt to their deployment environment, as when a pretrained model is introduced zero-shot or with little data into a new environment or task; as a controlled proxy we use a 5k-step checkpoint taken partway through training, and we assess the applicable range of the method with a 10k checkpoint and with LIBERO-plus, which contains distribution shifts.

The core of the proposed framework, **Verify, Act, Distill**, is to *repurpose the candidate-aggregation result already computed for execution as teacher data for the policy*. A fixed verifier scores the candidates generated by the VLA, the weighted aggregate is executed, and after a small number of

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.

adaptation episodes per task the observations and aggregated chunks of the successful trials update the LoRA parameters of the action expert with the standard flow-matching loss. Adaptation requires neither a separate RL specialist nor value-gradient optimization of teacher actions. Afterwards, the verifier can be removed for lightweight single-candidate inference or, when compute allows, reused for candidate aggregation once more. Our method differs from existing approaches that distill improved actions in **where the supervision comes from**: PA-RL trains on actions selected and optimized by a critic (Mark et al., 2024), PLD distills recovery actions from a residual RL specialist (Xiao et al., 2025), and Q-Planning updates the evaluator from experience while keeping the policy fixed (Giridhar et al., 2026). We instead recover supervision from routine candidate-aggregation execution under a fixed verifier and return the improvement to *the VLA that proposes the candidates*, turning the sampling-and-verification procedure into an adaptation process that raises both single-candidate performance and the performance of subsequent TTC.

Our contributions are threefold.

1. **Direct distillation of verification-based execution decisions into the VLA:** an adaptation method that extracts the aggregated chunks already produced during TTC execution from successful trajectories and trains the VLA on them without additional RL optimization.
2. **Retention and re-amplification of TTC gains:** on the 40 LIBERO tasks with SmolVLA 5k, 10 adaptation episodes per task raise verifier-free single-candidate success from 54.1% to 70.6% (about 74% of the eight-candidate gain), and re-applying aggregation to the distilled policy raises it from 76.4% to 82.3% (44.0% to 57.8% on Long). Single-candidate gains persist for a 10k checkpoint (83.2% to 86.2%) and under the distribution shifts of LIBERO-plus.
3. **Analysis of teacher selection and applicability:** we quantify the effect of the success filter and the teacher chunk, the behavior under distribution shift, and the dependence on the number of candidates and the adaptation budget.

Retaining the improvement without a verifier is the primary goal; re-amplifying subsequent TTC is an additional benefit.

2 Related Work

Inference-time improvement through candidate evaluation. Allocating additional inference-time compute to large models (Snell et al., 2024) has been widely applied to generative robot policies based on diffusion and flow matching (Chi et al., 2024; Lipman et al., 2023). V-GPS re-ranks candidate actions with a value function (Nakamoto et al., 2025), and RoboMonkey combines candidate generation from a VLA with a learned verifier (Kwok et al., 2025). RoVer uses a robot reward model for candidate evaluation (Dai et al., 2025), and TACO selects candidates based on an anti-exploration criterion (Yang et al., 2025). CoVer focuses on the consistency between the instruction and the action (Kwok et al., 2026), and VeriSpace evaluates actions based on spatial information (Zhao et al., 2026a). Other works use the VLA's own conditional likelihood as the selection criterion, as in MG-Select (Jang et al., 2026), or switch between selection and rejection depending on the quality of the candidate set, as in BOKBO (Singh et al., 2026). All of these focus on selecting or generating a good action to execute from a candidate set; our work differs in connecting that execution-time decision to the training of the candidate-generating policy itself.

Guidance and lookahead in the generative process. Inference-time techniques for improving generation include optimizing the initial noise (Nguyen et al., 2026), guiding the flow with a Q-function (Yang et al., 2026b), combining latent-space RL with adaptive guidance (Tan et al., 2026), and dynamically adjusting inference compute according to predictive uncertainty (Choi et al., 2026). Tree search has been introduced as in VLA-Reasoner (Guo et al., 2026), hierarchical policies have been combined with world models as in τ_0-VLA (Cai et al., 2026), and future-state prediction has been exploited through test-time scaling of World Action Models (Zhao et al., 2026b). SVA distills knowledge obtained from search into a Q-evaluator and evaluates the candidates of a frozen VLA at execution time (Xie et al., 2026). In contrast, we distill the evaluation of the current candidate set directly into the VLA itself, improving the base performance of single-candidate generation.

Post-deployment adaptation and reinforcement learning. DSRL learns a policy that controls the input noise of a generative policy (Wagenmaker et al., 2025), and TT-VLA and EVOLVE-VLA adapt VLAs using environment feedback from the deployment site (Liu et al., 2026a; Bai et al., 2025). Reinforcement-learning approaches to further training VLAs include ConRFT (Chen et al., 2025a), EXPO and EXPO-FT (Dong et al., 2026b;a), and VLA-RL, SimpleVLA-RL, and TGRPO (Lu et al., 2025; Li et al., 2025; Chen et al., 2025b), which have been studied intensively. The relationship between RL objectives and generalization (Liu et al., 2026b), and frameworks that combine action-chunked PPO with imitation of self-collected successful trajectories (Wang et al., 2025), have also been explored. Our method is distinctive in using the aggregated chunks obtained in the deployment environment directly as teacher data in the standard flow-matching loss.

Reward and value models as learning signals. Vision-Language-Action-Critic (Zhai et al., 2025), Large Reward Models (Wu et al., 2026), and SARM2 (Chen et al., 2026) provide evaluation signals that support the further training of robot policies. Q-VGM converts the action gradient of a critic into a supervision signal for the velocity field of a flow (Wang et al., 2026b). The verifier used in this work is trained in advance to rank candidates by their distance to the recorded actions of official demonstrations and successful rollouts. At adaptation time, the aggregated action based on this ranking is adopted as the teacher, which completely separates the training of the evaluator from the update of the policy.

Reusing improved actions and policy distillation. Expert Iteration is a representative framework that trains the policy on search results and reuses the updated policy in subsequent search (Anthony et al., 2017). PA-RL connects Q-function-based candidate selection and local optimization to supervised policy updates, and has been applied to OpenVLA and other models (Mark et al., 2024). PLD distills corrective actions generated by a residual RL specialist into the VLA (Xiao et al., 2025). Q-Planning combines weighted execution of action candidates with continual updates of the Q-function (Giridhar et al., 2026). By using the execution-time candidate-aggregation result under a fixed verifier as the teacher, our work makes the process that generates teacher data identical to ordinary candidate evaluation and execution.

Data-generation approaches for self-improvement include imagined trajectories from world models (Yang et al., 2026a), reflection on and reuse of execution experience (Li et al., 2026), and collection of successful rollouts for novel skills (Wang et al., 2026a). Refined Policy Distillation distills VLA knowledge into a small RL expert (Jülg et al., 2025), and DEAS improves long-horizon decision making and action prediction through value learning over action sequences (Kim et al., 2025).

3 VERIFY, ACT, DISTILL

Our method converts successful rollouts of a candidate-aggregation policy into teacher data for the same VLA; the policy and verifier are fixed during adaptation, and only the LoRA parameters are updated after episode collection (Figure 1).

3.1 PROBLEM SETTING

The observation o_t consists of two camera images, an 8-dimensional proprioceptive state, and a language instruction. A pretrained VLA π_{θ_0} maps an initial noise z and o_t to a normalized action chunk $A = \pi_{\theta_0}(o_t, z) \in \mathbb{R}^{H_1 \times d}$; we use SmolVLA (Shukor et al., 2025) with $H_1 = 50$ and $d = 32$. The leading 7 dimensions are applied to the environment through g, which undoes the normalization and applies the environment-specific coordinate transformation, and only the first $H_2 = 10$ steps $g(A_{0:H_2})$ (half-open interval) are executed at each replanning point; H_1 is the generation horizon and H_2 the execution horizon.

For each task we collect K adaptation episodes, update the parameters once, and evaluate under new reset seeds. The LoRA parameters and buffer are initialized per task, no updates occur during evaluation, and nothing is transferred across tasks. During adaptation the only available information is the observations, the generated candidates, the verifier scores, and the episode-level success outcome; offline pretraining uses newly collected successful rollouts for the policy and, additionally, the official LIBERO demonstrations for the verifier. To decide the executed action from candidates

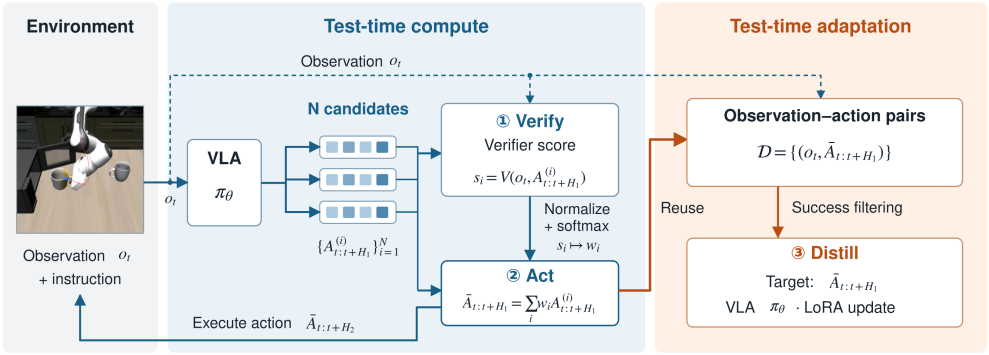


Figure 1: Overview of Verify, Act, Distill. Candidate chunks are scored and aggregated by the verifier and the leading $H_2 = 10$ steps are executed. Observations and the aggregated chunks of length $H_1 = 50$ from successful episodes are stored in a buffer and used to update the LoRA parameters. The verifier evaluates only the first H_2 steps of each candidate, whereas the teacher data include the entire planned chunk.

we use a weighted average (*candidate aggregation*) rather than the argmax; "Best-of-N" in Figure 1 denotes this operation, and for $N = 1$ a single candidate is executed without invoking the verifier.

3.2 Verify and Act: Candidate Aggregation with a Fixed Evaluator

Candidate generation. At replanning point b we draw N candidates ($N = 8$ by default) from independent noise: the first from $z^{(1)} \sim \mathcal{N}(0, I)$ and the rest from $z^{(i)} \sim \mathcal{N}(0, \alpha^2 I)$ with $\alpha = 1.2$, so that one action from the base policy's own distribution is always in the set (Appendix H compares α).

Verifier. Following RoboMonkey (Kwok et al., 2025), the action evaluator V_{act} maps the observation features and the environment actions of the first H_2 steps of a candidate to a scalar $s_i = V_{\text{act}}(o_b, g(A^{(i)}_{0:H_2}))$. Observation features are the frozen VLM connector outputs (two views, 64 tokens each, 960 dimensions) of a checkpoint trained for 20k steps, so even the 5k policy is evaluated with a stable representation; the evaluator is a 6-layer Transformer with hidden size 512, and candidate data are created and a separate verifier trained for each of the 5k, 10k, and 20k policies (Appendix A.5). The training data consist of eight candidates generated at each state of the official LIBERO demonstrations (all four suites) and of newly collected successful rollouts; letting d_i be the environment-space RMSE between the first H_2 steps of candidate i and the 10 recorded action steps of the source trajectory at that state (the demonstration actions for demonstration states and the rollout's own actions for rollout states), we minimize an all-pairs Bradley–Terry loss (Bradley, 1954) preferring i over j whenever $d_i < d_j$. The verifier thus learns to score candidates by their closeness to the recorded actions of successful trajectories. Training runs for 60k steps with batch size 16 and learning rate 10^{-4} (Appendix A.5).

Weighted execution. Scores are standardized and converted to softmax weights with temperature $\lambda = 0.3$:

$$w_i = \frac{\exp(u_i)}{\sum_{j=1}^{N} \exp(u_j)}, \qquad u_i = \frac{s_i - \max_j s_j}{\lambda(\sigma_s + \delta)}, \qquad \bar{A}_b = \sum_{i=1}^{N} w_i A^{(i)}, \tag{1}$$

where σ_s is the score standard deviation within the set and $\delta = 10^{-8}$ ($w_1 = 1$ for $N = 1$). Aggregation is performed in the normalized space and the first H_2 steps of $\bar{A}_b$ are executed. We denote weighted aggregation over N candidates by Agg-N (Agg-8 for $N = 8$) and argmax selection among eight by Sel-8; Appendix G compares the two execution rules.

3.3 DISTILL: SUPERVISED UPDATE FROM SUCCESSFUL ROLLOUTS

In each adaptation episode e, the pair $(o_b, \bar{A}_b)$ at every replanning point is stored temporarily, and with the success label $y_e \in \{0, 1\}$ the distillation buffer is

$$\mathcal{D} = \bigcup_{e:y_e=1} \{(o_b, \bar{A}_b) : b \text{ is a replanning point of episode } e\}, \tag{2}$$

with $K = 10$ episodes by default. If no episode succeeds, the update is skipped and the base policy is kept. Only the LoRA parameters ϕ of the action expert are updated; θ_0 and the verifier stay fixed. With $A_\tau = (1-\tau)\bar{A} + \tau\epsilon$, $\tau \sim U(0,1)$, $\epsilon \sim \mathcal{N}(0, I)$, the LoRA minimizes the flow-matching loss (Lipman et al., 2023; Hu et al., 2021)

$$\mathcal{L}(\phi) = \mathbb{E}_{(o,\bar{A})\sim\mathcal{D},\tau,\epsilon}\left[\left\|v_{\theta_0,\phi}(A_\tau, \tau \mid o) - (\epsilon - \bar{A})\right\|_F^2\right], \tag{3}$$

which is formulated in the data-to-noise direction and integrated in the reverse direction at inference. The teacher is the full H_1-step aggregated chunk: although only the first H_2 steps were executed, imitating the whole plan keeps the same input/output dimensions and objective as SmolVLA's pretraining (Shukor et al., 2025), and because the weights w_i are shared across time steps, the mixture chosen by evaluating the leading segment yields one coherent plan rather than a patchwork of candidates (Appendix B.1 compares a teacher masked to the executed H_2 steps). The LoRA has rank 8, scaling 16, and 0.37M trainable parameters, updated with Adam (learning rate 10^{-4}, batch 8) for 200 steps (Appendix A.7).

3.4 USAGE AFTER ADAPTATION AND COST

We evaluate the **distilled policy (single candidate)**, which runs with the same number of candidates and inference steps as the base policy and needs neither the verifier nor its observation encoder, and the **distilled policy + Agg-N**, which reuses the same fixed verifier to test the synergy between the update and candidate quality. When the K rollouts are collected during ordinary TTC execution, the only extra cost is storing the executed data and 200 LoRA steps (measured latency on real hardware remains to be examined).

4 EXPERIMENTAL RESULTS

We examine (1) whether the improvement is retained after removing the verifier, (2) whether reapplying aggregation to the distilled policy yields further gains, and (3) how the success filter and evaluation conditions affect the outcome.

4.1 SETUP AND PROTOCOL

We use the four LIBERO suites (Liu et al., 2023) (Spatial, Object, Goal, Long; 10 tasks each). SmolVLA is trained on the 5,225 successful episodes among 10,000 rollouts collected with $\pi_{0.5}$ (Intelligence et al., 2025); the main experiments use the 5k checkpoint as a proxy for an insufficiently adapted policy, and we also evaluate the 10k checkpoint and, with it, LIBERO-plus (Fei et al., 2025) under distribution shift. On clean LIBERO, each policy is adapted with reset seeds 0–9 and evaluated on held-out seeds 10–59 (50 episodes per task), which are also used for the unadapted Base and Agg-8 (data, training, compute, episode limits, and success criteria are given in Appendices A and N). We report suite means, the four-suite macro-average, per-task paired differences (mean $\pm$ SEM over tasks; Appendix N), and the number of tasks that improved / were unchanged / degraded (positive/zero/negative). On LIBERO-plus we use 105 tasks per suite (15 from each of seven perturbation categories) with adaptation seeds 0–9 and evaluation seeds 10–19; due to compute constraints, the distilled policy is evaluated on the first 35 tasks per suite (140 tasks), and Base and Agg-8 are aggregated on the same subset, which is not guaranteed to be balanced across categories.

Compared methods and notation. Base is the unadapted policy; Self distills Base's own successful rollouts without a verifier (Appendix A.3); Distill is the proposed method; and N-GRPO

Table 1: Success rates (%) on LIBERO (clean), SmolVLA 5k, evaluation seeds 10–59 (50 episodes per task, 500 per suite). Self, Distill, and N-GRPO use 10 adaptation episodes; Agg-8 is weighted aggregation with V_{act} ($\lambda = 0.3$).

Method	Spatial	Object	Goal	Long	Avg.
Base	57.6	80.4	61.0	17.4	54.1
Agg-8	80.6	94.4	86.8	44.0	76.4
N-GRPO	64.8	79.2	68.2	20.0	58.1
N-GRPO + Agg-8	84.4	95.0	86.6	50.6	79.2
Self	74.4	89.8	63.2	27.6	63.7
Self + Agg-8	86.8	**98.6**	82.6	55.2	80.8
Distill	81.8	88.0	79.4	33.2	70.6
Distill + Agg-8	**88.2**	95.4	**87.8**	**57.8**	**82.3**

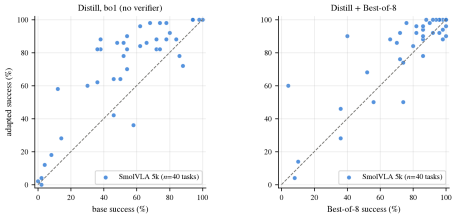


Figure 2: Per-task success rates (SmolVLA 5k, 40 tasks, 50 episodes per point, seeds 10–59). Left: Base vs. distilled policy with a single candidate (no verifier); right: Agg-8 vs. distilled policy + Agg-8. Points above the diagonal indicate improvement (left 34/1/5, right 24/6/10).

is noise-space GRPO, which keeps the VLA and verifier fixed and learns a perturbation distribution over the input noise by RL with the verifier ranking as reward, updating only from successful episodes (Appendices I and A.2). Names without a suffix denote single-candidate generation; +Agg-8 and +Sel-8 denote evaluation with weighted aggregation and argmax selection over eight candidates, respectively.

4.2 Main Results: Retention and Synergy with Candidate Aggregation

About 74% of the TTC gain is retained in a verifier-free policy. As Table 1 shows, aggregating eight candidates (Agg-8) raises the mean success rate substantially, from 54.1% to 76.4%. The policy distilled from the successful cases of this aggregated execution (Distill) then reaches 70.6% with verifier-free single-candidate generation. The per-task paired difference against Base averages $+16.5 \pm 2.7$ points; 34 of the 40 tasks improve, 1 is unchanged, and only 5 degrade (positive/zero/negative: 34/1/5). The fraction of the TTC gain that is retained, $(70.6-54.1)/(76.4-54.1)$, is about 74%.

Re-amplification of subsequent TTC through distillation. Re-applying candidate aggregation to the distilled policy ("Distill + Agg-8") yields a mean success rate of 82.3%, a paired difference of $+5.8 \pm 2.3$ points over the original Agg-8. The gain is especially large on the Long suite, which demands long-horizon planning: from 44.0% to 57.8%. At the task level, however, the breakdown is 24 improved, 6 tied, and 10 degraded, so some tasks do get worse (Figure 2). These results show that transferring the TTC gain into single-candidate generation and raising the ceiling of subsequent candidate aggregation can be achieved simultaneously.

Gains beyond self-imitation of successes. As a control, we evaluated "Self," which distills only the base policy's own successful rollouts without using the verifier (Table 1; implementation details in Appendix A.3). Self also improves single-candidate success from 54.1% to 63.7%, but Distill, which uses the verifier-aggregated chunks, reaches 70.6%, a further 6.9 points higher. By suite, Distill exceeds Self by 7.4 points on Spatial, 16.2 points on Goal, and 5.6 points on Long; only on Object is Self higher, by 1.8 points. With candidate aggregation as well, Distill records 82.3% against Self's 80.8%. Although simple self-imitation of successes gives some improvement, routing the teacher through the verifier's higher-quality aggregation consolidates the gain into the single-candidate policy more strongly, especially on suites such as Goal.

During the 10 adaptation episodes, the trial success rate was 53.5% for Self and 73.8% for Distill, and the mean number of transitions accumulated in the distillation buffer was 84 boundaries for Self versus 111 for Distill. For Self, 3 of the 40 tasks yielded no successful example during adaptation, and the base policy was used for evaluation as is. The observed additional improvement therefore reflects the combined effect of the whole adaptation process: more successful data collected and higher-quality aggregated chunks.

Table 2: Success rates (%) on LIBERO-plus, SmolVLA 10k, 105 tasks per suite, seeds 10–19 (n=1050 per suite). The lower block uses the first 35 tasks per suite (140 tasks, n=350), on which Base and Agg-8 are re-aggregated for comparison.

Method (10k, LIBERO-plus)	Spatial	Object	Goal	Long	Avg.
Base	35.5	18.8	13.0	16.2	20.9
Agg-8 (V_{act})	41.5	24.9	14.3	21.7	25.6
First 35 tasks per suite (common subset)					
Base	36.6	15.1	14.9	16.0	20.7
Agg-8 (V_{act})	40.6	**17.7**	**17.1**	**23.4**	24.7
Distill	41.1	**17.7**	16.6	22.6	24.5
Distill + Agg-8	**46.3**	16.9	15.7	22.6	**25.4**

Comparison with noise-space GRPO. Under the same adaptation/evaluation seeds and budget $K = 10$, we evaluated noise-space GRPO, which uses the verifier ranking as the reward signal. It reaches only 58.1% with single-candidate generation and 79.2% with candidate aggregation, consistently below the proposed distillation (70.6% / 82.3%). Under a limited adaptation-episode budget, distilling the aggregated action chunks as direct supervision for the VLA is far more sample-efficient than using the verifier ranking as an RL reward for a noise policy. Implementation details of the comparison are given in Appendix A.2, and additional evaluations with argmax execution and alternative advantage computations in Appendix L.

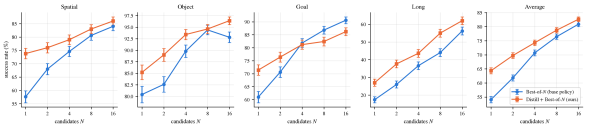
4.3 Retention of Improvement at the 10k Checkpoint

For the further-trained 10k checkpoint, the distilled single-candidate policy records 86.2% against Base's 83.2% (Table 8; paired difference $+3.0 \pm 1.4$ points). The primary goal of our method, obtaining the improvement without additional candidate generation or verifier inference, is thus also achieved for a more thoroughly trained policy. The retention effect is therefore not specific to the 5k condition, where Base's success rate was as low as 54.1%.

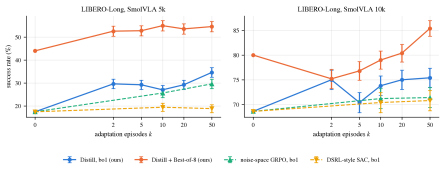
On the other hand, combining the distilled policy with candidate aggregation gives 89.8%, which does not exceed the 90.4% of the original candidate aggregation. At the 10k checkpoint, then, the effect of distillation appears mainly as improved single-candidate generation, and no re-amplification of TTC is observed. One possible factor is that a fixed verifier trained on distance to the recorded actions of successful trajectories has difficulty discriminating the small differences among the high-quality candidates generated by a sufficiently improved policy. Indeed, the verifier's validation pairwise accuracy drops from 0.82 for the 5k policy to 0.73 for the 10k policy. Because the candidate distribution itself also changes, however, this number alone does not allow us to conclude that reduced discriminative ability is the main cause. Other factors, such as reduced candidate diversity or a shift in the post-distillation generative distribution, may also contribute. Overall, retention of improvement is reliably obtained at 10k as well, but the additional amplification from candidate aggregation is more pronounced at 5k (Appendix D). Distillation and aggregation at the 20k checkpoint, comparisons with other adaptation methods, and the per-checkpoint reproducibility of the correction direction under re-sampled candidate sets are given in Appendices K, M, and J.

4.4 Ablation: Success Filter and Teacher Chunk

Distilling from all adaptation episodes without the success filter gives 64.4% / 78.8% (single candidate / aggregation), clearly below 70.6% / 82.3%; the paired differences in favor of the filter are $+6.2 \pm 1.6$ and $+3.5 \pm 1.7$ points (Appendix B). Using the highest-scoring candidate instead of the aggregated chunk as the teacher gives 67.8% / 82.5% (-2.8 ± 2.0 and $+0.2 \pm 1.8$ points relative to the aggregated teacher); we keep the aggregated chunk as the default because it matches the plan that was actually generated and executed during adaptation.



(a) Success rate vs. N for Agg-N (base policy) and distilled policy + Agg-N (one LoRA distilled with K=10); SmolVLA 5k, n=500 per point, seeds 10–59, binomial SE.



(b) Success rate vs. adaptation budget K on LIBERO-Long (n=500 per point; K=0 is Base and Agg-8); the evaluation seed set differs per K.

Figure 3: Dependence on the number of candidates and the adaptation budget. Both use the variant without the success filter; numbers are in Tables 6 and 7.

4.5 Distribution Shift and Scarce Successes

Table 2 reports the results on LIBERO-plus. Over all 105 tasks per suite, the 10k policy averages 20.9% for Base and 25.6% for Agg-8. On the common subset of 35 tasks per suite used to evaluate the distilled policy, Base reaches 20.7%, Agg-824.7%, the distilled policy (single candidate) 24.5%, and the distilled policy + Agg-825.4%. The paired difference of distilled single-candidate generation over Base is $+3.9 \pm 1.4$ points (positive/zero/negative: 43/74/23), and the difference of the aggregated variant over the original Agg-8is $+0.6 \pm 1.2$ points (positive/zero/negative: 34/76/30). Even under distribution shift with unseen perturbations, single-candidate generation improves, but the additional synergy with candidate aggregation is limited.

Moreover, in 66 of the 140 tasks no success was obtained during the adaptation trials, so no update was performed and the base policy was retained. That is, teacher data for distillation were recovered only from the remaining 74 tasks that succeeded at least once during adaptation. An evaluation in which the verifier input is replaced by latent variables predicted by a world model is given in Appendix E, and a comparison of feature spaces in Appendix F.

On the same subset, applying the distillation variant without the success filter yields a single-candidate paired difference over Base of only $+0.7 \pm 1.1$ points, so the improvement essentially vanishes.

4.6 Number of Candidates and Adaptation Budget

To characterize the dependence on the number of candidates N and the adaptation-episode budget K, we ran additional experiments with the variant that does not use the success filter (Appendix C). As shown in Figure 3a, for every number of candidates evaluated, $N \in \{1, 2, 4, 8, 16\}$, the four-suite mean success rate with the distilled policy consistently exceeds that with the original base policy. The improvement from distillation is larger in the regime of few candidates, and on the Long suite a 5.8-point improvement is maintained even with as many as $N = 16$ candidates. This indicates that the benefit of distillation is not overfitted to the number of candidates (eight) used during training. By suite, however, Goal shows saturation or a decline for $N \geq 4$.

In the adaptation-budget experiment (Figure 3b), on the Long suite with the 5k checkpoint the single-candidate success rate reaches 29.6% with only $K = 2$ and 34.6% at $K = 50$. Under the same budget $K = 10$, the proposed distilled policy achieves 27.0%, noise-space GRPO 25.6%, and DSRL-style SAC 19.4%. These trends corroborate the very high data efficiency of our approach, which recovers supervision from a handful of rollouts and adapts the policy. The evaluation seeds here are $K, \ldots, K + 49$, and the performance after collection at each budget is measured separately.

4.7 Qualitative Example

Figure 4 shows a case in which Base and ordinary aggregation (Agg-8) both fail from the same initial state while only the distilled policy + Agg-8 succeeds: where the former break down during grasping or manipulation, the higher-quality candidates of the distilled policy lead the rollout stably to completion.

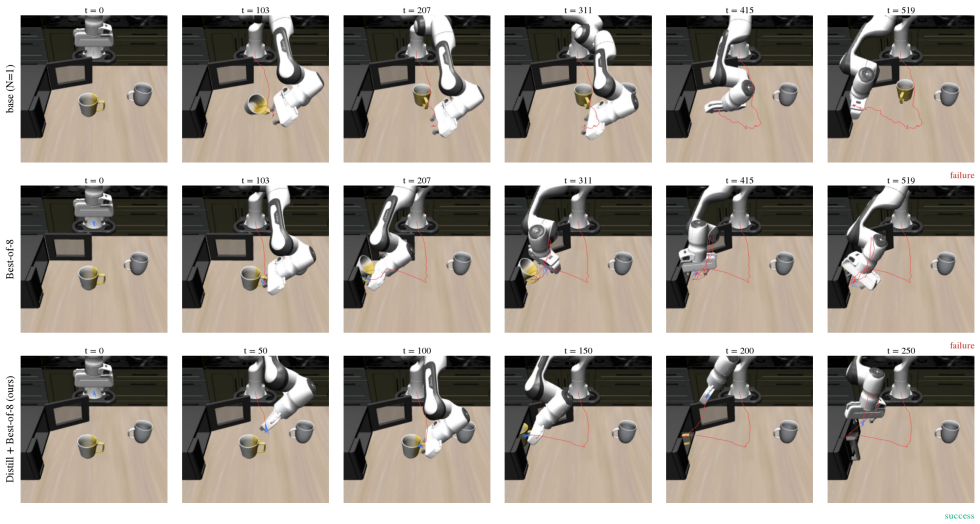


Figure 4: Rollouts on LIBERO-Long task 9 (SmolVLA 5k) from the same initial state (seed 12) for Base (N=1), Agg-8, and distilled policy + Agg-8; thin lines show each candidate's predicted trajectory at each chunk boundary and the thick line the executed weighted average, and rows show the progression of time. The seed is the first one, searching upward from seed 10, in which Base fails and the distilled policy + Agg-8 succeeds (Agg-8 alone also fails here).

5 Conclusion and Limitations

Verify, Act, Distill extends inference-time verification in VLAs from a one-off action selector into a learning signal for the policy. By extracting aggregated actions from successful candidate-aggregation rollouts and writing them back into the VLA with a single LoRA update, it retains about 74% of the TTC gain in verifier-free single-candidate generation on LIBERO and further improves subsequent candidate aggregation. TTC need not be compute consumed anew at every inference; it can generate the teacher data that improves the very policy responsible for future inference.

Limitations. Our method rests on two prerequisites: (1) the verifier must discriminate action candidates that lead to improvement, and (2) at least some adaptation episodes must succeed. The first ties the teacher to a fixed verifier trained on distances to recorded actions, so the "no additional data" property holds only for the adaptation phase, and distillation may pull the policy toward the style of those actions and reduce candidate diversity, which is consistent with the limited re-amplification at 10k (§4.3). The second binds under severe distribution shift: on LIBERO-plus, 66 of the 140 tasks yielded no success within 10 episodes and could not be updated (§4.5), and in the real world the success signal would come from a detector whose errors propagate directly into the distillation buffer. The update is also unconditional, with no validation step or fallback to the base policy, and 5 of the 40 tasks degrade with the distilled single-candidate policy and 10 when aggregation is re-applied (Figure 2). Finally, all results are in simulation with a single VLA architecture, a single training seed, and per-task LoRA parameters that are not transferred across tasks, and our cost argument counts candidates and evaluator calls rather than measured latency (§3.4). Addressing these points, in particular safeguards against regressions and real-robot validation, is left to future work.

AI use statement

We used generative AI tools (Claude and ChatGPT/Codex) to assist with implementing the experimental pipeline, aggregating results, interpreting experimental findings, and providing feedback on the methodology and experiments. Additionally, we used these tools to create plotting and analysis scripts, identify and summarize relevant literature, draft and edit the manuscript, and organize its structure. The rollout data used for training were generated by the pretrained robot policy described in Appendix A.1. The authors checked AI-assisted experimental code and numerical results, and

reviewed the AI-assisted text and interpretations against the experimental records and cited sources. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The benchmark environment, policy checkpoints, seeds used for adaptation and evaluation, and the procedure for aggregating success rates are specified in §4.1. The candidate generation and aggregation algorithm and the distillation objective are detailed in §3.2 and §3.3. The collection procedure for the pretraining data and the breakdown of its uses are given in Appendix A.1 and Table 3, the verifier's network configuration and training parameters in Appendix A.5, and the implementation details of the compared methods and the LoRA update hyperparameters in Appendices A.2 and A.7. All experiments were run on NVIDIA RTX PRO 6000 (96 GB) and RTX 6000 Ada (48 GB) GPUs (Appendix A.4).

ACKNOWLEDGMENTS

This work was supported by the Japan Science and Technology Agency (JST) through the ASPIRE program (JPMJAP2503), ACT-X program (JPMJAX25CR), and Moonshot Research and Development Program (JPMJMS263I).

REFERENCES

Thomas Anthony, Zheng Tian, and David Barber. Thinking fast and slow with deep learning and tree search, 2017. URL https://arxiv.org/abs/1705.08439.

Zechen Bai, Chen Gao, and Mike Zheng Shou. EVOLVE-VLA: Test-Time Training from Environment Feedback for Vision-Language-Action Models, December 2025. URL http://arxiv.org/abs/2512.14666. arXiv:2512.14666 [cs.RO].

Ralph Allan Bradley. Rank analysis of incomplete block designs: Ii. additional tables for the method of paired comparisons. *Biometrika*, 41(3/4):502–537, 1954. ISSN 00063444, 14643510. URL http://www.jstor.org/stable/2332730.

Xiaowei Cai, Yunuo Cai, Bingao Chen, Jingxiao Chen, Zhi Chen, Siyuan Feng, Tengyu Hou, Jingshun Huang, Han Jiang, Runkun Ju, Dong Li, Mingxiang Li, Shaowei Li, Xinchen Li, Yifan Li, Yi Liu, Zhongyuan Liu, Jianlan Luo, Junwen Miao, Ruiqi Ni, Buqing Nie, Mingjie Pan, Xinlin Ren, Jianheng Song, Jiaxu Wang, Peiqi Wang, Sen Wang, Xiaoyan Wang, Dafeng Wei, Dongming Wu, Pengwei Xie, Pu Yang, Hangjian Ye, Xiangyu Yue, Jinyu Zhang, Qinglin Zhang, Xueyong Zhao, Pengfei Zhou, and Yue Zhou. $_0$-VLA: a Hierarchical Robot Foundation Model with World-Model-Guided Test-Time Computation, August 2026. URL http://arxiv.org/abs/2608.16885. arXiv:2608.16885 [cs.RO].

Qianzhong Chen, Hau Zheng, Justin Yu, Suning Huang, Jiankai Sun, Ken Goldberg, Chuan Wen, Pieter Abbeel, Yide Shentu, Philipp Wu, and Mac Schwager. SARM2: Multi-Task Stage Aware Reward Modeling for Self Improving Robotic Manipulation, June 2026. URL http://arxiv.org/abs/2606.10305. arXiv:2606.10305 [cs.RO].

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, April 2025a. URL http://arxiv.org/abs/2502.05450. arXiv:2502.05450 [cs.RO].

Zengjue Chen, Runliang Niu, He Kong, Qi Wang, Qianli Xing, and Zipei Fan. TGRPO :Fine-tuning Vision-Language-Action Model via Trajectory-wise Group Relative Policy Optimization, September 2025b. URL http://arxiv.org/abs/2506.08440. arXiv:2506.08440 [cs.RO].

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, March 2024. URL http://arxiv.org/abs/2303.04137. arXiv:2303.04137 [cs.RO].

Hyeonbeom Choi, Daechul Ahn, Youhan Lee, Taewook Kang, Seongwon Cho, and Jonghyun Choi. SCALE: Self-uncertainty Conditioned Adaptive Looking and Execution for Vision-Language-Action Models, August 2026. URL http://arxiv.org/abs/2602.04208. arXiv:2602.04208 [cs.RO].

Mingtong Dai, Lingbo Liu, Yongjie Bai, Yang Liu, Zhouxia Wang, Rui SU, Chunjie Chen, Liang Lin, and Xinyu Wu. RoVer: Robot Reward Model as Test-Time Verifier for Vision-Language-Action Model, October 2025. URL http://arxiv.org/abs/2510.10975. arXiv:2510.10975 [cs.RO].

Perry Dong, Kuo-Han Hung, Tian Gao, Dorsa Sadigh, and Chelsea Finn. EXPO-FT: Sample-Efficient Reinforcement Learning Finetuning for Vision-Language-Action Models, August 2026a. URL http://arxiv.org/abs/2605.25477. arXiv:2605.25477 [cs.RO].

Perry Dong, Qiyang Li, Dorsa Sadigh, and Chelsea Finn. EXPO: Stable Reinforcement Learning with Expressive Policies, April 2026b. URL http://arxiv.org/abs/2507.07986. arXiv:2507.07986 [cs.LG].

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, Jinlan Fu, Jingjing Gong, and Xipeng Qiu. LIBERO-Plus: In-depth Robustness Analysis of Vision-Language-Action Models, December 2025. URL http://arxiv.org/abs/2510.13626. arXiv:2510.13626 [cs.RO].

Varun Giridhar, Anant Khandelwal, Jeremy A. Collins, Ignat Georgiev, and Animesh Garg. Beyond Imitation: Self-Improving Robot Policies via Off-Policy Q-Planning, August 2026. URL http://arxiv.org/abs/2608.21204. arXiv:2608.21204 [cs.RO].

Wenkai Guo, Guanxing Lu, Haoyuan Deng, Zhenyu Wu, Yansong Tang, and Ziwei Wang. VLA-Reasoner: Empowering Vision-Language-Action Models with Reasoning via Online Monte Carlo Tree Search, March 2026. URL http://arxiv.org/abs/2509.22643. arXiv:2509.22643 [cs.RO].

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models, October 2021. URL http://arxiv.org/abs/2106.09685. arXiv:2106.09685 [cs.CL].

Physical Intelligence, 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. $_{0.5}$: a Vision-Language-Action Model with Open-World Generalization, April 2025. URL http://arxiv.org/abs/2504.16054. arXiv:2504.16054 [cs.LG].

Suhyeok Jang, Dongyoung Kim, Changyeon Kim, Youngsuk Kim, and Jinwoo Shin. Verifier-free Test-Time Sampling for Vision-Language-Action Models, July 2026. URL http://arxiv.org/abs/2510.05681. arXiv:2510.05681 [cs.RO].

Tobias Jülg, Wolfram Burgard, and Florian Walter. Refined Policy Distillation: From VLA Generalists to RL Experts, August 2025. URL http://arxiv.org/abs/2503.05833. arXiv:2503.05833 [cs.RO].

Changyeon Kim, Haeone Lee, Younggyo Seo, Kimin Lee, and Yuke Zhu. DEAS: DEtached value learning with Action Sequence for Scalable Offline RL, October 2025. URL http://arxiv.org/abs/2510.07730. arXiv:2510.07730 [cs.LG].

Jacky Kwok, Christopher Agia, Rohan Sinha, Matt Foutter, Shulu Li, Ion Stoica, Azalia Mirhoseini, and Marco Pavone. RoboMonkey: Scaling Test-Time Sampling and Verification for Vision-Language-Action Models, July 2025. URL http://arxiv.org/abs/2506.17811. arXiv:2506.17811 [cs.RO].

Jacky Kwok, Xilun Zhang, Mengdi Xu, Yuejiang Liu, Azalia Mirhoseini, Chelsea Finn, and Marco Pavone. Scaling Verification Can Be More Effective than Scaling Policy Learning for Vision-Language-Action Alignment, February 2026. URL http://arxiv.org/abs/2602.12281. arXiv:2602.12281 [cs.RO].

Baicheng Li, Dong Wu, Zike Yan, Xinchen Liu, Lusong Li, Zecui Zeng, and Hongbin Zha. Reflection-Based Task Adaptation for Self-Improving VLA, April 2026. URL http://arxiv.org/abs/2510.12710. arXiv:2510.12710 [cs.RO].

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, September 2025. URL http://arxiv.org/abs/2509.09674. arXiv:2509.09674 [cs.RO].

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling, February 2023. URL http://arxiv.org/abs/2210.02747. arXiv:2210.02747 [cs.LG].

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking Knowledge Transfer for Lifelong Robot Learning, October 2023. URL http://arxiv.org/abs/2306.03310. arXiv:2306.03310 [cs.AI].

Changyu Liu, Yiyang Liu, Taowen Wang, Qiao Zhuang, James Chenhao Liang, Wenhao Yang, Renjing Xu, Qifan Wang, Dongfang Liu, and Cheng Han. On-the-Fly VLA Adaptation via Test-Time Reinforcement Learning, April 2026a. URL http://arxiv.org/abs/2601.06748. arXiv:2601.06748 [cs.RO].

Jijia Liu, Feng Gao, Bingwen Wei, Xinlei Chen, Qingmin Liao, Yi Wu, Chao Yu, and Yu Wang. What Can RL Bring to VLA Generalization? An Empirical Study, January 2026b. URL http://arxiv.org/abs/2505.19789. arXiv:2505.19789 [cs.LG].

Guanxing Lu, Wenkai Guo, Chubin Zhang, Yuheng Zhou, Haonan Jiang, Zifeng Gao, Yansong Tang, and Ziwei Wang. VLA-RL: Towards Masterful and General Robotic Manipulation with Scalable Reinforcement Learning, May 2025. URL http://arxiv.org/abs/2505.18719. arXiv:2505.18719 [cs.RO].

Max Sobol Mark, Tian Gao, Georgia Gabriela Sampaio, Mohan Kumar Srirama, Archit Sharma, Chelsea Finn, and Aviral Kumar. Policy agnostic rl: Offline rl and online rl fine-tuning of any class and backbone, 2024. URL https://arxiv.org/abs/2412.06685.

Mitsuhiko Nakamoto, Oier Mees, Aviral Kumar, and Sergey Levine. Steering Your Generalists: Improving Robotic Foundation Models via Value Guidance, February 2025. URL http://arxiv.org/abs/2410.13816. arXiv:2410.13816 [cs.RO].

Duc Minh Nguyen, Bao-Ngoc Dao, Tung M. Luu, Binh Gia Nguyen, Vinh Tong, Anji Liu, Vu N. Duong, Dung D. Le, Daniel Sonntag, Trung Le, Ngan Le, Jan Peter, An Thai Le, Minh Nhat Vu, Mathias Niepert, Khoa D. Doan, Duy M. H. Nguyen, and Vien Anh Ngo. Self-Improving VLA Policies: Selected Diffusion Noise for Spurious-Robust Action Smoothing, June 2026. URL http://arxiv.org/abs/2606.14084. arXiv:2606.14084 [cs.RO].

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. SmolVLA: A Vision-Language-Action Model for Affordable and Efficient Robotics, June 2025. URL http://arxiv.org/abs/2506.01844. arXiv:2506.01844 [cs.LG].

Anya Singh, Cabrel Happi, Jai Relan, Varun Nair, and Vidyut Baradwaj. BOKBO (Best of K Bad Options): Calibrated Abstention for VLA Policies, May 2026. URL http://arxiv.org/abs/2605.30660. arXiv:2605.30660 [cs.LG].

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters, August 2024. URL http://arxiv.org/abs/2408.03314. arXiv:2408.03314 [cs.LG].

Derek Ming Siang Tan, Shailesh Shailesh, Srikrishna Iyer, William Wei Jie Teo, Yuanliang Ju, Qiao Gu, and Guillaume Sartoretti. RL^2-VLA: Adaptive RL Latent Compositional Steering with Test-Time Scaling for Vision-Language-Action Models, July 2026. URL http://arxiv.org/abs/2607.26991. arXiv:2607.26991 [cs.RO].

Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering Your Diffusion Policy with Latent Space Reinforcement Learning, June 2025. URL http://arxiv.org/abs/2506.15799. arXiv:2506.15799 [cs.RO].

Maggie Wang, Lars Osterberg, Stephen Tian, Ola Shorinwa, Joseph Bowkett, Daniel Pastor, Jiajun Wu, and Mac Schwager. InSight: Self-Guided Skill Acquisition via Steerable VLAs, September 2026a. URL http://arxiv.org/abs/2606.24884. arXiv:2606.24884 [cs.RO].

Si-Cheng Wang, Tian-Yu Xiang, Xiao-Hu Zhou, Mei-Jiang Gui, Xiao-Liang Xie, Shi-Qi Liu, Shuang-Yi Wang, Ao-Qun Jin, and Zeng-Guang Hou. VLA Model Post-Training via Action-Chunked PPO and Self Behavior Cloning, September 2025. URL http://arxiv.org/abs/2509.25718. arXiv:2509.25718 [cs.RO].

Ziqian Wang, Rui Zhang, Yitian Liu, Xingjian Mao, Minqian Wang, and Yao Mu. Q-vgm: Q-guided value-gradient matching for offline-to-online rl of flow-matching vla policies, 2026b. URL https://arxiv.org/abs/2606.08015.

Yanru Wu, Weiduo Yuan, Ang Qi, Vitor Guizilini, Jiageng Mao, and Yue Wang. Large Reward Models: Generalizable Online Robot Reward Generation with Vision-Language Models, March 2026. URL http://arxiv.org/abs/2603.16065. arXiv:2603.16065 [cs.RO].

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, October 2025. URL http://arxiv.org/abs/2511.00091. arXiv:2511.00091 [cs.CV].

Xinyi Xie, Zican Hu, Zhanyu Liu, Yicheng Dong, Wenhao Wu, Zhenhong Sun, Haoran Li, Chunlin Chen, Zhi Wang, and Pichao Wang. Look before you leap: Distilling tree search into action evaluation for frozen vla models, 2026. URL https://arxiv.org/abs/2607.03751.

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, April 2026a. URL http://arxiv.org/abs/2602.11075. arXiv:2602.11075 [cs.RO].

Liuhaichen Yang, Zhuang Jiang, Chenchao Sheng, and Zezhi Tang. Guided Action Flow: Q-Guided Inference for Flow-Matching Vision-Language-Action Policies, August 2026b. URL http://arxiv.org/abs/2607.02092. arXiv:2607.02092 [cs.RO].

Siyuan Yang, Yang Zhang, Haoran He, Ling Pan, Xiu Li, Chenjia Bai, and Xuelong Li. Steering Vision-Language-Action Models as Anti-Exploration: A Test-Time Scaling Approach, December 2025. URL http://arxiv.org/abs/2512.02834. arXiv:2512.02834 [cs.RO].

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, September 2025. URL http://arxiv.org/abs/2509.15937. arXiv:2509.15937 [cs.RO].

Guiyu Zhao, Longteng Guo, Junyou Zhu, Jun Fu, Yanghong Mei, Bin Cao, Jie Jiang, Xingjian He, and Jing Liu. VeriSpace: Spatially Grounded Action Verification for Vision-Language-Action Models, June 2026a. URL http://arxiv.org/abs/2606.10568. arXiv:2606.10568 [cs.RO].

Zesen Zhao, Minkyoung Cho, Hui shen, Boyuan Zheng, Kunxiao Gao, Yulong Cao, and Z. Morley Mao. Test-Time Scaling for World Action Models via Zero-Shot Geometric Evaluation, July 2026b. URL http://arxiv.org/abs/2607.17454. arXiv:2607.17454 [cs.RO].

A Implementation Details

A.1 Data Collection and Policy Training

The policy training data consist of 10,000 episodes collected by running a pretrained $\pi_{0.5}$ checkpoint on every task of the four LIBERO suites (250 episodes per task = 100 in the standard environment + 150 in a failure-injection environment). Failure injection deliberately adds an offset to the 6 pose dimensions of the environment action that increases linearly within a chunk (ee_ramp, amplitude $r \in \{0.6, \ldots, 1.6\}$, tuned per suite so that the success rate falls between 0.14 and 0.76). Of the collected data, only the 5,225 successful episodes that completed their task (808,143 frames) were used for policy training (the official LIBERO demonstrations are not used for behavior cloning (BC) of the policy; they are used only for training the verifier and the world model described below). Preliminary experiments showed that mixing the official demonstrations with the collected rollouts for BC degraded policy performance, so policy pretraining uses only the 5,225 newly collected successful episodes.

SmolVLA is based on lerobot's `smolvla_base`; all parameters, including the vision encoder, were trained for 20k steps (batch size 32, lerobot's default learning rate, random seed 1000), and checkpoints were saved at 5k, 10k, 15k, and 20k steps.

Table 3 summarizes the use of training data for each trained component.

Table 3: Breakdown of offline training data by use. This is distinct from the success-filtered distillation buffer used in the adaptation phase.

Trained component	Data and supervision
SmolVLA (BC, 20k steps)	Only the 5,225 successful episodes among the newly collected rollouts. Official LIBERO demonstrations not used.
Verifier	Eight candidates generated at states from both the official LIBERO demonstrations (all four suites) and the newly collected successful rollouts. Supervision is the RMSE to the $H_2 = 10$ recorded action steps at each state. The 10k candidate dataset comprises 104,294 states.
World model (appendix)	Official LIBERO demonstrations and newly collected rollouts. For rollouts, both successful and failed data are used.

A.2 Hyperparameters of the Compared Methods

Noise-space GRPO (Table 1): Our own implementation of noise-space GRPO. The verifier is V_{act} (trained for 60k steps), the advantage is computed with the top1 scheme, and the noise is edited by additive perturbation. The initial value of $\log \sigma$ is -1, the learning rate is 10^{-4}, the success filter is applied, and no μ trust region is used. Adaptation uses $K = 10$ (seeds 0–9, softmax-aggregated execution), and evaluation uses seeds 10–59 with both a single candidate (one candidate generated from the adapted mean μ) and +Agg-8(weighted aggregation). **Noise-space GRPO and DSRL-style SAC (Table 7):** Adaptation seeds are $0, \ldots, K-1$, and evaluation uses the 50 seeds immediately following, with single-candidate generation. GRPO uses argmax-selection adaptation and the success filter. SAC is a port of the official DSRL implementation (dsrl_pi0), with noise-space replacement (pure scheme), a $-1/0$ reward (-1 per chunk and 0 at the terminal state upon task success), a TanhNormal distribution clipped to $[-1, 1]$, an ensemble mean over 10 critics, discount factor 0.99, actor / critic / temperature learning rates of 10^{-4} / 3×10^{-4} / 3×10^{-4}, batch size 16, UTD ratio 1, and updates starting after 300 transitions have accumulated. SAC does not use the verifier.

A.3 Setup of the Self-Imitation Control

"Self" in Table 1 runs the SmolVLA 5k checkpoint with $N = 1$ at every replanning boundary. Because the LoRA output weights are zero-initialized, the data-collection policy is identical to the base policy. A single input noise $z_0 \sim \mathcal{N}(0, I)$ is drawn from the standard normal distribution, and the candidate weight is trivially 1. The verifier is not used at all during the adaptation trials. The pair of observation o_b and generated chunk A_b at each boundary is stored in a temporary buffer, and

after the episode ends, only the pairs from episodes judged successful are added to the distillation buffer $\mathcal{D}$. The teacher is the full $H_1 = 50$ steps in the model's normalized space, of which the first $H_2 = 10$ steps are executed in the environment. This corresponds to a control in which the teacher of the proposed method (Distill), the aggregated chunk $\bar{A}_b$, is replaced by the chunk A_b generated by the base policy alone. The LoRA is distilled for 200 steps on the successes among the 10 collected episodes, and the 40 tasks are evaluated with seeds 10–59. For the 3 tasks whose distillation buffer was empty because no success was obtained, the base policy was evaluated without any weight update. Self is evaluated with single-candidate generation, and "Self + Agg-8" with weighted aggregation of eight candidates using the same V_{act} at evaluation time.

A.4 Compute Environment

All experimental evaluations were run on NVIDIA RTX PRO 6000 (96 GB) and RTX 6000 Ada (48 GB) GPUs, with 2–4 jobs placed in parallel on a single GPU. Adaptation for one task (10 episodes of Agg-8execution + 200 LoRA update steps) takes 5–10 minutes, and 100 evaluation episodes (50 single-candidate + 50 Agg-8) take 20–40 minutes. One row of the main experiment (Table 1) over all 40 tasks corresponds to roughly 20–40 GPU-hours.

A.5 Verifier Architecture and Training

The candidate dataset was built by generating eight candidates with the mixed-α-scale scheme, per target policy checkpoint, at states from both the official LIBERO demonstrations (all four suites) and the newly collected successful rollouts (104,294 states for the 10k policy dataset). The supervision is the ranking induced by the environment-space RMSE to the $H_2 = 10$ recorded action steps of the source trajectory at each state. Training uses an all-pairs Bradley–Terry loss for 60k steps with batch size 16, learning rate 10^{-4}, weight decay 0.01, 5% warmup, and gradient clipping at 1.0. The evaluator is a Transformer with hidden size 512 and 6 layers. The latent evaluator V_{latent} uses identical hyperparameters, with only the input replaced by latent variables predicted by the world model. For each evaluated checkpoint (5k / 10k / 20k), V_{act} and V_{latent} were trained separately under the same compute budget. The validation pairwise accuracy at the end of training (60k steps), reported as V_{act} / V_{latent}, was 0.82 / 0.70 for the 5k policy, 0.73 / 0.62 for the 10k policy, and 0.72 / 0.60 for the 20k policy. This pairwise accuracy measures agreement with the distance-based teacher ranking; the final improvement of the policy is evaluated by the closed-loop success rates reported in the main text.

A.6 World Model Architecture

The world model is "Pi05SplitLatentWorldModel," which takes a history of the past 3 latent frames, the proprioceptive state, and a candidate action chunk as input and outputs the predicted latent $\hat{\boldsymbol{z}}_{t+H_2}$ after executing the first H_2 steps. It has a two-branch residual structure with an action-conditioned branch and an action-independent branch, feature dimension 512, depth 4 (2 for the independent branch), 8 attention heads, and MLP hidden dimension 2048. It was trained for 20k steps on the official LIBERO demonstrations and the newly collected rollouts (both successes and failures). The L1 error on validation data is 3.09, a 36% improvement over the naive baseline that simply copies the previous state (error 4.86), with a cosine similarity of 0.96.

A.7 Distillation Details

The LoRA targets follow SmolVLA's standard fine-tuning configuration: the Query/Value projections of the action expert, the state/action input and output projections, and the time MLP layers, for a total of 74 tensors (0.37M parameters). The success buffer collected over $K = 10$ episodes averaged 111 boundaries per task in the 5k main experiment and 84 boundaries in the self-imitation control. Updates were run for 200 steps with batch size 8, the Adam optimizer (learning rate 10^{-4}), bfloat16 automatic mixed precision (autocast), and gradient-norm clipping at 1.0.

Table 4: Comparison of distillation teacher, success filter, and verifier (SmolVLA 5k, 10 adaptation episodes, evaluation seeds 10–59, 500 episodes per suite). Success rates (%) with (a) single-candidate generation and (b) weighted aggregation of eight candidates. "Mean" uses the weighted-average chunk and "Max" the highest-scoring candidate as the teacher. The main-method setting is "Mean teacher, filter on, V_{act}." The V_{latent} rows also use V_{latent} for candidate evaluation.

(a) Single-candidate generation

Teacher	Filter	Verifier	Spatial	Object	Goal	Long	Avg.
Mean	On	V_{act}	81.8	88.0	79.4	33.2	70.6
Max	On	V_{act}	78.4	86.8	78.2	27.8	67.8
Mean	Off	V_{act}	73.8	85.2	71.4	27.0	64.4
Max	Off	V_{act}	—	—	—	32.6	—
Mean	Off	V_{latent}	62.2	84.4	67.2	25.8	59.9

(b) Candidate aggregation (eight candidates)

Teacher	Filter	Verifier	Spatial	Object	Goal	Long	Avg.
Mean	On	V_{act}	88.2	95.4	87.8	57.8	82.3
Max	On	V_{act}	89.4	95.4	91.2	54.0	82.5
Mean	Off	V_{act}	83.0	94.6	82.4	55.0	78.8
Max	Off	V_{act}	—	—	—	56.2	—
Mean	Off	V_{latent}	79.2	87.0	79.4	43.8	72.4
No adaptation (V_{latent})			74.0	89.8	81.2	41.6	71.7

B Ablation on the Success Filter and Teacher Selection

Table 4 reports ablations that vary the type of teacher chunk, the presence of the success filter, and the verifier input. Results with single-candidate generation are in Table 4(a) and those with candidate aggregation (eight candidates) in Table 4(b).

Our standard setting is "aggregated-mean teacher, success filter on, V_{act}." Note that the distillation results with V_{latent} were measured without the success filter and should therefore be compared against the corresponding filter-off rows for V_{act}.

B.1 Distilling Only the Executed Segment (H_2-Only Teacher)

The main recipe (§3.3) imitates the full $H_1 = 50$-step aggregated chunk even though the verifier scores, and the environment executes, only its first $H_2 = 10$ steps. To test whether the unexecuted $H_1 - H_2$ steps are necessary as a teacher, we distilled from the same success buffers as the main experiment (SmolVLA 5k, $K = 10$, success filter on, V_{act}, 200 LoRA steps) while masking the flow-matching loss in Eq. equation 3 to the first H_2 steps of each chunk; the remaining 40 steps contribute no gradient. Everything else, including the evaluation seeds 10–59, is identical.

Table 5: Full-chunk teacher ($H_1 = 50$, main recipe) vs. H_2-only teacher (loss masked to the executed 10 steps). Success rates (%) on LIBERO (clean), SmolVLA 5k, seeds 10–59 (n=500 per suite). The full-chunk rows are those of Table 1.

Evaluation	Teacher	Spatial	Object	Goal	Long	Avg.
Single candidate	Full chunk (H_1)	**81.8**	88.0	79.4	33.2	70.6
	Executed steps (H_2)	79.0	**92.0**	**84.2**	**41.0**	**74.1**
+Agg-8	Full chunk (H_1)	88.2	95.4	87.8	57.8	82.3
	Executed steps (H_2)	**90.4**	**99.0**	**92.6**	**64.2**	**86.6**

As Table 5 shows, restricting the teacher to the executed segment does not hurt; on average it helps. With single-candidate generation, the H_2-only variant reaches 74.1% against 70.6% for the full-chunk teacher, with gains on Object (+4.0), Goal (+4.8), and Long (+7.8) and a loss on Spatial

Table 6: Numbers underlying Figure 3a: success rates (%) of Agg-N / distilled policy + Agg-N, SmolVLA 5k, n=500 per cell.

N	Spatial	Object	Goal	Long	Avg.
1	57.6 / **73.8**	80.4 / **85.2**	61.0 / **71.4**	17.4 / **27.0**	54.1 / **64.4**
2	68.0 / **76.0**	82.6 / **89.0**	70.6 / **76.4**	26.0 / **37.6**	61.8 / **69.8**
4	74.6 / **79.0**	89.8 / **93.4**	**81.8** / 81.2	36.6 / **43.6**	70.7 / **74.3**
8	80.6 / **83.0**	94.4 / **94.6**	**86.8** / 82.4	44.0 / **55.0**	76.4 / **78.8**
16	84.0 / **86.0**	92.8 / **96.4**	**90.6** / 86.2	56.2 / **62.0**	80.9 / **82.7**

Table 7: Success rate (%) on LIBERO-Long as a function of the adaptation-episode budget K (n=500). Parameters are frozen after K adaptation episodes and then evaluated. DSRL-style is our port of the official implementation (dsrl_pi0), with a $-1/0$ reward (time penalty), a TanhNormal distribution restricted to $[-1, 1]$, an ensemble of 10 critics, and prior-replacement composition. The GRPO rows are the comparison method (verifier-ranking reward, top1 advantage, success filter, argmax adaptation). The distilled policy is the variant without the success filter (Appendix B), with adaptation seeds $0, \ldots, K-1$ and the 50 seeds immediately following as evaluation seeds. Base performance is 17.4 at 5k and 68.6 at 10k (seeds 10–59); Agg-8is 44.0 at 5k and 80.0 at 10k.

Policy (Base)	Method	K=2	K=5	K=10	K=20	K=50
5k (17.4)	Distill	**29.6**	**29.2**	**27.0**	**29.2**	**34.6**
	Distill + Agg-8	52.6	52.8	55.0	53.6	54.6
	N-GRPO	—	—	25.6	—	29.6
	DSRL	—	—	19.4	—	18.8
10k (68.6)	Distill	**75.0**	70.4	**73.8**	**75.0**	**75.4**
	Distill + Agg-8	75.2	76.8	79.0	80.4	85.4
	N-GRPO	—	—	71.2	—	71.4
	DSRL	—	—	70.4	—	70.8

(-2.8); with candidate aggregation it is higher on every suite (86.6% vs. 82.3% on average, $+6.4$ on Long). The unexecuted tail of the aggregated chunk is therefore not needed for retention or re-amplification, and supervising only the steps that the verifier actually scored and the environment actually executed is at least as effective. A plausible reading is that the tail, which was never validated by execution and is re-planned away at the next boundary, adds noise to the target, whereas the full chunk mainly preserves the dimensions of SmolVLA's pretraining objective. We keep the full-chunk teacher in the main experiments for consistency with the rest of the paper; these numbers come from a single training seed, so the suite-level differences should be read as indicative, but the H_2-only variant is an attractive default and removes the need to imitate unexecuted actions.

C Supplementary Experiments on the Number of Candidates and the Adaptation Budget

Table 6 reports results for varying the number of candidates N, and Table 7 for varying the number of adaptation episodes K. They correspond to the plots in Figures 3a and 3b in the main text. Both were run with the variant that does not use the success filter; in the N sweep, the same distilled LoRA is evaluated while varying only the number of candidates at evaluation time.

Because the K sweep uses adaptation seeds $0, \ldots, K-1$ and evaluation seeds $K, \ldots, K+49$, the Base numbers evaluated on seeds 10–59 cannot be strictly compared as a common baseline for conditions other than $K = 10$. In addition, although DSRL-style SAC uses the same number of episodes, it does not use the recorded-action ranking supervision used to train the verifier, so the total amount of available supervision is not perfectly matched. Bold indicates the highest success rate in each column.

D Details of Retention at the 10k Checkpoint

Table 8 summarizes the success rates of the base policy, candidate aggregation, and the distilled policy at the 10k checkpoint.

Table 8: Success rates (%) on LIBERO (clean) with SmolVLA 10k. Measured under the same protocol as Table 1 (evaluation seeds 10–59, n=500 per suite).

Verifier	Method	Spatial	Object	Goal	Long	Avg.
—	Base	85.0	95.0	84.0	68.6	83.2
V_{act}	Agg-8	90.2	**99.6**	**91.8**	80.0	90.4
	Distill	89.4	96.0	85.4	73.8	86.2
	Distill + Agg-8	91.8	98.4	89.8	79.0	89.8
V_{latent}	Agg-8	**92.4**	99.0	91.4	**82.6**	**91.4**
	Distill	91.0	97.0	80.8	75.6	86.1
	Distill + Agg-8	90.2	99.4	86.8	80.8	89.3

As Table 8 shows, the single-candidate success rate of the distilled policy rises from 83.2% to 86.2%, so the improvement is reliably retained in inference without additional verifier evaluation. On the other hand, stacking candidate aggregation on the distilled policy does not exceed the original candidate aggregation (Agg-8), confirming that retention of improvement and re-amplification of subsequent TTC behave differently.

E Verifier Input Representation: Direct Action Evaluation vs. World-Model Latents

V_{latent} is a verifier whose input is the latent state $\hat{z}_{t+H_2}$, H_2 steps ahead, predicted from the candidate action by the trained world model (Appendix A.6) in the same $2 \times 64 \times 960$ token space as the observation. It shares the same encoder, the same candidate sets, the same supervision, the same BT loss, and the same number of training steps (60k) as V_{act}, differing only in the form of the input representation and in whether the world model is interposed. Table 9 compares the two.

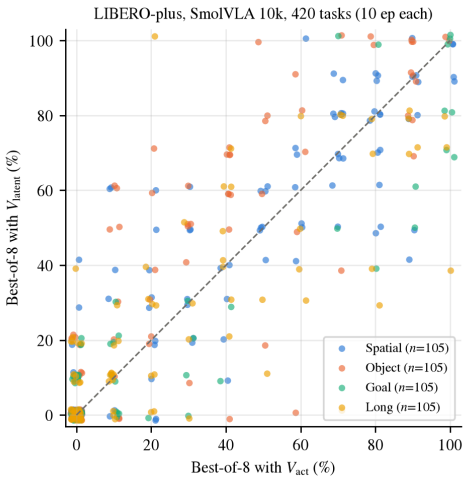


Figure 5: Scatter plot comparing Agg-8success rates on all 420 LIBERO-plus tasks (SmolVLA 10k): V_{act} (horizontal) vs. V_{latent} (vertical), 10 episodes per point (jitter of ± 1.5 points added to reduce overlap). Points above the diagonal favor V_{latent}. The upward shift is especially pronounced on the Object suite.

Table 9: Comparison of verifier inputs (success rate %; Agg-8is softmax aggregation with λ=0.3; each verifier trained for 60k steps). Clean uses evaluation seeds 10–59 (n=500 per suite); plus uses 105 tasks × 10 episodes (seeds 10–19). Δ is the per-task paired difference $V_{\text{latent}} - V_{\text{act}}$ (mean±SEM). The comparison as a distillation teacher is based on the variant without the success filter (K=10, single-candidate generation).

Method	Spatial	Object	Goal	Long	Avg.
clean 5k					
Base	57.6	80.4	61.0	17.4	54.1
Agg-8 V_{act}	**80.6**	**94.4**	**86.8**	**44.0**	**76.4**
Agg-8 V_{latent}	74.0	89.8	81.2	41.6	71.7
Δ	-6.6 ± 3.7	-4.6 ± 1.7	-5.6 ± 2.5	-2.4 ± 2.2	-4.8 ± 1.3
clean 5k, as distillation teacher					
Distill (V_{act})	73.8	85.2	71.4	27.0	64.4
Distill (V_{latent})	62.2	84.4	67.2	25.8	59.9
Δ	-11.6 ± 4.3	-0.8 ± 4.1	-4.2 ± 3.5	-1.2 ± 1.2	-4.5 ± 1.8
clean 10k					
Base	85.0	95.0	84.0	68.6	83.2
Agg-8 V_{act}	90.2	**99.6**	**91.8**	80.0	90.4
Agg-8 V_{latent}	**92.4**	99.0	91.4	**82.6**	**91.4**
Δ	$+2.2 \pm 1.5$	-0.6 ± 0.6	-0.4 ± 2.7	$+2.6 \pm 1.9$	$+0.9 \pm 0.9$
clean 10k, as distillation teacher					
Distill (V_{act})	89.4	96.0	85.4	73.8	86.2
Distill (V_{latent})	91.0	97.0	80.8	75.6	86.1
Δ	$+1.6 \pm 2.7$	$+1.0 \pm 1.2$	-4.6 ± 2.3	$+1.8 \pm 3.2$	-0.1 ± 1.3
plus 10k					
Base	35.5	18.8	13.0	16.2	20.9
Agg-8 V_{act}	41.5	24.9	**14.3**	**21.7**	25.6
Agg-8 V_{latent}	**43.6**	**32.5**	13.0	20.6	**27.4**
Δ	$+2.1 \pm 1.6$	$+7.6 \pm 1.7$	-1.2 ± 1.1	-1.1 ± 1.8	$+1.8 \pm 0.8$

The improved/tied/degraded task counts corresponding to the differences Δ in Table 9 are 7/9/24 on clean 5k, 16/13/11 on clean 10k, and 127/210/83 on LIBERO-plus for aggregated execution, and 14/4/22 on clean 5k and 15/8/17 on clean 10k for the distilled-policy evaluation. As Table 9 and Figure 5 show, V_{latent} yields a clear improvement on LIBERO-plus, particularly on the Object suite. However, because introducing the world model changes both the abstraction level of the input representation and the model structure, we have not yet identified the detailed mechanism behind this advantage.

F Encoder and Feature-Space Ablation

The feature-space comparison in this section uses a SmolVLA 4k checkpoint trained on successful rollouts from the Object and Long suites. The success rate of this base policy is 81% on Object and 36% on Long. Table 10 summarizes the differences between this setup and the main experiments, which use the 5k/10k/20k checkpoints trained on all four suites.

The verifier's candidate dataset was generated with $\alpha = 1.2$, whereas the closed-loop evaluation in Figure 6 used $\alpha = 2.0$. The feature-space comparison was also evaluated separately under the $\alpha = 1.2$ condition.

In Figure 6, the success rates on Object / Long are 81% / 36% for the base policy, 93% / 66% for V_{act}, 88% / 45% for V_{latent} with DINOv2 features, and 97% / 71% for V_{latent} with SmolVLA features. Under this comparison condition, candidate evaluation through a world model that predicts SmolVLA's own features achieved the highest success rates. Although DINOv2 $\geq$ SmolVLA held in every individual diagnostic evaluation of frame-level perturbation invariance, state-decoding accuracy, and the world model's action sensitivity, the ranking of these offline feature diagnostics did not match the ranking of final closed-loop success rates.

Table 10: Differences in setup between the feature-space comparison (Figure 6) and the main experiments (clean LIBERO). "a1.2" in the candidate-dataset names denotes the noise multiplier.

Item	Feature-space comparison (Figure 6)	Main experiments
Policy	BC-trained on the two suites Object and Long. 4k checkpoint (`run_bc_success/checkpoints/004000`)	BC-trained on successful rollouts from all four suites. 5k / 10k / 20k checkpoints of `run_bc_4suite`
Candidate data	`cand_4k_a1.2`. Object and Long candidates generated from the 4k policy	`cand_4suite_<ckpt>_a1.2`. Four suites, generated per policy checkpoint
Verifier training	20k steps	60k steps
World model	`run1` (SmolVLA features) and `run_dinov2` (DINOv2 features)	`run_4suite` (SmolVLA features, all four suites)
Candidate noise at evaluation	$\alpha = 2.0$	$\alpha = 1.2$
Execution rule	Select the highest-scoring of 8 candidates (argmax)	Softmax-weighted aggregation of 8 candidates ($\lambda = 0.3$)
Evaluation scale	Object and Long, 25 episodes per task (250 per suite). Seed set differs from the main experiments	All four suites, 50 episodes per task (500 per suite), seeds 10–59
Feature encoder	20k checkpoint of `run_bc_success`	20k checkpoint of `run_bc_4suite`

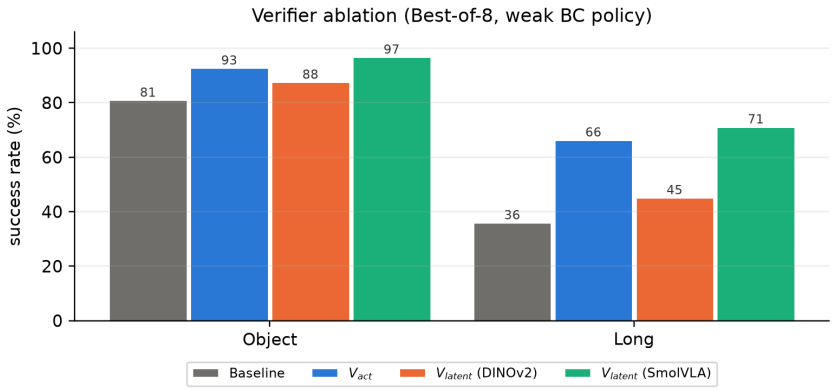


Figure 6: Comparison of verifier input representations with a SmolVLA 4k checkpoint trained on the Object and Long suites. Eight candidates, noise scale $\alpha = 2.0$, argmax selection, 25 evaluation episodes per task (250 per suite). Verifiers are trained for 20k steps. The base policy, V_{act} with direct action input, and V_{latent} predicting DINOv2 features or SmolVLA features through the world model are compared. See Table 10 for details of the setup.

G Softmax-Aggregated Execution vs. Argmax Selection

On the Long suite with the 5k checkpoint, evaluated with V_{act} over 50 episodes per task (seeds 0–49), argmax selection (Sel-8) achieved a success rate of 34.4%, whereas softmax aggregation ($\lambda = 0.3$, Agg-8) achieved 44.2% and softmax aggregation with V_{latent} achieved 43.2%. On the Long suite with the 10k checkpoint, argmax selection gave 77.0%, softmax aggregation 78.2%, and softmax aggregation with V_{latent} 82.2%. When GRPO adapted under argmax selection was evaluated with softmax aggregation, its success rate on 5k Long was 51.8% (53.4% under argmax evaluation).

H Effect of the Candidate Noise Scale α

Table 11 shows success rates on the Object and Long suites as the noise multiplier α used for candidate generation is varied. For every checkpoint, $\alpha = 1.2$ achieved the most consistently high success rate within the explored range $\{1.2, 1.5, 2.0, 3.0\}$.

Table 11: Success rate as a function of the candidate noise scale α (N=8, V_{latent}, n=100 per point, Long / Object suites). Bold marks the maximum in each row including the base policy (ties included). The explored range is $\{1.2, 1.5, 2.0, 3.0\}$.

Checkpoint	Suite	Base	α=1.2	1.5	2.0	3.0
5k	Object	**82**	**82**	59	39	44
5k	Long	17	**42**	21	7	6
10k	Object	96	**98**	93	64	71
10k	Long	67	**74**	60	27	21
20k	Object	**99**	98	91	84	75
20k	Long	**90**	79	68	33	30

I Detailed Formulation of Noise-Space GRPO

I.1 Formulation

N-GRPO edits the initial noise fed to the frozen VLA with a state-dependent perturbation vector η. The noise-editing policy takes as input a 1928-dimensional vector formed by concatenating patch-averaged features from the frozen observation encoder for the two views (2×960 dimensions) and the normalized proprioceptive state (8 dimensions). A 2-layer MLP with hidden width 64 outputs a 32-dimensional mean vector μ (zero-initialized), which together with a state-independent variance parameter $\log \sigma$ (initial value -1) defines a diagonal multivariate Gaussian. A perturbation η sampled from this distribution is broadcast along the time axis of the action chunk and added. Among the generated candidates, candidate 0 is an ordinary sample without perturbation, and candidates $i = 1, \ldots, 7$ are formed by adding the same η_i to independently generated noise. The advantage uses the top1 scheme ($+1$ for the candidate with the highest verifier score and $-1/(N-1)$ for the others), and updates use learning rate 10^{-4}, PPO clip ratio 0.2, replay buffer size 256, and 3 epochs. As update constraints, we introduce a "success filter," which defers updates until the end of the episode and applies them only upon success, and a "μ trust region," which restricts the parameter norm to $\|\mu\| \leq 1$.

The above is the basic configuration of the comparison implementation; the main experiment in Table 1 does not use the μ trust region, and the adaptation-budget experiment (Table 7) uses argmax-selection adaptation.

J Reproducibility of the Correction Direction under Re-Sampling of the Candidate Set

We examined how consistent the verifier's correction direction is when the candidate set is independently re-drawn at the same state. We loaded 32 fixed LIBERO-plus states (8 per suite, extracted from diag80 rollouts) at each of the 5k / 10k / 20k checkpoints and independently re-sampled the candidate set ($N = 8$, $\alpha = 1.2$) $M = 32$ times at each state. Let $A_m^{(i)}$ be the candidates obtained in the m-th draw, $\langle A \rangle_m = \frac{1}{N} \sum_i A_m^{(i)}$ their simple mean, and $w_{m,i}$ the softmax weights; we define the TTC correction vector as $\Delta_m = \sum_i w_{m,i}(A_m^{(i)} - \langle A \rangle_m)$ (350 dimensions = $H_1 = 50$ steps $\times$ 7 environment-action dimensions). We computed the following metrics: (i) the normalized resultant length $R = \|\frac{1}{M} \sum_m \Delta_m / \|\Delta_m\|\|$ (a measure of directional agreement whose expected value for completely uncorrelated directions is about $1/\sqrt{M} \approx 0.18$), (ii) the split-half cosine similarity (the cosine similarity between the mean correction vectors of a random two-way split of the M sets, averaged over 10 trials), (iii) SDR = $\|\frac{1}{M} \sum_m \Delta_m\| / \frac{1}{M} \sum_m \|\Delta_m\|$ (the ratio of the norm of the mean correction vector to the mean of the norms), and (iv) the effective number of candidates $N_{\text{eff}} = 1/\sum_i w_{m,i}^2$ (the number of candidates effectively averaged by the softmax).

The measurements (5k / 10k / 20k) were: resultant length R of 0.229 / 0.228 / 0.209, split-half cosine similarity of 0.22 / 0.22 / 0.13, SDR of 0.053 / 0.053 / 0.043, and an effective number of candidates $N_{\text{eff}} \approx 2.1$ in all conditions (Table 12). Bootstrap 95% confidence intervals for the paired

Table 12: Within-state reproducibility of the TTC correction vector Δ_m (32 states, 32 candidate-set draws per state). (a) Directional reproducibility metrics; (b) correction norm, effective number of candidates, and score standard deviation. R is the resultant length, Split-cos the split-half cosine similarity, and SDR the directional stability ratio.

(a) Directional reproducibility of the correction vector

Checkpoint	R	Split-cos	SDR
5k	0.229 ± 0.007	0.219 ± 0.025	0.053 ± 0.004
10k	0.228 ± 0.007	0.216 ± 0.027	0.053 ± 0.004
20k	0.209 ± 0.005	0.129 ± 0.023	0.043 ± 0.002

(b) Correction norm and candidate statistics

Checkpoint	$\lvert\Delta\rvert$/spread	N_{eff}	Score std. dev.
5k	0.649	2.13	2.65
10k	0.658	2.11	1.60
20k	0.647	2.14	1.36

differences between checkpoints at the same states (5k−20k and 10k−20k) excluded zero for every metric, indicating significant differences. In contrast, the split-half Frobenius cosine similarity of the second-moment structure (the weighted covariance $D_m = \sum_i (w_{m,i} - 1/N)\, e_{m,i} e_{m,i}^\top$ with $e_{m,i} = A_m^{(i)} - \langle A \rangle_m$) was $+0.02$ (the null reference with shuffled weights is 0) and showed no difference between checkpoints. These results reveal that at the most thoroughly trained 20k checkpoint, the variability of the correction direction across re-draws of the candidates at inference time increases markedly compared with 5k and 10k. The paired differences of 5k minus 20k (bootstrap 95% CI) were $+0.020$ [0.006, 0.034] for R, $+0.090$ [0.037, 0.143] for the split-half cosine similarity, and $+0.010$ [0.003, 0.019] for SDR. Likewise, the differences of 10k minus 20k were $+0.020$ [0.011, 0.029], $+0.087$ [0.047, 0.127], and $+0.010$ [0.005, 0.015], respectively (the null reference for R is 0.184).

K Behavior of Distillation and Candidate Aggregation at the 20k Checkpoint

The effect size of test-time compute (TTC) varies greatly with the training stage (checkpoint) of the base policy (see Tables 1, 8, and 13: on 5k Long, Agg-8improves performance by $+26.6$ points, whereas on 20k Long it degrades performance by -11 points). Yet when this 20k checkpoint is evaluated on LIBERO-plus, with its unseen perturbations, Agg-8again shows a clear improvement of $+6.6$ points ($n = 837$). Nevertheless, the additional gain from stacking candidate aggregation on top of an adaptation method remained very small (Table 13: for GRPO, softmax distillation, advantage-weighted flow matching, online LoRA, and preference optimization alike, the additional effect of combined aggregation was limited, and for the distillation methods it even decreased). On the other hand, the single-candidate performance of the distilled policy exceeds the base policy by $+6.9 \pm 2.1$ points (positive/zero/negative: 36/29/15), confirming that about 70% of the gain obtained through TTC ($+10.1$ points) is reliably transferred to lightweight single-candidate inference that requires no verifier.

To probe the background of this phenomenon, we analyzed in detail the within-state reproducibility of the verifier's correction direction, as described above (Appendix J). When the candidate set was re-sampled $M = 32$ times at the same state, the split-half cosine similarity of the correction vector Δ_m was 0.22 at 5k and 0.22 at 10k, but dropped to 0.13 at 20k. This suggests that at highly trained checkpoints, the correction direction varies widely across re-draws of the candidates, which may be one factor that hinders the synergy from candidate aggregation after adaptation.

Table 13: Success rates (%) on LIBERO-plus with SmolVLA 20k and per-task paired differences from Base (points, mean±SEM). Evaluation seeds 10–19. (a) Aggregated over the 837 Level-5 tasks on which Base has a positive success rate; (b) aggregated over a fixed set of 80 tasks (32 tasks for PRO only). VWFM is advantage-weighted flow matching over all candidates, Online is online LoRA updating, and PRO is preference optimization (Appendix M).

(a) All 837 tasks

Method	Success	Δ Base	Δ Agg-8
Base	37.3 ± 1.1	—	-6.6 ± 0.7
Agg-8	43.9 ± 1.2	$+6.6 \pm 0.7$	—
N-GRPO + Agg-8	43.8 ± 1.2	$+6.5 \pm 0.7$	-0.1 ± 0.6

(b) Fixed 80-task subset

Method	Success	Δ Base	Δ Agg-8
Base	38.8 ± 3.7	—	-10.1 ± 2.2
Agg-8	48.9 ± 3.7	$+10.1 \pm 2.2$	—
N-GRPO + Agg-8	46.1 ± 3.7	—	-2.8 ± 1.9
Distill	45.6 ± 1.8	$+6.9 \pm 2.1$	-3.3 ± 2.1
Distill + Agg-8	45.2 ± 1.8	$+6.4 \pm 2.2$	-3.6 ± 2.1
VWFM	42.9	$+4.1 \pm 2.3$	—
VWFM + Agg-8	49.1	—	$+0.2 \pm 2.1$
Online	42.0	$+3.2 \pm 2.1$	—
Online + Agg-8	49.0	—	$+0.1 \pm 1.8$
PRO	—	$+3.4 \pm 2.8$	—
PRO + Agg-8	—	—	-0.3 ± 2.5

L Additional Experiments on Noise-Space GRPO

L.1 Evaluation on Clean LIBERO with Argmax Selection

Table 14 evaluates with the argmax scheme (Sel-8), which selects and executes the highest-scoring candidate at each replanning point. The evaluation seeds for the base policy and the candidate-selection policy are 0–49; GRPO uses adaptation seeds 0–9 and evaluation seeds 10–59.

Table 14: GRPO success rates (%) on clean LIBERO, 50 episodes per task. Base and Sel-8use evaluation seeds 0–49 (argmax selection); the GRPO rows use adaptation seeds 0–9 and evaluation seeds 10–59.

Checkpoint	Method	Spatial	Object	Goal	Long
SmolVLA 5k	Base	62.6	80.0	59.6	17.4
	Sel-8	73.6	88.2	**81.8**	34.4
	N-GRPO	59.8	83.8	68.2	23.2
	N-GRPO+Sel-8	**80.2**	**88.2**	78.8	**53.4**
SmolVLA 10k	Base	85.0	96.4	84.0	65.6
	Sel-8	**89.6**	**97.8**	**86.8**	**77.0**
	N-GRPO	85.0	96.2	86.0	69.4
	N-GRPO+Sel-8	84.0	96.8	85.8	73.0

L.2 Evaluation of the 10k Checkpoint on LIBERO-plus

Table 15 reports the LIBERO-plus success rates of noise-space GRPO for the 10k checkpoint. We evaluate single-candidate generation and combined argmax selection, and also compare the type of verifier used for candidate evaluation (V_{act} vs. V_{latent}).

Table 15: GRPO success rates (%) on LIBERO-plus 10k, 105 tasks × 10 episodes, argmax selection (Sel-8). GRPO applies the success filter and the μ trust region. Parentheses indicate the verifier used for the reward and for candidate selection.

Method	Spatial	Object	Goal	Long	Avg.
N-GRPO (V_{act})	35.9	21.6	13.8	18.0	22.3
N-GRPO+Sel-8 (V_{act})	42.1	29.6	**15.3**	**21.3**	27.1
N-GRPO+Sel-8 (V_{latent})	42.4	32.5	13.4	16.7	26.3

L.3 Analysis of Degradation Causes and Protective Mechanisms

We found two main pathways for performance degradation during adaptation. (1) Harmful learning from failed episodes: introducing the success filter and updating only from successful trials completely prevents this degradation (single-candidate 5k Object improves from 75.4% to 83.8%, and 10k Long +Sel-8from 63.4% to 73.0%). On 5k Long, where the success rate is only about 20%, the adaptation gain was preserved even though 80% of the data were discarded for the update (52.6% $\rightarrow$ 53.4%). (2) Excessive divergence of the noise mean μ: this occurs specifically when V_{latent} is used as the reward ($\|\mu\|$ diverges to 2.7), and introducing the μ trust-region constraint reliably suppresses the divergence (success rate 18% $\rightarrow$ 63%). V_{act} tends to select directions that pull distributional drift back toward the center (restoring coefficient $-0.035 \rightarrow -0.053$, growing stronger as the policy degrades), whereas the restoring force of V_{latent} is only about 40% of that and does not act strongly even when the policy has degraded. We also examined a KL regularization term (coefficient $\beta_{\mathrm{KL}} \in \{0.03, 0.1, 0.3\}$), but it over-constrains the useful adaptation range and substantially reduces the adaptation gain (on 5k Long, the success rate drops from 53% to 46–49%).

L.4 Comparison of Advantage Computation Schemes

Table 16 compares four advantage computation schemes for noise-space GRPO: "top1," "zscore," "rank," and "tophalf." The comparison was run on 5k Long in clean LIBERO and on the Object and Long suites of LIBERO-plus.

Table 16: Comparison of advantage computation schemes (n=500 / n=1050). Assignment schemes that dilute the weight of the top candidates tend to weaken the correct update direction. Settings with V_{latent} reward and the μ trust region were also tested.

Advantage scheme	Clean 5k Long	Plus Object V_{act}	Plus Object V_{latent}	Plus Long V_{latent}
top1	**53.4**	29.6	32.5	**16.7**
zscore	52.6	**30.0**	30.8	16.4
rank	48.6	28.0	31.7	15.4
tophalf	48.4	27.4	**33.0**	14.7

L.5 Reproducibility of the Selected Input Noise and Output Actions

Over 25 states × 16 repetitions, the pairwise cosine similarity of the optimal noise vectors $\boldsymbol{z}^*$ selected by the verifier was 0.007 at 5k and 0.001 at 20k, and the resultant lengths were 0.26 / 0.25 (the expected value for random direction vectors is 0.25). Thus, at both checkpoints the directional agreement of the selected noise itself was at the level of random vectors. However, in the real-environment action space produced through that noise, the selected actions agreed closely with one another (cosine similarity 0.59 at 5k and 0.37 at 20k). The immediate improvement in the expected verifier score at boundaries where a policy update occurred was only $+0.028$ at 5k and $+0.011$ at 20k (less than 1% of the within-set score variance), so the local score improvement itself was very small. The update-dynamics indicators (coherence 0.72 / 0.72, gradient-clip rate, probability ratio, and the evolution of $\log \sigma$) were nearly identical between the two checkpoints.

M Comparison of Adaptation Methods at the 20k Checkpoint on LIBERO-plus

Paired comparisons were run with evaluation seeds 10–19 on the fixed 80-task set (Table 13) or on subsets of 16–32 tasks.

- **Distillation-based methods:** Distilling the softmax-aggregated chunk improved single-candidate generation by $+6.0 \pm 2.1$ points, advantage-weighted distillation (one sample drawn) by $+3.9 \pm 2.0$ points, and weighted distillation over all candidates (exact) by $+4.1 \pm 2.3$ points, but in every case the additional gain from combined candidate aggregation (+Agg-8) was at most $+0.2$ points.
- **Online LoRA updating:** A scheme that performs a one-pass update over the cumulative buffer each time a successful episode is obtained improved single-candidate generation by $+3.2 \pm 2.1$ points and +Agg-8by $+0.1 \pm 1.8$ points. An attempt to also update the verifier online with the new candidates during adaptation caused a large degradation of -5.0 ± 3.0 points.
- **Preference optimization:** A method using a FlowPRO-style objective ($\beta = 3.5$, L_{PRO} only, 32 tasks) showed no effect, with a +Agg-8improvement of -0.0 ± 2.8 points.
- **Iterative distillation:** An iterative scheme that regenerates the teacher data in each of 4 rounds yielded only $+1.2 \pm 6.0$ points (16 tasks), inferior to a single batch distillation.
- **Many candidates ($N = 64$):** Increasing the number of candidates to 64 improved the offline oracle-selection performance by $+26.5\%$, but the actual closed-loop TTC success rate decreased by -5.5%, and combined with TTA it decreased by -4.7 ± 4.2 points. Noise sampling by diagonal Gaussian expansion (RoboMonkey-style, $K = 64$) did not improve even the offline oracle, because the temporal structure of the chunk collapses.
- **Other RL variants:** We tested Flow-GRPO-style FPO (using the CFM loss ratio), Neighbor GRPO (introducing a locally geometry-normalized distance, $+1.9 \pm 4.6$ points), and a candidate-set hybrid (mixing the base policy with the distillation delta, -0.6 ± 2.9 points), but none produced a significant improvement.

N Evaluation Protocol Details

For paired differences d_j between two methods on the same task, the mean $\bar{d}$ and its standard error over T tasks are

$$\mathrm{SEM}(\bar{d}) = \frac{s_d}{\sqrt{T}}, \qquad s_d^2 = \frac{1}{T-1} \sum_{j=1}^{T} (d_j - \bar{d})^2. \tag{4}$$

Alongside the mean difference we report the number of tasks that improved / were unchanged / degraded (positive/zero/negative). The binomial standard errors shown as error bars in the figures treat each episode as independent; they do not account for within-task correlation between trials or for uncertainty due to random seeds. The maximum episode length is 280 environment steps for Spatial and Object, 300 for Goal, and 520 for Long, and success is determined by the environment's native criterion. Candidate generation and distillation use torch seed 0; variance across training seeds is not evaluated.