EMBODIEDHARNESS: ADAPTIVE HARNESS OPTIMIZATION FOR SELF-EVOLVING EMBODIED AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

An embodied agent's performance depends not only on its foundation model but also on the system built around it: how skills are organized, how context is managed, what is checked before a task is declared complete, and what happens after a failed action. In partially observable, long-horizon tasks, these model-external components, spanning skills, context, verification, and recovery and collectively termed the harness, decide whether a frozen model can operate in closed loop. We present EmbodiedHarness, a non-parametric harness optimization framework that improves the structure and content of the harness from a small number of target-environment rollouts while keeping planner and executor weights frozen. At the interaction level a frozen vision-language model acts through a modular harness; at the evolution level the same model serves as the optimizer and runs a diagnose-revise-validate loop that accepts a revision only if it fixes some episodes without losing any the parent already solved. The central difficulty in this loop is harness-structure evolution under sparse feedback: a failed episode returns one binary outcome, and every component admits a revision consistent with it, so the evolution direction cannot be read off the trajectory. We address this credit-assignment problem with anchored counterfactual replay, which restores the simulator to the deviation anchor, applies a single-component revision, and replays only the suffix, testing each candidate direction at a fraction of a rollout before budget is committed. On VLABench, EmbodiedHarness improves average success by +14.1 points over the frozen agent and, under the same backbone, +10.2 points over the sequential optimizer baseline methods; it gains +23.0 micro points over the frozen agent on ESI-Bench and surpasses all open-source train-free methods on EmbodiedBench Habitat and Navigation.

1 INTRODUCTION

Embodied AI is moving from isolated policy models toward complete agent systems. Vision-language-action and multimodal foundation models can perceive a scene, interpret an instruction, and act in simulated or physical environments (Kim et al., 2024; Black et al., 2024; Intelligence et al., 2025; Gemini Robotics Team et al., 2025; NVIDIA et al., 2025), and a growing line of work places such a model at the center of an agent that plans, calls controllers, and inspects what came back (Ahn et al., 2022; Liang et al., 2022; Hong et al., 2026; Zhang et al., 2024b; Yang et al., 2025). The agent's behavior is not set by the model alone: the skills it is handed, the observations kept in its context, the checks it runs before declaring a task done, and the rules it follows after a failed action all shape what it does. Recent work refers to these model-external components as the *harness* (Lin et al., 2026a; Wang et al., 2026d). In partially observable, long-horizon manipulation tasks the harness matters most, since a slipped grasp may surface in the observations only several steps later. Keeping such a harness improving in the target environment is therefore foundational to deploying frozen models in embodied settings.

The harness is text and code, so interaction experience can improve it without touching model parameters, a non-parametric route to adapting embodied agent systems. General agents already write their own skills, reflections, and context playbooks (Wang et al., 2023; Shinn et al., 2023; Sarch et al., 2024; Zhang et al., 2025b); in embodied settings, Zetta evolves runtime critics and recovery modules, EmbodiSkill evolves skills, Evo-Harness evolves execution wrappers, and GEPA rewrites every component at once (Ding et al., 2026a; Ju et al., 2026; Wei et al., 2026; Agrawal et al., 2025).

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.

The evolution itself, however, is organized in a limited way. Its direction is fixed before any trajectory is seen, by a preset order over components (Wang et al., 2026b), by restriction to one component type (Ding et al., 2026a), or by rewriting everything each round (Agrawal et al., 2025). The content of a revision comes from the model's post-hoc reflection, a hypothesis rather than tested evidence. Validation, where it exists, stays inside the family being evolved: Zetta runs post-update rollouts before it adopts a critic or recovery bundle, yet a skill edit is never weighed against the context edit the same budget could have bought. In all three cases, trajectory feedback decides how to revise, never what to revise.

The central question is how to sustain structural evolution of the harness under sparse feedback. A failed episode ends with one binary outcome that says nothing about which part is responsible, since a context, a skill and a verification revision can each be made consistent with the same failure. Credit assignment cannot be read from this signal, and post-hoc attribution by the model itself is unreliable at the step level (Shah, 2026).

To address this challenge, we present **EmbodiedHarness**, a non-parametric harness optimization framework for embodied agents that covers skills, context, verification, and recovery. At the interaction level, a frozen vision-language model acts through a modular harness; at the evolution level, the same frozen model switches to the role of optimizer, diagnosing failures, revising the harness, and validating improvements in a three-stage loop. EmbodiedHarness tests each candidate evolution direction empirically through *anchored counterfactual replay*: the simulator state and the recorded evidence log are restored to the first observable deviation in the trajectory (the deviation anchor), a single-component revision is applied for each candidate, and the parent harness and each candidate complete the remaining steps from the same restored state and seed. We organize this loop into three stages. First, diagnosis maps trajectory evidence to a failure type and a deviation anchor and narrows the candidates to at most two components. Then, revision constructs a bounded single-component edit for each candidate and selects among them by anchored counterfactual replay. Last, validation runs the selected candidate and its parent on the same episodes and seeds and accepts the edit only if it fixes some episodes without losing any the parent already solved.

EmbodiedHarness raises VLABench average success by +14.1 points over the frozen agent on the same backbone, and by +10.2 points over the sequential optimizer baseline methods. It gains +23.0 micro points over the frozen agent on ESI-Bench, and on EmbodiedBench it surpasses all open-source train-free methods on both Habitat and Navigation.

Our contributions are threefold. (1) We present EmbodiedHarness, a non-parametric harness optimization framework whose two-level architecture and diagnose-revise-validate loop let a frozen foundation model improve its harness from a few target-environment rollouts without parameter updates. (2) We identify harness-structure evolution under sparse feedback as its central credit-assignment problem and propose anchored counterfactual replay, which turns the choice of evolution direction from a post-hoc guess into a measured comparison of repairs that were run from the same restored state. (3) We conduct comprehensive experiments on VLABench, EmbodiedBench, and ESI-Bench. EmbodiedHarness outperforms every comparable open-source train-free method on all three.

2 Related work

Self-evolving embodied agents. Reasoning-action loops, verbal feedback and reusable skill libraries were established by ReAct, Reflexion and Voyager (Yao et al., 2022; Shinn et al., 2023; Wang et al., 2023), and later work turned experience into reusable examples (Sarch et al., 2024), evolving context playbooks (Suzgun et al., 2025; Zhang et al., 2025b), external memory (Xu et al., 2025; Chhikara et al., 2025; Liu et al., 2026b) and visual skill memory (Wang et al., 2026a). Embodied systems evolve narrower objects: skills on a fixed skill-then-harness schedule (Wang et al., 2026b), critic and recovery bundles behind post-update validation (Ding et al., 2026a), skill files distilled from experience (Wu et al., 2026a), skill content separated from execution lapses (Ju et al., 2026), candidates audited by paired trajectories (Gao et al., 2026), and single-shot context compiled into reusable harnesses (Wei et al., 2026). Each fixes the direction in advance and validates, where it validates at all, only inside the component family it already chose to evolve. EmbodiedHarness selects the direction itself by anchored counterfactual replay across skill, context, verifier and recovery edits under one budget.

Harness engineering for agents. Language-model pipelines became optimizable objects with DSPy (Khattab et al., 2023) and then whole systems with workflow optimizers and self-modifying agents (Zhang et al., 2024a; 2025a; Liu et al., 2026a), while GEPA reflects on traces to revise prompts and merge candidates (Agrawal et al., 2025). The harness view treats prompts, tools and context templates as an evolvable system around a fixed model (Lin et al., 2026a; Wang et al., 2026d), with a shared interface for skill proposals, execution and verification in EmbodiedSkills (Wang et al., 2026c). Causal Agent Replay attributes failures by re-executing intervened steps, but only in mocked tool environments without side effects (Shah, 2026). We bring that intervention into a physics simulator with exact state restoration and use it for cross-component credit assignment, so that a single-component suffix replay from a restored anchor estimates one repair direction at a fraction of a full rollout. Neither validation-gated single-type evolution nor post-hoc re-execution offers this selection across harness dimensions under a shared budget.

Train-free adaptation for embodied agents. Frozen language models reach grounded action through affordance scoring and executable programs (Ahn et al., 2022; Liang et al., 2022), and spatial evidence accumulates through tools and memory (Dai et al., 2026; Zhou et al., 2026). The parametric alternative trains the model: VLA models learn perception-action interfaces (Kim et al., 2024; Black et al., 2024; Intelligence et al., 2025; Li et al., 2024; NVIDIA et al., 2025; Gemini Robotics Team et al., 2025), action priors and temporal modeling address embodiment transfer and chunk horizons (Jing et al., 2026b;a; 2025), geometry-aware models add spatial representations (Lin et al., 2026b; Wu et al., 2026b), and reinforcement learning optimizes memory use or skill-augmented policies (Yan et al., 2025; Xia et al., 2026). We keep model and executor fixed within each comparison and adapt only the harness. ESI-Bench measures active spatial interaction, VLABench evaluates reasoning-intensive manipulation and EmbodiedBench covers several embodied interfaces (Hong et al., 2026; Zhang et al., 2024b; Yang et al., 2025); we leave their scoring rules unchanged, state observation privileges explicitly, and report scores published under other models or action spaces as reference only.

3 EMBODIEDHARNESS: ADAPTIVE HARNESS OPTIMIZATION

EmbodiedHarness turns failed rollouts into persistent harness revisions in three stages: diagnosis, component selection and paired validation (Figure 1). Sections 3.1 to 3.5 state the objective and the modular interface, map failed trajectories to evidence-cited diagnoses, turn a diagnosis into a revision target tested by anchored counterfactual replay, bound the edit, and give the acceptance rule.

3.1 PROBLEM FORMULATION

We model embodied interaction as a task-conditioned partially observable process $\mathcal{M}_x = (\mathcal{Q}, \mathcal{A}, \mathcal{O}, p, p_o, \mu_x, T)$, where $\mathcal{Q}$ is the physical state space, written so as not to be read as the skill artifact S. The hidden state $s_t \in \mathcal{Q}$ evolves under p; the agent sees only $o_t \sim p_o(\cdot \mid s_t, x)$ and public receipts r_t. A harness $H = (S, C, V, R)$ supplies skills, context selection, verification and recovery. With observable history $h_t = (x, o_{0:t}, a_{0:t-1}, r_{0:t-1})$, which carries the receipts alongside observations and actions, execution factorizes as

$$u_t \sim \pi_\theta(\cdot \mid C(h_t), S), \qquad a_t \sim K_{V,R}(\cdot \mid h_t, u_t), \tag{1}$$

$$s_{t+1} \sim p(\cdot \mid s_t, a_t), \qquad \theta = \text{const}. \tag{2}$$

Here u_t is a proposed command and $K_{V,R}$ applies the fixed executor contract with verification and bounded recovery. Neither the model nor a learned actor is updated, and artifacts can neither expose privileged state nor alter scoring.

For a task distribution $\mathcal{D}$ and evaluator-assigned success $y(\tau) \in \{0, 1\}$, the target is

$$\begin{aligned} \max_{H \in \mathcal{H}} \quad & J(H) := \mathbb{E}_{x \sim \mathcal{D},\, \tau \sim P_\theta^H(\cdot|x)}[y(\tau)] \\ \text{s.t.} \quad & \sum_{k=0}^{K-1} c_k \le B_{\text{acq}}, \quad \mathbb{E}[c_{\text{infer}}(\tau)] \le B_{\text{infer}}, \quad \rho(H, H_0) \le \epsilon. \end{aligned} \tag{3}$$

Acquisition cost c_k covers diagnosis, proposals, rejected candidates, replays and validation. The regression risk ρ is the probability of losing a reference success under a declared paired protocol.

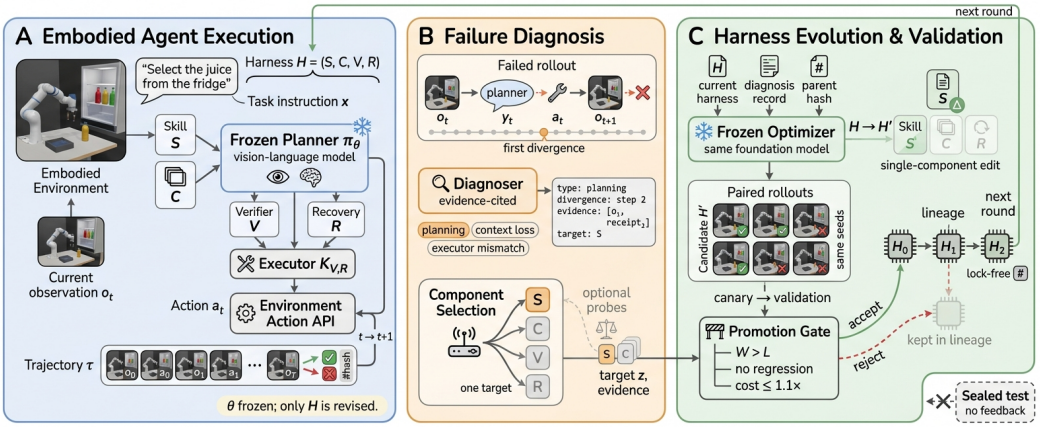


Figure 1: Overview of EmbodiedHarness. (A) A frozen vision-language planner interacts with the environment through a modular harness $H = (S, C, V, R)$. (B) Trajectory evidence supports failure diagnosis and component selection via anchored counterfactual replay. (C) A bounded revision undergoes paired canary and validation rollouts before promotion and persistent reuse. Illustrated outcomes are schematic.

3.2 Failure Diagnosis

A binary outcome cannot say what to change, so every turn is recorded: instruction, observation references, selected context, planner output, executed action, receipt, next observation, termination and resource usage, each tagged with the harness hash and episode identifier. The optimizer sees permitted observations and development feedback only.

From a failed trajectory τ with $y(\tau) = 0$, the diagnoser produces a structured record

$$d(\tau) = \big(f,\ t^\star,\ E,\ \kappa,\ z\big), \tag{4}$$

with f a failure type from a fixed taxonomy (evidence acquisition, context retention, planning, planner-executor mismatch, verification, recovery), $E \subseteq \tau$ evidence references that must resolve to existing trajectory entries, $\kappa \in [0, 1]$ the diagnoser's confidence, and $z \in \mathcal{Z} = \{S, C, V, R\}$ the proposed target. The deviation anchor $t^\star$ is the earliest step at which recorded evidence shows the executed trajectory departing from what the task requires:

$$t^\star = \min\{t : \exists\, e \in \mathcal{E}(h_t) \text{ s.t. } e \text{ shows } \tau_{0:t} \text{ departing from the task requirement}\}. \tag{5}$$

Two lags separate this index from the decision that caused the failure. A grasp slipping at t_{phys} becomes visible only at $t_{\text{obs}} > t_{\text{phys}}$, when the arm moves away and the object stays behind; if the receipt chain identifies the dispatch step that produced the slip, that step becomes the anchor, and otherwise the anchor stays at t_{obs} and the replay measures only how far a candidate can still repair the episode. A semantic error, grasping an orange where the instruction required an apple, is the other case: every receipt reports success and the evaluator disagrees only at T. The anchor is then the dispatch step whose own record already names the wrong target, which is observable even though nothing later contradicts it; a trace with no such step is excluded from replay rather than anchored at T. A diagnosis that cites no observations stays uncertain and cannot drive a confident attribution. A receipt-grounded rule diagnoser reads the same evidence at no model cost, and Appendix E reports the agreement matrix between the two.

3.3 Component Selection via Anchored Counterfactual Replay

Component selection allocates the next revision to one artifact. The default policy follows the diagnosis and its confidence and abstains when evidence is weak; fixed rules, random selection and sequential optimization are the alternatives we compare against.

The intervention variant, *anchored counterfactual replay*, tests competing targets instead of trusting the diagnosis. For each of at most two plausible components the optimizer writes one minimal

candidate from the same evidence, differing from the parent in that component alone. The simulator state $s_{t^\star-1}$ and the recorded evidence log $h_{t^\star-1}$ are restored, and candidate and parent complete the remaining $T - t^\star$ turns, with $(s_{-1}, h_{-1}) \equiv (s_0, h_0)$ when the anchor falls on the first turn. What is restored is the unpruned log rather than the parent's selected context, so each candidate's own selector re-filters $h_{t^\star-1}$ and a context revision can act on prefix evidence the parent had dropped. Formally, the suffix trajectory under candidate $H' = H \oplus \Delta_z$ is drawn from

$$\tau^{H'}_{t^\star:T} \sim P^{H'}_\theta\big(\cdot \mid s_{t^\star-1}, h_{t^\star-1}\big), \tag{6}$$

while the parent suffix is drawn from $P^H_\theta(\cdot \mid s_{t^\star-1}, h_{t^\star-1})$ with the same restored state and seed. Because both share the prefix $(s_{0:t^\star-1}, h_{0:t^\star-1})$, the paired suffix-return difference

$$\widehat{g}_z = \frac{1}{|P|} \sum_{(i,\rho)\in P} [y_{i\rho}(H \oplus \Delta_z) - y_{i\rho}(H)] \tag{7}$$

over the replicate pairs (i, ρ) is insulated from the variance before the anchor, unlike a full-rollout comparison where both runs start from $s_0 \sim \mu_x$, and each replay costs a fraction $(T - t^\star)/T$ of an episode. When state restoration is unavailable, evaluation restarts from the initial condition; recorded actions are never replayed as observations under an altered policy.

The policy takes the target with the best observed gain subject to cost and regression constraints, breaking ties by lower cost and smaller edit. Replay rollouts are charged to EmbodiedHarness, competing methods get the same total budget for further proposals, and Table 4 checks whether the replays pay for themselves.

3.4 Harness revision

Once a target z is selected, the optimizer rewrites that artifact. A valid revision must satisfy the single-component constraint

$$H' = H \oplus \Delta_z, \qquad H'_{-z} = H_{-z}, \qquad d_z(H'_z, H_z) \le b_z, \tag{8}$$

where H_{-z} is every component except z and d_z is a declared schema or content-size bound, at most 40 changed lines and 2,000 characters for every component type (Table 5). Parent hashes enforce the equality of unchanged components and bind a candidate to its evaluated parent, so a candidate cannot be applied to a harness it was not evaluated against.

The optimizer receives the parent artifact, a task-independent failure summary, cited evidence and the schema, and returns a replacement with a rationale and evidence references. No revision may reach privileged state or the benchmark success function, and every added observation, verifier call and recovery action counts toward the episode limits baselines also face; Appendix A gives the contracts. Because components interact, edit-all enters as a control (Table 4).

3.5 Validation and promotion

A revision that repairs the trajectory that motivated it has not yet earned a place in the harness. Static checks reject malformed schemas, unsupported fields, stale parents and out-of-type edits; canary episodes then catch obvious regressions before the larger validation run, which compares candidate and parent on identical episode identifiers and seeds.

For n matched episodes, write y_i, y'_i for parent and candidate outcomes and define

$$W = \sum_i (1 - y_i) y'_i, \qquad L = \sum_i y_i (1 - y'_i), \qquad \widehat{\Delta} = (W - L)/n. \tag{9}$$

$$\mathrm{accept}(H', H) = \mathbf{1}\{\widehat{\Delta} > 0,\ L \le \ell,\ N_{\mathrm{cat}} = 0,\ C' \le \alpha C_0,\ C' \le B_{\mathrm{infer}}\}, \tag{10}$$

with defaults $\ell = 0$ and $\alpha = 1.1$. Note that $\widehat{\Delta} > 0$ is equivalent to $W > L$. A pair counts toward N_{cat} when the candidate raises an event the parent did not raise on that episode and that leaves it unscorable or the scene unusable: a simulator or worker crash, an action rejected as out of bounds or beyond a joint limit, an object driven off the workspace, an unhandled tool exception. An ordinary failure is not catastrophic. The cost check uses the root harness $C_0 = C(H_0)$ rather than the parent, so K accepted revisions cannot compound to α^K, and the absolute bound B_{infer} is enforced in the same test.

Property 1 (observed paired monotonicity). For complete paired binary outcomes and the gate in Equation 10 with $\ell = 0$, acceptance implies $\widehat{J}(H') > \widehat{J}(H)$ and $y_i = 1 \Rightarrow y'_i = 1$ for every evaluated pair. It is a statement about the evaluated pairs and the immediate parent; Appendix B gives the accounting across rounds. The derivation is immediate: each pair contributes $+1$, -1 or 0 to the success-count difference, so $n(\widehat{J}(H') - \widehat{J}(H)) = W - L$, and acceptance with $\ell = 0$ forces $L = 0$ and $W > L$.

Algorithm 1 in Appendix B states the loop, including the abstention and static-check branches that leave the harness unchanged and the guard that spends validation budget only when a replay showed a positive gain. Accepted candidates become immutable versions; rejected ones stay in the lineage with their evidence, and rollback restores an earlier version without deleting its successor. Final-test feedback cannot trigger a patch: the bundle is evaluated only after code and configuration freeze.

4 EXPERIMENTS

We evaluate EmbodiedHarness on VLABench (reasoning-intensive manipulation), EmbodiedBench (visual interaction and navigation) and ESI-Bench (active spatial exploration), asking whether harness optimization alone improves on the frozen agent, whether it beats a preset evolution direction, and which mechanism carries the gain.

4.1 EXPERIMENTAL SETUP

Benchmarks and metrics. **VLABench** (Zhang et al., 2024b) evaluates reasoning-intensive manipulation across five capability categories: in-distribution generalization (InD), cross-category transfer (XCat), commonsense reasoning (CS), instruction following (Inst) and texture recognition (Tex). The planner acts through six skill primitives (pick, place, press, open, lift, pull), and every command is realised by π_0, the executor the published train-free results also use. **EmbodiedBench** (Yang et al., 2025) evaluates multi-modal models as embodied agents; we test EB-Habitat and EB-Navigation, 300 tasks each. **ESI-Bench** (Hong et al., 2026) evaluates active spatial exploration across ten categories. Test identifiers are disjoint from optimization and validation, the test manifest is sealed before the harness is frozen, and the metric is task success rate throughout.

Models. Planner and optimizer both use Qwen3.8-27B Max by default, written Qwen3.8-27B in the tables; on VLABench we additionally run Qwen3.6-27B and on EmbodiedBench Qwen3-VL-32B, matching the parameter count of the competing open-source frameworks. Parameters stay fixed throughout, and every method in a comparison runs on the same backbone through our own harness.

Baselines. We compare against three groups. *VLA-level*: π_0 and π_0-FAST act end to end without an upper-level planner (Black et al., 2024). *Test-time scaling*: VLA+VOTE, Seed Agent+MG-Select and Seed Agent+VOTE aggregate several inference calls under a matched deployment budget (Lin et al., 2025; Jang et al., 2025). *Train-free harness adaptation*: Seed Agent, Skill Evolution and SHAPER on VLABench, all reported by Wang et al. (2026b), and Practice (Bai et al., 2026), EmbodiSkill (Ju et al., 2026) and MemCompiler (Ding et al., 2026b) on EmbodiedBench, the latter three on Qwen3-VL-32B. External numbers are reproduced as published.

Evolution protocol. Each evolving method runs four rounds. A round samples four episodes from the optimization pool, diagnoses the failures among them, and then spends further rollouts on anchored replay, the canary and the paired validation batch; paired methods share initializations and environment seeds, and the gate uses $\ell = 0$ and $\alpha = 1.1$. The selected harness is then frozen and evaluated on the sealed test manifest (Appendix C).

4.2 MAIN RESULTS

VLABench. Table 1 reports sealed-test results on VLABench. EmbodiedHarness with Qwen3.8-27B achieves the best average success rate of 48.9%, outperforming the frozen agent (34.8%) by +14.1 points.

Table 1: VLABench sealed-test success rate (%) by capability category; best **bold** and second best underlined among our methods. *Exec.* is the low-level executor; every row acts through π_0, so backbone and execution substrate are matched across the table. †: these methods publish C1–C4 only, so their C1–C5 average is undefined and every comparison with them is read from the C1–C4 column.

Method	Backbone	Exec.	Success (%)					Avg	
			InD	XCat	CS	Inst	Tex	C1–C4	C1–C5
VLA-level baselines									
π_0	PaliGemma	π_0	47.0	21.2	29.1	17.3	32.2	28.7	29.4
π_0-FAST	PaliGemma+FAST	π_0	56.2	31.0	38.0	35.0	39.0	40.1	39.8
Test-time scaling†									
VLA + VOTE	π_0	π_0	26.0	12.5	33.0	16.5	–	22.0	–
Seed Agt + MG-Select	Qwen3.6-27B	π_0	35.0	15.0	40.5	7.0	–	24.4	–
Seed Agt + VOTE	Qwen3.6-27B	π_0	38.0	13.0	34.0	9.0	–	23.5	–
Train-free harness adaptation†									
Seed Agent	Qwen3.6-27B	π_0	40.0	20.0	40.0	13.0	–	28.3	–
Skill Evolution	Qwen3.6-27B	π_0	42.0	23.5	50.0	18.5	–	33.5	–
SHAPER	Qwen3.6-27B	π_0	42.5	26.0	50.0	19.5	–	34.5	–
Ours (same executor and, on Qwen3.6-27B, same backbone as the block above)									
Frozen agent	Qwen3.6-27B	π_0	47.0	19.2	26.7	29.4	31.1	30.6	30.7
Frozen agent	Qwen3.8-27B	π_0	50.6	23.4	33.5	33.0	33.6	35.1	34.8
EmbodiedHarness (probes)	Qwen3.6-27B	π_0	61.0	38.3	40.3	39.3	43.9	44.7	44.6
EmbodiedHarness (probes)	Qwen3.8-27B	π_0	**62.9**	**43.2**	**48.3**	**42.9**	**47.2**	**49.3**	**48.9**

Table 2: EmbodiedBench test success rate (%). B: Base; C: Common; Cx: Complex; V: Visual; S: Spatial; L: Long-horizon. Best open-source result **bold**, second best underlined; closed-source rows are context and are not ranked. All four train-free frameworks share the Qwen3-VL-32B backbone. ‡: reproduced as published, where the average is task-weighted over the 300 tasks and so differs slightly from the mean of the subset scores.

Method	Backbone	EB-Habitat (%)							EB-Navigation (%)					
		Avg	B	C	Cx	V	S	L	Avg	B	C	Cx	V	L
Zero-shot closed-source														
GPT-4o	GPT-4o	59.0	86	44	56	68	36	64	57.7	55.0	60.0	58.3	60.0	55.0
Claude-3.5	Claude-3.5	68.0	96	68	78	70	38	58	44.7	66.7	51.7	41.7	36.7	26.7
Zero-shot open-source														
InternVL3-78B	InternVL3	55.0	84	58	60	56	32	40	53.7	66.7	63.3	61.7	45.0	31.7
Qwen2.5-VL-72B	Qwen2.5	35.7	70	30	36	32	28	18	40.0	46.7	46.7	46.7	26.7	33.3
Open-source $\leq$32B + train-free frameworks														
Practice	Qwen3-VL-32B	58.3	82	**68**	60	**76**	42	22	–	–	–	–	–	–
EmbodiSkill	Qwen3-VL-32B	52.3	**96**	44	52	62	36	24	61.3‡	67.0	**70.0**	70.0	**68.0**	32.0
MemCompiler	Qwen3-VL-32B	55.7	90	58	44	64	40	38	–	–	–	–	–	–
Frozen agent (H_0)	Qwen3-VL-32B	45.0	78	40	44	52	30	26	49.3	55.0	53.3	51.7	48.3	38.3
EmbodiedHarness	Qwen3-VL-32B	**62.0**	92	56	**64**	72	**46**	**42**	**65.3**	**76.7**	68.3	**76.7**	63.3	**41.7**

On the four capability categories that external train-free methods publish, EmbodiedHarness reaches 44.7% under the Qwen3.6-27B backbone SHAPER uses, against SHAPER's 34.5%, a gain of +10.2 points; with Qwen3.8-27B the same subset reaches 49.3%. Every category improves (Figure 2a), but not evenly: cross-category transfer gains most, from 23.4% to 43.2%, and it is the category on which the frozen agent is weakest, while instruction following gains least, from 33.0% to 42.9%.

EmbodiedBench. Table 2 reports EB-Habitat and EB-Navigation. On Habitat, EmbodiedHarness with Qwen3-VL-32B reaches 62.0%, ahead of every open-source train-free method, Practice (58.3%), MemCompiler (55.7%) and EmbodiSkill (52.3%), and of GPT-4o as a direct agent

Table 3: ESI-Bench test accuracy (%) by category; micro averages over questions, macro over categories. The seed agent and SHAPER are published results on Qwen3.6-27B, which differs from our default but matches our own Qwen3.6-27B row, the paired comparison.

Method	PhyS	PhyD	Refl	Grnd	Metr	Enum	SpRel	CogM	Temp	ActS	Micro	Macro
n	21	10	20	49	21	18	51	17	19	5	231	10
Non-adaptive agents												
Frozen agent (Qwen3.6-27B)	23.8	30.0	15.0	59.2	33.3	27.8	33.3	17.6	52.6	40.0	36.4	33.3
Frozen agent (Qwen3.8-27B)	28.6	40.0	20.0	55.1	38.1	33.3	37.3	23.5	57.9	60.0	39.8	39.4
Published reference results (Seed agent and SHAPER use Qwen3.6-27B, matching our Qwen3.6-27B row)												
Seed agent (Qwen3.6-27B)	*14.3*	*40.0*	*10.0*	*53.1*	*33.3*	*16.7*	*27.5*	*29.4*	*47.4*	*40.0*	*32.5*	*31.2*
SHAPER (Qwen3.6-27B)	*38.1*	*40.0*	*60.0*	*69.4*	*47.6*	*27.8*	*54.9*	*23.5*	*47.4*	*20.0*	*49.8*	*42.9*
GPT-5 (active)	*51.4*	*46.1*	*53.0*	*24.9*	*48.2*	*10.7*	*31.0*	*60.4*	*40.6*	*36.4*	–	*40.3*
Our method												
EmbodiedHarness (Qwen3.6-27B)	52.4	50.0	50.0	77.6	57.1	33.3	54.9	35.3	52.6	60.0	55.8	52.3
EmbodiedHarness (Qwen3.8-27B)	47.6	60.0	65.0	77.6	61.9	44.4	66.7	47.1	63.2	60.0	**62.8**	**59.4**

(59.0%); on Navigation it reaches 65.3%, ahead of EmbodiSkill (61.3%) and every listed closed-source agent including GPT-4o (57.7%). The unadapted agent on the same model scores 45.0% and 49.3%, so the gain is read against the same backbone.

The per-subset breakdown places the improvement in Complex and Long-horizon, at +7 and +10 points over EmbodiSkill on EB-Navigation and +12 and +18 on EB-Habitat. Against the unadapted agent the same pattern holds: Complex instruction rises from 44 to 64 on Habitat and from 51.7% to 76.7% on Navigation. Long-horizon navigation is the one subset that barely moves, from 38.3% to 41.7%, because those episodes terminate on the step budget rather than on a recoverable error.

ESI-Bench. Table 3 reports ESI-Bench. EmbodiedHarness with Qwen3.8-27B reaches 62.8% micro and 59.4% macro accuracy, above SHAPER (49.8% micro, 42.9% macro) by +13.0 and +16.5 points and above GPT-5 as an active agent (40.3% macro) by +19.1. Gains concentrate in categories that reward deliberate re-observation: specular reflection (+45.0 points over the frozen agent), spatial relations (+29.4) and cognitive mapping (+23.6), where context and verifier revisions take effect. With the same Qwen3.6-27B backbone SHAPER uses, EmbodiedHarness still reaches 55.8% micro and 52.3% macro, exceeding SHAPER by +6.0 and +9.4 points, confirming that the gain comes from harness optimization rather than a stronger backbone. The two categories that barely move are the two the frozen agent already handles best, active search at 60.0% and temporal reasoning at 57.9%, and both are settled by a single view, so there is little for a re-observation policy to add.

4.3 Ablation study

Table 4 removes one mechanism at a time on VLABench under a fixed optimization schedule, so rows differ in what they do with a round rather than in how much budget they receive. Choosing the target matters more than editing broadly: replacing the diagnosis by a random target is the largest single loss at 15.0 points, while edit-all, which patches every component under the same content budget, loses 8.7, so editing more of the harness is not a substitute for editing the right part of it. Testing that choice matters in turn. Keeping the diagnosis but removing the repair probes reaches 50.6%, so trying a direction before committing to it is worth +3.9 points beyond what the diagnosis alone recovers, and restarting the probes from the initial state rather than the recorded anchor costs a further 3.2 points, since each replay then re-executes a prefix both sides already agree on.

No single artifact carries the result. Freezing one family at a time costs 6.8 points for skills, 5.0 for verifier and recovery and 3.1 for context. Skills matter most, which matches the diagnoser's own distribution of attributions (Appendix E), but removing any one family still leaves most of the gain, so the four artifacts contribute jointly rather than through one dominant channel. What decides whether those edits accumulate is the gate. Targeting raises promoted revisions per run from 1.3 under random selection to 3.5, and dropping the gate raises the count further still, to 4.2, while raising the fraction of sealed-test episodes that H_0 solved and the final harness does not from 1.6% to 9.2%. Promotions are not the objective; validated promotions are, and the rows read together

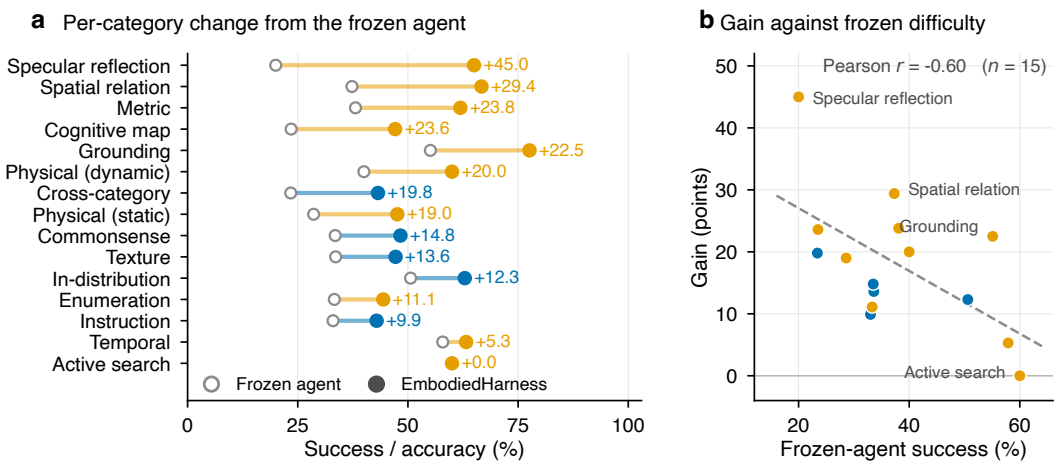


Figure 2: Per-category improvement over the frozen agent (Qwen3.8-27B) on VLABench (blue) and ESI-Bench (orange); open markers are the frozen agent. (a) Every category improves, and the ranking follows how much re-observation it demands. (b) Gain against frozen-agent success: $r = -0.60$ raw, -0.34 and -0.22 once the bounded scale and the baseline-difference coupling are corrected for (Appendix D).

Table 4: Ablation on VLABench average success, averaged over task families, which weights each family equally and differs from the capability-category average of Table 1. The optimization schedule is held fixed across rows (Appendix E). *Regr.* is the fraction of sealed-test episodes that H_0 solved and the final harness does not; *Rounds* and *Prom./run* are the average rounds executed and revisions promoted per run.

Variant	Avg (%)	Δ	Regr. (%)	Rounds	Prom./run
EmbodiedHarness (full)	54.5	–	1.6	4.0	3.5
− state anchoring (probes restart from reset)	51.3	−3.2	2.4	4.0	2.8
− repair probes (diagnosis only)	50.6	−3.9	2.1	4.0	2.8
− diagnosis (random target)	39.5	−15.0	6.2	4.0	1.3
− validation gate (accept every proposal)	47.3	−7.2	9.2	4.6	4.2
− skill evolution	47.7	−6.8	2.9	4.0	2.9
− context evolution	51.4	−3.1	2.2	4.0	3.2
− verifier/recovery evolution	49.5	−5.0	2.7	4.0	3.0
edit-all (one patch per component)	45.8	−8.7	5.4	4.0	1.7

say that the gain comes from making each update targeted, tested and validated rather than from updating the harness more often.

5 Conclusion

We introduced EmbodiedHarness, a non-parametric framework in which a frozen model evolves the harness around it through a diagnose-revise-validate loop. Anchored counterfactual replay tests each candidate direction from a restored deviation anchor, so budget reaches a repair only after that repair has shown it works. Paired validation then admits the winner only when it regresses nothing in the batch that compares it against its parent. Across VLABench, EmbodiedBench and ESI-Bench this beats every comparable open-source train-free method, raising VLABench success by +14.1 points over the same frozen agent. The anchor is the part that travels. Wherever an environment can be restored to the step at which a trajectory first left the task, the same suffix comparison ranks repairs without a reward model, and that is where we expect harness optimization to generalize. The method touches only text and code, so an evolved harness can be archived, inspected and carried to a new backbone for a few hundred rollouts rather than a training run.

AI USE STATEMENT

Large language models were used to assist with code development, literature retrieval, and polishing of the manuscript. All results, analyses, and claims were verified by the authors.

ETHICS STATEMENT

This work studies simulated embodied agents. It involves no human subjects, no personally identifying data, and no release of a new dataset. All experiments run inside published benchmark simulators under their own licences, and the patches we apply to upstream benchmark code are released with the supplementary material. The optimizer is denied access to the benchmark success function and to privileged simulator state, so an evolved artifact cannot encode evaluation labels, and the sealed test manifest is fixed before any harness is frozen. The harness artifacts we release contain rules expressed over public receipts and observations rather than scene-specific answers. Because the method adapts agent systems that act in physical environments, the promotion gate rejects any revision that raises a new catastrophic event on a validation episode, which is how safety-relevant changes are screened before an artifact is retained.

REPRODUCIBILITY STATEMENT

The code will be released when the paper is accepted. Appendix C lists the shared configuration and Appendix A the prompts and artifact contracts. Credentials and third-party simulator assets are excluded; the patches applied to upstream benchmark code are included.

REFERENCES

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective Prompt Evolution Can Outperform Reinforcement Learning, 2025. URL https://arxiv.org/abs/2507.19457.

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Mengyuan Yan, and Andy Zeng. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances, 2022. URL https://arxiv.org/abs/2204.01691.

Ziyi Bai, Siqi Li, Tinglei Huang, and Börje F. Karlsson. PRACTICE: From Experience to Expertise in Self-Evolving Embodied Agents, 2026. URL https://arxiv.org/abs/2608.30760.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A Vision-Language-Action Flow Model for General Robot Control, 2024. URL https://arxiv.org/abs/2410.24164.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building Production-Ready AI Agents with Scalable Long-Term Memory, 2025. URL https://arxiv.org/abs/2504.19413.

Yalun Dai, Hao Li, Shulin Tian, Runmao Yao, Yuhao Dong, Fangzhou Hong, Zhaoxi Chen, Fangfu Liu, Tao Wang, Kim-Hui Yap, and Ziwei Liu. S-Agent: Spatial Tool-Use Elicits Reasoning for Spatial Intelligence, 2026. URL https://arxiv.org/abs/2606.20515.

Xin Ding, Liang Mi, Mingzhe Huang, Zixuan Wang, Chao Zhang, Zixu Hao, Fu Chen, Xiangyu Li, Yikai Zheng, Yaoyu Guo, Weijun Wang, Kun Li, Hao Wu, Yunxin Liu, and Ting Cao. Zetta ζ: An Efficient Closed-Loop Embodied Harness for Self-Evolving Physical Intelligence, 2026a. URL https://arxiv.org/abs/2608.16590.

Xin Ding, Xinrui Wang, Yifan Yang, Hao Wu, Shiqi Jiang, Qianxi Zhang, Liang Mi, Hanxin Zhu, Kun Li, Yunxin Liu, Zhibo Chen, and Ting Cao. MemCompiler: Compile, Don't Inject – State-Conditioned Memory for Embodied Agents, 2026b. URL https://arxiv.org/abs/2605.07594.

Haowen Gao, Haoran Chen, Can Wang, Shasha Guo, Liang Pang, Zhaoyang Liu, Huawei Shen, and Xueqi Cheng. Skillaudit: Ground-truth-free skill evolution via paired trajectory auditing, 2026. URL https://arxiv.org/abs/2606.14239.

Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, Steven Bohez, Konstantinos Bousmalis, Anthony Brohan, Thomas Buschmann, Arunkumar Byravan, Serkan Cabi, Ken Caluwaerts, Federico Casarini, Oscar Chang, Jose Enrique Chen, Xi Chen, Hao-Tien Lewis Chiang, Krzysztof Choromanski, David D'Ambrosio, Sudeep Dasari, Todor Davchev, Coline Devin, Norman Di Palo, Tianli Ding, Adil Dostmohamed, Danny Driess, Yilun Du, Debidatta Dwibedi, Michael Elabd, Claudio Fantacci, Cody Fong, Erik Frey, Chuyuan Fu, Marissa Giustina, Keerthana Gopalakrishnan, Laura Graesser, Leonard Hasenclever, Nicolas Heess, Brandon Hernaez, Alexander Herzog, R. Alex Hofer, Jan Humplik, Atil Iscen, Mithun George Jacob, Deepali Jain, Ryan Julian, Dmitry Kalashnikov, M. Emre Karagozler, Stefani Karp, Chase Kew, Jerad Kirkland, Sean Kirmani, Yuheng Kuang, Thomas Lampe, Antoine Laurens, Isabel Leal, Alex X. Lee, Tsang-Wei Edward Lee, Jacky Liang, Yixin Lin, Sharath Maddineni, Anirudha Majumdar, Assaf Hurwitz Michaely, Robert Moreno, Michael Neunert, Francesco Nori, Carolina Parada, Emilio Parisotto, Peter Pastor, Acorn Pooley, Kanishka Rao, Krista Reymann, Dorsa Sadigh, Stefano Saliceti, Pannag Sanketi, Pierre Sermanet, Dhruv Shah, Mohit Sharma, Kathryn Shea, Charles Shu, Vikas Sindhwani, Sumeet Singh, Radu Soricut, Jost Tobias Springenberg, Rachel Sterneck, Razvan Surdulescu, Jie Tan, Jonathan Tompson, Vincent Vanhoucke, Jake Varley, Grace Vesom, Giulia Vezzani, Oriol Vinyals, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Fei Xia, Ted Xiao, Annie Xie, Jinyu Xie, Peng Xu, Sichun Xu, Ying Xu, Zhuo Xu, Yuxiang Yang, Rui Yao, Sergey Yaroshenko, Wenhao Yu, Wentao Yuan, Jingwei Zhang, Tingnan Zhang, Allan Zhou, and Yuxiang Zhou. Gemini Robotics: Bringing AI into the Physical World, 2025. URL https://arxiv.org/abs/2503.20020.

Yining Hong, Jiageng Liu, Han Yin, Manling Li, Leonidas Guibas, Li Fei-Fei, Jiajun Wu, and Yejin Choi. ESI-Bench: Towards Embodied Spatial Intelligence that Closes the Perception-Action Loop, 2026. URL https://arxiv.org/abs/2605.18746.

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, 2025. URL https://arxiv.org/abs/2504.16054.

Suhyeok Jang, Dongyoung Kim, Changyeon Kim, Youngsuk Kim, and Jinwoo Shin. Verifier-free Test-Time Sampling for Vision-Language-Action Models, 2025. URL https://arxiv.org/abs/2510.05681.

Dong Jing, Gang Wang, Jiaqi Liu, Weiliang Tang, Zelong Sun, Yunchao Yao, Zhenyu Wei, Yunhui Liu, Zhiwu Lu, and Mingyu Ding. Mixture of Horizons in Action Chunking, 2025. URL https://arxiv.org/abs/2511.19433. Accepted at ICML 2026.

Dong Jing, Jingchen Nie, Tianqi Zhang, Jiaqi Liu, Huaxiu Yao, Zhiwu Lu, and Mingyu Ding. TempoVLA: Learning Speed-Controllable Vision-Language-Action Policies, 2026a. URL https://arxiv.org/abs/2606.06491.

Dong Jing, Tianqi Zhang, Jiaqi Liu, Jinman Zhao, Zelong Sun, Li Erran Li, Zhiwu Lu, and Mingyu Ding. Learning Action Priors for Cross-embodiment Robot Manipulation, 2026b. URL https://arxiv.org/abs/2606.26095.

Ruofei Ju, Xinrui Wang, Xin Ding, Yifan Yang, Hao Wu, Shiqi Jiang, Qianxi Zhang, Hao Wen, Xiangyu Li, Weijun Wang, Kun Li, Yunxin Liu, Haipeng Dai, Wei Wang, and Ting Cao. Embodiskill: Skill-aware reflection for self-evolving embodied agents, 2026. URL https://arxiv.org/abs/2605.10332.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. DSPy: Compiling Declarative Language Model Calls into Self-Improving Pipelines, 2023. URL https://arxiv.org/abs/2310.03714.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An Open-Source Vision-Language-Action Model, 2024. URL https://arxiv.org/abs/2406.09246.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, Xiaofan Wang, Bei Liu, Jianlong Fu, Jianmin Bao, Dong Chen, Yuanchun Shi, Jiaolong Yang, and Baining Guo. CogACT: A Foundational Vision-Language-Action Model for Synergizing Cognition and Action in Robotic Manipulation, 2024. URL https://arxiv.org/abs/2411.19650.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as Policies: Language Model Programs for Embodied Control, 2022. URL https://arxiv.org/abs/2209.07753.

Jiahang Lin, Shichun Liu, Chengjun Pan, Lizhi Lin, Shihan Dou, Zhiheng Xi, Xuanjing Huang, Hang Yan, Zhenhua Han, Tao Gui, and Yu-Gang Jiang. Agentic harness engineering: Observability-driven automatic evolution of coding-agent harnesses, 2026a. URL https://arxiv.org/abs/2604.25850.

Juyi Lin, Amir Taherin, Arash Akbari, Arman Akbari, Lei Lu, Guangyu Chen, Taskin Padir, Xiaomeng Yang, Weiwei Chen, Yiqian Li, Xue Lin, David Kaeli, Pu Zhao, and Yanzhi Wang. VOTE: Vision-Language-Action Optimization with Trajectory Ensemble Voting, 2025. URL https://arxiv.org/abs/2507.05116.

Lucy Lin, Ayush Jain, Yifan Liu, and Katerina Fragkiadaki. Qwen-3D: A Generalist 3D Vision-Language Model for Spatial Understanding, 2026b. URL https://arxiv.org/abs/2608.02980.

Jiaqi Liu, Shi Qiu, Mairui Li, Bingzhou Li, Haonian Ji, Siwei Han, Xinyu Ye, Peng Xia, Zihan Dong, Meng Chen, Congyu Zhang, Letian Zhang, Guiming Chen, Haoqin Tu, Xinyu Yang, Lu Feng, Xujiang Zhao, Haifeng Chen, Jiawei Zhou, Xiao Wang, Weitong Zhang, Hongtu Zhu, Yun Li, Jieru Mei, Hongliang Fei, Jiaheng Zhang, Linjie Li, Linjun Zhang, Yuyin Zhou, Sheng Wang, Caiming Xiong, James Zou, Zeyu Zheng, Cihang Xie, Mingyu Ding, and Huaxiu Yao. AutoResearchClaw: Self-Reinforcing Autonomous Research with Human-AI Collaboration, 2026a. URL https://arxiv.org/abs/2605.20025.

Jiaqi Liu, Yaofeng Su, Peng Xia, Siwei Han, Zeyu Zheng, Cihang Xie, Mingyu Ding, and Huaxiu Yao. SimpleMem: Efficient Lifelong Memory for LLM Agents, 2026b. URL https://arxiv.org/abs/2601.02553.

NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng,

and Yuke Zhu. GR00T N1: An Open Foundation Model for Generalist Humanoid Robots, 2025. URL https://arxiv.org/abs/2503.14734.

Gabriel Sarch, Lawrence Jang, Michael J. Tarr, William W. Cohen, Kenneth Marino, and Katerina Fragkiadaki. VLM Agents Generate Their Own Memories: Distilling Experience into Embodied Programs of Thought, 2024. URL https://arxiv.org/abs/2406.14596.

Jaineet Shah. Causal agent replay: Counterfactual attribution for llm-agent failures, 2026. URL https://arxiv.org/abs/2606.08275.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language Agents with Verbal Reinforcement Learning, 2023. URL https://arxiv.org/abs/2303.11366.

Mirac Suzgun, Mert Yuksekgonul, Federico Bianchi, Dan Jurafsky, and James Zou. Dynamic Cheatsheet: Test-Time Learning with Adaptive Memory, 2025. URL https://arxiv.org/abs/2504.07952.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An Open-Ended Embodied Agent with Large Language Models, 2023. URL https://arxiv.org/abs/2305.16291.

Pan Wang, Yihao Hu, Xiujin Liu, Jingchu Yang, Hang Wang, and Zhihao Wen. AtlasVA: Self-Evolving Visual Skill Memory for Teacher-Free VLM Agents, 2026a. URL https://arxiv.org/abs/2605.17933.

Peidong Wang, Zhiming Ma, Ying Chang, Xufang Luo, Yiqun Zhang, Zihan Wang, Xiaocui Yang, Shi Feng, Yuqing Yang, and Dongsheng Li. Self-Evolving Embodied Agents via Skill-Harness Evolution, 2026b. URL https://arxiv.org/abs/2608.11350.

Wei Wang, Wenqiao Zhang, Yutong Lin, Yuqian Yuan, Tianwei Lin, Jinhao Mao, Zhenxuan Fan, Mingjian Gao, Yang Dai, Wentong Li, Zheqi Lv, Zheng Dong, Yingjie Niu, Jiaqi Zhu, Jun Xiao, Chao Li, and Yueting Zhuang. EmbodiedSkills: A Unified Framework for Orchestrating, Training, and Deploying VLA Agents, 2026c. URL https://arxiv.org/abs/2609.01281.

Xiaoxuan Wang, Haixin Wang, Alexander Taylor, Jason Cong, Yizhou Sun, and Wei Wang. Harnessbridge: Learnable bidirectional controller for llm agent harness, 2026d. URL https://arxiv.org/abs/2606.12882.

Tianxin Wei, Zhan Shi, Minhua Lin, Bing He, Zewen Liu, Yisi Sang, Yuanchen Bei, Xuying Ning, Jiaru Zou, Ting-Wei Li, Xiao Lin, Yanjun Zhao, Chi Wang, Benoit Dumoulin, Dakuo Wang, Jingrui He, and Hanqing Lu. Evo-harness: Context-to-harness skill compilation for self-evolving agents, 2026. URL https://arxiv.org/abs/2608.15071.

Aodi Wu, Haodong Han, Xubo Luo, Ruisuo Wang, Shan He, and Xue Wan. SpaceMind: A Modular and Self-Evolving Embodied Vision-Language Agent Framework for Autonomous On-orbit Servicing, 2026a. URL https://arxiv.org/abs/2604.14399.

Jing Wu, Jianhua Wu, Jiayi Guan, Jiahong Chen, Jinghui Lu, Hangjun Ye, Bingzhao Gao, and Long Chen. SpatioLM: Towards General Physical Spatial Intelligence in Vision-Language Models, 2026b. URL https://arxiv.org/abs/2608.01899.

Peng Xia, Jianwen Chen, Hanyang Wang, Jiaqi Liu, Kaide Zeng, Yu Wang, Siwei Han, Yiyang Zhou, Xujiang Zhao, Haifeng Chen, Zeyu Zheng, Cihang Xie, and Huaxiu Yao. SkillRL: Evolving Agents via Recursive Skill-Augmented Reinforcement Learning, 2026. URL https://arxiv.org/abs/2602.08234.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-MEM: Agentic Memory for LLM Agents, 2025. URL https://arxiv.org/abs/2502.12110.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Jinhe Bi, Kristian Kersting, Jeff Z. Pan, Hinrich Schütze, Volker Tresp, and Yunpu Ma. Memory-R1: Enhancing Large Language Model Agents to Manage and Utilize Memories via Reinforcement Learning, 2025. URL https://arxiv.org/abs/2508.19828.

Rui Yang, Hanyang Chen, Junyu Zhang, Mark Zhao, Cheng Qian, Kangrui Wang, Qineng Wang, Teja Venkat Koripella, Marziyeh Movahedi, Manling Li, Heng Ji, Huan Zhang, and Tong Zhang. EmbodiedBench: Comprehensive Benchmarking Multi-modal Large Language Models for Vision-Driven Embodied Agents, 2025. URL https://arxiv.org/abs/2502.09560.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models, 2022. URL https://arxiv.org/abs/2210.03629.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin Godel Machine: Open-Ended Evolution of Self-Improving Agents, 2025a. URL https://arxiv.org/abs/2505.22954.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. AFlow: Automating Agentic Workflow Generation, 2024a. URL https://arxiv.org/abs/2410.10762.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Zou, and Kunle Olukotun. Agentic Context Engineering: Evolving Contexts for Self-Improving Language Models, 2025b. URL https://arxiv.org/abs/2510.04618.

Shiduo Zhang, Zhe Xu, Peiju Liu, Xiaopeng Yu, Yuan Li, Qinghui Gao, Zhaoye Fei, Zhangyue Yin, Zuxuan Wu, Yu-Gang Jiang, and Xipeng Qiu. VLABench: A Large-Scale Benchmark for Language-Conditioned Robotics Manipulation with Long-Horizon Reasoning Tasks, 2024b. URL https://arxiv.org/abs/2412.18194.

Yang Zhou, Zixuan Huang, Sunzhu Li, Zhuo Yang, Chen Zhang, Shunian Chen, Caijun Yan, Jianyao Xu, Shunyu Liu, Weijie Fu, Peiliang Li, Xiaozhi Chen, and Yuxiang Cai. SpatialCLI: Learning to Reason With Spatial Tools, Then Without Them, 2026. URL https://arxiv.org/abs/2607.27703.

A PROMPTS AND ARTIFACT CONTRACTS

The runtime prompts are versioned with the protocol hash. Only permitted development evidence is inserted; the templates below illustrate the schemas used at each stage of the evolution pipeline.

A.1 DIAGNOSIS PROMPTS

Diagnoser System Prompt

```
Analyze this embodied development trajectory using only supplied
observations, selected context, actions, and public receipts.
Identify the earliest observable divergence from task requirements.
Distinguish acquisition, retention, planning, interface mismatch,
verification, and recovery.  Cite stable evidence identifiers.  State
alternatives and uncertainty.  Never invent hidden state or treat
model confidence as proof of success.  Recommend one repairable
artifact or abstain.  A recommendation is a hypothesis, not an
established cause.
```

Diagnoser Structured Output Schema

```
{"failure_type":  "context_retention",
"first_divergence_step":  7,
"evidence_refs":  ["observation_2", "context_7"],
"alternative_causes":  ["planning_error"],
"target":  "context", "confidence":  0.6}
```

Rule diagnoser. The receipt-grounded rule diagnoser reads the same evidence and applies five deterministic rules: a repeated failed request maps to recovery; completion without a subsequent observation to the verifier; a failed request without re-observation to context; a failed request followed by a replan to skill; and operational success with task failure to skill. It runs at zero model cost and provides the second attribution used in Appendix E.

A.2 REVISION PROMPTS

Optimizer System Prompt (all component types)

```
Revise only the requested artifact.  Parent hash, schema, and edit
budget are binding.  Derive a reusable rule from cited development
evidence.  Avoid memorized answers and scene-specific identifiers.
Return replacement, rationale, expected behavior change, and
rollback condition.  Do not change the evaluator, action privileges,
checkpoint, or other artifacts.  Return no proposal when evidence does
not support a repair.
```

Skill Revision User Template

```
Target artifact:  skill
Parent hash:  {parent_hash}
Parent content:  {parent_skill_json}

Failed episodes (development only):
{for each episode:  task instruction, trajectory summary,
diagnosis record (failure_type, divergence_step,
evidence_refs, confidence), selected observations}

Schema constraints:  {skill_schema}
Edit budget:  {max_diff_lines}

Return:  replacement JSON, rationale, expected change, rollback
condition.
```

Context Revision User Template

```
Target artifact: context
Parent hash: {parent_hash}
Parent content: {parent_context_json}

Failed episodes (development only):
{for each episode: retained vs. dropped evidence items,
context budget utilization, trajectory summary,
diagnosis record with evidence_refs}

Context budget: {B_ctx} images, {B_txt} text items
Schema constraints: {context_schema}

Return: replacement JSON, rationale, expected change, rollback
condition.
```

Verifier Revision User Template

```
Target artifact: verifier
Parent hash: {parent_hash}
Parent content: {parent_verifier_json}

Failed episodes with diagnosis_type ∈ {verification,
premature_completion}:
{for each: completion decision, subsequent observation
showing unmet requirement, evidence_refs}

Verifier contract: 1--3 review calls, 0--2 additional
observations, bounded visual criterion.

Return: replacement JSON, rationale, expected change, rollback
condition.
```

Recovery Revision User Template

```
Target artifact: recovery
Parent hash: {parent_hash}
Parent content: {parent_recovery_json}

Failed episodes with repeated failed actions:
{for each: repeated action sequence, controller receipts,
available alternative actions, step budget remaining}

Recovery contract: threshold 1--3 failures, 0--3
interventions (re-observe, switch approach, stop).

Return: replacement JSON, rationale, expected change, rollback
condition.
```

A.3 Artifact contracts

Verifier contract. The verifier consumes public runtime evidence and returns continue, request observation, recover, or permit completion. Permission is not benchmark success. Requests count toward episode limits and cannot create unbounded loops. A closed gripper alone does not prove object transport.

Context contract. The current observation must remain available. Selection obeys a common image/text budget and retains provenance. Historical evidence can be selected, but scoring-only metadata is inaccessible. Dropped-frame logs distinguish acquisition from retention failures. Additional visual novelty computation is costed and shared with relevant controls.

Executable artifact configuration. Verifier JSON configures an enable flag, one to three review calls, zero to two additional observations, and a bounded visual criterion. Recovery JSON configures an enable flag, a repeated-failure threshold of one to three, zero to three interventions, and a bounded response to repeated failure (re-observe, switch the grasp approach, or stop). Context JSON configures the number of retained main-camera turns, whether the initial view and current wrist view are included, and how many failed-receipt records are retained. Skill JSON configures a list of named rules, each with a trigger condition expressed over public receipts or observation fields, an ordered command template drawn from the six controllers, and an optional precondition on entity attributes; it carries no free-text field that the planner sees outside these rules. The initial common bundle disables the optional interventions; candidate evolution can change these fields.

Every component type shares one edit bound: a candidate may change at most 40 lines and 2,000 characters relative to its parent, which is the b_z of Equation 8 and the `max_diff_lines` slot of the revision templates above. The bound is passed to the optimizer in the prompt and re-checked statically after generation, so a candidate that exceeds it is discarded before any rollout is spent. The skill candidate of Figure 6 changes 1,074 characters and passes.

B Development loop

Algorithm 1 states the loop of Section 3 in full. Three branches leave the harness unchanged and each consumes its round: the round abstains when no component reaches the confidence threshold $\kappa_{\min}$, it stops when every proposal fails a static check, and it stops when no probed target returns a positive gain. Only the last of these is a policy choice rather than a guard, and it exists so that validation rollouts, the most expensive step in the round, are never spent on an edit the replay already found inert. The canary is a second early exit of the same kind: a candidate that regresses a canary pair is rejected without paying for the full validation batch.

C Experimental configuration

Optimization, canary, validation, and test identifiers are fixed before any run; the test manifest is sealed and the selected harness of every method is frozen before test evaluation. All methods share the configuration in Table 5.

D Statistical treatment of the per-category gain

Figure 2a ranks the per-category improvement over the frozen agent across VLABench and ESI-Bench. All 15 categories improve, the largest gains falling on specular reflection (+45.0), spatial relations (+29.4) and metric estimation (+23.8) and the smallest on categories a single view settles, active search (+0.0) and temporal reasoning (+5.3). Panel (b) requires two corrections before it supports an interpretation. Correlating a gain $Y - X$ with its own baseline X is not a neutral operation: the identity $\mathrm{Cov}(Y - X, X) = \mathrm{Cov}(Y, X) - \mathrm{Var}(X)$ drags the coefficient negative whenever the baseline varies at all, an artifact usually named after Oldham. A bounded percentage scale adds a second one, since a category sitting at 60.0% has at most 40.0 points of headroom while one at 20.0% has 80.0. We therefore report the raw coefficient $r = -0.60$ alongside two that remove them: normalising by available headroom, $\Delta_{\mathrm{norm}} = (Y - X)/(100 - X)$, gives -0.34, and correlating $Y - X$ against the average $(Y + X)/2$ rather than against X gives -0.22. Both stay negative under either correction, and both are about half the raw coefficient.

E Diagnoser agreement and ablation cost accounting

What Table 4 holds fixed across rows is the optimization schedule, namely the round cap, the number of failed traces sampled per round, the probe replicates and the validation batch. Two rows depart from it for a reason. Restarting probes from the initial state turns every replay into a full episode, so the same schedule costs more there than under anchored replay. Removing the gate runs no validation batch at all, so the reported row is the cost-matched variant that puts the freed budget into further rounds; it executes 4.6 rounds on average, which is why it records 4.2 promotions per run under a nominal four-round cap.

Algorithm 1 EmbodiedHarness development loop

Require: Frozen planner π_θ; initial harness $H_0 = (S, C, V, R)$; development splits $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{val}}, \mathcal{D}_{\text{can}}$ disjoint from the sealed test; acquisition budget B_{acq}; gate parameters (ℓ, α); probe replicates r; abstention threshold $\kappa_{\min}$

$k \leftarrow 0$; $c \leftarrow 0$; $\mathcal{L} \leftarrow \{H_0\}$ {immutable lineage; distinct from the scalar L of Eq. 9}
while $k < K$ **and** $c < B_{\text{acq}}$ **do**
  Sample $x_{1:m} \sim \mathcal{D}_{\text{train}}$; roll out $\tau_i \sim P_\theta^{H_k}(\cdot \mid x_i)$, recording anchors (s_t, h_t) at every turn t
  $\mathcal{F} \leftarrow \{(\tau_i, d_i) : y(\tau_i) = 0\}$, $d_i = \text{Diag}(\tau_i) = (f_i, t_i^\star, E_i, \kappa_i, z_i)$ with $E_i \subseteq \tau_i$ {Eq. 4}
  $w(z) \leftarrow \sum_i \kappa_i \mathbf{1}[z_i = z]$
  if $\max_z w(z) < \kappa_{\min}$ **then**
    $H_{k+1} \leftarrow H_k$; $c \leftarrow c + c_k$; $k \leftarrow k + 1$; **continue** {abstain: evidence too weak to name a target}
  end if
  $\mathcal{T} \leftarrow$ target of highest weight, plus the runner-up if $w > 0$, else the most cited `alternative_causes` entry, else the next component in the fixed order (S, C, V, R)
  for $z \in \mathcal{T}$ **do**
    $\Delta_z \leftarrow \text{Propose}(H_k, z, \mathcal{F})$; drop z from $\mathcal{T}$ if schema-invalid or $\text{parent}(\Delta_z) \neq \text{hash}(H_k)$
    $P \leftarrow \{(i, \rho) : (\tau_i, d_i) \in \mathcal{F},\ \rho < r\}$; restore $(s_{t_i^\star - 1}, h_{t_i^\star - 1})$ and replay suffixes under H_k and $H_k \oplus \Delta_z$
    $\widehat{g}_z \leftarrow \frac{1}{|P|} \sum_{(i,\rho) \in P} \left[y_{i\rho}(H_k \oplus \Delta_z) - y_{i\rho}(H_k) \right]$ {Eq. 7; $\widehat{g}_z \leftarrow w(z)$ without replay}
  end for
  if $\mathcal{T} = \emptyset$ **then**
    $H_{k+1} \leftarrow H_k$; $c \leftarrow c + c_k$; $k \leftarrow k + 1$; **continue** {every proposal failed a static check}
  end if
  $z_k \leftarrow \arg\max_{z \in \mathcal{T}} \widehat{g}_z$ (ties: lower cost, smaller edit)
  if $\widehat{g}_{z_k} \leq 0$ **then**
    $H_{k+1} \leftarrow H_k$; $c \leftarrow c + c_k$; $k \leftarrow k + 1$; **continue** {spend no validation budget on an edit the replay did not support}
  end if
  $H'_k \leftarrow H_k \oplus \Delta_{z_k}$
  Paired canary on $\mathcal{D}_{\text{can}}$; **if** a canary pair regresses **then** reject H'_k and skip the validation run {early exit}
  Paired validation on $\{(x_i, \sigma_i)\}_{i=1}^n \subset \mathcal{D}_{\text{val}}$ with $y_i \sim P_\theta^{H_k}$, $y'_i \sim P_\theta^{H'_k}$; compute W, L, N_{cat}, C' {Eq. 9}
  $H_{k+1} \leftarrow H'_k$ **if** $\text{accept}(H'_k, H_k) = 1$ **else** H_k {Eq. 10}
  $\mathcal{L} \leftarrow \mathcal{L} \cup \{H'_k\}$; $c \leftarrow c + c_k$; $k \leftarrow k + 1$ {c_k charges diagnosis, proposals, replays, canary, validation}
end while
Freeze $H^\star \leftarrow H_k$; evaluate $\widehat{J}_{\text{test}}(H^\star)$ on the sealed manifest

Table 5: Recorded configuration shared by all methods.

Field	Setting
Model	Qwen3.8-27B Max for planning and optimization; temperature 0; identifier and reasoning setting recorded in every receipt
Images	Four rendered views plus a numbered overlay per observation, JPEG transport, at most 24 images per optimizer call
Environment	VLABench pinned revision, six fixed skill primitives (pick, place, press, open, lift, pull) executed by π_0, 1,500 steps per command, 6,000 per episode; EmbodiedBench EB-Habitat and EB-Navigation as released; ESI-Bench interface as released
Budget	10 planner calls and 10 turns per episode; four rounds, four optimization episodes sampled per round; probe replicates 2; identical request cap for every method
Splits	15/1/24/40 optimization/canary/validation/test episodes per VLABench family; 300 tasks each for EB-Habitat and EB-Navigation; 10/10/231 questions for ESI-Bench
Selection	Gate $\ell = 0$ and $\alpha = 1.1$, with the cost check made against $C(H_0)$ rather than the parent; abstention threshold $\kappa_{\min} = 0.30$; two probed targets per round, the second taken from the most cited `alternative_causes` entry when only one component has positive weight and from the fixed order (S, C, V, R) when that field is empty; ties resolved by cost, then edit size; validation is entered only when $\widehat{g}_{z_k} > 0$
Edit bounds	At most 40 changed lines and 2,000 characters per candidate, for every component type
Catastrophic events	Simulator or worker crash, action rejected as out of bounds or beyond a joint limit, object driven off the workspace, unhandled tool exception; an ordinary unsuccessful completion is not one
Excluded pairs	A probe pair is dropped only when the runner raises a worker crash, a simulator start failure, or a transport timeout on either side; a malformed candidate action returns a failed receipt and the episode continues

Section 3.2 reads every failed trajectory twice, once with the model diagnoser and once with the receipt-grounded rule diagnoser, and Table 6 reports how often the two name the same artifact on the 102 confident development failures. The two agree on 76 of 102 attributions, a raw agreement of 74.5% and Cohen's $\kappa = 0.65$. Agreement is highest on skill, where a failed receipt followed by a replan is easy for both to read, and lowest on context, where the rule diagnoser can only see that no re-observation was requested and cannot tell a retention failure from a planning one. This is the reason the two are not merged into a single label: the rule diagnoser is a free second opinion that flags disagreement, not a ground truth, and a disagreement is what lowers κ_i and pushes the round toward abstention.

Table 6: Attribution agreement on the 102 confident development failures: rows are the model diagnoser, columns the receipt-grounded rule diagnoser.

Model \ Rule	Skill	Context	Verifier	Recovery
Skill	31	4	3	2
Context	3	14	2	1
Verifier	2	3	19	2
Recovery	1	1	2	12

F CASE STUDY

This section follows EmbodiedHarness through four recorded cases: an episode the frozen agent cannot escape, the single-component revision that solves it outright, the round in which the gate blocks a revision that would have regressed a solved task, and the revision the gate promotes under a cost-matched comparison. Together they show the method working where the frozen harness fails, and show each stage of the loop contributing to that outcome.

Frames, controller receipts, planner outputs, probe outcomes and gate decisions are taken unchanged from the run directories and the harness registry; no command sequence in this section was written

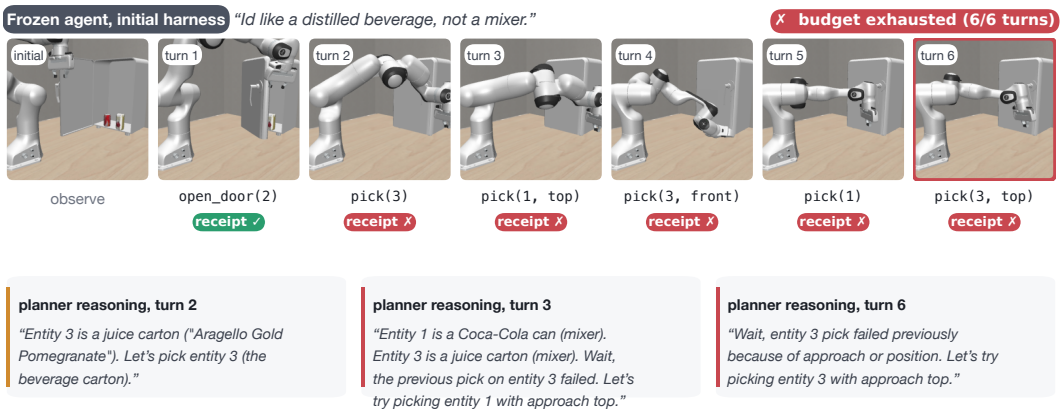


Figure 3: A recorded VLABench development episode (drink selection, commonsense, example 5) under the initial harness, with Qwen3.8-27B Max as the planner. Top: the planner view after each turn, the issued command and the controller receipt. Bottom: verbatim excerpts of the planner reasoning. The door opens, all five grasp requests return a failed receipt, and the episode ends at the six-turn budget. This development manifest runs under a six-turn limit, which both sides of the paired comparison share. Instructions are reproduced exactly as the benchmark ships them, including their own punctuation.

by hand. Every run in this section uses the same frozen planner as the main experiments, Qwen3.8-27B Max, and every comparison is paired: the episode, the seed, the executor and the planner are held fixed, and only the harness differs, so every difference reported below is attributable to the harness alone.

F.1 A RECORDED FAILURE OF THE FROZEN AGENT

Figure 3 records an episode the frozen harness cannot solve, and it fails in two ways at once. The first is the repetition of failed requests. After the first failed grasp the planner alternates between the two visible drinks and among the grasp approaches, but it never requests a new observation and never changes the skill; five consecutive failed receipts consume the entire budget and the episode scores zero. The initial recovery artifact is disabled, so nothing intercepts the repetition, and this is the pattern the rule diagnoser maps to a recovery revision (Appendix A). The second is unresolved grounding. In turn 3 the reasoning labels both visible drinks as mixers and still selects one of them, so even a successful grasp would not have satisfied the instruction.

Neither failure is in the model. Resampling the same planner reproduces both, while a bounded edit to the skill, the context or the recovery artifact reaches each of them, which is why the harness rather than the weights is the object worth optimizing here. The trajectory supports at least two such edits, and its binary outcome does not say which to try first. The next subsection shows EmbodiedHarness selecting one of them and turning this exact class of failure from a zero into a success, and the two after it show the gate deciding which such successes are allowed to persist.

F.2 A REPAIR PRODUCED BY A SINGLE-COMPONENT REVISION

Figure 4 is the paired counterpart of Figure 3: an episode that the parent harness cannot solve and that one revision repairs. The instruction asks for the cola to be taken out of the open fridge. Under the parent the planner cycles through the same request with the default, top and front approaches for eight turns; every receipt reports a failed grasp, and the episode ends when the planner reaches the token threshold at 93k tokens. Under the candidate the first grasp succeeds, a lift receipt reports failure, and the planner responds by grasping again from above, which carries the bottle out of the enclosure and satisfies the success condition after three turns and 23k tokens, a quarter of the parent cost. The two rows share the episode, the seed, the controller and the planner, and the context, verifier and recovery artifacts are byte-identical, so the difference is attributable to the skill edit

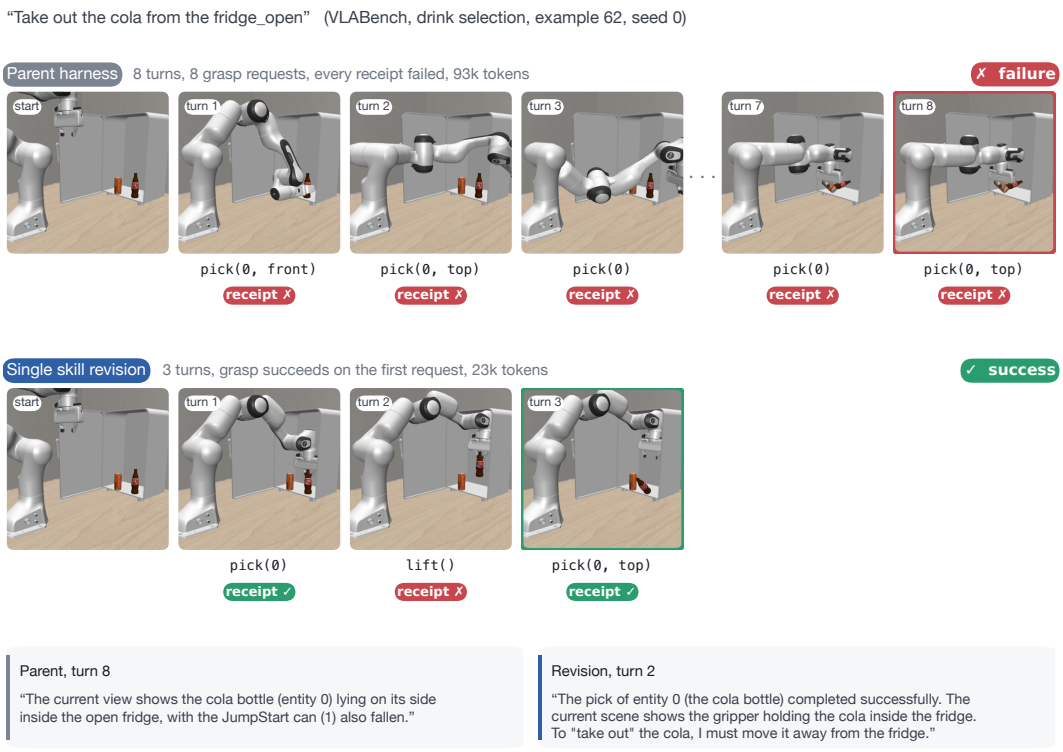


Figure 4: The same VLABench episode (drink selection, example 62, seed 0) under the parent harness and under a candidate that differs from it only in the skill artifact, with Qwen3.8-27B Max as the frozen planner in both rows. Both rollouts are recorded; commands, receipts and reasoning come from the run directories. The parent issues eight grasp requests that all return failed receipts and stops at the token budget; the candidate grasps the bottle on its first request and completes the task in three turns.

alone. The edit that produced it is the candidate in Figure 6; its first rule tells the planner to match the requested attributes against the numbered candidates rather than repeat a failing request, and its second rule fixes the grasp and transport sequence. Both rules target the failure patterns of Figure 3 directly, and together they convert an episode that exhausted the budget into one solved in three turns at a quarter of the cost.

A repaired episode is not yet a promoted revision. The next subsection records the gate enforcing that distinction on this very candidate.

F.3 One recorded development round

Figures 5 and 6 follow the same round from evidence to decision. All four diagnoses attribute the failure to planning and propose a skill revision, three of them at the first step. No other component receives any weight in this round, so the second probed target comes from the tie-breaking rule of Algorithm 1 rather than from the diagnoses: the optimizer wrote a context candidate that retains five instead of three recent views, taken from the most cited entry of the `alternative_causes` field.

Replay separates the two. The context candidate changes no outcome and also exceeds the execution cost limit. The skill candidate turns the drink trace into a success and proceeds to validation with an estimated gain of $+0.2$ over five complete pairs. That number is worth reading carefully, because it is a paired quantity and the pairing is what a cell of Figure 5b cannot show on its own. A pair enters P only when the parent run and the candidate run both completed: the candidate succeeds on the drink trace under both seeds, but the parent run of the second seed was broken by an infrastructure error, so that pair is excluded and the candidate's success in it is not counted. Five of the eight

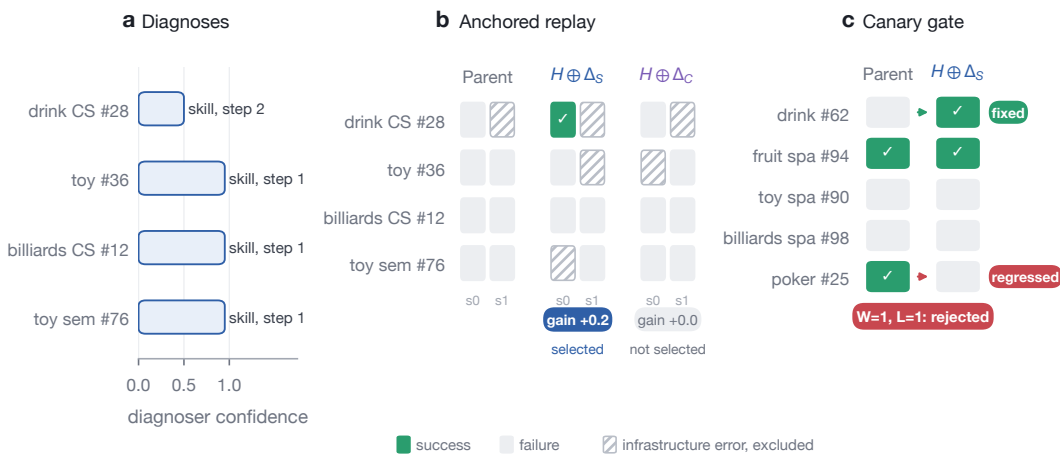


Figure 5: One recorded round of the probe campaign on VLABench (seed 0, round r01), with Qwen3.8-27B Max as planner and optimizer. (a) Diagnoses of the four failed training traces sampled for the round. (b) Anchored replay of the parent and of the two single-component candidates from each divergence anchor, with two seeds per site. (c) Paired canary of the selected skill candidate against the parent on five episodes, which includes the repair of Figure 4. Every cell is a recorded outcome.

Parent skill S hash 2b3e44

```
Use public observations and exact tool
schemas. Treat tool receipts as evidence
about the attempted operation. Identify
the requested object from the numbered
views and its role, pick it, and place
it into the requested container or onto
the requested surface before finishing.
Observe uncertain outcomes before
declaring completion.
```

Round record

other components
context, verifier, recovery unchanged

edit size
1074 characters

anchored replay
gain +0.2 over 5 complete pairs

canary gate
W = 1, L = 1

rejected, kept in lineage

Candidate skill S′ hash 3b3cf3

```
Use public observations and exact tool schemas.
Treat receipts as evidence about the attempted
operation, not task success.

1. Semantic Selection: Parse the instruction for
specific attributes (e.g., 'shapeshifting',
'classic', 'refreshing') and map them to visual
entities in the numbered views. Do not arbitrarily
pick the first object; verify its visual
properties match the request.

2. Execution Sequence: To move an object, use
`pick` followed immediately by `place`.
   • `pick`: Use `entity_id` of the target. If a
   receipt shows `operational_success: false`, try
   `approach: "top"` or adjust distance if
   supported, but do not assume failure implies a
   physical cause without observation.
   • `place`: Use `entity_id` of the destination
   container/surface.

3. Avoid Invalid Actions: Do not call `lift` after
`pick` unless explicitly required by a complex
sequence; `lift` often fails when the gripper is
already holding an object or has no target. The
`place` action handles the transfer.

4. Verification: Observe the scene after `place`.
Ensure the object is visually inside the container
or on the surface before declaring completion.
```

Figure 6: The skill candidate probed and validated in the round of Figure 5, beside its parent skill, rendered from the registry artifacts. Highlighted lines differ from the parent. Context, verifier and recovery are identical to the parent. List markers are typeset as bullets.

pairs survive and one of them is a win, which is $\hat{g}_S = 1/5 = +0.2$ by Equation 7. Exclusion is decided by the runner, not by the optimizer, and only for faults the runner itself raises, namely worker

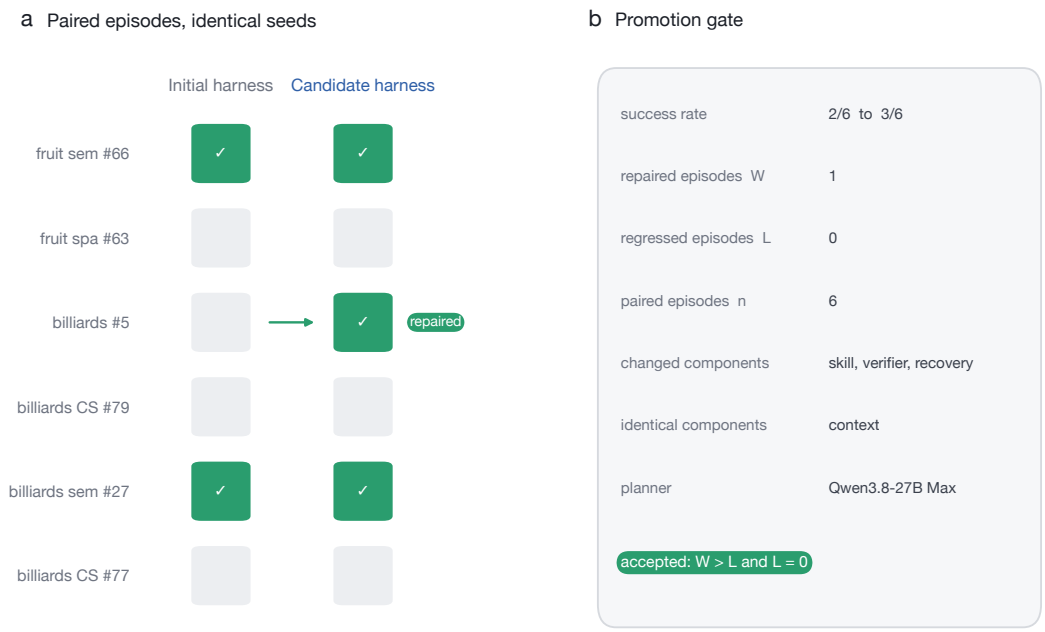


Figure 7: A paired evaluation run on the six reachable validation episodes of the development manifest, with Qwen3.8-27B Max as the frozen planner. Both columns use the same manifest, the same protocol and the same seeds, and each side spent 24 planner calls. (a) Per-episode outcomes. (b) The promotion gate applied to those outcomes. The grounded harness repairs one episode and regresses none, so the gate accepts it.

crashes, simulator start failures, and transport timeouts. A malformed action from a candidate is not an infrastructure error; it is returned as a failed receipt and the episode continues, so an unusable candidate cannot remove its own evidence. Of the three pairs lost by the skill candidate, one broke on the parent side and two on the candidate side; the context candidate lost one on each side and kept six pairs. The exclusion criterion, rather than the count, is what the protocol fixes.

The canary then shows the gate doing the work it exists for. The candidate fixes the cola episode of Figure 4 but loses a playing card episode that the parent solved, and the gate rejects it with $W = L = 1$. The candidate text shows why: it rewrites the whole skill, changing 1,074 characters, and adds a general instruction not to call lift after pick, whereas the playing card episode is scored by grasping the card and lifting it to a target height. The regression falls outside the probed set, so no replay could have exposed it and no amount of reflection on the repaired trace would have predicted it; only running the candidate against its parent on held-out episodes does.

This is the step competing loops omit. A method that adopts a revision as soon as it is written would have shipped this regression into the active harness and carried it forward for the rest of the run. EmbodiedHarness keeps the candidate out and retains it in the lineage with its evidence, which is the behaviour Property 1 guarantees on evaluated pairs. The no-gate row of Table 4 is the aggregate form of the same effect: without the gate the out-of-sample regression rate rises to 9.2%.

F.4 A CANDIDATE THAT THE GATE ACCEPTS

Figures 7 and 8 record a candidate that passes the acceptance rule of Equation 10. Success rises from two of six to three of six, a relative gain of 50 percent, with one episode repaired and none regressed, so $W > L$ with $L = 0$. The gain is not bought with extra inference. The comparison is cost matched rather than merely budget matched: both sides issued 24 planner calls and consumed 722,874 and 723,734 tokens respectively, a difference of 860 tokens, or 0.12 percent.

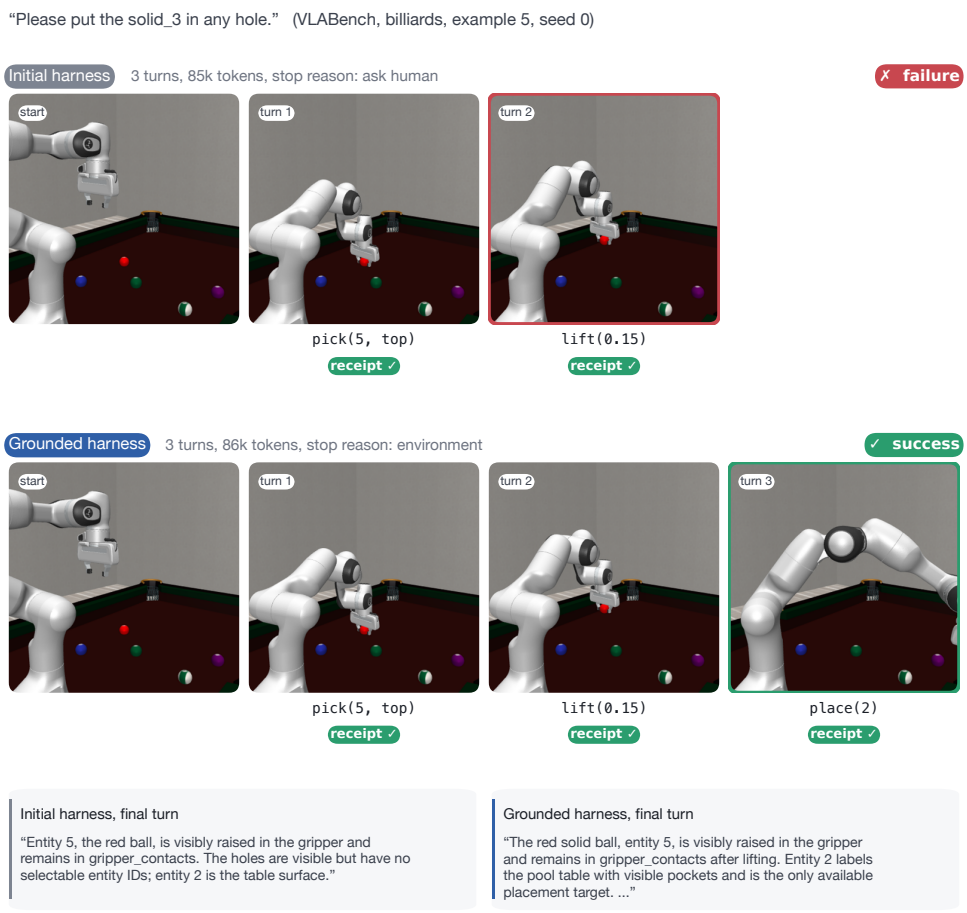


Figure 8: The episode repaired in Figure 7, recorded under both harnesses. The first two commands and their receipts are identical. The initial harness then halts and asks for human help, which the evaluator scores as a failure; the grounded harness issues the placement and the evaluator scores a success. Frames are the planner views of the two runs.

Figure 8 shows where the two rollouts diverge. Both grasp the requested ball from above and lift it by 0.15 metres, and both receipts report operational success, so the difference is not in motor control. It lies in what the planner concludes about the placement target. The recorded reasoning of the final turn makes this explicit. Under the initial harness the planner writes that the holes are visible but have no selectable entity identifier and that entity 2 is the table surface, and it then asks for human help while holding the ball, which ends the episode as a failure. Under the grounded harness the planner writes that entity 2 labels the pool table with its visible pockets and is the only available placement target, issues the placement, and satisfies the success condition. All four episodes that the initial harness fails in this run end in the same way, by asking for human help, so on these tasks the binding constraint is how the agent resolves the requested end state rather than whether the executor can act.

The accepted bundle changes the skill, verifier and recovery artifacts together, so the figure records the acceptance rule operating on a bundle rather than a single-component ablation; Table 4 separates the components.

The six episodes evaluated here are the ones on which any harness could matter. A host-side audit tried every visible candidate on the ten development validation episodes of this run under the same planner and executor and found six of them solvable; placing a toy into its box and lifting a playing card did not succeed for any candidate in the remaining four, so those four are out of reach of the executor rather than of the harness. Measured against that ceiling the frozen harness reaches two

of the six reachable episodes and the evolved harness three, so harness optimization alone recovers a quarter of the headroom the frozen harness leaves on the table, under identical planner calls and a token difference of 0.12 percent. This does not contradict the canary of Figure 5c, where the parent solves a playing card episode: that episode is `poker #25` of a different manifest, and its scored end state is reachable by the command sequence the parent happened to issue. Solvability is a property of an episode together with the executor and the command sequence tried, not of a task family. The audit is run by the host, and its results never enter a diagnosis, a probe or a gate.