RISE: RECOVERY-INFORMED SCALING VIA EVENTS FOR TOOL-USING LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Tool-using language models operate on mutable external state, where detecting an anomaly does not ensure that a subsequent action restores progress. We study Failed Recovery Decision: an incorrect decision during recovery with an attributable adverse outcome. Its retrospective adjudication is separated from the Recovery-risk Disagreement Signal, which uses public execution evidence. Building on this separation, RISE uses public evidence between completed rollouts to direct fresh executions, retain an accepted trajectory, and construct source-constrained event credit from eligible candidate comparisons. Selection determines which execution to retain; bilateral credit identifies structures to preserve or avoid and issues to recheck. Thus, a rejected proposal can guide a later execution without replacing the accepted trajectory. Across four benchmarks and three models, RISE improves over Best-of-N under a shared four-rollout limit, including increases from 82.73 to 91.37 on AppWorld Test-Challenge scenario completion and from 70.68 to 76.96 on OccuBench PASS@1 with GPT-5.6-Sol.[1]

1 INTRODUCTION

Language models increasingly complete tasks by calling external tools, some of which modify persistent state. Recent systems demonstrate strong capabilities in this setting, from learned tool invocation (Schick et al., 2023) and interleaved reasoning and acting (Yao et al., 2023b) to empirically tuned tool-using agents (Zhu et al., 2025a). Test-time compute strategies generate multiple attempts and select among them; verifier-based Best-of-N sampling provides a standard selection baseline (Snell et al., 2025; Zhu et al., 2025b). A tool call that writes to a database, books an itinerary, or edits a file can change the preconditions of dependent actions. When a failed call blocks a required dependency, recovery is needed before those actions can proceed, making recovery decisions important for reliable tool use.

Detecting an anomaly does not determine how to repair it. Intrinsic self-correction has documented limitations without external feedback (Huang et al., 2024), while tool-interactive feedback can support revision (Gou et al., 2024). In tool execution, recovery must additionally target the correct object, ground arguments, respect dependencies, and verify the relevant effects. Figure 1 shows a real case from AppWorld. The model calls show_drafts() at step A08 without an access token and receives an authentication error. It correctly diagnoses the missing credential and logs in at A36, obtaining a valid token. At A50 it retries the original read and omits the token again, reproducing the identical error. The authentication step succeeds, but the dependent

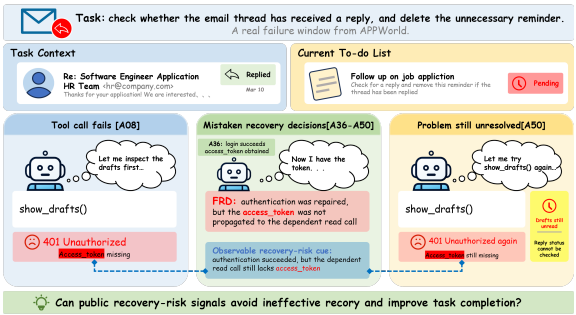


Figure 1: Entering recovery does not restore progress. The model obtains a valid token at A36 but omits it at A50, reproducing the error from A08.

[1]Code is available in the supplementary materials.

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.

read still fails because the newly obtained prerequisite is not applied. We call this a *Failed Recovery Decision*, which can arise from a wrong target, ungrounded arguments, an invalid action order, a skipped verification, or a premature stop. Therefore, the decisive question is not whether a model notices an error but whether the action it takes in response restores the execution.

Attributing a recovery outcome to an earlier decision may require evidence beyond the current observation. Local checks can confirm particular effects without establishing the full recovery outcome. We therefore distinguish retrospective failure adjudication from public recovery-risk evidence available for control. Selection-only sampling generates candidates independently before choosing among them (Zhu et al., 2025b), whereas reflection-based methods already carry verbal feedback into retries (Shinn et al., 2023; Madaan et al., 2023). Our focus is recovery-directed, source-constrained feedback: identifying what the next execution should recheck and which structures from selected and rejected candidates may inform it.

To this end, we propose RISE, a test-time scaling framework that combines complete-trajectory acceptance with event-level recovery guidance. As illustrated in Figure 3, a recovery frontier identifies what the next fresh execution should recheck from public risk, current event credit, and permitted history. An eligible comparison determines whether to replace the accepted trajectory and supplies feedback for later proposals. Preserve credit is drawn from the selected side and avoid credit from the rejected side; retained event shapes omit literal argument values. This separates two uses of the same comparison: choosing which execution to return and guiding how to generate the next one. A rejected candidate can therefore contribute evidence without becoming the search state.

We evaluate RISE against Vanilla and Best-of-N on four benchmarks, matching RISE and Best-of-N at a maximum of four complete rollouts. Experimental results show that RISE improves the reported metrics, raising AppWorld Test-Challenge scenario completion from 82.73 to 91.37 and OccuBench PASS@1 from 70.68 to 76.96 with GPT-5.6-Sol. In summary, our key contributions are three-fold.

- **Problem Formulation**: To the best of our knowledge, we are the first to characterize failed recovery as a distinct failure mode in tool-using language models. We formalize Failed Recovery Decision as an evidence-based retrospective label and distinguish it from public recovery-risk evidence available during execution.
- **Method**: We propose RISE, which directs fresh rollouts by public recovery risk, retains an accepted trajectory, and converts bilateral comparison into source-constrained structural credit, so that rejected candidates still shape later attempts.
- **Strong Performance**: Across four tool-calling benchmarks, RISE outperforms Best-of-N under a shared rollout limit. More importantly, this advantage widens on metrics that aggregate related tasks, such as scenario goal completion on AppWorld Test-Challenge.

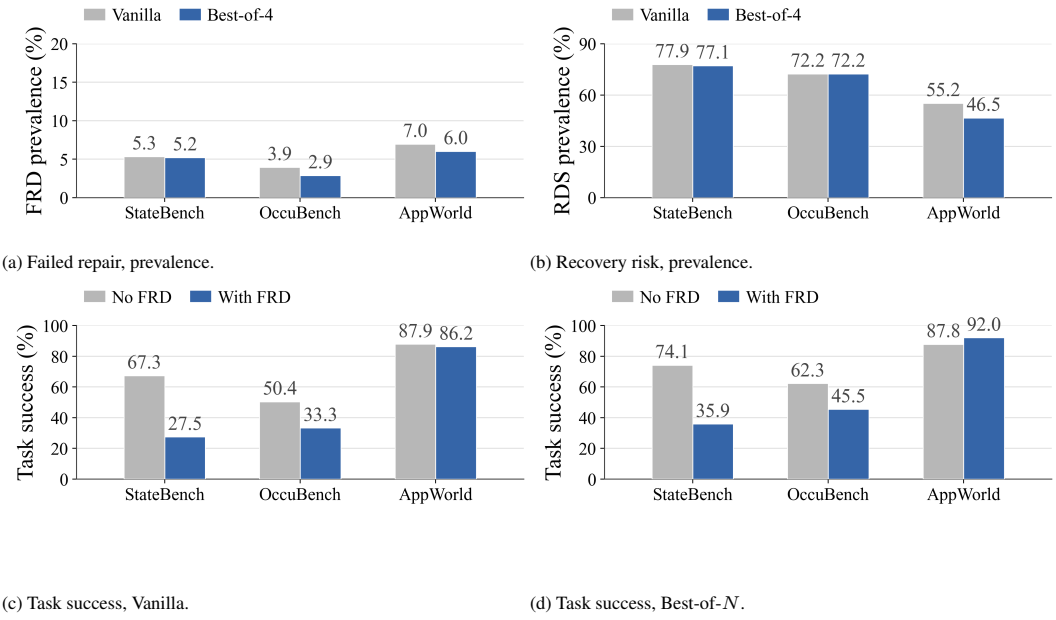
2 Why Does Repair Fail After the Model Detects an Error?

This section characterizes failed repair and distinguishes its retrospective adjudication from public recovery-risk evidence. Appendix C states the adjudication procedure, and the analyses use DeepSeek-v4.1-Flash.

Retrospective Label Let $\tau = (a_1, o_1, \ldots, a_T, o_T, r)$ denote a tool-calling trajectory with actions a_t, public observations o_t, and final response r. For each audited recovery episode w, the adjudicator checks four conditions in order: a recovery need F_w, a concrete recovery attempt D_w, an incorrect decision under the available task or tool evidence I_w, and an adverse outcome attributable to that decision L_w. A Failed Recovery Decision is

$$\mathrm{FRD}(\tau) = \mathbf{1}[\exists w \in \mathcal{W}(\tau) : F_w D_w I_w L_w = 1], \tag{1}$$

where $\mathcal{W}(\tau)$ is the set of recovery episodes in τ. Adjudication begins only when the public trace shows a concrete recovery entry, such as diagnosis, planning, a retry, a corrective call, or verification addressing a failed call, inconsistent state, or user-reported problem. An episode is labelled positive only when all four conditions are supported, including attribution of the adverse outcome to the incorrect decision. Recovery status is recorded separately from final task completion: a later successful repair does not erase an earlier failed decision. Appendix C provides the full protocol and audit coverage.



(a) Failed repair, prevalence.

(b) Recovery risk, prevalence.

(c) Task success, Vanilla.

(d) Task success, Best-of-N.

Figure 2: Failed repair is common and often harmful. With DeepSeek-v4.1-Flash, retrospective labels and public signals persist after Best-of-N selection (a, b); FRD-labelled trajectories complete fewer tasks in five of six benchmark–baseline comparisons (c, d).

Public Recovery-Risk Signal For control, the Recovery-risk Disagreement Signal marks public evidence requiring rechecking, without retrospective labels. Its single-trajectory component for task x and public prefix through action j is

$$R_{\tau,j} = f_{\text{risk}}(x, \tau_{\leq j}), \tag{2}$$

where the benchmark-specific f_{risk} extracts supported failure boundaries, repeated mutations, missing read-back, and explicit tool errors. These cues are observable during execution; RISE consumes them between completed rollouts. Candidate-pair disagreement is computed after a proposal completes and gates selection (Section 3.3). Neither kind of evidence constitutes the retrospective judgment in Equation 1.

Prevalence and Harm We measure both quantities on Vanilla and Best-of-N trajectories from StateBench, AppWorld, and OccuBench. Figure 2(a,b) shows that public recovery risk occurs in every benchmark and persists after selection; the retrospective FRD label is rarer because it requires an attributable adverse outcome. In Figure 2(c,d), FRD-labelled trajectories complete fewer tasks in five of the six benchmark–baseline comparisons. The exception is AppWorld under Best-of-N, where a failed recovery episode can precede eventual task completion. These associations motivate using public recovery evidence to guide later rollouts; Equation 2 specifies the single-trajectory signal available to that guidance.

3 RISE

Section 2 distinguishes retrospective adjudication from public recovery-risk evidence. RISE addresses this distinction by organizing three updates into a single search over complete executions. Generation and acceptance operate on complete trajectories, while analysis and feedback operate on public events. Every execution runs in a fresh environment.

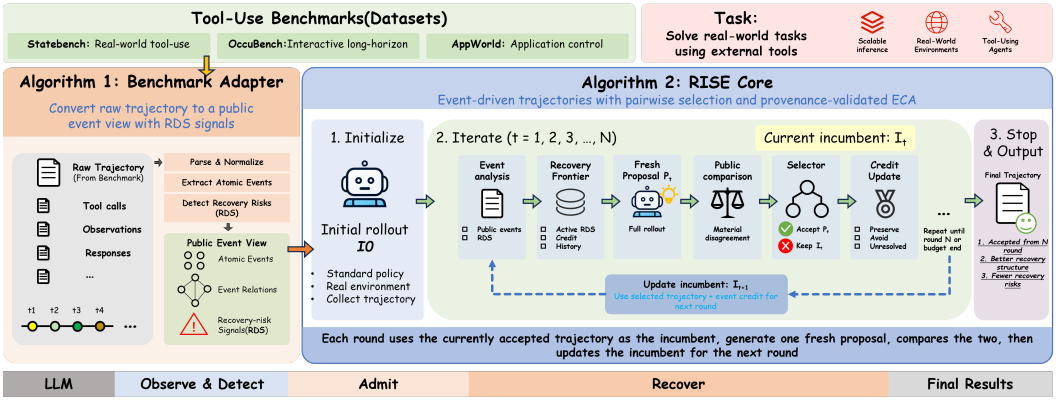


Figure 3: RISE combines complete-trajectory selection with recovery guidance. EDS denotes event-driven scaling and ECA denotes event credit assignment. A frontier directs a fresh rollout; eligible comparisons update bilateral credit. The final incumbent may originate from an earlier round. The unified procedure is given in Algorithm A.1.

3.1 Accepted Trajectory as the Search State

To avoid unconditionally replacing the current candidate, RISE maintains an accepted trajectory across rounds. The search begins from an unguided rollout and maintains an accepted trajectory I_t, current event credit C_t, and history H_t, with credit initially empty. Each round executes the original task in a fresh environment, so cross-round guidance may carry event structure while every execution grounds its own identifiers and values.

3.2 Recovery Frontier for Directed Rollouts

To convert public risk into an actionable instruction, RISE compiles the current trajectory into an event view and builds a frontier that names what the next attempt must recheck. The event view $V_t = f_{\text{event}}(I_t)$ contains atomic events E_t, supported groups G_t, and public recovery risk R_t; DeepPlanning also retains final-answer text. The frontier and the resulting proposal are

$$\mathcal{F}_t = f_{\text{front}}(V_t, C_t, H_t), \qquad P_t = f_{\text{roll}}(x, \mathcal{F}_t), \tag{3}$$

where f_{front} combines public evidence, current credit, and adapter-maintained history. f_{roll} executes a complete rollout in a fresh environment. The frontier carries guidance, not partially executed states (Appendix A).

3.3 Material Disagreement as a Selection Gate

RISE computes $D_t = f_{\text{cmp}}(f_{\text{event}}(I_t), f_{\text{event}}(P_t))$ using benchmark-specific predicates over transaction structure, ordered API signatures, or tool-event sequences and final-answer text (Appendix A). When $D_t = 0$, it retains the accepted trajectory and skips selection. Otherwise, the selector compares the complete candidates through their public representations. Let η_t denote its full response and $q_t = \text{choice}(\eta_t) \in \{\text{acc}, \text{prop}, \perp\}$ its selection field; nomination-based adapters also return credit information in η_t. The accepted trajectory is updated as

$$I_{t+1} = \begin{cases} P_t, & D_t = 1 \text{ and } q_t = \text{prop}, \\ I_t, & \text{otherwise}, \end{cases} \tag{4}$$

where the selector weighs task coverage, grounding, action order, public effects, recovery, verification, and stopping. A fallback retains the accepted trajectory under adapter-specific state updates.

3.4 Bilateral Credit with Source Constraints

To extract information from the candidate that was rejected, RISE derives guidance from both sides of a valid comparison under a provenance rule. Let V_t^s and V_t^r denote the selected and rejected

event views. The update is

$$C_{t+1} = f_{\text{credit}}(V_t^s, V_t^r, D_t, \eta_t, C_t) = (C_{t+1}^+, C_{t+1}^-, z_{t+1}), \tag{5}$$

where C^+ and C^- hold preserve and avoid credit, and z records an unresolved issue. Preserve and avoid events come only from the selected and rejected sides, respectively. For StateBench, let E_s and E_r be their event sets and $\mathcal{B}$ denote rejected, failed, and error results. Before sorting and truncation, preserve eligibility is

$$\begin{aligned}\mathcal{P}(E_s, E_r) = \{e \in E_s : \text{result}(e) \notin \mathcal{B} \wedge \\ [\text{shape}(e) \notin \text{Shape}(E_r) \ \vee \ \text{class}(e) \in \{\text{mutation}, \text{verification}\}]\}.\end{aligned} \tag{6}$$

A shape retains operation, event class, tool, entity field names, argument keys, result class, and interaction type, excluding literal argument values. StateBench draws avoid candidates from rejected-side risk identifiers, failure results, structural differences under the same projection, and mutation/verification rules. These candidates identify structures for scrutiny in the next rollout. Rejected-side shapes can also change preserve credit without replacing the incumbent. OccuBench, AppWorld, and DeepPlanning instead use joint selector nominations with source validation and structural projection. These constraints establish provenance, not event-level correctness. Appendix D gives post-processing details and cross-round examples. The next frontier uses the updated trajectory, credit, and permitted history; after K proposals, RISE returns the final incumbent using $K+1$ complete rollouts (Algorithm A.1, Appendix A).

4 Experiments

4.1 Benchmarks

We evaluate on four benchmarks that cover representative forms of complex tool calling, namely StateBench[2] for stateful conversational tool use, AppWorld (Trivedi et al., 2024) for persistent cross-application interaction, OccuBench (Hu et al., 2026) for professional tool use, and DeepPlanning Travel (Zhang et al., 2026b) for constrained planning. Task counts, evaluation cohorts, and metric definitions appear in Appendix B.

4.2 Setup

We compare RISE with Vanilla, which executes a single rollout, and with Best-of-N, the most representative way of spending extra computation at inference time. Drawing several complete attempts and selecting among them with a verifier is the form of test-time scaling that has been studied and released most systematically for tool-using agents (Snell et al., 2025; Zhu et al., 2025a;b), and it is what released agent systems adopt by default, so it is the baseline RISE has to beat at an equal budget. We set $N = 4$ following its released configuration and cap RISE at the same number of complete rollouts, so the N candidates of the baseline correspond to one initial trajectory followed by $K = 3$ guided proposals. This makes N the shared upper bound on both the sampling width of the baseline and the iteration depth of RISE. We evaluate DeepSeek-v4.1-Flash, Qwen3.8-Max, and GPT-5.6-Sol on all four benchmarks. Tasks, tools, simulators, and evaluators are held fixed across methods within each cohort, and every online decision reads the public trace.

4.3 Main Results

To determine whether directed rollouts outperform independent sampling at equal cost, we compare all three settings under an identical budget of complete rollouts. As shown in Tables 1–4, RISE improves over both baselines across all four benchmarks and all evaluated models, raising OccuBench PASS@1 from 68.85 to 74.35 with Qwen3.8-Max (Table 2) and AppWorld Test-Challenge scenario completion from 82.73 to 91.37 with GPT-5.6-Sol (Table 3). Best-of-N already captures most of what independent sampling offers, lifting OccuBench PASS@1 from 57.85 to 68.85. RISE adds a further 5.50 points at the same rollout budget, a gain sample diversity cannot explain because the number of complete executions is identical. More importantly, unlike independent sampling, RISE

[2]https://github.com/microsoft/STATE-Bench

Table 1: Results on StateBench. PASS^5 requires completion in all five independent runs and UX is rated from 1 to 5. **Bold** marks the best within each model.

Method	DeepSeek-v4.1-Flash			Qwen3.8-Max			GPT-5.6-Sol		
	PASS@1	PASS^5	UX	PASS@1	PASS^5	UX	PASS@1	PASS^5	UX
Vanilla	65.20	38.67	3.9709	69.33	41.33	4.0217	71.33	42.00	4.0658
Best-of-N	72.13	46.67	4.1428	76.00	49.33	4.1893	78.00	50.67	4.2374
RISE	**72.67**	**48.67**	**4.1499**	**81.33**	**54.67**	**4.3512**	**82.67**	**56.67**	**4.3981**

Table 2: Results on OccuBench. PASS@1 is the average completion rate over the evaluation instances. **Bold** marks the best within each model.

Method	DeepSeek-v4.1-Flash	Qwen3.8-Max	GPT-5.6-Sol
Vanilla	49.74	57.85	59.69
Best-of-N	61.78	68.85	70.68
RISE	**63.35**	**74.35**	**76.96**

Table 3: Results on AppWorld. 'TGC' and 'SGC' are task and scenario goal completion, and 'Normal' and 'Challenge' are the Test-Normal and Test-Challenge splits. **Bold** marks the best within each model.

Method	DeepSeek-v4.1-Flash				Qwen3.8-Max				GPT-5.6-Sol			
	Normal		Challenge		Normal		Challenge		Normal		Challenge	
	TGC	SGC	TGC	SGC	TGC	SGC	TGC	SGC	TGC	SGC	TGC	SGC
Vanilla	91.07	80.36	87.77	75.54	91.07	82.14	89.93	79.14	92.26	83.93	90.65	80.58
Best-of-N	91.07	82.14	88.01	77.70	92.26	83.93	90.41	81.29	93.45	85.71	91.13	82.73
RISE	**92.26**	**85.71**	**89.69**	**82.01**	**96.43**	**89.29**	**94.72**	**88.49**	**97.62**	**92.86**	**95.68**	**91.37**

constrains where each new rollout spends its attempt and carries the outcome of each comparison forward, so later rollouts are directed, not independent draws.

The margin over Best-of-N varies with a property of the failure distribution, and reading it this way explains an otherwise puzzling result. Independent sampling protects against *stochastic* failure, where independent draws rarely go wrong in the same place, so the maximum over the pool improves as the pool grows. It offers no protection against *systematic* failure, where a model omits the same prerequisite in every draw, because enlarging a pool of identically biased samples leaves the maximum unchanged. For instance, on AppWorld Test-Normal with DeepSeek-v4.1-Flash, Best-of-N leaves task completion unchanged at 91.07. Its failures there are the repeatable kind, and RISE lifts the same metric to 92.26 because an avoided structure recorded from one rejected candidate is unavailable to every later attempt.

This reading also predicts which metric should move first. Scenario goal completion requires every task in a related group to pass, so a defect that recurs across the group costs the entire score while task-level completion loses only the affected members. A metric that aggregates over correlated tasks is therefore more sensitive to systematic failure than to stochastic failure. On AppWorld Test-Challenge, RISE improves scenario completion over Best-of-N by 4.31, 7.20, and 8.64 points across the three models, against 1.68, 4.17, and 4.55 points for task completion on the same split, and the ordering holds for every evaluated model, which is what the mechanism predicts. Overall, RISE's strong results under a matched budget highlight its advantage in converting evidence about repeatable failure into a constraint, which is information that no amount of independent sampling can supply.

5 Analysis

Every analysis below uses DeepSeek-v4.1-Flash. We first isolate which component of RISE carries the gain on OccuBench, then trace the mechanism through one AppWorld execution, and then

Table 4: Results on DeepPlanning Travel. 'Deliv.' is plan delivery, 'Comm.' and 'Pers.' are constraint satisfaction, and 'Comp.' is the arithmetic mean of Commonsense and Personalized scores. **Bold** marks the best within each model and language.

Method	Chinese				English			
	Deliv.	Comm.	Pers.	Comp.	Deliv.	Comm.	Pers.	Comp.
DeepSeek-v4.1-Flash								
Vanilla	85.83	39.27	27.94	33.61	90.83	45.62	25.83	35.73
Best-of-N	89.17	43.18	30.86	37.02	93.33	49.37	28.94	39.16
RISE	**95.00**	**49.86**	**36.41**	**43.14**	**97.50**	**56.13**	**34.28**	**45.21**
Qwen3.8-Max								
Vanilla	89.17	41.63	29.73	35.68	94.17	47.86	27.64	37.75
Best-of-N	92.50	45.82	32.85	39.34	95.83	51.74	30.92	41.33
RISE	**95.83**	**51.47**	**38.64**	**45.06**	**97.50**	**58.08**	**36.18**	**47.13**
GPT-5.6-Sol								
Vanilla	90.83	42.75	30.58	36.67	95.00	48.71	28.37	38.54
Best-of-N	93.33	46.91	33.72	40.32	96.67	52.63	31.59	42.11
RISE	**98.33**	**53.36**	**39.46**	**46.41**	**99.17**	**59.41**	**37.73**	**48.57**

measure the targeted failure mode and the distribution of gains on StateBench, AppWorld, and OccuBench, matching the setting of Section 2.

5.1 Ablation Study on Guidance Components

To assess the contribution of the guidance components within RISE, we remove each in turn on OccuBench while holding the task manifest, model, selector budget, and evaluation protocol fixed. **No recovery focus** withholds the explicit target supplied by current public risk while retaining events and other feedback. **No event credit** keeps candidate comparison and acceptance but blocks credit from reaching later rollouts. **Pairwise summary** replaces structural credit with a free-form summary of the two candidates.

Table 5: Ablation on OccuBench with DeepSeek-v4.1-Flash. Each variant alters one guidance component under a fixed rollout budget. **Bold** marks the best result.

Method	PASS@1
Vanilla	49.74
Best-of-N	61.78
No recovery focus	53.14
No event credit	57.07
Pairwise summary	60.73
RISE	**63.35**

Table 5 shows a 10.21-point drop when recovery focus is withheld despite retaining trajectory information, and a 6.28-point drop when event credit is blocked while selection and acceptance are retained. These contrasts support both explicit recovery targeting and feedback across rounds in this configuration. A free-form pairwise summary trails full RISE by 2.62 points, indicating that the tested feedback formulation matters beyond providing a retry. All three variants also underperform Best-of-N under the same rollout budget, indicating that search requires both relevant guidance and a clear direction.

5.2 Case Study on a Directed Repair

To trace how the intended mechanism fires within a single task, we follow one AppWorld execution end to end. The first two comparisons retain the accepted trajectory, so neither proposal replaces it. The third proposal receives a frontier naming the failed authenticated read and begins in a fresh environment. It initially repeats unauthenticated reads, then consults the tool specification, supplies the missing credential on the dependent call, and passes every official check. Overall, as shown in Figure 4, the repair succeeds at the point the frontier identified, which highlights that guidance localizes the defect to the call the baseline left unrepaired.

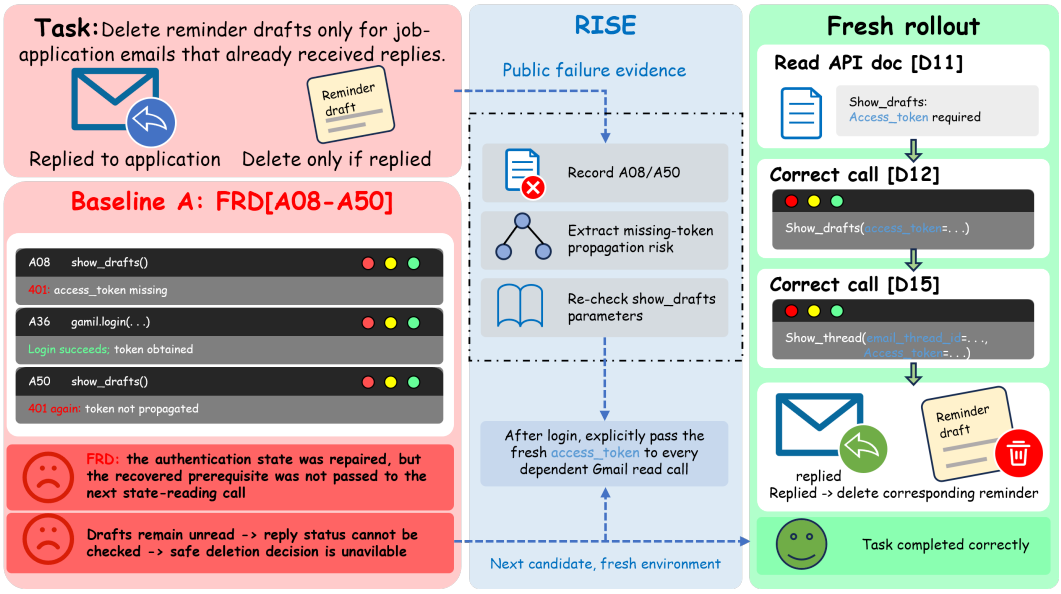
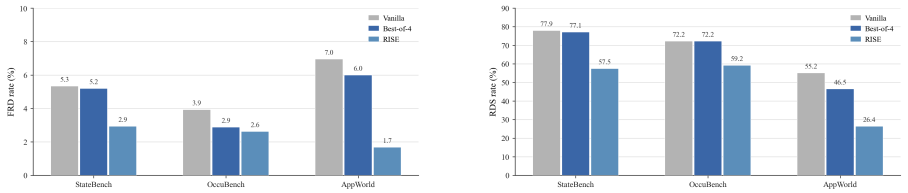


Figure 4: Directed recovery with DeepSeek-v4.1-Flash on AppWorld. The directed candidate supplies the missing credential at the retry the baseline left unauthenticated, then completes every official check.



(a) Adjudicated failed repair.

(b) Public recovery risk.

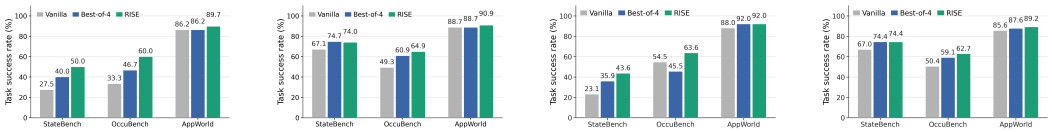
Figure 5: Recovery failures and public risk are less frequent in RISE's returned trajectories. All methods use DeepSeek-v4.1-Flash.

5.3 Reduction of Failed Repair

To measure the target failure mode in final outputs, we compute both the adjudicated label and the public signal on the trajectories each method returns. Figure 5 shows lower rates for RISE on both measures across the three benchmarks. The decrease in public risk is larger in percentage points; that signal captures a broader set of observable cues, whereas the FRD label requires an attributable adverse outcome. Together, the panels show fewer detected recovery failures and public risk signals in the returned trajectories.

5.4 Effectiveness on Risk-Positive Tasks

To locate where the gains fall, we fix cohorts of tasks whose baseline trajectories already exhibit recovery risk and evaluate every method on that identical task set, which keeps the denominator constant across methods. As shown in Figure 6, RISE improves over Vanilla on both the label-positive and signal-positive cohorts across all three benchmarks, with the widest margins on the label-positive cohort where the targeted defect is known to be present. The same procedure applied to Best-of-N cohorts yields the same conclusion, with the strongest result on the label-positive AppWorld cohort. In particular, RISE begins from its own unguided rollout instead of continuing the trajectory that defined the cohort, so these gains come from a fresh directed search. Overall, the cohort results highlight that the margin grows with the density of the defect the method was built to address.



(a) Label-positive, Vanilla.

(b) Signal-positive, Vanilla.

(c) Label-positive, Best-of-N.

(d) Signal-positive, Best-of-N.

Figure 6: Performance on fixed baseline-defined recovery-risk cohorts with DeepSeek-v4.1-Flash. All methods share the task set within each cohort; improvements vary across benchmarks and cohort definitions.

6 Related Work

Tool Calling and Test-Time Selection Research on tool-using language models divides into strengthening the controller and spending additional computation at inference time. ReAct established the interleaving of reasoning, tool calls, and observations (Yao et al., 2023b), and subsequent systems study design choices and reproducible comparison (Zhu et al., 2025a). A companion study of test-time compute evaluates parallel sampling, sequential revision, verification, and result merging, and its released configuration combines Best-of-N sampling with list-wise verification (Zhu et al., 2025b), while the verifier that ranks candidates and the benchmarks that place an agent against a mutable backend are themselves active subjects of study (Lightman et al., 2024; Yao et al., 2024). Related efforts improve the controller through verified tool-use data, tutorial-guided trajectory synthesis, and world models of action-conditioned transitions (Gao et al., 2025; Xu et al., 2025; Chae et al., 2025). These approaches address controller capability, observation quality, and inference-time computation.

Iterative Refinement at Inference Time A second line treats inference as a repeated process in which later attempts may consume the results of earlier ones. Self-consistency samples multiple reasoning paths and aggregates them (Wang et al., 2023), Tree of Thoughts expands and evaluates intermediate states (Yao et al., 2023a), tree search over agent trajectories adds value estimates and self-reflection (Zhou et al., 2024), and Reflexion and Self-Refine feed a verbal critique into a later attempt (Shinn et al., 2023; Madaan et al., 2023). Layered proposal and aggregation frameworks extend the premise across stages (Wang et al., 2025). Complementary work studies dynamic recovery in ToolMaze and contract-based runtime verification in ToolGate (Zhu et al., 2026; Liu et al., 2026); RISE instead uses public recovery evidence to guide subsequent complete executions.

Positioning ExpeL extracts insights from successful and failed experiences (Zhao et al., 2024), and ReasoningBank couples experience-derived memory with test-time scaling (Ouyang et al., 2026). RISE focuses on recovery within a task: public risk directs fresh execution, selection retains a complete trajectory, and source-constrained bilateral credit can change without replacement. Its contribution is this coupling of recovery focus, accepted-trajectory search, and event-level feedback, rather than iteration or experience reuse alone. Our experiments establish gains over Best-of-N and the tested internal variants, not over all iterative methods; Appendix E extends the comparison of designs.

7 Conclusion

We identify failed repair as a distinct failure mode in tool-using language models, where a model detects an anomaly, acts on it, and still leaves the task broken. Because such a failure is adjudicable only after execution ends, we build RISE on the public evidence available during a run. Recovery risk directs each fresh rollout, an accepted trajectory preserves progress, and bilateral credit turns every comparison into a constraint on what follows.

The results support a view of test-time compute as a choice about what a budget purchases. Sampling purchases variance and pays off against failures that differ between attempts, while RISE purchases information about failures that repeat, which is why its margin reaches 8.64 points over Best-of-N on metrics that aggregate over correlated tasks. More generally, a larger budget becomes a better one once the evidence of each comparison constrains the execution that follows.

AI USE STATEMENT

In this work, we used generative AI tools only for final-stage language polishing and grammar refinement of the manuscript. We have not used generative AI tools for generating research ideas, designing methods, conducting experiments, implementing code, analyzing data, interpreting results, or producing scientific claims, and other generative AI-assisted tasks requiring disclosure are not applicable to this work. We have reviewed all AI-assisted revisions to ensure their correctness and consistency with the authors' intended content. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We have made efforts to ensure the reproducibility of this work by providing detailed descriptions of the proposed method, experimental settings, implementation details, and evaluation protocols in the main paper and supplementary materials. The supplementary materials include the source code required to reproduce our experiments, and we will publicly release the code after the review process to further facilitate verification and future research. The content of this paper can be completely reproduced.

REFERENCES

Hyungjoo Chae, Namyoung Kim, Kai Tzu-iunn Ong, Minju Gwak, Gwanwoo Song, Jihoon Kim, Sunghwan Kim, Dongha Lee, and Jinyoung Yeo. Web agents with world models: Learning and leveraging environment dynamics in web navigation. In *International Conference on Learning Representations*, 2025.

Zhi Gao, Bofei Zhang, Pengxiang Li, Xiaojian Ma, Tao Yuan, Yue Fan, Yuwei Wu, Yunde Jia, Song-Chun Zhu, and Qing Li. Multi-modal agent tuning: Building a vlm-driven agent for efficient tool usage. In *International Conference on Learning Representations*, 2025.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *International Conference on Learning Representations*, 2024.

Samira Hajizadeh and Suman Jana. EffiPair: Improving the Efficiency of LLM-generated Code with Relative Contrastive Feedback, 2026. URL https://arxiv.org/abs/2604.05137.

Xiaomeng Hu, Yinger Zhang, Fei Huang, Jianhong Tu, Yang Su, Lianghao Deng, Yuxuan Liu, Yantao Liu, Dayiheng Liu, and Tsung-Yi Ho. OccuBench: Evaluating AI agents on real-world professional tasks via language environment simulation, 2026. URL https://arxiv.org/abs/2604.10866.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *International Conference on Learning Representations*, 2024.

Joongwon Kim, Wannan Yang, Kelvin Niu, Hongming Zhang, Yun Zhu, Eryk Helenowski, Ruan Silva, Zhengxing Chen, Srinivasan Iyer, Manzil Zaheer, Daniel Fried, Hannaneh Hajishirzi, Sanjeev Arora, Gabriel Synnaeve, Ruslan Salakhutdinov, and Anirudh Goyal. Scaling Test-Time Compute for Agentic Coding, 2026. URL https://arxiv.org/abs/2604.16529.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

Jiaye Lin, Yifu Guo, Yuzhen Han, Sen Hu, Ziyi Ni, Licheng Wang, Mingguang Chen, Hongzhang Liu, Ronghao Chen, Yangfan He, Daxin Jiang, Binxing Jiao, Chen Hu, and Huacan Wang. SE-Agent: Self-Evolution Trajectory Optimization in Multi-Step Reasoning with LLM-Based Agents, 2025. URL https://arxiv.org/abs/2508.02085.

Yanming Liu, Xinyue Peng, Jiannan Cao, Xinyi Wang, Songhang Deng, Jintao Chen, Jianwei Yin, and Xuhong Zhang. ToolGate: Contract-grounded and verified tool execution for LLMs. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 9653–9684, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.470. URL https://aclanthology.org/2026.findings-acl.470/.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, 2023.

Microsoft. STATE-Bench: Benchmark for enterprise workflows. Official project repository, 2026. URL https://github.com/microsoft/STATE-Bench.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T. Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. ReasoningBank: Scaling agent self-evolving with reasoning memory. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=jL7fwchScm.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, 2023. URL https://papers.nips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

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

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations*, pp. 10131–10165, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/1b623663fd9b874366f3ce019fdfdd44-Paper-Conference.pdf.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. Appworld: A controllable world of apps and people for benchmarking interactive coding agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, 2024.

Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. Mixture-of-agents enhances large language model capabilities. In *International Conference on Learning Representations*, pp. 33944–33963, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/5434be94e82c54327bb9dcaf7fca52b6-Paper-Conference.pdf.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *International Conference on Learning Representations*, 2023. URL https://iclr.cc/virtual/2023/poster/11718.

Yiheng Xu, Dunjie Lu, Zhennan Shen, Junli Wang, Zekun Wang, Yuchen Mao, Caiming Xiong, and Tao Yu. Agenttrek: Agent trajectory synthesis via guiding replay with web tutorials. In *International Conference on Learning Representations*, 2025.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 2023a. URL https://papers.nips.cc/paper_files/paper/2023/hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract-Conference.html.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *International Conference on Learning Representations*, 2023b.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024.

Tuo Zhang, Alin-Ionut Popa, Yan Xu, Rui Song, and Dimitrios Dimitriadis. PIVOT: Bridging Planning and Execution in LLM Agents via Trajectory Refinement, 2026a. URL https://arxiv.org/abs/2605.11225.

Yinger Zhang, Shutong Jiang, Renhao Li, Jianhong Tu, Yang Su, Lianghao Deng, Xudong Guo, Chenxu Lv, and Junyang Lin. DeepPlanning: Benchmarking long-horizon agentic planning with verifiable constraints, 2026b. URL https://arxiv.org/abs/2601.18137.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 19632–19642, 2024. doi: 10.1609/aaai.v38i17.29936. URL https://ojs.aaai.org/index.php/AAAI/article/view/29936.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning, acting, and planning in language models. In *International Conference on Machine Learning*, 2024.

Dongsheng Zhu, Xuchen Ma, Yucheng Shen, Xiang Li, Yukun Zhao, Shuaiqiang Wang, Lingyong Yan, and Dawei Yin. When Tools Fail: Benchmarking Dynamic Replanning and Anomaly Recovery in LLM Agents, 2026. URL https://arxiv.org/abs/2606.05806.

He Zhu, Tianrui Qin, King Zhu, Heyuan Huang, Yeyi Guan, Jinxiang Xia, Yi Yao, Hanhao Li, Ningning Wang, Pai Liu, Tianhao Peng, Xin Gui, Xiaowan Li, Yuhui Liu, Yuchen Eleanor Jiang, Jun Wang, Changwang Zhang, Xiangru Tang, Ge Zhang, Jian Yang, Minghao Liu, Xitong Gao, Jiaheng Liu, and Wangchunshu Zhou. Oagents: An empirical study of building effective agents, 2025a. URL https://arxiv.org/abs/2506.15741.

King Zhu, Hanhao Li, Siwei Wu, Tianshun Xing, Dehua Ma, Xiangru Tang, Minghao Liu, Jian Yang, Jiaheng Liu, Yuchen Eleanor Jiang, Changwang Zhang, Chenghua Lin, Jun Wang, Ge Zhang, and Wangchunshu Zhou. Scaling test-time compute for llm agents, 2025b. URL https://arxiv.org/abs/2506.12928.

A Implementation Details

Unified RISE Framework. RISE begins with an unguided rollout and maintains an accepted trajectory and event credit. Public event evidence, recovery risk, current credit, and permitted history guide a fresh complete proposal. After execution, a material-disagreement gate determines whether to invoke pairwise selection. A valid comparison can update the accepted trajectory and event credit, which inform subsequent generation. The procedure repeats within the rollout budget and returns the final accepted trajectory, not necessarily the last proposal (Algorithm A.1). Benchmark adapters provide the public representations and predicates used by this loop.

Public Recovery Risk and Timing. Public recovery evidence is available to the search without evaluator labels. Equation 2 denotes evidence extracted from a single public trace; benchmark adapters may also expose candidate differences after both rollouts finish. RISE constructs each frontier between complete executions, and Section 3.3 uses material disagreement to gate pairwise selection. The next proposal restarts the original task in a fresh environment, rather than continuing a previous environment state. We adjudicate Failed Recovery Decision after search and do not supply it to generation, selection, or credit construction.

Table 6 summarizes the four benchmark adapters used by the shared search procedure. For AppWorld, a routine business rejection is not an API failure solely because its text says rejected. DeepPlanning's optional domain-enhanced view is not assumed in the reported experiments; under its default predicate, changes only to literal argument values do not trigger comparison.

Table 6: Benchmark adapters in RISE. Each uses public evidence to guide a fresh complete rollout; the event view, comparison predicate, and credit rule differ.

Benchmark	Public view and frontier	Material disagreement	Credit construction
StateBench	Tool events and transaction groups; current failure, repeated mutation, or missing read-back takes priority over a prior unresolved issue.	Transaction content and order.	Rules after selection; preserve eligibility follows Equation 6.
OccuBench	Action–observation events and transaction risk in the StateBench-compatible view.	Transaction content and order.	Joint selector nominations, source checks, then structural projection.
AppWorld	API events, explicit failures, and unresolved issues from prior comparisons.	Ordered API signatures.	Joint nominations, source checks, then structural projection.
DeepPlanning	Tool events and final-answer text; at most twelve prioritized structural events plus credit.	Ordered tool, argument-key, and result-category tuples, or final-answer text.	Joint nominations, source checks, then structural projection.

Control Flow and Runner Semantics. Algorithm A.1 gives the shared search flow. No-disagreement rounds and selector fallbacks retain the incumbent under adapter-specific updates; runner errors retain their existing retry and resume behavior. Where supported, credit nominations share the selector response. Credit persistence and history updates remain adapter-specific. After K proposals, RISE has used $K + 1$ complete rollouts; the returned incumbent need not be the last proposal, and its true task score need not improve monotonically.

Algorithm A.1 RISE: Recovery-Guided Search over Complete Rollouts

Require: Task x, proposal schedule $\mathcal{S}$ of K stages
$I \leftarrow f_{\mathrm{roll}}(x, \varnothing)$; $C \leftarrow \varnothing$; initialize H
for stage $s \in \mathcal{S}$ **do**
  $V_I \leftarrow f_{\mathrm{event}}(I)$; $\mathcal{F} \leftarrow f_{\mathrm{front}}(V_I, C, H)$
  $P \leftarrow f_{\mathrm{roll}}(x, \mathcal{F})$; $V_P \leftarrow f_{\mathrm{event}}(P)$
  $D \leftarrow f_{\mathrm{cmp}}(V_I, V_P)$
  if $D = 0$ **then**
    retain I; apply adapter no-disagreement update; continue
  end if
  $\eta \leftarrow f_{\mathrm{sel}}(x, V_I, V_P)$; $q \leftarrow \mathrm{choice}(\eta)$
  if q is a fallback **then**
    retain I; apply adapter fallback update; continue
  end if
  $(V^s, V^r) \leftarrow$ selected and rejected views under q
  $C \leftarrow f_{\mathrm{credit}}(V^s, V^r, D, \eta, C)$
  if $q = \mathrm{prop}$ **then**
    $I \leftarrow P$
  end if
  $H \leftarrow f_{\mathrm{hist}}(H, D, \eta)$
end for
return I

B Benchmark Details and Paired Evaluation

StateBench[??] evaluates stateful conversational tool use across travel, customer support, and shopping assistance. We use its held-out test split of 50 tasks per domain, totaling 150 tasks. It reports PASS@1 for average completion, PASS^5 for completion in all five independent runs, and UX for user experience rated from one to five. The five runs use five generation, selector, and judge seeds; we evaluate every method under the same seed set.

AppWorld (Trivedi et al., 2024) evaluates persistent Python and API interaction across applications, comprising 168 tasks over 56 scenarios in Test-Normal and 417 tasks over 139 scenarios in Test-Challenge. It reports task goal completion and scenario goal completion, where a scenario passes only if every task in its group passes.

OccuBench (Hu et al., 2026) evaluates tool calling in professional settings spanning ten industry categories, with tool responses and fault conditions supplied by language environment simulators. Our evaluation cohort contains 382 instances, and we report PASS@1 over this cohort.

DeepPlanning Travel (Zhang et al., 2026b) evaluates travel planning under commonsense and personalized constraints. We evaluate 120 distinct tasks, each in Chinese and English, yielding 240 task–language instances. We report Delivery, Commonsense, Personalized, and Composite scores separately for each language, with Composite defined as $(\text{Commonsense} + \text{Personalized})/2$.

Paired Evaluation. Table 7 matches DeepSeek-v4.1-Flash outcomes by task and seed: 750 pairs on StateBench, 382 on OccuBench, and 417 on AppWorld Test-Challenge. W/L/T counts first-method wins, losses, and ties in binary task completion; $\Delta = 100(W - L)/n$ is the net pass-rate difference, not a count of repaired FRD episodes. The 95% intervals use 10,000 paired bootstrap resamples, grouped by task for StateBench and OccuBench and by scenario for AppWorld. They concern StateBench PASS@1 and AppWorld task goal completion, and do not represent additional model runs.

Table 7: Paired task-completion results for DeepSeek-v4.1-Flash. Differences and 95% paired bootstrap intervals are in percentage points. W/L/T denotes wins, losses, and ties of the first method against the second.

Comparison	Pairs	Δ	95% CI	W/L/T
StateBench				
RISE vs. Vanilla	750	+7.47	[3.60, 11.33]	111/55/584
RISE vs. Best-of-N	750	+0.53	[-2.67, 3.73]	74/70/606
Best-of-N vs. Vanilla	750	+6.93	[4.27, 9.73]	75/23/652
OccuBench				
RISE vs. Vanilla	382	+13.61	[8.38, 18.85]	81/29/272
RISE vs. Best-of-N	382	+1.57	[-3.66, 6.81]	54/48/280
Best-of-N vs. Vanilla	382	+12.04	[7.33, 16.75]	69/23/290
AppWorld Test-Challenge				
RISE vs. Vanilla	417	+1.92	[-0.24, 4.32]	15/7/395
RISE vs. Best-of-N	417	+1.68	[-0.96, 4.08]	18/11/388
Best-of-N vs. Vanilla	417	+0.24	[-2.16, 2.88]	16/15/386

RISE has positive point estimates against both baselines on all three benchmarks. The StateBench and OccuBench intervals against Vanilla exclude zero; the AppWorld interval and all comparisons against Best-of-N include zero. These intervals apply only to DeepSeek-v4.1-Flash on the listed binary task outcomes, not to other models or DeepPlanning Composite scores. The StateBench 0.53-point difference uses paired outcomes before rounding aggregate rates.

C Adjudication Protocol and Audit Coverage

Protocol. An episode enters review when a public trace shows a recovery attempt addressing an anomaly or user-reported problem. Equation 1 requires a recovery need, a concrete attempt, an incorrect decision, and an attributable adverse outcome. Linked actions may span tools and non-adjacent calls. Task failure, missing read-back, or a routine business rejection alone does not establish FRD; later success does not erase an earlier failed decision. Public risk is measured separately from this retrospective label.

Audit Scope. We screen the final returned DeepSeek-v4.1-Flash trajectory for each task/seed on StateBench, OccuBench, and AppWorld Test-Challenge. StateBench checks policy and preview constraints, OccuBench checks case-level action–observation evidence, and AppWorld checks API failures and tracebacks for linked recovery attempts. The supplementary screening in Table 8 differs from the annotations in Figures 2, 5, and 6; its counts do not replace those plots.

Table 8: Supplementary FRD screening for DeepSeek-v4.1-Flash. We count each trajectory once. 'No witness' means the audit found no FRD witness; it does not certify an FRD-negative trajectory.

Method	Trajectories	FRD-positive	No witness
StateBench			
Vanilla	750	40	710
Best-of-N	750	39	711
RISE	750	22	728
OccuBench			
Vanilla	382	15	367
Best-of-N	382	11	371
RISE	382	10	372
AppWorld Test-Challenge			
Vanilla	417	68	349
Best-of-N	417	40	377
RISE	417	3	414

The table covers 4,647 trajectories; the 4,399 without a witness are not adjudicated negatives. No independent blinded multi-annotator agreement is available. We extract public risk separately from single trajectories and define positive cohorts using the source baseline's labels. In the supplementary AppWorld FRD cohorts, RISE completes 65 of 68 Vanilla-positive tasks versus Vanilla's 66, and 39 of 40 Best-of-N-positive tasks versus 36 for Best-of-N. These cohorts therefore differ from those in Figure 6.

D Credit Construction

- **StateBench rule.** Equation 6 gives preserve eligibility. StateBench keeps at most six preserve and six avoid events; if preserve is empty, it uses the selected side's last three non-failure events. UNKNOWN is non-failure, not verified success. Avoid candidates come from rejected-side risks, failures, and structural differences; these are targets for scrutiny, not proven local errors.
- **Other adapters.** OccuBench, AppWorld, and DeepPlanning nominate credit in the same response that selects a trajectory. Preserve events must come from the selected side and avoid events from the rejected side. Source checks precede projection to event shapes without literal argument values; they establish provenance, not event correctness.
- **Credit without replacement.** For StateBench, let the unchanged selected side contain a non-failure read e_o and verification e_v. Under Equation 6, rejected sides that respectively contain and omit $\mathrm{shape}(e_o)$ yield preserve sets $\{e_v\}$ and $\{e_o, e_v\}$. Thus, rejected-side evidence can change credit without changing the incumbent.
- **Observed feedback path.** In AppWorld task 7264edc_3, proposal B becomes the incumbent; comparison with rejected proposal C supplies preserve/avoid credit and an unresolved recovery issue to the next generation request D. B remains accepted. These letters denote successive rounds, not a limit on the number of rounds, and the record does not isolate credit's effect on D's task outcome.

E Relation to Trajectory Reuse and Refinement

SE-Agent revises earlier trajectories (Lin et al., 2025), and Kim et al. (2026) use structured rollout summaries for selection and fresh execution. EffiPair uses paired execution profiles (Hajizadeh & Jana, 2026); PIVOT uses plan-execution discrepancies and preference-based acceptance (Zhang et al., 2026a). RISE combines public recovery risk, accepted-trajectory search, and source-constrained event credit. Our experiments compare Best-of-N and internal variants, not these related methods directly.