REFLECT, RECOVER, INTERNALIZE: SELF-IMPROVING VLAS FROM EXECUTION FEEDBACK

Anonymous authors
Paper under double-blind review

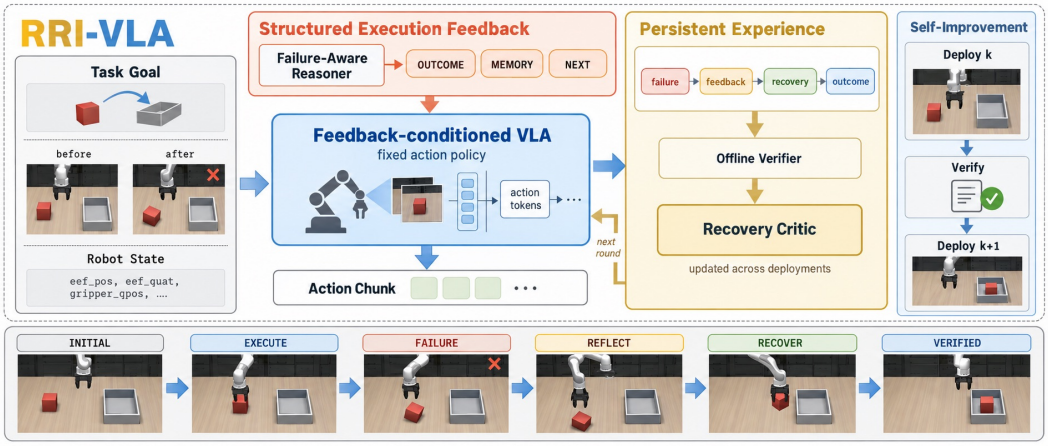

Figure 1: Overview of RRI-VLA. Two feedback-driven loops share a common execution-feedback interface: the within-episode *reflect–recover* loop interprets each action chunk and authorizes bounded correction for immediate recovery, while the across-episode *collect–verify–internalize* loop turns the resulting records into outcome-verified experience that updates the Recovery Critic, keeping the action-producing VLA fixed during deployment and subsequent self-improvement rounds.

ABSTRACT

Vision-language-action (VLA) models are usually trained from fixed demonstrations and discard the consequences and corrections observed during deployment. We introduce RRI-VLA, a *reflect–recover–internalize* framework that uses execution feedback for both immediate recovery and persistent improvement. At each action boundary, a frozen multimodal reasoner summarizes the preceding chunk as an outcome, running episode memory, and next adjustment. These states condition the next action, while a governed orchestrator combines semantic, physical, and learned evidence to authorize one bounded corrective chunk. An offline verifier then converts completed rollouts into verified failure–recovery records that train an action-conditioned Recovery Critic across deployments. After one-time feedback-interface initialization, only the critic is updated and the action-producing VLA remains frozen. On LIBERO-Plus and LIBERO-PRO, RRI-VLA reaches 92.47% and 57.84% task success. Relative to online-only recovery, internalization adds 3.12 and 5.64 points while retaining 95.30% success on unperturbed tasks; governed recovery resolves 74.2% of eligible failures with 1.10% harmful flips. Ablations identify verified experience, causal feedback alignment, and bounded governance as the key sources of improvement. Execution feedback thus provides a shared interface for within-episode recovery and policy-preserving learning across deployments.

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.

1 INTRODUCTION

Large vision-language-action (VLA) models unify perception, instruction following, and continuous control in reusable robot policies (Brohan et al., 2022; 2023; O'Neill et al., 2024; Kim et al., 2024; Team et al., 2024). Yet they are usually trained from fixed data and deployed as static policies. Deployment exposes action consequences, failure states, and successful corrections, but this evidence is typically discarded after each episode. Existing work addresses this limitation through three routes. *Execution-time reflection, failure supervision, and agent harnesses* use language feedback, external memory, learned failure predictors, physical-feasibility checks, or retries to repair the current rollout (Huang et al., 2022; Yao et al., 2022; Shinn et al., 2023; Yang et al., 2026b; Lin et al., 2025; Ma et al., 2026; Zhang et al., 2026; Yang et al., 2026a). These approaches may rely on trained auxiliary models or external state, but generally do not turn each newly verified deployment correction into persistent, iteratively updated experience. *Reasoning and action-consequence conditioning* provides intermediate reasoning or recent interaction history to improve local adaptation (Zhao et al., 2025; Zhong et al., 2026; Lian et al., 2026). It does not establish which corrections should become verified long-term experience. *Training-based self-improvement* uses policy-induced data, specialists, reinforcement learning, or failure-derived rewards to obtain lasting gains (Ross et al., 2011; Bousmalis et al., 2023; Xiao et al., 2026; Li et al., 2025a). These methods often require additional supervision or direct policy updates, and their data generation is separated from deployment feedback. A unified record connecting actions, consequences, recoveries, and outcomes therefore remains missing.

Our design follows a simple principle: the feedback used to correct an ongoing rollout should also become verified experience for future deployments. We therefore introduce RRI-VLA, a *reflect–recover–internalize* framework that connects a within-episode correction loop with an across-deployment learning loop. The former uses action consequences to recover immediately, while the latter internalizes validated recovery experience without changing the generalist action policy. This separation allows the agent to improve when it invokes recovery while largely preserving its existing motor competence.

Following this design, RRI-VLA connects the two loops through the pipeline shown in Figure 1. After the fixed VLA executes an action chunk, the Failure-Aware Reasoner compares the task goal with the executed chunk, resulting observations, and boundary robot states. It converts the observed consequence into Structured Execution Feedback with three components: outcome, running episode memory, and next adjustment. The feedback-conditioned VLA consumes these components at the next action boundary. When the evidence indicates failure, the Recovery Orchestrator authorizes a bounded corrective chunk and prevents repeated or non-progressing interventions. The interaction is then stored as a localized failure–feedback–recovery–outcome record, as illustrated by the bottom sequence from initial execution to verified recovery. Across deployments, the Offline Verifier checks whether the attempted correction produced the intended physical outcome. Verified records train an action-conditioned Recovery Critic that carries this experience into the next deployment and learns when to activate recovery. Repeating this collect–verify–update cycle accumulates persistent recovery knowledge from the agent's own experience while the deployed action-producing VLA remains fixed after the one-time interface initialization.

RRI-VLA achieves 92.47% success on LIBERO-Plus and 57.84% on LIBERO-PRO, improving over its online-only counterpart by 3.12 and 5.64 percentage points while achieving 95.30% success on unperturbed tasks. Under full recovery governance, it resolves 74.2% of eligible failures with only 1.10% harmful flips. Controlled ablations further show that verified failure–recovery pairs retain most of the full-mixture performance, whereas shuffled targets reduce success by 4.82 and 8.79 points. Together, these results attribute the gains to causally aligned feedback, outcome verification, and bounded recovery rather than additional motor-policy training.

Our main contributions are:

- We formulate execution feedback as a shared interface for two-timescale improvement, supporting within-episode recovery and across-deployment experience.
- We introduce RRI-VLA, which combines Structured Execution Feedback, governed one-chunk recovery, offline verification, and a Recovery Critic to convert deployment failures into supervision while keeping the deployed VLA fixed after interface initialization.

- We validate the framework on LIBERO-Plus and LIBERO-PRO, showing persistent improvement from self-collected recovery experience and clarifying the roles of feedback alignment and governance.

2 Related Work

Generalist VLA policies and action reasoning. Large language-conditioned robot transformers scaled instruction-following control (Brohan et al., 2022; 2023), and subsequent cross-embodiment, open-source, and flow-based foundation models broadened transfer across tasks and platforms (O'Neill et al., 2024; Team et al., 2024; Kim et al., 2024; Black et al., 2024; Intelligence et al., 2025). Complementary lines improve action modeling through multimodal prompting, spatial representations, and expressive continuous action distributions (Jiang et al., 2023; Chi et al., 2025; Qu et al., 2025), or augment action prediction with visual, linguistic, or action-space reasoning before execution (Zhao et al., 2025; Zhong et al., 2026; Zheng et al., 2025). Other approaches incorporate latent world models to enrich predictive representations for action generation (Sun et al., 2026). These advances strengthen action generation from pre-action context, but do not specify how the physical consequences observed afterward should become reusable experience. We retain the coarse-to-fine action interface of Zhong et al. (2026) and add a post-action feedback pathway that serves both the next decision and future deployments.

Execution-time reflection and action consequences. A broad line of work shows that language feedback and stored reflections can revise sequential decisions without changing a base model (Huang et al., 2022; Yao et al., 2022; Shinn et al., 2023). For manipulation, related systems add visual-language failure supervision, backtracking, external memory, learned failure prediction, and physical-feasibility checks to detect failures and retry execution (Yang et al., 2026b; Lin et al., 2025; Ma et al., 2026; Zhang et al., 2026; Yang et al., 2026a). Complementary interactive-imitation methods use uncertainty estimates to query expert intervention during policy rollouts and collect corrective data (Lee et al., 2025), while other approaches directly condition action prediction on observation–action–consequence history to adapt to deployment-specific dynamics (Lian et al., 2026). However, such improvements usually remain in transient context or external state, or require separately collected intervention data; individual deployment corrections are not verified and accumulated through a persistent self-improvement loop. RRI-VLA instead represents each action boundary with Structured Execution Feedback and converts governed recoveries into outcome-verified records shared by the acting and learning loops.

Training-based self-improvement. Interactive imitation and autonomous data-generation methods improve a policy from states it induces during deployment (Ross et al., 2011; Kelly et al., 2019; Bousmalis et al., 2023). More recent VLA post-training pipelines explore failure regions with reinforcement-learning specialists and distill curated trajectories back into the generalist (Xiao et al., 2026), use world-model rollouts and verified rewards for reinforcement fine-tuning (Li et al., 2025b), or derive rewards from failure analysis and combine reinforcement learning with supervised fine-tuning (Li et al., 2025a). These methods achieve persistent improvement, but often rely on expert labels, separate specialists, synthesized or simulator-derived rewards, or direct updates to the action policy. RRI-VLA reuses the deployed agent's own feedback-mediated recovery attempts and updates only the Recovery Critic, isolating improved recovery decisions from changes to the generalist motor policy.

3 Method

The method is organized into four parts. Section 3.1 introduces the problem formulation and two-timescale framework. Section 3.2 describes the feedback-conditioned action backbone. Section 3.3 presents the online reflect-and-recover loop for causal feedback and governed correction, while Sec. 3.4 details the offline verify-and-internalize loop for outcome verification and Recovery Critic updates.

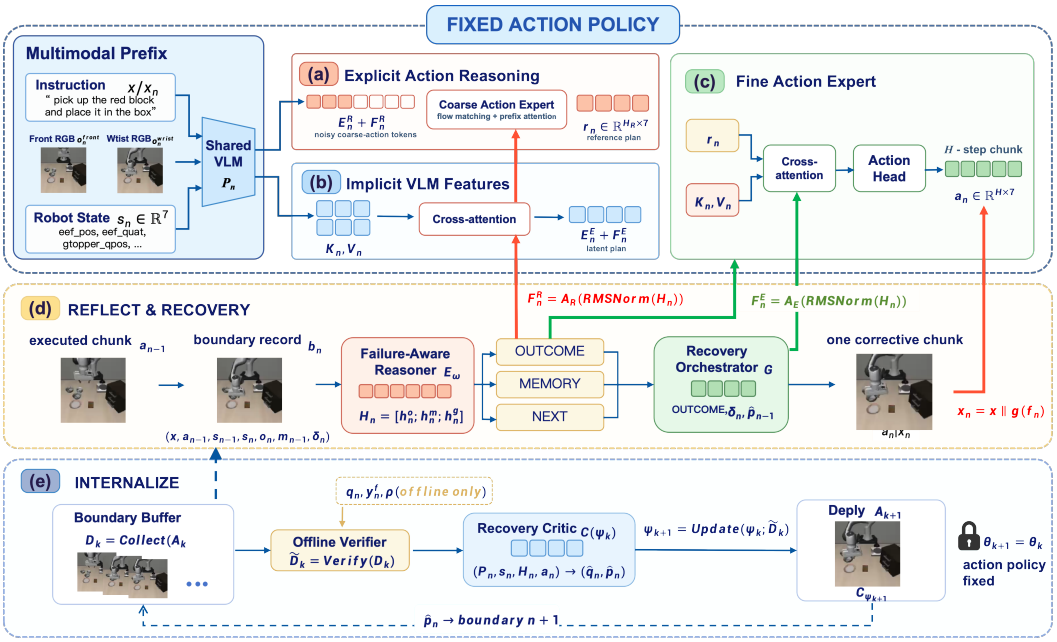

Figure 2: Detailed implementation of RRI-VLA. At each action boundary, the frozen Failure-Aware Reasoner converts the execution record into typed feedback, which is injected into the coarse and fine action streams for the next decision. The Recovery Orchestrator combines semantic feedback, physical diagnostics, and the delayed assessment of the Recovery Critic to authorize one bounded correction. Across deployments, the Offline Verifier converts collected interactions into verified experience for updating the Recovery Critic while the action-producing VLA remains fixed after interface initialization.

3.1 Overview and Problem Formulation

RRI-VLA couples two feedback-driven loops operating at different timescales. Within an episode, the fixed action policy executes one chunk at a time, the Failure-Aware Reasoner E_ω interprets the resulting consequence, and the Recovery Orchestrator G may authorize one bounded correction. Across deployments, these interactions are verified and used to update the Recovery Critic C_ψ, which improves subsequent recovery decisions without modifying the action policy. Figure 2 summarizes this organization: the top path is the feedback-conditioned action backbone, the middle path is the online reflect-and-recover loop, and the bottom path is the offline verify-and-internalize loop.

Let x be a language instruction, $o_n = (o_n^{\text{front}}, o_n^{\text{wrist}})$ the RGB observations at action boundary n, $s_n \in \mathbb{R}^7$ the end-effector pose and gripper state, and $u_t \in \mathbb{R}^7$ the low-level action at control step t. With a fixed chunk horizon H, the next action chunk is $a_n = (u_{nH}, \dots, u_{(n+1)H-1}) \in \mathbb{R}^{H\times 7}$. We replan only after the complete chunk is executed. A feedback-conditioned action policy is

$$a_n \sim \pi_\theta(\cdot \mid o_n, s_n, x_n, F_n), \qquad F_n = (F_n^R, F_n^E), \tag{1}$$

where x_n is the boundary-specific instruction defined below, and F_n^R and F_n^E condition the coarse and fine streams. Because the feedback describes a_{n-1}, F_0 is unavailable and masked.

At deployment round k, the complete improvement cycle is

$$\begin{aligned} \mathcal{D}_k &= \text{Collect}(\mathcal{A}_k), \\ \widetilde{\mathcal{D}}_k &= \text{Verify}(\mathcal{D}_k), \\ \psi_{k+1} &= \text{Update}(\psi_k; \widetilde{\mathcal{D}}_k), \qquad \theta_{k+1} = \theta_k, \end{aligned} \tag{2}$$

where $\mathcal{A}_k = (\pi_\theta, E_\omega, G, C_{\psi_k})$. Only ψ is updated; θ, ω, and G remain fixed. Equation 2 therefore separates transient within-episode context from persistent critic updates and isolates recovery learning from motor-policy changes.

3.2 Feedback-Conditioned Action Backbone

Following ACoT-VLA (Zhong et al., 2026), a shared VLM prefix encodes the instruction, front and wrist images, and robot state, yielding pooled representation P_n and intermediate key–value states (K_n, V_n). An explicit reasoner predicts a coarse trajectory $r_n \in \mathbb{R}^{H_R \times 7}$ with coarse-plan horizon H_R, while an implicit reasoner extracts complementary guidance from (K_n, V_n). Let E_n^R and E_n^E be the noisy action-token suffixes of the coarse reasoner and fine expert. Cross-attention aligns both streams, and the fine expert produces the executed H-step chunk.

Both action experts use conditional flow matching. For action target y, noise $\epsilon \sim \mathcal{N}(0, I)$, and time $\tau \in (0, 1]$,

$$y_\tau = \tau\epsilon + (1-\tau)y, \qquad u_\tau = \epsilon - y. \tag{3}$$

The coarse and fine experts estimate the corresponding velocity fields with losses $\mathcal{L}_R$ and $\mathcal{L}_E$. During feedback-interface initialization, the action stack and feedback adapters are trained with

$$\mathcal{L}_{\text{VLA}} = \mathcal{L}_R + \mathcal{L}_E, \tag{4}$$

while the language backbone remains frozen. This initialization is not counted as a self-improvement round; it creates the interface through which future execution experience can affect behavior.

After this one-time interface initialization, the complete action-producing stack, the feedback adapters, and the multimodal reasoner are frozen. Unless explicitly stated otherwise for the action-interface-tuning ablation, all subsequent deployment and self-improvement rounds update only the Recovery Critic parameters ψ. Thus, throughout the paper, "fixed VLA" refers to the post-initialization deployment stage.

3.3 Online Reflect-and-Recover Loop

The online loop converts the consequence of each completed action chunk into causal context for the next boundary, then decides whether that evidence warrants intervention.

Boundary record. After executing a_{n-1}, the Multimodal Interface constructs

$$b_n = (x, a_{n-1}, s_{n-1}, s_n, o_n^{\text{front}}, o_n^{\text{wrist}}, m_{n-1}, \delta_n), \tag{5}$$

where m_{n-1} is running memory carried across boundaries within the current episode and δ_n contains deployment-available endpoint diagnostics: end-effector translation and rotation, gripper displacement, requested translation, and executed-to-requested motion ratio. Online sanitization removes task success, reward, simulator progress, contact state, object pose, and ground-truth task state.

Typed reflection bottleneck. A frozen multimodal reasoner E_ω receives the compact boundary record, including the current front and wrist observations. It emits three fixed-schema fields:

`OUTCOME` – observable consequence; `MEMORY` – fact to carry forward within the current episode; `NEXT` – one actionable adjustment.

Let $H_n = [h_n^o; h_n^m; h_n^g] \in \mathbb{R}^{3\times D}$ denote the final-layer states at the closing markers of `OUTCOME`, `MEMORY`, and `NEXT`, where D is the reasoner hidden width. The decoded `MEMORY` becomes m_n in the next record. The full interface exposes two distinct channels. The fixed-size latent states H_n condition the action experts through the residual adapters, whereas the decoded fields are consumed by the Recovery Orchestrator for semantic failure evidence and are retained for auditing. Decoded text is not directly inserted into the action-token stream. The same representation organizes failure–recovery records offline.

Causal feedback injection. Because the coarse and fine action experts occupy different embedding spaces, two residual adapters A_R and A_E map normalized feedback states into their respective widths:

$$F_n^R = A_R(\text{RMSNorm}(H_n)), \qquad F_n^E = A_E(\text{RMSNorm}(H_n)). \tag{6}$$

The three states are prepended to each action suffix,

$$\widetilde{E}_n^R = [F_n^R; E_n^R], \qquad \widetilde{E}_n^E = [F_n^E; E_n^E]. \tag{7}$$

where $[\,\cdot\,;\,\cdot\,]$ denotes concatenation along the token dimension. Action tokens can attend to the shared multimodal prefix, all valid feedback states, and earlier tokens in their own action block. Feedback cannot attend to future actions, and all feedback is masked for the first chunk. Consequently, evidence from a_{n-1} can affect a_n but never the action that produced it.

Governed recovery. Reflection alone does not guarantee a productive recovery. A stale or incorrect diagnosis can repeatedly interrupt progress and contaminate the collected dataset. The Recovery Orchestrator therefore separates failure evidence from intervention authority. At boundary n, it combines: (i) semantic patterns in `OUTCOME`; (ii) physical diagnostics such as low execution ratio under a nontrivial command; and (iii) the Recovery Critic's delayed risk estimate $\hat{p}_{n-1}$ for the previously executed chunk.

If recovery is admitted, the original goal is retained verbatim and a class-specific clause is appended for one chunk. Here f_n is the detected failure class and g maps it to a bounded corrective instruction:

$$x_n = x \,\|\, \text{“For the next action chunk only: } g(f_n)\text{. Continue the original task.”} \tag{8}$$

Otherwise, $x_n = x$; the original instruction is never replaced. The governor uses carry-aware logic, short semantic persistence, post-intervention cooldown, progress-aware suppression, no-progress abandonment, and hard retry budgets. These mechanisms are not introduced as a substitute for learning. Their purpose is to produce temporally localized interventions whose outcomes can be attributed and verified: a failed chunk, a specific recovery decision, one corrective chunk, and the subsequent boundary.

Every decision writes an auditable record containing the evidence source, failure class, prompt, action source, intervention state, and next observation. Thus an online recovery mechanism also becomes an automatic experience generator. Without recovery, a deviation often yields only a terminal failure; with bounded recovery, the same rollout can reveal whether a corrective behavior already present in the VLA is effective from the visited off-trajectory state.

3.4 Offline Verify-and-Internalize Loop

The offline loop turns the auditable records produced during deployment into grounded supervision, then uses that supervision to improve recovery decisions in subsequent rounds.

Outcome verification. The deployed agent stores boundary records containing observations, robot state, feedback states, coarse and fine actions, recovery metadata, and episode termination. Privileged progress is never present in Equation 5; when available in simulation, it is attached only after rollout termination for offline verification.

The verifier produces three targets. A normalized recovery-value target $q_n \in [0, 1]$ represents subsequent physical progress or successful continuation. A binary failure target $y_n^f \in \{0, 1\}$ marks whether the action at boundary n led to a detected or terminal failure. A pairwise relation ρ links a failed attempt to a verified correction from the same local episode context. A correction is considered verified only if a subsequent boundary exhibits measurable progress, clears the relevant failure under independent evidence, or completes the task. Failed episodes without an explicit detector event receive conservative labels only near termination. The reasoner's generated text is never used as its own ground-truth target.

This verifier transforms raw self-generated trajectories into four useful strata: stable success, failure without recovery, failure followed by verified recovery, and failure after attempted recovery. The fail–recover strata are especially valuable because they compare actions on the policy's own off-trajectory state distribution rather than on states selected by an external demonstrator.

Recovery Critic and objective. The Recovery Critic scores the action proposed by the fixed VLA. Let P_n be the pooled VLM prefix representation. Linear projections map P_n, s_n, the feedback states H_n, and each candidate action token into a common width. The action tokens attend to the multimodal and feedback context and are pooled across time into an action-context representation v_n, which feeds two prediction heads:

$$\hat{q}_n = \sigma(f_q(v_n)), \qquad \hat{p}_n = \sigma(f_f(v_n)) \equiv \widehat{\Pr}_\psi\big(y_n^f = 1 \mid P_n, s_n, H_n, a_n\big)\,. \tag{9}$$

The internalization objective, averaged over $\widetilde{\mathcal{D}}_k$, is

$$\mathcal{L}_{\text{int}} = \lambda_q \mathcal{L}_{\text{Huber}}(\hat{q}, q) + \lambda_f \mathcal{L}_{\text{BCE}}(\hat{p}, y^f) + \lambda_r \,\text{softplus}\big(\gamma - (\hat{q}^+ - \hat{q}^-)\big)\,, \tag{10}$$

where $\gamma > 0$ is the ranking margin, and $(\hat{q}^+, \hat{q}^-)$ are the predicted values of a correction–failure pair linked by ρ. Optimization updates only ψ; the VLM, action experts, action heads, and feedback adapters remain fixed. In subsequent deployments, $\hat{p}_n$ is retained until boundary $n+1$ and supplied to the Recovery Orchestrator as delayed learned evidence; $\hat{q}_n$ supports diagnostics or candidate ranking when diverse candidates are available.

Unlike transient reflection, updating ψ changes recovery decisions in future episodes. Freezing the action stack isolates this persistent gain from motor-policy changes and avoids the risks of continual motor-policy updates.

4 Experiments

The experiments cover four questions: how performance changes under distribution shift, how online recovery differs from critic internalization, which experience should be retained, and which feedback and governance components are necessary. Sections 4.1–4.3 present the protocol, main comparisons, and controlled ablations; iterative self-improvement and systems analysis appear in Appendix A.

4.1 Experimental Setup

We report point estimates from a frozen evaluation protocol. All controlled comparisons in Table 3 and the appendix use the same manifests, seeds, and metric definitions. Tables 1 and 2 instead reproduce results reported by prior work under the corresponding official benchmark protocols; they are not treated as paired evaluations with shared episode seeds, and rates with different denominators are never differenced.

We evaluate on two robustness benchmarks built on LIBERO (Liu et al., 2023). LIBERO-Plus covers camera viewpoint, robot initial state, language instruction, lighting, background texture, sensor noise, and object layout (Fei et al., 2026); LIBERO-PRO covers object, position/swap, semantic, task, and environment shifts (Zhou et al., 2025). We use each official aggregate and a separate unperturbed retention manifest. Training rollouts and evaluation episodes use disjoint task instances and seeds, and held-out-family evaluation additionally excludes entire perturbation families from critic training.

All controlled systems share the initial action checkpoint, action horizon, observation resolution, environment version, and evaluation seeds. Appendix B gives the component-wise matrix and identifies the information available to each system.

The primary metric is the official aggregate task success rate (SR). Conditional recovery is the fraction of detected or injected failures after which progress resumes; harm is the paired success-to-failure flip rate on common episode keys. We additionally report repeated interventions per episode, PR-AUC for the Recovery Critic's failure target, and retention on unperturbed tasks. For no-recovery variants, recovery, harm, and repeat columns are zero because they refer specifically to orchestrator interventions; feedback-induced action changes remain reflected in SR.

4.2 Main Results

Robustness under distribution shift. Tables 1 and 2 compare the complete critic-internalized system with representative reported baselines.

Table 1: LIBERO-Plus success rates (%) under seven distribution shifts. Column maxima are bold.

Family	Method	Camera	Robot	Language	Light	Background	Noise	Layout	Total
VLA	OpenVLA	0.8	3.5	23.0	8.1	34.8	15.2	28.5	15.6
	π_0	13.8	6.0	58.8	85.0	81.4	79.0	68.9	53.6
	$\pi_{0.5}$	64.8	71.8	83.0	93.5	92.2	78.8	85.5	81.4
	VLA-JEPA	63.3	67.1	85.4	95.6	93.6	66.3	85.1	79.5
WAM	Fast-WAM	14.3	41.0	65.6	77.8	51.8	37.0	60.0	48.1
	Faster-WAM	53.8	71.6	94.7	96.3	61.3	63.6	79.1	73.6
	Enfold	59.2	61.0	92.1	89.7	48.7	35.8	76.6	66.2
	DC-WAM	23.9	51.7	83.4	91.7	61.3	54.2	69.8	60.9
	Cosmos-Policy	75.8	63.3	81.7	96.5	88.9	92.7	82.2	82.2
	ARPS	88.6	62.8	**97.8**	97.5	91.5	95.0	81.8	87.3
Ours	RRI-VLA	**97.81**	**82.06**	83.28	**99.04**	**99.26**	**98.63**	**87.21**	**92.47**

Table 2: LIBERO-PRO success rates (%); columns average four official suites. "–" is unreported; maxima are bold.

Family	Method	Object	Position/Swap	Semantic	Task	Environment	Average
VLA	OpenVLA	93.0	0.0	**97.3**	0.0	**68.0**	52.0
	π_0	90.5	0.0	90.5	0.0	38.8	44.0
	$\pi_{0.5}$	**96.0**	20.8	95.8	0.8	52.8	53.0
	MolmoAct	76.0	1.5	85.8	1.5	–	41.0
	NORA	70.5	0.0	86.3	0.0	–	40.0
	X-VLA	79.0	0.8	96.8	6.8	–	46.0
Ours	RRI-VLA	84.35	**27.90**	94.80	**26.40**	55.75	**57.84**

RRI-VLA reaches 92.47% on LIBERO-Plus, 5.17 points above the strongest listed aggregate baseline, and leads six of seven families; language is the exception (83.28% versus ARPS at 97.8%). Its 57.84% LIBERO-PRO average leads by 4.84 points. Position/Swap (27.90%) and Task (26.40%) remain hardest, suggesting that observable execution deviations are easier to recover from than shifts that invalidate the task-level action sequence.

Table 3: Controlled results (%). Bold marks the best feedback-enabled frozen-action system; tuning and oracle rows are diagnostic.

System	Plus SR	PRO SR	Recovery	Harm	Retention
Static VLA	85.60	46.80	0.0	0.00	95.70
External reflection	89.80	53.20	64.5	2.40	95.40
RRI-VLA online	89.35	52.20	63.8	2.05	**95.55**
RRI-VLA shuffled-label	87.65	49.05	46.0	4.45	91.85
RRI-VLA internalized	**92.47**	**57.84**	**74.2**	**1.10**	95.30
Action-interface tuning	93.35	59.60	76.0	1.35	93.85
Oracle trigger	94.85	63.20	84.0	0.45	95.55

Controlled system comparison. Internalization improves the static VLA by 6.87/11.04 points on Plus/PRO and the online-only system by 3.12/5.64 points. It raises recovery from 63.8% to 74.2%, lowers harm from 2.05% to 1.10%, and changes retention by only 0.25 points. The oracle gap (2.38/5.36 points) identifies remaining trigger headroom; action-interface tuning gains 0.88/1.76 points but reduces retention to 93.85%. At the same checkpoint, the learned trigger alone adds 2.77/5.19 points; Appendix C provides the full and held-out-family analyses.

4.3 Ablation Study

What experience should be internalized? All variants in Table 4 use the same sample count, critic initialization, optimizer budget, batch composition, and evaluation seeds. Deltas are relative to the full verified mixture.

Table 4: Experience-composition ablation. SR, harm, and retention are percentages.

Training experience	Plus	ΔPlus	PRO	ΔPRO	PR-AUC	Harm	Retention
Success only	89.35	-3.12	52.10	-5.74	0.641	2.65	95.45
Failure only	90.10	-2.37	54.25	-3.59	0.821	3.35	93.90
Attempted recovery	90.88	-1.59	55.32	-2.52	0.842	2.28	94.60
Verified fail–recover pairs	92.05	-0.42	57.18	-0.66	0.876	1.35	95.05
Full verified mixture	**92.47**	**0.00**	**57.84**	**0.00**	**0.889**	**1.10**	95.30
Shuffled targets	87.65	-4.82	49.05	-8.79	0.421	4.45	91.85

Verified fail–recover pairs trail the full mixture by only 0.42/0.66 points, showing the value of localized failed-proposal/verified-correction supervision. Failure-only training over-triggers (3.35% harm), whereas shuffled targets cause the largest drop (4.82/8.79 points; PR-AUC 0.421). Appendix C shows that equal-size expert and uniform sources also trail self-collected, policy-aligned transitions.

Feedback and recovery. Table 5 combines feedback controls with recovery-scope controls under a fixed verified mixture and critic.

Table 5: Feedback and recovery-scope ablations. All rates are percentages.

Variant	Plus SR	PRO SR	Recovery	Harm flips	Repeats/ep.
Zero feedback	87.95	49.75	44.8	2.65	0.18
Shuffled feedback	86.80	48.10	41.2	4.25	0.27
Online feedback, no recovery	89.25	52.05	0.0	0.00	0.00
Persistent unrestricted correction	90.35	54.18	72.8	4.85	0.74
One-chunk correction	91.45	55.80	67.9	2.18	0.20
Full governed recovery	**92.47**	**57.84**	**74.2**	**1.10**	**0.12**

Correct feedback exceeds shuffled feedback by 5.67/9.74 points on Plus/PRO. Unrestricted correction reaches 72.8% recovery but incurs 4.85% harm and 0.74 repeats per episode, versus 1.10% and 0.12 with full governance. Relative to one-chunk correction, the complete governor adds 1.02/2.04 points and lowers harm by 1.08 points. Appendix C contains the field, training–inference, and rule-level ablations.

5 Conclusion

We presented RRI-VLA, a *reflect–recover–internalize* framework that treats execution feedback as a shared interface for two-timescale improvement of vision-language-action policies. Rather than discarding deployment consequences, our design converts each action boundary into a structured record that serves two roles: it provides causal context for the next decision through a within-episode loop pairing a frozen reasoner with a governed Recovery Orchestrator, and it seeds an across-deployment loop where an offline verifier retains only confirmed corrections to train a Recovery Critic. Because only the critic is updated while the VLA stays frozen, the agent's motor competence is preserved by construction. Controlled experiments confirm that self-collected policy-aligned experience outperforms expert or uniform data, that causal feedback alignment is essential, and that bounded governance keeps recovery productive while suppressing harmful interventions—suggesting that execution feedback offers a lightweight pathway to persistent improvement.

AI USE STATEMENT

In this work, we used generative AI tools for two tasks in the required-disclosure category: assisting with translation and supporting the interpretation and presentation of author-provided experimental results during manuscript revision. We did not use generative AI tools to generate datasets, develop theoretical models or conceptual frameworks, formulate or prove mathematical claims, propose or refine hypotheses, design methodology or experiments, implement methods, clean or reformat datasets, or perform qualitative or thematic data analysis. Additionally, we used generative AI tools for tasks in the recommended-disclosure category: editing text to improve readability, drafting and revising prose, suggesting document structure, formatting references, and creating or modifying draft figures and illustrative diagrams. The authors checked all AI-assisted interpretations against the underlying experimental outputs and fixed evaluation tables, and reviewed all AI-assisted text and figures for accuracy and originality. We take full responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies execution-time recovery and offline self-improvement for vision-language-action policies. It is evaluated entirely in simulation on the publicly available LIBERO-Plus and LIBERO-PRO benchmarks under their respective terms, involves no experiments on human subjects, and collects no personally identifiable or sensitive information. We acknowledge that an agent that improves its own recovery behavior from self-collected experience could reinforce systematic errors when the verifier, feedback model, and recovery critic share the same blind spot, and that physical deployment would raise additional safety concerns. To mitigate these risks we treat verification as a distinct offline stage, retain raw observations for audits, and require held-out perturbation and retention gates, while bounded recovery, immutable task text, cooldown, and intervention budgets limit the consequences of an incorrect reflection; physical deployment would additionally require collision avoidance, hardware safety controllers, calibrated out-of-distribution detection, and human override. The authors have read and adhere to the ICLR Code of Ethics.

REPRODUCIBILITY STATEMENT

The main paper and appendix specify the model components, two feedback loops, boundary-record contents, Recovery Critic objective, evaluation benchmarks, controlled-comparison protocol, and metric definitions needed to understand and independently implement the proposed framework. Appendix B further describes the immutable self-improvement round structure and the fields stored in each boundary record, while Appendix C reports the feedback-content, experience-source, training–inference, and recovery-governance ablations.

Exact reproduction of the reported point estimates additionally requires the frozen manifests, perturbation instances, feedback prompts, optimization configurations, checkpoint hashes, and evaluation scripts. These artifacts, together with the collection and verification code, training mixtures, and paired statistical analyses, will be released with the final submission.

REFERENCES

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

Konstantinos Bousmalis, Giulia Vezzani, Dushyant Rao, Coline Devin, Alex X Lee, Maria Bauzá, Todor Davchev, Yuxiang Zhou, Agrim Gupta, Akhil Raju, et al. Robocat: A self-improving generalist agent for robotic manipulation. *arXiv preprint arXiv:2306.11706*, 2023.

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.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, et al. Libero-plus: A progressive robustness benchmark for visual-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 38574–38583, 2026.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.

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

Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with multimodal prompts. In *Fortieth International Conference on Machine Learning*, 2023.

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

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.

Sung-Wook Lee, Xuhui Kang, and Yen-Ling Kuo. Diff-dagger: Uncertainty estimation with diffusion policy for robotic manipulation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4845–4852. IEEE, 2025.

Baicheng Li, Dong Wu, Zike Yan, Xinchen Liu, Lusong Li, Zecui Zeng, and Hongbin Zha. Reflection-based task adaptation for self-improving vla. *arXiv preprint arXiv:2510.12710*, 2025a.

Hengtao Li, Pengxiang Ding, Runze Suo, Yihao Wang, Zirui Ge, Dongyuan Zang, Kexian Yu, Mingyang Sun, Hongyin Zhang, Donglin Wang, et al. Vla-rft: Vision-language-action reinforcement fine-tuning with verified rewards in world simulators. *arXiv preprint arXiv:2510.00406*, 2025b.

Qing Lian, Kent Yu, and Lei Zhang. Reflective vla: In-context action consequences make vlas generalize. *arXiv preprint arXiv:2606.25215*, 2026.

Zijun Lin, Jiafei Duan, Haoquan Fang, Dieter Fox, Ranjay Krishna, Cheston Tan, and Bihan Wen. Failsafe: Reasoning and recovery from failures in vision-language-action models. *arXiv preprint arXiv:2510.01642*, 2025.

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.

Chenyang Ma, Kai Lu, Guangyu Yang, Jiuming Liu, Shitong Xu, Bill Byrne, Ioannis Havoutis, Niki Trigoni, and Andrew Markham. Cyclevla: Proactive self-correcting vision-language-action models via subtask backtracking and minimum bayes risk decoding. *arXiv preprint arXiv:2601.02295*, 2026.

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: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

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 visual-language-action model. *arXiv preprint arXiv:2501.15830*, 2025.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

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.

Jingwen Sun, Wenyao Zhang, Zekun Qi, Shaojie Ren, Zezhi Liu, Hanxin Zhu, Guangzhong Sun, Xin Jin, and Zhibo Chen. Vla-jepa: Enhancing vision-language-action model with latent world model. In *European Conference on Computer Vision*, pp. 478–497. Springer, 2026.

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.

Wenli Xiao, Haotian Lin, Andy Peng, Haoru Xue, Tairan He, Zhengyi Luo, Yuqi Xie, Fengyuan Hu, Jim Fan, Guanya Shi, et al. Self-improving vision-language-action models with data generation via residual rl. In *International Conference on Learning Representations*, volume 2026, pp. 13209–13236, 2026.

Jiayu Yang, Tao Yang, Weijun Li, Xiang Chang, Fei Chao, Changjing Shang, and Qiang Shen. Physreflect-vla: Physical feasibility and self-reflective regulation for reliable vision-language-action policies. *arXiv preprint arXiv:2606.27146*, 2026a.

Yifan Yang, Zhixiang Duan, Tianshi Xie, Fuyu Cao, Pinxi Shen, Peili Song, Chenyang Zhao, Piaopiao Jin, Guokang Sun, Shaoqing Xu, et al. Fpc-vla: A vision-language-action framework with a supervisor for failure prediction and correction. *Expert Systems with Applications*, pp. 131742, 2026b.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Yixian Zhang, Huanming Zhang, Feng Gao, Xiao Li, Zhihao Liu, Chunyang Zhu, Jiaxing Qiu, Yuchen Yan, Jiyuan Liu, Wenhao Tang, et al. Harness vla: Steering frozen vlas into reliable manipulation primitives via memory-guided agents. *arXiv preprint arXiv:2607.08448*, 2026.

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1702–1713. IEEE, 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. In *International Conference on Learning Representations*, volume 2025, pp. 54277–54296, 2025.

Linqing Zhong, Yi Liu, Yifei Wei, Ziyu Xiong, Si Liu, and Guanghui Ren. Acot-vla: Action chain-of-thought for vision-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8152–8162, 2026.

Xueyang Zhou, Yangming Xu, Guiyao Tie, Yongchao Chen, Guowen Zhang, Duanfeng Chu, Pan Zhou, and Lichao Sun. Libero-pro: Towards robust and fair evaluation of vision-language-action models beyond memorization. *arXiv preprint arXiv:2510.03827*, 2025.

A Self-Improvement and Systems Analysis

Table 6: Collect–verify–internalize rounds. SR, harm, and retention are percentages.

Round	Persistent update	Plus	PRO	PR-AUC	Harm	Retention
0	none; online recovery	89.35	52.20	0.742	2.05	**95.55**
1	critic update 1	91.35	55.45	0.847	1.42	95.40
2	critic update 2	**92.47**	**57.84**	**0.889**	**1.10**	95.30

Multiple self-improvement rounds. The first round improves Plus and PRO by 2.00 and 3.25 points; the second adds 1.12 and 2.39 points. Gains therefore continue but diminish, while PR-AUC rises monotonically and harm falls from 2.05% to 1.10%. The total two-round gain of 3.12 and 5.64 points costs only 0.25 points of unperturbed retention.

Critic-only internalization is our conservative default: action-interface tuning adds 0.88 and 1.76 points on Plus and PRO but lowers retention by 1.45 points and raises harm by 0.25 points; the complete update-scope comparison is reported in Appendix C.

B Experimental Protocol

B.1 Round structure

Each self-improvement round follows the same immutable sequence: select and hash the current checkpoint; collect rollouts only on designated training perturbations; finalize episodes before offline verification; construct immutable boundary records and recovery pairs; train the Recovery Critic with a fixed optimization budget; evaluate on locked LIBERO-Plus and LIBERO-PRO manifests; and apply retention and harmful-intervention gates. Evaluation trajectories never enter the training buffer, and hyperparameters are selected on a disjoint development manifest.

B.2 Controlled comparison matrix

Table 7 summarizes the controlled systems used throughout the evaluation and identifies the component changed by each comparison.

Table 7: Controlled comparison matrix. "Persistent update" is the component changed for the next deployment.

System	Feedback	Recovery	Self-data	Persistent update	Purpose
Static VLA	no	no	no	none	frozen action baseline
External reflection	text	agent/tool	no	memory only	training-free baseline
RRI-VLA online	latent+text	governed	collected	none	immediate recovery
RRI-VLA internalized	latent+text	governed	verified	critic	main system
RRI-VLA shuffled-label	latent+text	governed	shuffled	critic	information-free control
Action-interface tuning	latent+text	governed	verified	action subset	policy-update branch
Oracle trigger	latent+text	governed	no	none	recoverability upper bound

B.3 Boundary-record schema

Every stored boundary contains the task identifier and instruction; current front and wrist observations; current and previous robot states; the previously executed fine-action chunk and coarse trajectory; feedback hidden states and decoded audit text; semantic, physical, and learned failure

evidence; orchestrator state, corrective prompt, cooldown, and budgets; and the subsequent observation, termination status, and offline-only verifier fields. Dataset exports carry a schema version so records from incompatible feedback interfaces cannot be silently mixed.

B.4 Result definitions

All tables report measured point estimates. Plus SR and PRO SR use the official benchmark aggregates. Recovery is conditional on a detected or injected failure, harm is the common-key success-to-failure flip rate, and retention is measured on the corresponding unperturbed manifest. For iteration k, persistent improvement on the fixed manifest is

$$\Delta_k = \mathrm{SR}(\mathcal{A}_{k+1}) - \mathrm{SR}(\mathcal{A}_k). \tag{11}$$

A same-checkpoint trigger-on/off comparison isolates activation by the learned critic from changes to the underlying action policy.

C Detailed Ablation Results

C.1 Failure-evidence attribution

We isolate the evidence used by the Recovery Orchestrator while keeping the action-producing VLA, Recovery Critic architecture, recovery governance, evaluation manifests, and seeds fixed. At boundary n, the orchestrator can use semantic patterns in `OUTCOME`, physical diagnostics, and the Recovery Critic's delayed risk estimate $\hat{p}_{n-1}$ for the previously executed chunk. Thresholds are selected on a disjoint development manifest and then frozen for evaluation. All entries report measured point estimates; Plus SR, PRO SR, and harm are percentages.

Table 8: Failure-evidence attribution (%). Each variant changes only the evidence available for recovery admission; harm is the common-key success-to-failure flip rate.

Recovery evidence	Plus SR	PRO SR	Harm
Random score	87.85	49.30	4.20
Untrained critic	88.40	50.20	3.55
Semantic + physical evidence only	89.70	52.65	**0.35**
Learned critic evidence only	91.05	55.55	1.75
Semantic + physical + learned evidence	**92.47**	**57.84**	1.10

The trained critic provides non-redundant action-conditioned evidence. Relative to the untrained critic, learned critic evidence alone raises Plus SR by 2.65 points and PRO SR by 5.35 points while lowering harm by 1.80 points. Relative to semantic and physical evidence alone, learned critic evidence alone adds 1.35 and 2.90 points on Plus and PRO, respectively, at a 1.40-point increase in harm. This comparison indicates that the learned signal improves recovery admission beyond the deterministic evidence. Using it alone also produces more harmful interventions. Combining all three evidence sources yields the highest measured SR and reduces harm relative to learned critic evidence alone. The combined variant adds 1.42 points on Plus and 2.29 points on PRO while reducing harm by 0.65 points. Relative to semantic and physical evidence alone, it adds 2.77 and 5.19 points, with harm increasing from 0.35% to 1.10%. These complementary effects are consistent with the Recovery Critic expanding useful recovery admission while semantic and physical evidence constrains harmful over-triggering. The Oracle trigger in Table 3 remains 2.38 points higher on Plus and 5.36 points higher on PRO than the combined learned system. This gap identifies remaining headroom in recovery-admission quality. On the separate held-out-family manifest, the system with learned evidence obtains 90.65% Plus SR, 54.90% PRO SR, and 1.55% harm.

C.2 Internalization-objective ablation

We ablate the three terms in Equation 10 while fixing the verified training data, critic initialization, optimization budget, frozen action-producing VLA, Recovery Orchestrator, and evaluation manifests. The $\hat{p}$ head is trained with the BCE term, the $\hat{q}$ head is trained with the Huber regression term,

and the pairwise ranking term operates on correction–failure pairs linked by ρ. For variants without the $\hat{p}$ head, $1 - \hat{q}$ is calibrated as a risk score on the disjoint development manifest and the resulting threshold is frozen for evaluation. Because removing the $\hat{q}$ head leaves the BCE-only objective, that configuration is reported once.

Table 9: Internalization-objective ablation (%). We vary the terms in Equation 10; all other training and evaluation settings are fixed.

Internalization objective	Plus SR	PRO SR	Harm
Huber only	90.45	54.25	2.20
Huber + pairwise ranking (no $\hat{p}$ head)	90.95	55.05	1.90
BCE only (no $\hat{q}$ head)	91.30	55.75	1.62
BCE + Huber (no ranking)	91.85	56.70	1.35
Full objective	**92.47**	**57.84**	**1.10**

The full internalization objective attains the highest SR and the lowest harm. Comparing the full objective with Huber plus pairwise ranking isolates the end-to-end contribution of the $\hat{p}$/BCE pathway: Plus SR increases by 1.52 points, PRO SR increases by 2.79 points, and harm decreases by 0.80 points. The remaining comparisons separate the measured contributions of recovery-value regression and pairwise ranking. Adding Huber regression to BCE increases Plus SR by 0.55 points and PRO SR by 0.95 points while reducing harm by 0.27 points. Adding pairwise ranking to BCE and Huber further increases Plus SR by 0.62 points and PRO SR by 1.14 points while reducing harm by 0.25 points. Together, the two ablations show that the action-conditioned Recovery Critic contributes learned failure evidence beyond semantic and physical evidence, and that the Recovery Orchestrator benefits from combining these sources. Within the critic, BCE provides the deployed failure-risk pathway, while recovery-value regression and pairwise ranking provide additional measured improvements in the success–harm trade-off. With the action-producing VLA frozen, these gains arise from improved recovery decisions.

C.3 Source of internalization experience

Table 10 fixes sample count, label construction, optimization budget, and critic architecture while changing only the source of transitions.

Table 10: Experience-source ablation. Deltas are relative to self-collected, policy-aligned data; SR, harm, and retention are percentages.

Experience source	Plus	ΔPlus	PRO	ΔPRO	PR-AUC	Harm	Retention
Self-collected, policy-aligned	**92.47**	**0.00**	**57.84**	**0.00**	**0.889**	**1.10**	95.30
Equal-size expert transitions	90.74	-1.73	54.68	-3.16	0.808	1.95	**95.55**
Equal-size uniform transitions	89.66	-2.81	52.92	-4.92	0.753	2.62	94.85

Expert transitions retain nominal competence slightly better, but trail self-collected experience by 1.73 and 3.16 points because they under-sample boundaries induced by the current policy's own failures. Uniform transitions lack both failure density and local fail–recover correspondence, producing the weakest critic and the highest harm among the three sources. This supports policy alignment, rather than sample count alone, as the source of the internalization benefit.

C.4 FEEDBACK CONTENT AND REPRESENTATION

Table 11: Feedback-content ablation with the verified mixture, Recovery Critic, and full governor fixed. All rates are percentages.

Feedback setting	Plus	PRO	Recovery	Harm	Repeats/ep.
Online latent+text	**92.47**	**57.84**	**74.2**	**1.10**	**0.12**
Zeroed	87.95	49.75	44.8	2.65	0.18
Shuffled within task	86.80	48.10	41.2	4.25	0.27
One-boundary delayed	89.42	52.60	55.9	2.45	0.21
Text only	90.63	54.55	61.8	1.98	0.17
Latent only	91.78	56.42	68.6	1.42	0.14
Outcome only	90.92	54.93	63.7	1.84	0.16
Matched continuation-trained null	88.35	50.55	49.6	2.71	0.20

In Table 11, "Online latent+text" denotes the full two-channel interface: latent states condition the action experts, while decoded text is available to the Recovery Orchestrator. "Latent only" removes decoded semantic evidence from the orchestrator but retains latent action conditioning. "Text only" masks the latent feedback adapters while retaining decoded semantic evidence and any resulting bounded corrective prompt.

Latent-only feedback is closest to the full interface, trailing by 0.69 and 1.42 points, because the action experts consume the fixed-size latent states directly. Removing decoded semantics weakens failure categorization and governance, however, increasing harm from 1.10% to 1.42%. Outcome-only feedback omits running episode memory and the next adjustment, while delayed feedback arrives after the most relevant correction boundary. Shuffled feedback is worse than zeroed feedback because it injects incorrect causal content rather than merely removing information.

C.5 FEEDBACK TRAINING–INFERENCE CROSS

Table 12: Matched 2×3 feedback training–inference experiment (SR in %).

Training setting	Inference feedback	Plus	PRO
Feedback-trained	Correct online	**92.47**	**57.84**
	Zeroed	87.95	49.75
	Shuffled within task	86.80	48.10
Matched null continuation	Correct online	88.35	50.55
	Zeroed	88.10	50.20
	Shuffled within task	87.70	49.55

The difference-in-differences interaction between feedback training and correct content is 5.02 points on Plus, $(92.47-86.80)-(88.35-87.70)$, and 8.74 points on PRO, $(57.84-48.10)-(50.55-49.55)$. With matched null continuation, correct feedback exceeds shuffled feedback by only 0.65 and 1.00 points. The dominant gain therefore requires the learned causal feedback interface rather than an inference-time text or recovery prompt alone.

C.6 RECOVERY-GOVERNANCE RULES

Table 13: Leave-one-rule-out governance ablation. Deltas are relative to the full governor; rates are percentages.

Governance	Plus	ΔPlus	PRO	ΔPRO	Recovery	Harm	Repeats/ep.
Full governed recovery	**92.47**	**0.00**	**57.84**	**0.00**	74.2	**1.10**	**0.12**
No carry awareness	91.62	-0.85	56.28	-1.56	69.5	1.92	0.21
No cooldown	91.74	-0.73	56.53	-1.31	74.6	2.25	0.38
No progress suppression	91.18	-1.29	55.62	-2.22	72.5	2.95	0.32
No no-progress abandonment	91.86	-0.61	56.65	-1.19	73.9	1.87	0.28
No hard retry budgets	90.72	-1.75	54.98	-2.86	**75.1**	3.92	0.61

Hard retry budgets are the most important individual rule for the success–safety trade-off. Removing them raises nominal recovery from 74.2% to 75.1%, but lowers Plus and PRO by 1.75 and 2.86 points and increases harm from 1.10% to 3.92%. Progress suppression is second most consequential, with losses of 1.29 and 2.22 points. Cooldown and no-progress abandonment affect repeats more than raw recovery, confirming that a high conditional recovery count is insufficient when repeated interventions create new failures.

C.7 SCOPE OF PERSISTENT UPDATING

Table 14: Persistent-update scope (%).

Persistent update	Plus SR	PRO SR	Harm	Retention
No update	89.35	52.20	2.05	**95.55**
Critic only	92.47	57.84	**1.10**	95.30
Action interface	**93.35**	**59.60**	1.35	93.85

Updating the action interface improves *how* the model recovers, adding 0.88 points on Plus and 1.76 on PRO beyond critic-only internalization. It also lowers retention by 1.45 points and raises harm by 0.25 points. We therefore use critic-only internalization as the main method: it captures most of the attainable success gain while leaving the action-producing stack fixed and preserving nominal behavior.

C.8 COMPUTATIONAL OVERHEAD AND LATENCY

We profile the computational overhead of RRI-VLA by separating action generation from multimodal reflection. Under matched action checkpoints, evaluation manifests, and seeds on LIBERO-Plus, online Structured Execution Feedback changes action-chunk inference latency by only -0.17% relative to zero feedback. This result shows that feedback conditioning introduces effectively no additional latency to our fixed action policy. Our frozen Failure-Aware Reasoner E_ω operates once after each complete action chunk rather than at every low-level control step, isolating multimodal reflection from the high-frequency action-execution loop.

We further measure the incremental cost of critic internalization over the online reflect-and-recover system. The Recovery Critic adds approximately 1.4% computation per action boundary, while enabling its learned trigger changes the measured per-call latency of the action policy and Failure-Aware Reasoner by -2.47% and -1.65%, respectively. At the episode level, recorded module time increases by 12.2% because admitted corrections increase action-policy and reasoner calls by 22.3% and 24.3%, rather than because individual calls become slower. The on-disk checkpoint size increases by 14.3% after critic internalization.

Overall, our measurements show that RRI-VLA preserves the per-call inference latency of the action-producing VLA. End-to-end episode-level module time nevertheless increases by 12.2% be-

cause admitted corrections invoke additional action-policy and reasoner calls. The direct boundary-level cost of critic internalization remains only 1.4%. We therefore isolate the incremental computation to bounded, chunk-level reflection and recovery decisions without slowing low-level action generation.

D Limitations and Future Work

Scope of internalization. RRI-VLA deliberately internalizes recovery competence—predicting when an action is likely to fail and when a bounded correction is beneficial—while keeping the generalist action policy fixed. The resulting gains therefore reflect persistent improvement in recovery decisions rather than unrestricted acquisition of new motor skills. This conservative scope preserves 95.30% success on unperturbed tasks and limits harmful interventions to 1.10%. Future work will study retention-gated updates of the action interface, extending internalization to correction quality only when nominal competence can be preserved.

Verification and transfer. Our experiments use controlled perturbations in LIBERO-Plus and LIBERO-PRO, where simulator-available progress signals support offline outcome verification; these signals are never exposed to the deployed action policy. The present results therefore establish persistent recovery learning under controlled robustness shifts rather than physical-robot safety. Future work will replace simulator-derived labels with calibrated visual outcome detectors and uncertainty-aware verification, and evaluate the complete loop on real robots with hardware safety controllers and out-of-distribution safeguards.