ACTIONCRITIC: ACTION-AWARE SELF-CRITICISM FOR RELIABLE VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-Language-Action (VLA) models provide a scalable interface for mapping multimodal observations and language instructions to robot actions. However, most VLA policies execute predicted actions without explicitly assessing whether those actions remain appropriate under the current scene configuration, where a locally incorrect command can cause an irreversible task failure. In this paper, we introduce ActionCritic, a shared-parameter VLA policy that internalizes action verification, corrective instruction generation, and dynamic re-planning within a single multimodal representation. Rather than relying on computationally expensive external simulators or a separate learned correction policy, ActionCritic proactively evaluates the physical suitability of proposed event-triggered key actions against the observation history and task semantics prior to execution. Unsuitable action proposals are preempted and translated into concise atomic corrections, which directly condition subsequent action generation to enable bounded, local replanning. Training follows a two-stage schedule that first establishes critic capability with QA supervision and then mixes planning and correction-conditioned replanning while retaining a low-weight QA objective. Furthermore, ActionCritic learns a deployment-oriented decision boundary by leveraging both successful and failed trajectories while excluding failed motor commands from imitation targets. Extensive evaluations across the SIMPLER, LIBERO, and LIBERO-Plus benchmarks, together with selected real-world ALOHA tasks, show improvements in task success and critic discrimination in the evaluated settings; the event-triggered design limits verification to selected key actions.

1 INTRODUCTION

Vision-Language-Action (VLA) models have emerged as a promising foundation for general-purpose robotic manipulation. By adapting large-scale vision-language representations to action prediction, these models connect semantic task understanding, visual perception, and low-level control within a common architecture. RT-1 and RT-2 demonstrated that transformer policies can scale across tasks and transfer knowledge from heterogeneous visual and linguistic data to robotic control Brohan et al. (2022); Zitkovich et al. (2023). Open X-Embodiment and Octo broadened cross-embodiment training O'Neill et al. (2024); Octo Model Team et al. (2024), while recent work has investigated open model development, cognition–action decomposition, spatial representation, and temporal memory Kim et al. (2024); Li et al. (2024a); Qu et al. (2025); Shi et al. (2025). Collectively, these advances establish VLA models as an increasingly capable interface for instruction-conditioned visuomotor control.

However, a standard VLA prediction is treated as a control output rather than as a proposal whose physical suitability should first be examined. This design is vulnerable to the distribution shift inherent in closed-loop manipulation, where small errors in object localization, approach direction, gripper timing, or contact estimation can move the robot away from the demonstrated state distribution, after which subsequent actions may compound the initial error. The problem is especially acute at ***key actions***, including grasp, release, insertion, and other contact transitions. Whereas an inaccurate free-space motion can often be corrected later, an incorrectly timed closure or release may alter the scene irreversibly and eliminate a viable recovery path. Reliable manipulation therefore requires not only generating a plausible key action, but also verifying that it remains appropriate for the current observation history and task context before execution.

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.

As illustrated in Figure 1, existing approaches typically obtain corrective signals from either an external supervisor or an auxiliary predictive or recovery model, using verbal reflection, explicit failure reasoning, motion-based correction, or dedicated recovery policies Shinn et al. (2023); Xia et al. (2025); Dai et al. (2024); Duan et al. (2024). VLA-in-the-Loop, for example, couples a world model with online policy correction for robotic grasping Xu et al. (2026). These systems demonstrate the value of structured feedback, but their third-party correctors introduce an additional pathway alongside the acting policy, increasing intervention latency and creating another source of compounding error.

To overcome the described limitations, we introduce ActionCritic, a shared-parameter policy with an event-triggered verification wrapper that investigates a complementary question: *can a VLA model evaluate and revise its own proposed actions before a consequential error is executed?* The vision-language backbone of a VLA already contains representations for interpreting scene geometry, task semantics, and action-relevant context; restricting this backbone to one-way action generation leaves these capabilities underutilized.

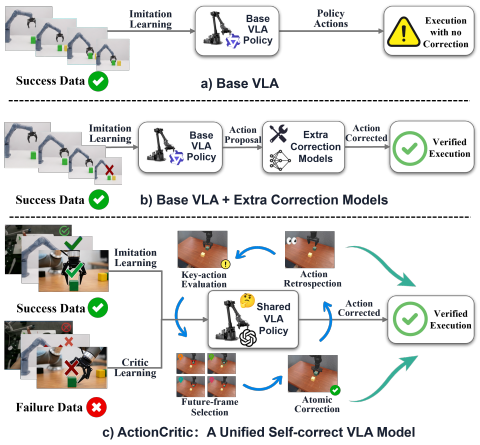


Figure 1: Comparison of VLA execution and correction paradigms. (a) A standard VLA policy executes predicted actions without explicit verification. (b) Modular correction methods pass action proposals to additional correction models before execution. (c) ActionCritic learns planning from successful trajectories and criticism from both successful and failed trajectories, enabling a shared VLA policy to verify and correct its own actions before execution.

We therefore formulate action verification as an additional conditional reasoning task within the same model, combining *action planning, key-action evaluation, atomic correction generation, and corrective re-planning*. To preserve computational efficiency, ActionCritic further employs an *event-triggered* verification mechanism in which only proposed action chunks containing *key actions* undergo proactive evaluation; the trigger is a deployment-time component rather than a learned correction policy.

Learning a useful critic requires more than successful demonstrations or binary classification of outcome images. Conventional policy learning primarily retains successful trajectories, leaving many failed rollouts unused even though they expose precisely the state–action configurations that cause deployment errors. From these complementary signals, we formulate four QA tasks for critic training. *Action retrospection* and *future-frame selection* establish a bidirectional correspondence between actions and visual state changes, while *key-action evaluation* and *atomic-correction QA* teach the model to assess a proposed action and prescribe a concise local correction. Together, these tasks ground criticism in the conditional relationship among visual history, candidate action, and physical consequence, equipping the shared VLA with both action-evaluation and corrective capabilities. Moreover, ActionCritic is compatible with different VLA action representations. Planning retains the continuous, discretized, or token-based decoder of the underlying policy, while criticism and correction are expressed through constrained multimodal interfaces.

Our contributions are summarized as follows:

- We introduce event-triggered self-criticism within a shared VLA policy loop, allowing the agent to verify key actions and avoid a separate learned corrector pathway.
- We propose a compact atomic corrective interface that enables interpretable local replanning without requiring computationally heavy predictive models.
- We develop four complementary spatial-transition QA tasks and a two-stage critic-then-policy curriculum training scheme that leverages massive failure data to refine decision boundaries without corrupting imitation targets.

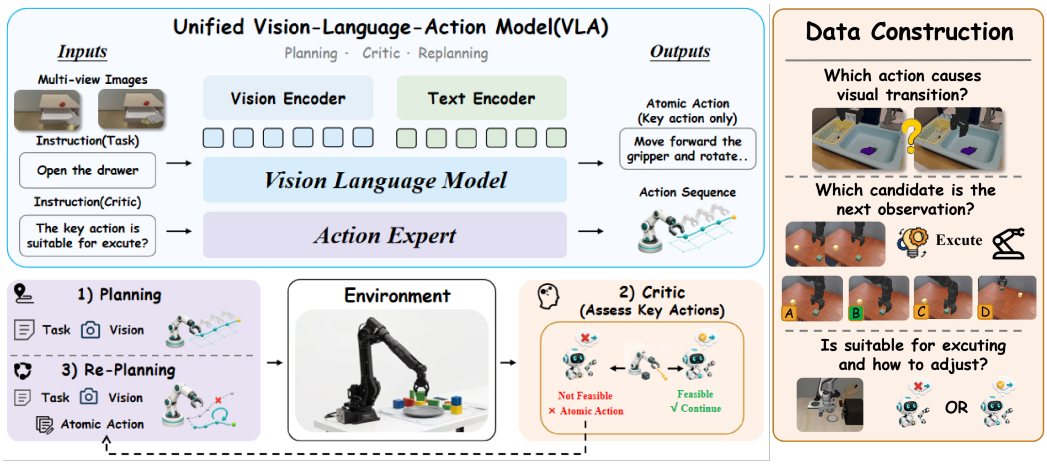


Figure 2: Overview of ActionCritic. The shared VLA proposes an action chunk and verifies its earliest key action before execution. Accepted actions proceed to the environment. Rejected actions are converted into an atomic corrective instruction, which conditions a revised proposal. Four complementary QA tasks first establish critic capability; mixed planning and re-planning then retain QA supervision at weight 0.1.

- We establish a multi-benchmark protocol that evaluates task success, critic discrimination, correction behavior, and inference efficiency across simulated and selected physical tasks.

2 Related Work

Vision-Language-Action Models. VLA models couple large-scale vision-language representations with action prediction, transferring semantic and visual knowledge into instruction-conditioned robotic control. RT-1 established scalable transformer policies over diverse manipulation tasks, while RT-2 demonstrated that knowledge acquired from web-scale vision-language data can transfer to robot actions Brohan et al. (2022); Zitkovich et al. (2023). Recent work improves policy expressiveness along complementary dimensions: CogACT separates cognition from action generation, SpatialVLA introduces explicit spatial representations, and MemoryVLA incorporates perceptual-cognitive memory Li et al. (2024a); Qu et al. (2025); Shi et al. (2025); π_0 and DexVLA employ flow-based or diffusion action experts for continuous control Black et al. (2024); Wen et al. (2025). Reasoning-oriented policies such as CoT-VLA further show that intermediate visual reasoning can benefit action generation Zhao et al. (2025). Despite these advances, the dominant objective remains action generation, with each prediction typically treated as an output to execute rather than a proposal to verify. ActionCritic is complementary to these advances: it augments the training interface and execution loop so that the shared policy can evaluate and revise its own key actions before commitment.

Reflection and Failure Recovery. Reflection and recovery methods improve an agent by feeding information about unsuccessful or suboptimal behavior back into subsequent decisions. Reflexion established verbal self-feedback as a mechanism for iterative improvement in language agents Shinn et al. (2023). Robotics systems instantiate this principle through different feedback sources and correction interfaces: Yell at Your Robot incorporates online human language corrections Shi et al. (2024); Phoenix performs motion-based self-reflection for fine-grained action correction Xia et al. (2025); RACER learns a language-guided recovery policy from rich failure descriptions Dai et al. (2024); and AHA employs a vision-language model to detect and reason about manipulation failures Duan et al. (2024). Collectively, these approaches demonstrate that structured feedback can improve robustness, but the corrective signal is typically supplied by a human or processed through a third-party reflection, failure-analysis, or recovery pathway distinct from the acting policy. In contrast, ActionCritic trains the shared VLA itself to evaluate a proposed key action before commitment and to express the required local adjustment as an atomic correction that directly conditions

re-planning. This unification couples evaluation and recovery to the policy representation while avoiding continuous human supervision or a separate corrective policy at deployment.

3 METHOD

We introduce ActionCritic, a shared-parameter Vision-Language-Action policy that treats consequential actions as proposals to be verified before execution. As illustrated in Figure 2, a shared model provides four prompt-conditioned interfaces: action planning, key-action evaluation, atomic-correction generation, and correction-conditioned re-planning. At inference, ordinary motions follow the receding-horizon execution path of the underlying VLA, while a deployment-time event trigger requests verification only for key actions such as grasping, releasing, and contact transitions. A rejected proposal is converted into a compact local correction that conditions the same policy to re-plan. By sharing the visual-semantic representation across these interfaces, ActionCritic provides short-horizon pre-execution verification and local replanning without online video generation or an external inverse-dynamics model.

3.1 PROBLEM FORMULATION

We formulate language-conditioned manipulation as a partially observable sequential decision problem. At time t, the robot receives

$$o_t = (I_t, s_t) \in \mathcal{O}, \tag{1}$$

where I_t is the visual observation and s_t is the available proprioceptive state. Given a task instruction L, the policy conditions on a finite history $H_t = (o_{t-h+1}, \ldots, o_t)$ and predicts an action chunk

$$A_t = \pi_\theta(H_t, L) = (a_{t,0}, \ldots, a_{t,K-1}). \tag{2}$$

The parameterization of $a_{t,j}$ is inherited from the underlying VLA and may include translational, rotational, and gripper commands in continuous, discretized, or tokenized form.

Imitation learning optimizes the policy primarily on histories induced by successful demonstrations, whereas deployment exposes it to histories generated by its own predictions. This covariate shift is particularly costly when an action changes contact or object state. We refer to such consequential decisions as *key actions*. The suitability of a proposed command is never an intrinsic property of an isolated image or action. Instead, physical validity depends entirely on the interplay among the recent observation history, the task instruction, and the specific predicted control.

Let $G(a_{t,j-1}, a_{t,j}) \in \{0, 1\}$ denote a task-specific event detector. For grasp-centric tasks, G detects an open-to-close gripper transition; the same abstraction supports release, insertion and tool-use events. The earliest key-action index is

$$j^\star = \min\{j \mid G(a_{t,j-1}, a_{t,j}) = 1\}. \tag{3}$$

If no event is detected, the policy follows its standard receding-horizon controller. Otherwise, execution stops before $a_{t,j^\star}$. After executing any non-critical prefix, the robot re-observes the environment and evaluates the pending key action from the refreshed history. The critic therefore reasons about the realized pre-action state rather than a stale observation or imagined future.

3.2 SHARED PLANNING, CRITICISM, AND CORRECTION

All capabilities are implemented by a shared multimodal model $\mathcal{M}_\theta$ and selected using task prompts. Let C_t denote the active instruction context. Outside a correction episode, $C_t = L$; after rejection, it contains the original instruction and the latest atomic correction. Planning uses

$$A_t = \mathcal{M}_\theta(H_t, C_t; q_{\text{plan}}). \tag{4}$$

For a detected key action a_t^{key}, the critic predicts

$$z_t = \mathcal{M}_\theta(H_t, C_t, a_t^{\text{key}}; q_{\text{critic}}), \qquad z_t \in \{\text{S}, \text{U}\}. \tag{5}$$

If the proposal is rejected, the correction interface generates

$$u_t = \mathcal{M}_\theta(H_t, C_t, a_t^{\text{key}}; q_{\text{corr}}), \tag{6}$$

and updates the local context to $C'_t = L \oplus u_t$. The same policy then produces

$$A'_t = \mathcal{M}_\theta(H_t, C'_t; q_{\text{plan}}). \tag{7}$$

Here, L preserves the long-horizon goal, whereas u_t specifies only the local adjustment required before the key action is attempted again. The critic is explicitly conditioned on $(H_t, C_t, a_t^{\text{key}})$ and directly generates a structured SUITABLE/UNSUITABLE decision; it does not convert language outputs into a calibrated probability or apply a numerical threshold. This action conditioning distinguishes, for example, a valid approach motion from a premature gripper closure in the same scene.

For a rejected proposal, u_t acts as an information bottleneck between criticism and control: it identifies one supported translation, height or orientation, gripper, or timing adjustment while leaving low-level realization to the shared policy. The context remains active until a key action is accepted or the bounded intervention terminates. Detailed output constraints and fallback behavior are provided in the supplementary material.

3.3 LEARNING SELF-CRITICISM

TRAJECTORY DATA AND SUPERVISION BOUNDARIES

Let $\mathcal{D}^+$ and $\mathcal{D}^-$ denote successful and unsuccessful trajectory sets. A trajectory

$$\tau = (L, o_1, a_1, \ldots, o_T, a_T) \tag{8}$$

contains the task instruction, observations, and executed actions. We extract key-action indices $\mathcal{K}(\tau)$ using Eq. 3. For each key event, we retain the pre-action history, candidate action, and post-action observations used only to construct supervision.

Successful trajectories supervise transition reasoning, positive key-action evaluation, atomic-correction construction, planning, and re-planning. Failed trajectories are restricted to negative key-action evaluation. Their state–action pairs reveal deployment-relevant failure modes, but their motor commands are excluded from both planning and re-planning targets. All data splits are performed at the trajectory level to prevent leakage between training and evaluation.

Binary suitability labels alone do not explain how an action changes the scene and may encourage shortcuts based on object identity or episode progress. We therefore formulate four complementary QA tasks that couple visual history, proposed actions, and physical transitions. The first two establish backward and forward transition understanding; the latter two teach the model to judge and correct consequential actions.

ACTION RETROSPECTION

Action Retrospection develops inverse action–transition reasoning. Given observations separated by k control steps, the model recovers the intervening action sequence:

$$\widehat{A}_{t:t+k-1} = Q_{\text{retro}}(o_t, o_{t+k}, L). \tag{9}$$

Data Generation. Targets are extracted from successful trajectories. We sample observation pairs at multiple temporal offsets and retain the executed action sequence between them. Short offsets emphasize local end-effector displacement, whereas longer offsets expose the sequence that leads to a contact transition. This supervision asks the model to explain visual change through its physical cause without requiring explicit pose regression.

Training Procedure. We serialize each pair as a multimodal question and train the model to generate the recorded action sequence:

> ***User: Task: [TASK]. Observation at the earlier step: `<image>`. Observation at the later step: `<image>`. Which action sequence moved the robot from the first observation to the second? Answer using only the supported robot actions.***

The resulting task grounds backward reasoning in the same language-oriented interface used by the VLA.

FUTURE-FRAME SELECTION

The complementary forward task asks which visual state is consistent with the recent history and a candidate action:

$$\widehat{n} = Q_{\text{future}}\left(H_t, a_t, \{\widetilde{o}_{t+1}^{n}\}_{n=1}^{N}\right). \tag{10}$$

Data Generation. Each candidate set contains the recorded successor, temporally adjacent hard negatives with similar appearance, and frames from trajectories that share the task but differ in configuration. The task therefore requires action-conditioned discrimination rather than object or instruction matching. Together, Eqs. 9 and 10 learn a bidirectional action–transition correspondence without generating future pixels.

Training Procedure. We cast forward transition grounding as multiple-choice visual QA:

> ***User: Task: [TASK]. Recent observations: <image> ... <image>. Candidate action: [ACTION]. Which candidate is the true next observation? Option A: <image>. Option B: <image>. Option C: <image>. Option D: <image>. Select the correct option.***

The training target is the option index corresponding to the recorded successor, and we maximize its autoregressive likelihood under the multiple-choice prompt. Distinguishing it from visually similar but transition-inconsistent distractors requires the model to ground the candidate action in its expected change to the scene rather than rely on static appearance cues.

KEY-ACTION EVALUATION

Key-Action Evaluation teaches the model to decide whether a consequential proposal is executable from the current pre-action state. Each example is $(H_t, L, a_t^{\text{key}}, y_t)$ with $y_t \in \{\text{S}, \text{U}\}$. A proposal is suitable when it can produce the intended stable transition and unsuitable when the gripper is premature, misaligned, aimed at the wrong object, or unable to establish the intended contact.

Data Generation. Successful interaction states provide positive examples, while temporally misaligned states and naturally occurring failures provide hard negatives. Post-action observations are used to establish supervision but are not included as future evidence in the critic input. The training samples are balanced by event type and decision label during training. More details are provided in Appendix. A.

Training Procedure. The JSON conversations serialize the task, two consecutive observations and actions, and the intended gripper event in the following form:

> ***User: Task: [TASK]. Observation t-1: <image>. Action t-1→t: [ACTION]. Observation t: <image>. Action t: [ACTION]. Intended key action: close the gripper to grasp, or open the gripper to release. Is it suitable to execute the intended key action now? If not, provide the next instruction for replanning.***

For a suitable proposal, the target response terminates after `Decision: yes` and a brief reason. For an unsuitable proposal, it begins with `Decision: no` and additionally exposes the correction target described next.

ATOMIC-CORRECTION GENERATION

A rejection identifies that the proposed key action should not be executed, but a useful critic must also state how to reach a suitable interaction state. Atomic-Correction Generation therefore maps an unsuitable proposal to a concise local instruction:

$$u_i = Q_{\text{corr}}(H_i, L, a_i^{-}). \tag{11}$$

Data Generation. Correction targets are grounded in successful interactions so that each instruction has a feasible realization. We pair an earlier history H_i with a later successful interaction state o_j, $i < j$, and use a key action that is valid near o_j but premature or misaligned at o_i as the counterfactual proposal a_i^{-}. An offline vision-language teacher describes the required local displacement, and we retain outputs only when they agree with the recorded end-effector motion, reference visible entities, and avoid unsupported precision.

Training Procedure. Evaluation and correction are jointly serialized in QA data. A negative example uses the same user question above and follows this structured answer format:

> ***User:*** *Task: put the small spoon from the basket to the tray. The current gripper-close proposal is unsuitable. Provide the next instruction for replanning.*
> ***Assistant:*** *Decision: no. Reason: the gripper is not yet at the grasp pose. Replan: move the gripper to center its opening over the spoon handle, rotate the fingers across the handle, and lower the open gripper around it before closing.*

The corresponding successful action segment supplies a separate re-planning target:

$$\widehat{A}_{i:j} = \pi_\theta(H_i, L \oplus u_i). \tag{12}$$

Equation 11 trains the model to generate a correction, whereas Eq. 12 trains the policy to perform correction-conditioned local replanning.

TWO-STAGE TRAINING OBJECTIVE

Let $\mathcal{Q} = \{\text{retro}, \text{future}, \text{critic}, \text{corr}\}$ denote the four QA tasks. Their aggregate objective is as follows:

$$\mathcal{L}_{\text{QA}} = \sum_{q \in \mathcal{Q}} \alpha_q \mathbb{E}_{x \sim \mathcal{D}_q}[\ell_q(x; \theta)], \qquad \sum_{q \in \mathcal{Q}} \alpha_q = 1. \tag{13}$$

The coefficients α_q control within-QA sampling. Each task loss is averaged independently before aggregation so that dataset size does not implicitly determine its contribution.

Starting from a planning-capable checkpoint $\theta^{(0)}$, Stage 1 establishes critic and correction capabilities:

$$\theta^{(1)} = \arg\min_\theta \mathcal{L}^{(1)}(\theta), \qquad \mathcal{L}^{(1)} = \mathcal{L}_{\text{QA}}. \tag{14}$$

Stage 2 initializes from $\theta^{(1)}$ and jointly trains planning, correction-conditioned re-planning, and retained QA examples:

$$\mathcal{L}^{(2)} = \mathcal{L}_{\text{plan}} + \mathcal{L}_{\text{replan}} + \beta \mathcal{L}_{\text{QA}}, \qquad \beta = 0.1. \tag{15}$$

Planning and re-planning therefore receive unit weight, while the aggregate QA objective receives weight 0.1 to mitigate catastrophic forgetting. Failed motor commands remain excluded from both action losses. These two training stages produce a single model whose prompt-conditioned interfaces are composed through the event-triggered online inference protocol detailed in the supplementary material.

4 EXPERIMENTS

4.1 DATASETS AND IMPLEMENTATION DETAILS

We evaluate ActionCritic on SIMPLER Li et al. (2024b;c), LIBERO and LIBERO-Plus Liu et al. (2023), and four real-world ALOHA tasks. These settings cover two simulated embodiments, multi-task and long-horizon manipulation, controlled visual and language shifts, and physical grasp and task success. We instantiate StarVLA with Qwen3.5 0.8B and 2B checkpoints and initialize ActionCritic from the corresponding planner. Our training follows the two-stage procedure in Section 3.3. Controlled comparisons keep the backbone, observations, action representation, and demonstrations fixed. More details are provided in Appendix.

4.2 MAIN RESULTS

SIMPLER. Tables 1 and 2 show consistent gains across embodiments and model scales. On Google Robot, ActionCritic-0.8B improves StarVLA from 54.1% to 57.1% overall, with gains of $+1.9$ points on VM and $+2.9$ points on VA. The available 2B entries improve from 63.7% to 67.0% overall, with gains of $+3.1$ points on VM and $+3.5$ points on VA. On WidowX, average success rises from 61.5% to 67.7% at 0.8B and from 69.8% to 78.1% at 2B. The largest 2B gains occur on spoon placement and cube stacking ($+12.5$ each), where interaction success is sensitive to approach geometry. Comparable improvements across model scales and embodiments suggest that the benefit stems from event-triggered verification and correction rather than additional capacity or embodiment-specific tuning.

Table 1: **Performance comparison on SimplerEnv-Fractal (Li et al., 2024c) with Google robot.** Success rates (%) are reported for *Visual Matching (VM)* and *Variant Aggregation (VA)* suites. O./C. denotes Open/Close, and * follows Tab. 2.

Method	Visual Matching (VM)					Variant Aggregation (VA)					Overall
	Coke Can	Move Near	O. / C. Drawer	Put in Drawer	Avg.	Coke Can	Move Near	O. / C. Drawer	Put in Drawer	Avg.	
RT-1-X (O'Neill et al., 2024)	56.7	31.7	59.7	21.3	42.4	49.0	32.3	29.4	10.1	30.2	36.3
OpenVLA (Kim et al., 2024)	18.0	56.3	63.0	0.0	34.3	60.8	67.7	28.8	0.0	39.3	36.8
RoboVLMs (Liu et al., 2025a)	76.3	79.0	44.9	27.8	57.0	50.7	62.5	10.3	0.0	30.9	44.0
TraceVLA (Zheng et al., 2024)	45.0	63.8	63.1	11.1	45.8	64.3	60.6	61.6	12.5	49.8	47.8
RT-2-X (O'Neill et al., 2024)	78.7	77.9	25.0	3.7	46.3	82.3	79.2	35.5	20.6	54.4	50.4
Magma (Yang et al., 2025)	75.0	53.0	58.9	8.3	48.8	68.6	78.5	59.0	24.0	57.5	53.2
SpatialVLA (Qu et al., 2025)	79.3	90.0	54.6	0.0	56.0	78.7	83.0	39.2	6.3	51.8	53.9
π_0-Uniform* (Black et al., 2024)	88.0	80.3	56.0	52.2	69.1	–	–	–	–	–	–
π_0-Beta* (Black et al., 2024)	97.9	78.7	62.3	46.6	71.4	–	–	–	–	–	–
CogACT-7B (Li et al., 2024a)	91.3	85.0	71.8	50.9	<u>74.8</u>	89.6	80.8	28.3	46.6	<u>61.3</u>	<u>68.1</u>
StarVLA-0.8B (StarVLA Community, 2026)	89.7	80.4	47.7	39.8	64.4	63.2	50.5	29.1	37.0	45.0	54.1
ActionCritic -0.8B(Ours)	92.3	82.9	50.0	45.4	66.3	65.6	52.8	32.8	40.7	47.9	57.1
StarVLA-2B (StarVLA Community, 2026)	94.0	87.1	61.1	47.2	72.3	70.5	59.2	50.4	40.2	55.1	63.7
ActionCritic -2B(Ours)	96.3	90.0	66.2	48.9	75.4	75.9	61.3	53.4	43.9	58.6	67.0

Table 2: **Performance comparison on SimplerEnv-Bridge (Li et al., 2024c) with WidowX robot.**

Method	Spoon on Towel	Carrot on Plate	Stack Cube	Eggplant in Basket	Avg. Success
RT-1-X (O'Neill et al., 2024)	0.0	4.2	0.0	0.0	1.1
OpenVLA (Kim et al., 2024)	4.2	0.0	0.0	12.5	4.2
Octo-Base (Octo Model Team et al., 2024)	15.8	12.5	0.0	41.7	17.5
TraceVLA (Zheng et al., 2024)	12.5	16.6	16.6	65.0	27.7
RoboVLMs (Liu et al., 2025a)	45.8	20.8	4.2	79.2	37.5
SpatialVLA (Qu et al., 2025)	16.7	25.0	29.2	100.0	42.7
Magma (Yang et al., 2025)	37.5	29.2	20.8	91.7	44.8
CogACT-Base (Li et al., 2024a)	71.7	50.8	15.0	67.5	51.3
π_0-Uniform* (Black et al., 2024)	63.3	58.8	21.3	79.2	55.7
CogACT-Large (Li et al., 2024a)	58.3	45.8	29.2	95.8	<u>57.3</u>
CronusVLA (Li et al., 2025)	66.7	54.2	20.8	100.0	60.4
π_0-Beta* (Black et al., 2024)	84.6	55.8	47.9	85.4	68.4
StarVLA-0.8B (StarVLA Community, 2026)	79.2	41.7	37.5	87.5	61.5
ActionCritic-0.8B	87.5	50.0	37.5	95.8	**67.7**
StarVLA-2B (StarVLA Community, 2026)	70.8	45.8	62.5	100.0	69.8
ActionCritic-2B	83.3	54.2	75.0	100	**78.1**

Table 3: **Performance comparison on LIBERO (Liu et al., 2023) with Franka robot.**

Method	Spatial	Object	Goal	Long-10	Avg. Success
Diffusion Policy (Chi et al., 2023)	78.3	92.5	68.3	50.5	72.4
Octo (Octo Model Team et al., 2024)	78.9	85.7	84.6	51.1	75.1
UniACT (Zheng et al., 2025)	77.0	87.0	77.0	70.0	76.8
SpatialVLA (Qu et al., 2025)	88.2	89.9	78.6	55.5	71.7
TraceVLA (Zheng et al., 2024)	84.6	85.2	75.1	54.1	74.8
OpenVLA (Kim et al., 2024)	84.7	88.4	79.2	53.7	75.9
CoT-VLA (Zhao et al., 2025)	87.5	91.6	87.6	69.0	81.1
π_0-FAST* (Pertsch et al., 2025)	96.4	96.8	88.6	60.2	85.0
CronusVLA (Li et al., 2025)	90.1	94.7	91.3	68.7	86.2
TriVLA (Liu et al., 2025b)	91.2	93.8	89.8	73.2	87.0
4D-VLA (Zhang et al., 2025)	93.8	92.8	95.6	86.5	92.2
CogACT (Li et al., 2024a)	97.2	98.0	90.2	88.8	<u>93.2</u>
π_0* (Black et al., 2024)	96.8	98.8	95.8	85.2	94.2
StarVLA-0.8B	92.0	99.6	95.8	92.4	95.0
ActionCritic 0.8B(Ours)	92.6	99.0	98.0	95.4	**96.4**
StarVLA-2B	93.8	99.6	97.4	93.4	96.1
ActionCritic 2B(Ours)	95.8	99.6	98.4	96.2	**97.5**

Table 4: Comparison on the LIBERO-Plus benchmark. Methods under *Zero-Shot Transfer* are trained on LIBERO and directly evaluated on LIBERO-Plus. An asterisk (*) denotes results reproduced from officially released checkpoints.**Bold**: Best.

Methods	Camera	Robot	Language	Light	Background	Noise	Layout	Avg.
Zero-Shot Transfer								
WorldVLA (Cen et al., 2025)	0.1	27.9	41.6	43.7	17.1	10.9	38.0	25.0
OpenVLA (Kim et al., 2024)	0.8	3.5	23.0	8.1	34.8	15.2	28.5	15.6
NORA (Hung et al., 2025)	2.2	37.0	65.1	45.7	58.6	12.8	62.1	39.0
UniVLA (Bu et al., 2025)	1.8	46.2	69.6	69.0	81.0	21.2	31.9	42.9
π_0-Fast (Pertsch et al., 2025)	65.1	21.6	61.0	73.2	73.2	74.4	68.8	61.6
RIPT-VLA (Tan et al., 2025)	55.2	31.2	77.6	88.4	91.6	73.5	74.2	68.4
OpenVLA-OFT (Kim et al., 2025)	56.4	31.9	79.5	88.7	93.3	75.8	74.2	69.6
π_0^* (Black et al., 2024)	61.0	40.8	63.5	89.3	84.1	80.1	76.4	69.4
$\pi_{0.5}^*$ (Physical Intelligence et al., 2025)	**75.8**	79.4	83.3	95.5	95.0	**89.6**	87.0	85.7
StarVLA-2B	61.0	40.1	80.6	89.4	86.1	78.0	79.6	73.6
ActionCritic-2B(Ours)	61.6	41.0	83.0	89.8	86.5	81.5	79.9	74.8

LIBERO and LIBERO-Plus. On LIBERO (Table 3), ActionCritic-0.8B reaches 96.4%, 1.4 points above StarVLA-0.8B, including 95.4% on Long-10. Thus, verification remains useful when early errors compound across longer tasks. Spatial is the weakest suite (92.6%), indicating that local correction does not eliminate fine-grained grounding ambiguity. Under the controlled shifts of LIBERO-Plus (Table 4), ActionCritic-2B improves StarVLA-2B from 73.6% to 74.8% overall and improves on every listed shift, but remains below $\pi_{0.5}^*$ on all seven shifts.

Table 5 shows diminishing returns beyond two corrections. At that budget, the 0.8B model improves WidowX and Google Robot by 6.2 and 3.0 points, while the 2B model gains 8.3 and 3.3 points. Unlimited retries improve success slightly but increase execution steps. Critic and re-planning latency for both model sizes, together with the profiling protocol, are reported in the supplementary ap-

Real-World Deployment On the ALOHA robot, Figure 3 reports four physical tasks and their average. From the plotted averages, ActionCritic reaches 88.5% grasp success and 81.2% end-to-end success, compared with 79.2% and 76.0% for StarVLA-2B. The figure also includes CogACT, which reaches 70.8% grasp success and 63.5% end-to-end success. These results indicate that the event-triggered verification and local replanning loop transfers to the evaluated physical tasks; the figure is the source of the reported real-world numbers.

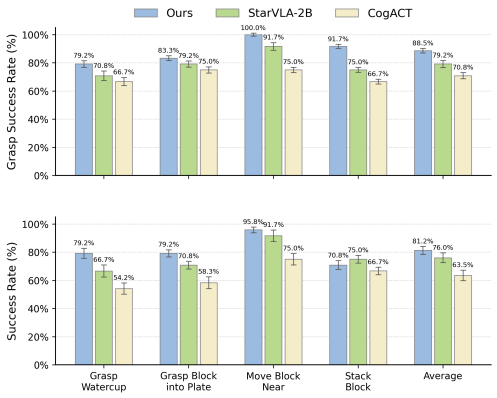


Figure 3: **Real-world evaluation on ALOHA.**

Table 5: **Ablation on maximum correction attempts.** Additional critic time is reserved for matched profiling and is currently unreported.

Model	Max Corrections	Success Rate		Efficiency	
		WidowX	Google	Avg. Latency	Avg. Steps
StarVLA-0.8B(Base)	-	61.5	54.1	12.4	71.1
ActionCritic-0.8B	1	64.6	55.7	11.9	65.6
	2	67.7	57.1	12.2	63.7
	3	68.4	56.4	13.1	66.4
	∞	68.4	56.9	14.7	79.9
StarVLA-2B(Base)	-	69.8	63.7	11.7	58.9
ActionCritic-2B	1	72.9	65.3	11.3	57.9
	2	78.1	67.0	11.9	57.8
	3	79.2	67.5	13.7	59.4
	∞	83.3	68.1	15.5	62.1

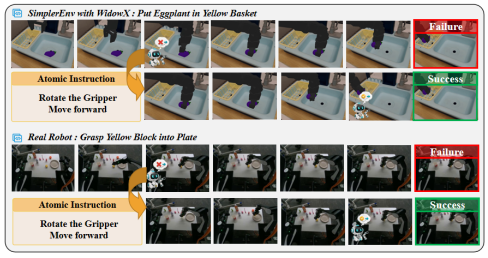


Table 6: **Event-triggered self-correction.** An unsuitable proposal is rejected, locally corrected, and successfully re-planned across simulated and real embodiments.

pendix. We therefore use two corrections as the operating point that captures most recovery gains without repeated re-planning overhead.

4.3 Qualitative Correction Analysis

Figure 6 shows the same proposal–evaluate–correct–replan sequence on WidowX, Google Robot, and ALOHA. Despite different viewpoints and action spaces, each correction changes only the deficient local interaction and is consumed by the shared policy. Across examples, intervention adjusts the approach geometry before irreversible contact rather than attempting post-failure recovery, providing a plausible explanation for the gains on grasp-sensitive tasks. This consistency supports language as a compact, embodiment-agnostic correction interface.

5 Conclusion

We presented ActionCritic, a shared VLA policy with event-triggered verification that evaluates consequential actions before execution and converts rejected proposals into atomic instructions for local replanning. Bidirectional action–transition QA and complementary success/failure data improve the evaluated simulated and real-world tasks without imitating failed commands or requiring online video prediction.

Limitations. The method remains limited to detectable, visually observable, and locally repairable errors; contact-rich or hidden-state failures may require force sensing or explicit dynamics. Language corrections may also lack metric precision. Accordingly, deployment must retain bounded retries, collision checking, hardware safeguards, and task-specific operational limits.

REFERENCES

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. pi-0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

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

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Univla: Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2505.06111*, 2025.

Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, et al. WorldVLA: Towards autoregressive action world models. *arXiv preprint arXiv:2506.21539*, 2025.

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*, pp. 02783649241273668, 2023.

Yinpei Dai, Jayjun Lee, Nima Fazeli, and Joyce Chai. RACER: Rich language-guided failure recovery policies for imitation learning. *arXiv preprint arXiv:2409.14674*, 2024.

Jiafei Duan, Wilbert Pumacay, Nishanth Kumar, Yi Ru Wang, Shulin Tian, Wentao Yuan, Ranjay Krishna, Dieter Fox, Ajay Mandlekar, and Yijie Guo. AHA: A vision-language model for detecting and reasoning over failures in robotic manipulation. *arXiv preprint arXiv:2410.00371*, 2024.

Chia-Yu Hung, Qi Sun, Pengfei Hong, Amir Zadeh, Chuan Li, U-Xuan Tan, Navonil Majumder, and Soujanya Poria. NORA: A small open-sourced generalist vision language action model for embodied tasks. *arXiv preprint arXiv:2504.19854*, 2025.

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

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

Hao Li, Shuai Yang, Yilun Chen, Yang Tian, Xiaoda Yang, Xinyi Chen, Hanqing Wang, Tai Wang, Feng Zhao, Dahua Lin, et al. Cronusvla: Transferring latent motion across time for multi-frame prediction in manipulation. *arXiv preprint arXiv:2506.19816*, 2025.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, et al. CogACT: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024a.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating real-world robot manipulation policies in simulation. In *Conference on Robot Learning*, 2024b.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, et al. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024c.

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

Huaping Liu, Xinghang Li, Peiyan Li, Minghuan Liu, Dong Wang, Jirong Liu, Bingyi Kang, Xiao Ma, Tao Kong, and Hanbo Zhang. Towards generalist robot policies: What matters in building vision-language-action models. *arXiv preprint arXiv:2412.14058*, 2025a.

Zhenyang Liu, Yongchong Gu, Sixiao Zheng, Xiangyang Xue, and Yanwei Fu. Trivla: A unified triple-system-based unified vision-language-action model for general robot control. *arXiv preprint arXiv:2507.01424*, 2025b.

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

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open X-embodiment: Robotic learning datasets and RT-X models. In *IEEE International Conference on Robotics and Automation*, pp. 6892–6903, 2024.

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

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. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, Jiayuan Gu, Bin Zhao, Dong Wang, et al. SpatialVLA: Exploring spatial representations for vision-language-action models. *arXiv preprint arXiv:2501.15830*, 2025.

Hao Shi, Bin Xie, Yingfei Liu, Lin Sun, Fengrong Liu, Tiancai Wang, Erjin Zhou, Haoqiang Fan, Xiangyu Zhang, and Gao Huang. MemoryVLA: Perceptual-cognitive memory in vision-language-action models for robotic manipulation. *arXiv preprint arXiv:2508.19236*, 2025.

Lucy Xiaoyang Shi, Zheyuan Hu, Tony Z. Zhao, Archit Sharma, Karl Pertsch, Jianlan Luo, Sergey Levine, and Chelsea Finn. Yell at your robot: Improving on-the-fly from language corrections. *arXiv preprint arXiv:2403.12910*, 2024.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652, 2023.

StarVLA Community. StarVLA: A lego-like codebase for vision-language-action model developing. *arXiv preprint arXiv:2604.05014*, 2026.

Shuhan Tan, Kairan Dou, Yue Zhao, and Philipp Krähenbühl. Interactive post-training for vision-language-action models. *arXiv preprint arXiv:2505.17016*, 2025.

Junjie Wen, Yichen Zhu, Jinming Li, Zhibin Tang, Chaomin Shen, and Feifei Feng. Dexvla: Vision-language model with plug-in diffusion expert for general robot control. *arXiv preprint arXiv:2502.05855*, 2025.

Wenke Xia, Ruoxuan Feng, Dong Wang, and Di Hu. Phoenix: A motion-based self-reflection framework for fine-grained robotic action correction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6981–6990, 2025.

Shaoqing Xu, Fang Li, Zhixiang Duan, Yifan Yang, Tianshi Xie, and Zhi-Xin Yang. VLA-IN-THE-LOOP: ONLINE POLICY CORRECTION WITH WORLD MODELS FOR ROBUST ROBOTIC GRASPING, 2026. URL https://openreview.net/forum?id=aT4LG8c6DE.

Jianwei Yang, Reuben Tan, Qianhui Wu, Ruijie Zheng, Baolin Peng, Yongyuan Liang, Yu Gu, Mu Cai, Seonghyeon Ye, Joel Jang, et al. Magma: A foundation model for multimodal ai agents. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 14203–14214, 2025.

Jiahui Zhang, Yurui Chen, Yueming Xu, Ze Huang, Yanpeng Zhou, Yu-Jie Yuan, Xinyue Cai, Guowei Huang, Xingyue Quan, Hang Xu, et al. 4d-vla: Spatiotemporal vision-language-action pretraining with cross-scene calibration. *arXiv preprint arXiv:2506.22242*, 2025.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. CoT-VLA: Visual chain-of-thought reasoning for vision-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1702–1713, 2025.

Jinliang Zheng, Jianxiong Li, Dongxiu Liu, Yinan Zheng, Zhihao Wang, Zhonghong Ou, Yu Liu, Jingjing Liu, Ya-Qin Zhang, and Xianyuan Zhan. Universal actions for enhanced embodied foundation models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 22508–22519, 2025.

Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. *arXiv preprint arXiv:2412.10345*, 2024.

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