LET IT GO: NARRATIVE LOCK-IN IN OPEN-ENDED AI AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-horizon, tool-using agents must maintain an understanding of task goals, evidence, and current state across many actions. Yet we observe that when an early, plausible but incorrect interpretation becomes the task state on which later decisions are based, agents may continue to plan, verify, use tools, and produce deliverables around that error even as counterevidence accumulates. We call this trajectory-level failure Narrative Lock-in (NLI). We study how these interpretations shape subsequent execution and whether correcting them produces behavioral recovery. To characterize this phenomenon, we analyze 6,035 trajectories from four benchmarks and five models using a five-dimension rubric that spans narrative formation and behavioral propagation. Within the same task and model, NLI trajectories score lower in 34 of 42 matched groups, with mean completion of 0.207 compared with 0.623 for non-NLI trajectories. We then evaluate six correction strategies on a complete matched cohort of 728 selected cases. Strict recovery ranges from 10.7% to 36.4%, while verbal acknowledgment substantially overestimates behavioral recovery. This gap suggests that asking an agent to reconsider often leaves the task state governing its later actions unchanged. Motivated by this finding, we evaluate Nexus, a unified external state-supervision loop that uses preserved evidence to identify conflicts and guide task-state revision. On 75 ClawMark cases, Nexus records 44 wins, 15 ties, and 16 losses against the original trajectories. Together, these results characterize NLI as a measurable failure of long-horizon agent execution and show that evidence-grounded state supervision offers a practical response.

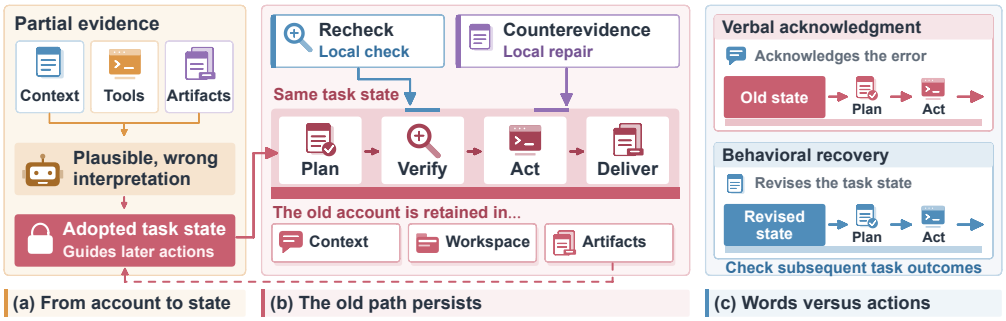


Figure 1: **Narrative Lock-in as persistent task state.** **(a)** A plausible but incorrect account becomes task state. **(b)** The old task state persists across actions and artifacts despite local checks and repairs. **(c)** Recovery requires state revision to propagate into execution; acknowledgment alone is insufficient.

1 INTRODUCTION

Long-horizon, tool-using agents are moving beyond short-form question answering toward real-world tasks that require them to read context, call tools, modify environments, and maintain an understanding of task goals, evidence, and current state across many actions. Yet across practical development and several agent benchmarks, we repeatedly observe a persistent failure. From real but

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.

partial evidence, an agent forms a plausible but incorrect interpretation and adopts it as the task state on which later decisions are based. The agent then continues to plan, verify, use tools, and produce deliverables around that interpretation rather than the original task. Counterevidence may prompt verbal acknowledgment or a local repair while subsequent behavior remains aligned with the same account. We call this trajectory-level failure *Narrative Lock-in* (NLI): the error is not only retained in text, but becomes the state from which the agent interprets evidence, chooses actions, and judges progress (Figure 1).

Prior work has shown that models can lose track of tasks in multi-turn or long-context interaction (Liu et al., 2024; Laban et al., 2026), and has examined context drift, memory contamination, belief inertia, and failures in agent execution (Wang et al., 2026; Dongre et al., 2025; Zou et al., 2026; Sunil et al., 2026; Zhang et al., 2025a;b). Self-checking, reflection, and external correction are commonly used to help agents recover. This motivates a trajectory-level question: when an incorrect interpretation has entered plans, tool calls, and deliverables, does correction change subsequent execution or only the agent's description of it? Answering this question requires following the interpretation from its adoption through later actions, task outcomes, and attempted recovery.

Motivated by these observations, we study NLI as a trajectory-level failure. We analyze 6,035 trajectories from four benchmarks and five models using a five-dimension rubric that spans narrative formation and behavioral propagation, and relate NLI to task outcomes within matched task–model groups. We then compare six correction strategies on a complete matched cohort of 728 selected cases, distinguishing verbal acknowledgment and local repair from genuine behavioral recovery. Finally, we test a direct response to the failure: maintaining observable evidence and unresolved task requirements outside the agent's own narrative. We build Nexus as a unified external state-supervision loop and evaluate whether it can help agents recover when their current account diverges from the observable task state.

Our contributions are threefold:

(1) Defining NLI. We define and operationalize NLI as a trajectory-level failure in which an erroneous interpretation enters task state, organizes behavior, and resists counterevidence.

(2) Empirical characterization. We characterize its formation, behavioral propagation, outcome association, and correction across four benchmarks and five models.

(3) Correction and state supervision. We identify the gap between linguistic correction and behavioral recovery, and test evidence-grounded state supervision through Nexus.

2 Problem Setup and Study Design

We formalize how an early, incorrect interpretation enters task state and organizes execution despite counterevidence. Final-answer correctness alone cannot reveal this process: we must observe the interpretation, its adoption in action, the available counterevidence, and subsequent behavioral updating. These requirements motivate the trajectory representation, NLI rubric, and detection, outcome, and intervention analyses below.

2.1 Tool-Mediated Trajectories and Operative Task State

Consider a long-horizon task with objective g. At step t, an agent receives task observations and tool feedback, expresses a judgment or plan, and produces a textual response, tool call, or environment action. We represent the complete trajectory as

$$\tau = (g, e_1, q_1, a_1, f_1, \ldots, e_T, q_T, a_T, f_T, d_T), \tag{1}$$

where e_t denotes evidence supported by the task specification, environment, or tool outputs; q_t denotes an explicitly expressed judgment, diagnosis, or plan; a_t denotes a task action; f_t denotes environment or tool feedback; and d_T denotes the final answer, state, or artifact. A task-native evaluator maps the resulting answer, state, or artifact to an outcome $y(\tau)$.

Rather than reconstructing the task from raw evidence at every step, agents maintain an account of the current situation, remaining work, and next action. We call this account the *operative task state* n_t: an interpretation reconstructed from expressed judgments, plans, tool choices, and actions that demonstrably organizes execution, rather than an unobserved internal belief.

New evidence should preserve, extend, or revise n_t. Yet a plausible but incorrect account formed from real, partial evidence may absorb later evidence as local exceptions. Planning, verification, tool calls, and deliverables then continue around that account without revising the state governing the trajectory.

2.2 Operationalizing Narrative Lock-in

We define *Narrative Lock-in* as a trajectory-level execution failure in which an early, coherent but incorrect interpretation enters the operative task state, remains substantively unrevised after counterevidence appears, and continues to organize later plans, actions, or artifacts. To identify this structure consistently across tasks, we represent each trajectory through five linked diagnostic dimensions:

$$r(\tau) = \big(F_{\text{form}}, F_{\text{adopt}}, F_{\text{counter}}, F_{\text{nonupdate}}, F_{\text{persist}}\big), \qquad F_k \in \{0, 1, 2, 3\}. \tag{2}$$

The five dimensions follow the trajectory from interpretation to consequence. **Narrative Formation** records whether a coherent but incorrect account appears early. **State Adoption** asks whether that account enters the agent's plan and actions. **Counterevidence** records whether contrary evidence becomes available and how it is handled. **Behavioral Non-update** asks whether later behavior changes in response, and **Persistence** records whether the same organization of the task continues or escalates across turns. Each dimension is scored from 0 to 3; the dimension definitions, ordinal anchors, and annotation protocol appear in Appendix B.8.

This persistent task-state structure distinguishes NLI from an isolated wrong answer, tool failure, or temporary planning error. Final task failure is not required: task-native evaluators supply outcomes independently, allowing us to measure NLI's association with completion.

2.3 From Problem Definition to Study Design

Neither a final answer nor a single action identifies NLI. We must trace **narrative formation** from partial evidence, **behavioral propagation** into plans, tool use, actions, and artifacts, and **updating under counterevidence**: state revision versus acknowledgment, patching, or rephrasing. Completion is evaluated independently of NLI detection. These requirements lead to three stages of analysis.

Detection and characterization. We reconstruct the five dimensions for each trajectory through permissive screening and stricter confirmation using the complete trace, tool feedback, and environment observations. Excluding unsolvable tasks, environmental confounds, and insufficient evidence, we examine NLI across benchmarks and models, from narrative formation to multi-step behavior.

Outcome association. We compare native outcomes for confirmed-NLI and non-NLI trajectories within the same task and model, controlling task difficulty and model composition. Rubric-defined damage separately records the adopted narrative's downstream effects rather than replacing official task outcomes.

Correction and behavioral updating. We continue each case from the same first identified lock-in state under six conditions: self-checking, reflection, and increasingly specific external corrections. Following replies, plans, actions, and task results distinguishes verbal acceptance, local plan revision, behavioral updating, and recovery.

Together, these stages organize the main study around two questions:

Q1. How does NLI form and propagate across benchmarks and models, and how is it associated with task outcomes (Section 3)?

Q2. To what extent do common corrections reconstruct the operative task state, and how does lock-in persist when they fail (Section 4)?

Only after answering these questions do we consider a state-supervision mechanism that does not inherit the agent's current narrative. Figure 2 connects the task sources and model coverage to annotation, matched correction, and evaluation of continued execution.

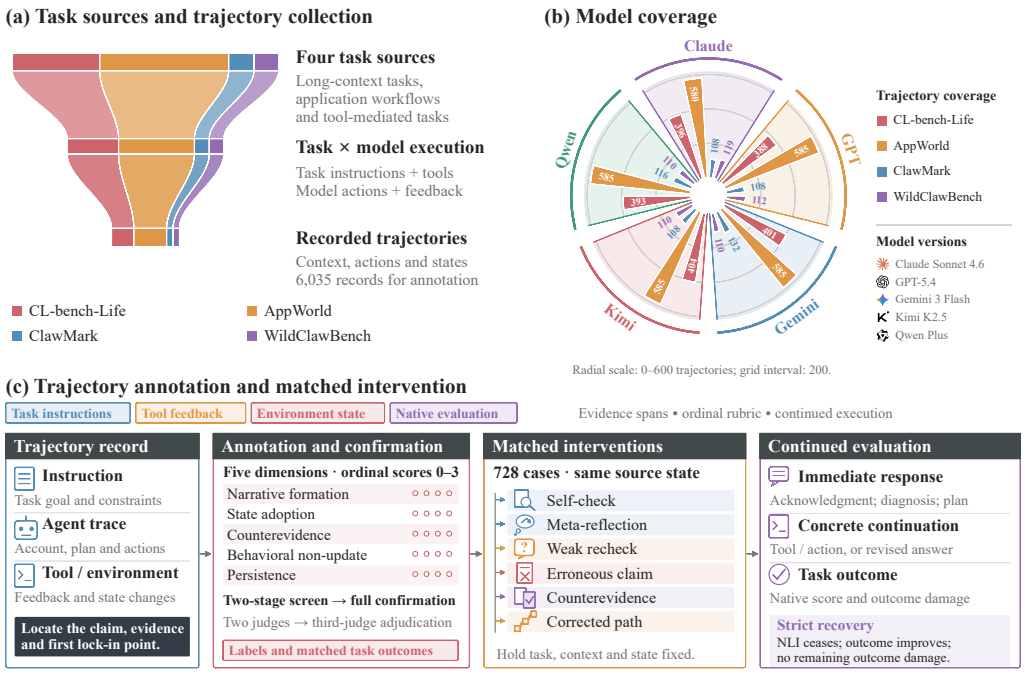


Figure 2: **From task collection to continued-execution evaluation.** **(a)** Task sources and trajectory collection; ribbon contraction is schematic, not sample attrition. **(b)** Model–benchmark coverage. **(c)** Five-dimension annotation and confirmation lead to matched interventions evaluated beyond the immediate reply. Task outcomes are measured separately from NLI confirmation.

2.4 Experimental Instantiation

Tasks and models. Our 6,035 trajectories cover four complementary benchmarks (Figure 3). CL-bench-Life (Dou et al., 2026) supports *formation-stage* analysis in long-context reasoning: Narrative Formation, State Adoption, and Counterevidence are observable; behavioral dimensions are scored only with a substantive post-answer turn. AppWorld (Trivedi et al., 2024), WildClawBench (Ding et al., 2026), and ClawMark (Meng et al., 2026) supply multi-turn plans, tool calls, environment feedback, and final states for *behavioral-stage* analysis of narrative propagation into actions and artifacts.

We evaluate GPT-5.4, Kimi-K2.5, Gemini-3-Flash, Qwen-Plus, and Claude-Sonnet-4-6. Benchmark-macro summaries and within-benchmark model comparisons prevent the largest environment from dominating. Appendix B provides the complete inventory, prompts, step limits, and run-level details.

Annotation and confirmation. Every trajectory undergoes two-stage screening with the shared rubric. Step 1 retains trajectories scoring at least 1 on any dimension for full evaluation. Step 2 requires verified solvability, at least three dimensions scored at least 2, Narrative Formation, coverage of Behavioral Non-update or Persistence, and no explicit environmental confound. For formation-stage CL-bench-Life, confirmation instead requires Narrative Formation, State Adoption, and Counterevidence each scored at least 2. Records lacking a definitive Step 2 decision remain in the inventory but are excluded from binary NLI–non-NLI comparisons.

Two DeepSeek V4 Pro calls independently assign 0–3 scores; a third adjudicates disagreements using both scores and rationales. Auxiliary fields record damage, confidence, primary error type, solvability, and manifestation. Solvability, execution validity, and NLI status are determined without downstream task scores. Appendix B.8 gives the evaluation inputs, scoring anchors, agreement statistics, and validity checks.

Outcomes and matched comparisons. We report continuous outcomes for confirmed-NLI and non-NLI trajectories in AppWorld, ClawMark, and WildClawBench; CL-bench-Life's answer-level outcomes use a different scale and are analyzed separately. We then compare within-group means

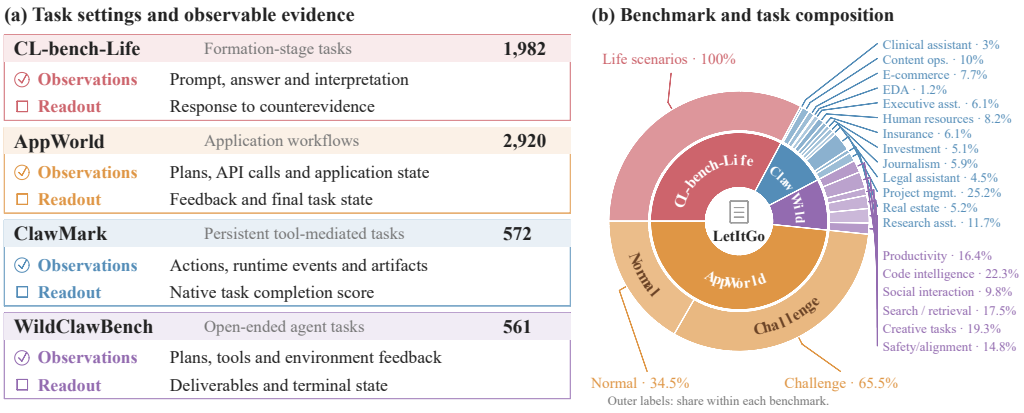


Figure 3: **Task settings, observable evidence, and composition.** **(a)** Task interpretation and execution evidence by benchmark. **(b)** Inner sectors: benchmark shares of all trajectories; outer sectors: task categories, labeled by within-benchmark percentage.

for groups matched by exact task ID and model and containing both labels. Official outcomes, rubric-defined damage, and NLI scores separately measure completion, downstream effects, and trajectory structure.

Correction experiments. The complete matched cohort contains 728 selected cases, each continued under six conditions from the same task context and environment state. Stage A records acknowledgment, revised diagnosis, or a new plan; Stage B requires a concrete continuation—a tool call or action for agent benchmarks, or a revised answer and justified next step for CL-bench-Life. Section 4 details the conditions and distinguishes strict recovery from partial responses. Appendix B.9 provides the complete prompts, continuation protocol, and recovery criteria.

3 Empirical Characterization of Narrative Lock-in

We first answer Q1 by asking whether the trajectory structure defined in Section 2.2 appears across tasks and models, and whether independently measured task outcomes change when that structure is present. Table 1 reports confirmation rates for every model and benchmark; Figure 4 relates these labels to matched outcomes, solvability-controlled scores, and downstream damage.

Table 1: **Narrative Lock-in across models and environments.** Trajectory counts (N) and confirmed-NLI rates (%) for formation-stage CL-bench-Life and three behavioral benchmarks. Totals pool models within each benchmark.

	CL-bench-Life		AppWorld		ClawMark		WildClawBench	
Model	N	NLI (%)	N	NLI (%)	N	NLI (%)	N	NLI (%)
Claude-Sonnet-4-6	396	9.3	580	0.7	108	56.5	119	5.9
GPT-5.4	388	11.3	585	14.2	108	16.7	112	17.9
Gemini-3-Flash	401	7.5	585	5.8	132	8.3	110	30.9
Kimi-K2.5	404	15.3	585	15.2	108	13.0	110	8.2
Qwen-Plus	393	15.5	585	12.8	116	30.2	110	26.4
All models	**1,982**	**11.8**	**2,920**	**9.8**	**572**	**24.3**	**561**	**17.6**

3.1 NLI Appears Across Benchmarks and Models

Formation-stage lock-in is not the same as a broad screening signal. The permissive first pass retains 1,954 of 1,982 CL-bench-Life trajectories (98.6%) for full evaluation. This number measures recall-oriented screening, not NLI prevalence. Under the conservative confirmation criteria, 234 trajectories (11.8%) exhibit formation, adoption, and counterevidence handling at the required

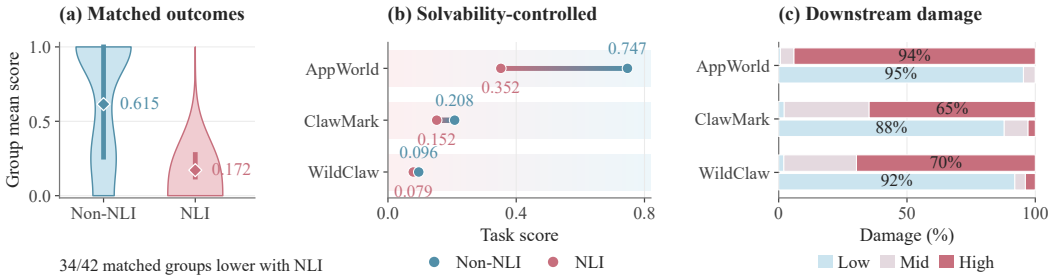


Figure 4: **Empirical characterization of Narrative Lock-in. (a)** Within-group mean scores for 42 matched task–model groups; diamonds: medians; vertical segments: interquartile ranges. **(b)** Means after solvability selection ($t = .5$), certifier removal, and ClawMark execution-validity filtering. **(c)** Damage distributions: NLI above and non-NLI below in each pair; two AppWorld non-NLI records without damage labels are excluded.

strength. Confirmed cases occur under all five evaluated models (Table 1), showing that early task-state adoption is not confined to one model.

Behavioral lock-in appears in each agent environment. Table 1 also reports behavioral lock-in across 4,053 AppWorld, ClawMark, and WildClawBench trajectories. The permissive screen retains 33.5% for full review. Conservative confirmation identifies 285 AppWorld trajectories (9.8%), 139 ClawMark trajectories (24.3%), and 99 WildClawBench trajectories (17.6%), for 523 confirmed behavioral-stage cases. These cases span all five models and contain observable propagation from an adopted interpretation into planning, tool use, environment interaction, or artifacts. Appendix B.8 gives the complete screening criteria, agreement analysis, and construct-validity checks.

3.2 NLI Is Associated with Worse Task Outcomes

Task outcomes are supplied by native evaluators and are not used to assign NLI labels. Of the 4,053 behavioral-stage trajectories, 3,217 receive a definitive binary NLI decision within the continuous-score protocol; excluding six non-final or execution-invalid attempts leaves 3,211 scored trajectories. The complete population flow and benchmark-specific score coverage appear in Appendix B.3.

The score gap remains after restricting the analysis to tasks with stronger solvability evidence (Figure 4b). At threshold 0.5, mean completion is 0.352 versus 0.747 for confirmed-NLI and non-NLI AppWorld trajectories, 0.079 versus 0.096 for WildClawBench, and 0.152 versus 0.208 for execution-valid ClawMark trajectories. Because pooled benchmark means can still reflect task composition, we additionally compare labels within the same task and model. NLI trajectories score lower in 34 of 42 eligible groups (Figure 4a), with trajectory-weighted means of 0.207 versus 0.623. Appendix B.5 documents the threshold construction, and Appendix B.4 reports the matching unit and aggregation; Appendix E explains raw and conditional reversals.

3.3 Locked Trajectories Show Greater Downstream Damage

Official completion and rubric-defined damage answer different questions. The first measures whether the benchmark task was completed; the second records how far the adopted narrative changed the attempted task, from a contained local error to a trajectory-wide effect on actions or artifacts. Across the three behavioral benchmarks, confirmed-NLI trajectories concentrate at higher damage levels (Figure 4c), although the severity distribution varies by environment. This separation matters: a trajectory can exhibit the NLI structure without ending in total failure, and a failed task need not exhibit NLI. Appendix B.7 defines the damage bins and reports the full benchmark-level distributions.

4 Intervening on Narrative Lock-in

The characterization results show that NLI is associated with lower task completion, but they do not establish how readily the adopted state can be revised. We therefore turn to Q2. If the failure were primarily a local mistake, a prompt to reconsider or a direct correction should redirect the

remaining trajectory. If the erroneous interpretation has become the state that organizes execution, language-level acknowledgment may occur without a corresponding change in plans, actions, or artifacts. Figure 5 reports the benchmark-specific recovery, response-only evaluation, and reset results; Table 2 gives the complete numerical comparison.

Table 2: **Strict recovery under matched corrections.** All six conditions use the same 728 cases. Benchmark columns report strict recovery percentages; the last columns report recovered cases and the overall rate. Bold marks the highest value in each column.

	Formation	Behavioral execution			All cases	
Correction	CL-Life	AppWorld	ClawMark	WildClaw	Recovered	Rate (%)
Cases per condition	297	222	121	88	728	
Self-check	19.5	4.5	3.3	6.8	78/728	10.7
Meta-reflection	30.0	19.8	9.1	9.1	152/728	20.9
L1: Weak recheck	30.6	9.0	13.2	10.2	136/728	18.7
L2: Claim only	40.4	20.7	7.4	9.1	183/728	25.1
L3: Concrete evidence	**49.2**	**40.1**	17.4	10.2	**265/728**	**36.4**
L4: Explicit path	40.4	19.8	**20.7**	**19.3**	206/728	28.3

4.1 Matched Intervention Design

The intervention study uses a complete matched cohort of 728 selected cases from screening and targeted trajectory review. Each case is continued from the same task context, environment state, and first identified lock-in point under six conditions. Two **trigger** conditions ask the agent to self-check or to reflect on evidence that could disprove its current approach. Four **correction** conditions provide progressively more task-specific information: a weak request to recheck, a warning about the current claim, concrete contradictory evidence, or an explicit corrected path with a next action. This matched design changes the correction signal while holding the source trajectory fixed.

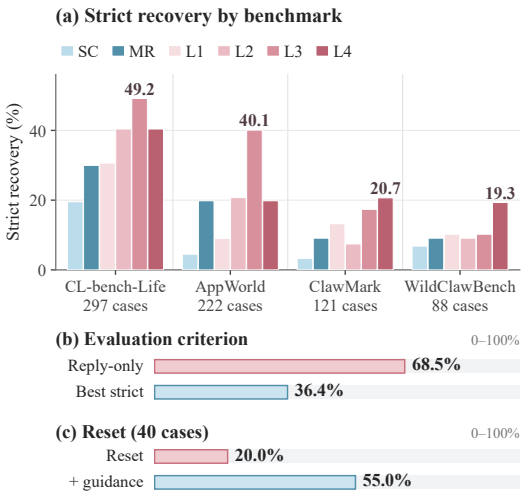


Figure 5: **Correction and recovery.** **(a)** Strict recovery across 728 matched cases; SC/MR: self-check/meta-reflection. CL-bench-Life covers formation; the others cover execution. **(b)** Reply-only and best strict rates use different endpoints. **(c)** Reset with/without guidance on the same 40 cases. Full rates: Table 2.

Evaluation continues beyond the immediate reply. A structured evaluator records whether the agent acknowledges the problem, revises its diagnosis or plan, changes subsequent behavior, and improves the task result. Strict recovery requires the locked narrative to cease, the outcome to improve, and no remaining outcome damage. A verbal acknowledgment, an unexecuted plan, or a surface rewrite is therefore recorded as a partial response rather than task recovery. Appendix B.9 provides the prompts, continuation rules, complete-panel selection rule, and evaluation schema.

4.2 Correction Helps, but Recovery Remains Limited

All six conditions recover some cases, but none recovers a majority of the complete panel (Table 2). Self-check recovers 78 of 728 cases (10.7%), while meta-reflection recovers 152 (20.9%). Recovery under the four correction conditions is 136 cases (18.7%) for weak recheck, 183 (25.1%) for claim-only correction, 265 (36.4%) for concrete counterevidence, and 206 (28.3%) for an explicit solution path. The increase through concrete evidence shows that specific contradiction is more useful than a

generic request to reconsider. The most prescriptive condition has lower aggregate recovery, although it is stronger on ClawMark and WildClawBench. More explicit guidance therefore does not yield a uniform gain across these environments.

The benchmark differences suggest why correction may become harder during execution (Figure 5a). A formation-stage claim can sometimes be reopened before it drives a long action sequence. Once the same interpretation has accumulated tool calls, intermediate results, workspace changes, subgoals, and deliverables, a local correction must reverse a longer execution path. The intervention results therefore characterize both the value and the limit of correction within an already established trajectory.

A separate 40-case stress test examines whether removing the inherited trajectory changes this picture (Figure 5c). Starting a new session without loading the original trajectory recovers 8 of 40 cases (20.0%). A fresh task packet with corrective guidance and ordered recovery steps recovers 22 of 40 (55.0%). The stronger recovery under guided reinitialization motivates rebuilding task state alongside changing the execution context. The two initialization formats and recovery criteria appear in Appendix B.10.

4.3 Language-Level Correction Overstates Behavioral Recovery

An auxiliary response-only evaluation classifies 68.5% of immediate post-intervention replies as recovered, highlighting the importance of the readout (Figure 5b). Appendix B.9 specifies this auxiliary evaluation; Table 25 reports field-level changes on the common 728-case cohort. Examination of the subsequent trajectories reveals three recurring partial-response patterns. In **verbal acknowledgment**, the agent accepts the correction while its plans and actions remain aligned with the prior account. In **local patching**, it repairs the flagged detail but leaves the broader task state unchanged. In **surface rephrasing**, the wording changes while the same claims and action choices are preserved.

These patterns clarify why rechecking and counterevidence should be considered separately. A generic recheck may examine one recent step or confirm a locally plausible signal without reopening the assumptions that organize the task. Concrete counterevidence more directly exposes a conflict, but it can still be absorbed as an exception or a narrow repair unless the conflict is propagated to the plan, environment state, and remaining requirements. By the time NLI is visible in behavior, the earlier account may already be preserved in context, workspace modifications, intermediate artifacts, and completed actions. Effective correction therefore requires a way to preserve observable evidence, identify state-level conflicts, and carry a revised state into subsequent decisions.

5 Evidence-Grounded Intervention

The correction and reset results suggest that recovery requires more than appending information to an established trajectory. We therefore test whether preserving evidence outside the agent's narrative can support state revision and redirect subsequent execution.

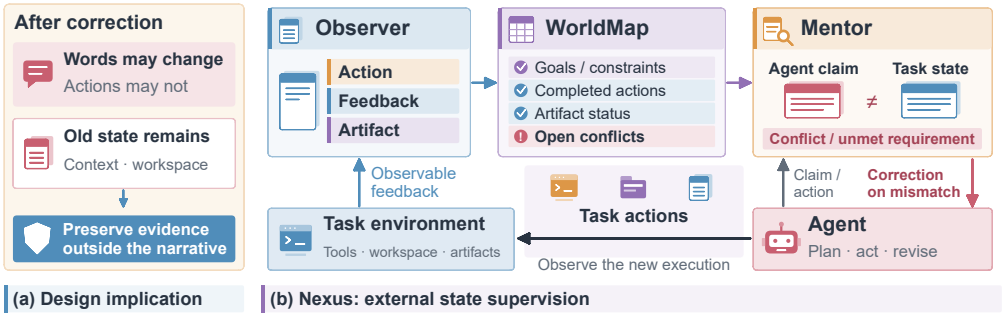


Figure 6: **Nexus maintains evidence outside the agent's narrative.** **(a)** Partial correction motivates separate evidence preservation. **(b)** Observer, WorldMap, and Mentor connect execution evidence, task state, and mismatch-triggered correction; subsequent actions close the feedback loop.

Intervention design. In Nexus (Figure 6), Observer records actions and feedback; WorldMap maintains goals, constraints, completed actions, conflicts, and artifact status; Mentor checks the proposed claim or action against this state and supplies a correction on mismatch. Subsequent actions return evidence to the loop. Appendix C details the updates and intervention logic.

Table 3: **Task outcomes with evidence-grounded intervention.** Mean ClawMark scores (%) on 75 cases, 15 per model; parentheses give percentage-point changes and W/T/L denotes wins, ties, and losses. Bold marks mean-score improvements over the original trajectories.

Condition	Claude Sonnet 4.6	GPT 5.4	Gemini 3 Flash	Kimi K2.5	Qwen Plus	All
Original	15.5	29.0	20.6	29.6	23.5	23.6
+ Nexus	8.0 (–7.4)	**63.8** (+34.8)	**33.9** (+13.3)	**40.5** (+10.8)	**27.7** (+4.2)	**34.8** (+11.1)
W/T/L	6/7/2	12/1/2	9/2/4	8/3/4	9/2/4	**44/15/16**

Targeted evaluation. We test this intervention in persistent, tool-mediated execution using 75 ClawMark cases: 15 tasks crossed with five models (Table 3). Against the corresponding original trajectories, Nexus records 44 wins, 15 ties, and 16 losses, with mean score gain $+0.111$. On the 29-case subset with all six correction records, Nexus improves 19/29 task scores (65.5%); L3 has 11/29 judged outcome improvements (37.9%). Appendix Table 21 distinguishes full-task score gains from post-correction judgments. Measured amortized token overhead is 30.7%. Appendix C gives formation-stage, subgroup, and runtime results. These model-dependent gains support evidence-grounded state maintenance as a practical response to the observed execution failures.

6 Related Work

Long-horizon agents. Tool-using and interactive agents (Schick et al., 2023; Yao et al., 2023; Wang et al., 2023), memory-augmented architectures (Packer et al., 2023; Lewis et al., 2020), and self-critique methods (Madaan et al., 2023; Shinn et al., 2023; Gou et al., 2024) combine reasoning, tools, memory, and environment interaction to support extended planning and execution. We study the reliability of the task understanding maintained throughout that execution, particularly when a plausible interpretation persists despite conflicting evidence.

Agent failure analysis. Complementary work studies hallucination and sycophancy (Augenstein et al., 2024; Maynez et al., 2020; Sharma et al., 2024; Perez et al., 2023), attention and task loss in long-context or multi-turn interaction (Liu et al., 2024; Laban et al., 2026), context drift and memory pollution (Dongre et al., 2025; Zou et al., 2026; Sunil et al., 2026), belief inertia (Wang et al., 2026), and failures in agent planning, tool use, and execution (Zhang et al., 2025a;b; Cemri et al., 2025; Zhu et al., 2025). These studies establish that errors can persist beyond individual answers or actions. We trace an incorrect interpretation through adoption, tool use, state changes, and correction, linking it to task outcomes and evaluating recovery through subsequent behavior rather than immediate replies. Appendix Table 23 compares study settings, analysis units, and intervention designs; Appendix G relates these observations to self-correction and task-ledger methods.

7 Conclusion

Narrative Lock-in is a measurable trajectory-level failure: an early erroneous interpretation becomes task state, organizes execution, and resists correction. Across 6,035 trajectories from four benchmarks and five models, matched comparisons associate NLI with substantially lower completion. Six correction strategies on 728 selected cases reveal a gap between verbal acknowledgment and behavioral recovery: adding a correction need not reconstruct the state governing subsequent actions. Nexus addresses this gap by preserving observable evidence and unresolved requirements outside the agent's narrative to guide state revision. Its gains on persistent ClawMark tasks support evidence-grounded supervision as a practical response, connecting this identifiable long-horizon failure to its downstream consequences and a direction for more reliable execution.

AI Assistance Disclosure

AI tools assisted language and figure polishing, code and data-processing scripts, and draft audits for clarity, consistency, and completeness. The authors reviewed and approved all AI-assisted outputs and retain full responsibility for the paper.

Ethical Considerations

All benchmarks used in this work (CL-bench-Life, AppWorld, WildClawBench, ClawMark) are publicly available; their original distribution terms apply. No human subjects or crowd workers were recruited for benchmark data collection. The blinded label check was conducted by a coauthor as described in Appendix B.8. The study concerns reliability failures in controlled agent tasks. In deployment, an external supervisor such as Nexus may omit relevant evidence or impose an incorrect task-state schema; its interventions should therefore remain auditable and should not replace human review in high-stakes settings.

References

Isabelle Augenstein, Timothy Baldwin, Meeyoung Cha, Tanmoy Chakraborty, Giovanni Luca Ciampaglia, David Corney, Renee DiResta, Emilio Ferrara, Scott Hale, Alon Halevy, Eduard Hovy, Heng Ji, Filippo Menczer, Ruben Miguez, Preslav Nakov, Dietram Scheufele, Shivam Sharma, and Giovanni Zagni. Factuality challenges in the era of large language models and opportunities for fact-checking. *Nature Machine Intelligence*, 6(8):852–863, 2024.

Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei A Zaharia, Joseph E Gonzalez, and Ion Stoica. Why do multi-agent LLM systems fail? *Advances in Neural Information Processing Systems*, 38, 2025.

Shuangrui Ding, Xuanlang Dai, Long Xing, Shengyuan Ding, Ziyu Liu, Yang JingYi, Penghui Yang, Zhixiong Zhang, Xilin Wei, Xinyu Fang, Yubo Ma, Haodong Duan, Jing Shao, Jiaqi Wang, Dahua Lin, Kai Chen, and Yuhang Zang. WildClawBench: A benchmark for real-world, long-horizon agent evaluation. *arXiv:2605.10912*, 2026.

Vardhan Dongre, Ryan A Rossi, Viet Dac Lai, David Seunghyun Yoon, Dilek Hakkani-Tür, and Trung Bui. Drift no more? context equilibria in multi-turn LLM interactions. *arXiv:2510.07777*, 2025.

Shihan Dou, Yujiong Shen, Chenhao Huang, Junjie Ye, Jiayi Chen, Junzhe Wang, Qianyu He, Shichun Liu, Changze Lv, Jiahang Lin, Jiazheng Zhang, Ming Zhang, Shaofan Liu, Tao Ji, Zhangyue Yin, Cheng Zhang, Huaibing Xie, Jianglu Hu, Jingcheng Deng, Lincheng Li, Minda Hu, Shaolei Wang, Syrus Zhao, Weichao Wang, Yan Lei, Yang Liu, Yanling Xiao, Yiting Liu, Zenan Xu, Zhen Guo, Ziliang Zhao, Pluto Zhou, Tao Gui, Qi Zhang, Xuanjing Huang, Yu-Gang Jiang, Di Wang, and Shunyu Yao. CL-bench Life: Can language models learn from real-life context? *arXiv:2604.27043*, 2026.

Adam Fourney, Gagan Bansal, Hussein Mozannar, Cheng Tan, Eduardo Salinas, Erkang Zhu, Friederike Niedtner, Grace Proebsting, Griffin Bassman, Jack Gerrits, Jacob Alber, Peter Chang, Ricky Loynd, Robert West, Victor Dibia, Ahmed Awadallah, Ece Kamar, Rafah Hosn, and Saleema Amershi. Magentic-One: A generalist multi-agent system for solving complex tasks. *arXiv:2411.04468*, 2024.

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. URL https://openreview.net/forum?id=Sx038qxjek.

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. URL https://openreview.net/forum?id=IkmD3fKBPQ.

Ryo Kamoi, Yusen Zhang, Nan Zhang, Jiawei Han, and Rui Zhang. When can LLMs actually correct their own mistakes? a critical survey of self-correction of LLMs. *Transactions of the Association for Computational Linguistics*, 12:1417–1440, 2024. doi: 10.1162/tacl_a_00713.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. LLMs get lost in multi-turn conversation. In *International Conference on Learning Repre-*

sentations, 2026. URL https://www.microsoft.com/en-us/research/publication/llms-get-lost-in-multi-turn-conversation/.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the association for computational linguistics*, 12:157–173, 2024.

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. *Advances in neural information processing systems*, 36: 46534–46594, 2023.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 1906–1919. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.173.

Fanqing Meng, Lingxiao Du, Zijian Wu, Guanzheng Chen, Xiangyan Liu, Jiaqi Liao, Chonghe Jiang, Zhenglin Wan, Jiawei Gu, Pengfei Zhou, Rui Huang, Ziqi Zhao, Shengyuan Ding, Ailing Yu, Bo Peng, Bowei Xia, Hao Sun, Haotian Liang, Ji Xie, Jiajun Chen, Jiajun Song, Liu Yang, Ming Xu, Qionglin Qiu, Runhao Fu, Shengfang Zhai, Shijian Wang, Tengfei Ma, Tianyi Wu, Weiyang Jin, Yan Wang, Yang Dai, Yao Lai, Youwei Shu, Yue Liu, Yunzhuo Hao, Yuwei Niu, Jinkai Huang, Jiayuan Zhuo, Zhennan Shen, Linyu Wu, Hannah Yao, Charles Chen, Cihang Xie, Yuyin Zhou, Jiaheng Zhang, Zeyu Zheng, Mengkang Hu, and Michael Qizhe Shieh. ClawMark: A living-world benchmark for multi-turn, multi-day, multimodal coworker agents. *arXiv:2604.23781*, 2026.

Charles Packer, Vivian Fang, Shishir G. Patil, Kevin Lin, Sarah Wooders, and Joseph E. Gonzalez. MemGPT: towards LLMs as operating systems. *arXiv:2310.08560*, 2023.

Ethan Perez, Sam Ringer, Kamilė Lukošiūtė, Karina Nguyen, Edwin Chen, Scott Heiner, Craig Pettit, Catherine Olsson, Sandipan Kundu, Saurav Kadavath, Andy Jones, Anna Chen, Ben Mann, Brian Israel, Bryan Seethor, Cameron McKinnon, Christopher Olah, Da Yan, Daniela Amodei, Dario Amodei, Dawn Drain, Dustin Li, Eli Tran-Johnson, Guro Khundadze, Jackson Kernion, James Landis, Jamie Kerr, Jared Mueller, Jeeyoon Hyun, Joshua Landau, Kamal Ndousse, Landon Goldberg, Liane Lovitt, Martin Lucas, Michael Sellitto, Miranda Zhang, Neerav Kingsland, Nelson Elhage, Nicholas Joseph, Noemí Mercado, Nova DasSarma, Oliver Rausch, Robin Larson, Sam McCandlish, Scott Johnston, Shauna Kravec, Sheer El Showk, Tamera Lanham, Timothy Telleen-Lawton, Tom Brown, Tom Henighan, Tristan Hume, Yuntao Bai, Zac Hatfield-Dodds, Jack Clark, Samuel R. Bowman, Amanda Askell, Roger Grosse, Danny Hernandez, Deep Ganguli, Evan Hubinger, Nicholas Schiefer, and Jared Kaplan. Discovering language model behaviors with model-written evaluations. In *Findings of the association for computational linguistics: ACL 2023*, pp. 13387–13434, 2023.

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. *Advances in neural information processing systems*, 36:68539–68551, 2023.

Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R. Bowman, Newton Cheng, Esin Durmus, Zac Hatfield-Dodds, Scott R. Johnston, Shauna Kravec, Timothy Maxwell, Sam McCandlish, Kamal Ndousse, Oliver Rausch, Nicholas Schiefer, Da Yan, Miranda Zhang, and Ethan Perez. Towards understanding sycophancy in language models. In *International Conference on Learning Representations*, 2024.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

Balachandra Devarangadi Sunil, Isheeta Sinha, Piyush Maheshwari, Shantanu Todmal, Shreyan Mallik, and Shuchi Mishra. Memory poisoning attack and defense on memory based LLM-agents. *arXiv:2601.05504*, 2026.

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 (Volume 1: Long Papers), pp. 16022–16076, 2024.

Gladys Tyen, Hassan Mansoor, Victor Carbune, Peter Chen, and Tony Mak. LLMs cannot find reasoning errors, but can correct them given the error location. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 13894–13908, 2024. doi: 10.18653/v1/2024.findings-acl.826.

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. *arXiv:2305.16291*, 2023.

Hanlin Wang, Chak Tou Leong, Jian Wang, and Wenjie Li. Seeing isn't believing: Mitigating belief inertia via active intervention in embodied agents. *arXiv:2604.17252*, 2026.

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

Guibin Zhang, Junhao Wang, Junjie Chen, Wangchunshu Zhou, Kun Wang, and Shuicheng Yan. AgenTracer: Who is inducing failure in the LLM agentic systems? *arXiv:2509.03312*, 2025a.

Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A. Smith. How language model hallucinations can snowball. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 59670–59684, 2024.

Shaokun Zhang, Ming Yin, Jieyu Zhang, Jiale Liu, Zhiguang Han, Jingyang Zhang, Beibin Li, Chi Wang, Huazheng Wang, Yiran Chen, and Qingyun Wu. Which agent causes task failures and when? on automated failure attribution of LLM multi-agent systems. *arXiv:2505.00212*, 2025b.

Kunlun Zhu, Zijia Liu, Bingxuan Li, Muxin Tian, Yingxuan Yang, Jiaxun Zhang, Pengrui Han, Qipeng Xie, Fuyang Cui, Weijia Zhang, Xiaoteng Ma, Xiaodong Yu, Gowtham Ramesh, Jialian Wu, Zicheng Liu, Pan Lu, James Zou, and Jiaxuan You. Where LLM agents fail and how they can learn from failures. *arXiv:2509.25370*, 2025.

Wei Zou, Mingwen Dong, Miguel Romero Calvo, Shuaichen Chang, Jiang Guo, Dongkyu Lee, Xing Niu, Xiaofei Ma, Yanjun Qi, and Jiarong Jiang. Poison once, exploit forever: Environment-injected memory poisoning attacks on web agents. *arXiv:2604.02623*, 2026.

A LIMITATIONS

Behavioral evidence. NLI is identified from observable trajectories: expressed interpretations, plans, tool calls, environment feedback, and artifacts. This supports a reproducible behavioral characterization but does not by itself identify the internal mechanism that produces lock-in. Written claims, latent model state, path dependence, and residual environment changes may all contribute. The mechanisms discussed in the appendix should therefore be read as hypotheses for future representation-level and controlled intervention studies.

Evaluation coverage. The study spans four benchmarks and five models, but the matched correction cohort and the Nexus evaluation remain finite. Recovery estimates may change with lock-in severity, interaction horizon, agent architecture, or the ambiguity of corrective evidence. The Nexus results establish the effect of the integrated supervision loop in a persistent, tool-mediated setting; broader estimates across repeated runs, task families, and alternative state-supervision implementations, component ablations, and fresh-run baseline comparisons remain open. The models are accessed through hosted APIs; the reported task/model repetitions are not a controlled multi-seed study. Provider-side sampling and model revisions can affect repeated execution.

State observability and deployment cost. Nexus assumes that task-relevant evidence and unresolved requirements can be represented in a structured external state. This assumption is strongest in tool-mediated environments with explicit feedback and weaker in subjective or partially observable tasks. The current implementation also adds a median 683 ms and approximately 6,295 component tokens per complete trigger, with 30.7% measured amortized token overhead. Reducing this cost and learning task-state schemas with less manual engineering are important deployment directions.

B DATA, ANNOTATION, AND REPRODUCIBILITY

B.1 TRAJECTORY INVENTORY

The corpus contains 6,035 trajectories across four benchmarks and five models. The final audit preserves trajectory IDs, task and model assignments, trajectory contents, and NLI labels. Table 4 reports every benchmark–model cell.

Table 4: **Trajectory inventory.** Counts sum to 6,035.

Benchmark	Claude	GPT	Gemini	Kimi	Qwen	Total
AppWorld	580	585	585	585	585	2,920
CL-bench-Life	396	388	401	404	393	1,982
ClawMark	108	108	132	108	116	572
WildClawBench	119	112	110	110	110	561
Total	**1,203**	**1,193**	**1,228**	**1,207**	**1,204**	**6,035**

B.2 BENCHMARK ROLES IN THE STUDY

The four benchmarks are not pooled as interchangeable tasks. They expose different portions of the trajectory and therefore support different parts of the analysis. Table 5 records this division explicitly.

B.3 OUTCOME-SCORE COVERAGE AND AGGREGATION

Official continuous task scores are available for AppWorld, ClawMark, and WildClawBench. CL-bench-Life is not combined with them because its answer-level outcome is not on the same continuous scale. Table 6 separates the trajectory inventory, definitive binary decisions, and final official-score population. Of 4,053 behavioral-benchmark trajectories, 836 do not receive a definitive binary decision for this comparison. Six additional WildClawBench attempts are excluded before scoring: four superseded attempts and two explicit 1,200-second execution timeouts. The resulting 3,211 rows equal the 520 confirmed-NLI and 2,691 non-NLI rows in Table 8.

Table 5: **Benchmark roles and observable evidence.**

Benchmark	Primary role	Observable trace	Outcome use
CL-bench-Life	Formation-stage analysis	Prompt, answer, stated interpretation, counterevidence, and substantive follow-up when present	Separate answer-level analysis
AppWorld	Behavioral propagation	Plans, API calls, application state, feedback, and final task state	Continuous native score
ClawMark	Persistent tool-mediated execution	Plans, tool calls, runtime events, artifacts, and task completion	Continuous native score; Nexus
WildClawBench	Behavioral propagation across open-ended tasks	Plans, tool calls, environment feedback, and terminal state	Continuous native score

Table 6: **Population flow for continuous task outcomes.** The Outside column denotes inventory rows without a definitive binary decision for this comparison; exclusions are documented non-final or execution-invalid runs.

Benchmark	Inventory	Binary	Outside	Scored
AppWorld	2,920	2,210	710	2,210
ClawMark	572	494	78	494
WildClawBench	561	513	48	507
Total	**4,053**	**3,217**	**836**	**3,211**

AppWorld task scores are evaluated from preserved final states with the official evaluator, independently of the trajectory labels. Table 7 reports included official-score rows and unresolved missing outputs for every benchmark–model cell. Its denominator is the final binary outcome-analysis population rather than the full trajectory inventory; zero denotes no unresolved evaluator output within that population.

Table 7: **Official-score coverage by benchmark and model.** Each cell is included rows / unresolved missing outputs. N/A denotes a benchmark without a comparable continuous official score.

Benchmark	Claude	GPT	Gemini	Kimi	Qwen	Total
AppWorld	464/0	490/0	462/0	419/0	375/0	2,210/0
CL-bench-Life	N/A	N/A	N/A	N/A	N/A	N/A
ClawMark	107/0	71/0	114/0	96/0	106/0	494/0
WildClawBench	114/0	102/0	91/0	108/0	92/0	507/0

Table 8: **Official task outcomes with complete score coverage.** Each cell reports trajectories / mean official score.

Benchmark	Conf-NLI	Non-NLI
AppWorld	285 / 0.398	1,925 / 0.763
ClawMark	139 / 0.128	355 / 0.097
WildClawBench	96 / 0.075	411 / 0.060
Continuous micro	**520 / 0.266**	**2,691 / 0.568**

For benchmark b and group g, the mean is the sum of included official scores divided by the number of included trajectories. The continuous micro average is

$$\mathrm{Micro}(g) = \frac{\sum_b n_{b,g}\,\bar{s}_{b,g}}{\sum_b n_{b,g}}. \tag{3}$$

Pooling the included trajectories gives 0.266 for confirmed NLI ($n = 520$) and 0.568 for non-NLI ($n = 2{,}691$). All aggregates are computed before rounding.

B.4 Matched Outcome Analysis

Groups are formed by exact task ID and model. We retain groups containing both confirmed-NLI and non-NLI runs and compute the two within-group means. Table 9 weights those means by the number of trajectories of the corresponding label in each group. Thirty-eight AppWorld groups and four ClawMark groups meet this criterion; no WildClawBench group in the final population contains both labels under the exact key.

Table 9: **Same-task, same-model outcome comparison.** Means are trajectory-weighted within the eligible groups; NLI-lower counts give one vote per group. AppWorld contributes 44 NLI / 49 non-NLI trajectories and ClawMark contributes 12 / 8, totaling 56 / 57.

Benchmark	Groups	NLI	Non	NLI lower
AppWorld	38	0.220	0.673	32/38
ClawMark	4	0.157	0.316	2/4
All	**42**	**0.207**	**0.623**	**34/42**

B.5 Solvability and Certifier Removal

At threshold t, a task is retained when at least one scored run reaches t. We remove one maximum-scoring run that certifies each retained task, resolving score ties by trajectory ID, and recompute both group means. The completed AppWorld official-score population is used throughout.

Table 10: **Solvability analysis after certifier removal.** Cells report N / mean for confirmed NLI (C) and non-NLI (N). The highlighted threshold is used in the main figure.

t	App C	App N	Wild C	Wild N	Claw C	Claw N
.5	**248/.352**	**1,752/.747**	**34/.079**	**106/.096**	**82/.149**	**185/.100**
.6	243/.355	1,731/.751	27/.086	68/.147	72/.152	169/.101
.7	237/.361	1,715/.755	27/.086	60/.166	35/.143	84/.110
.8	232/.364	1,698/.758	25/.060	56/.178	16/.131	33/.077
.9	217/.367	1,661/.767	25/.060	49/.187	12/.144	11/.135

B.6 ClawMark Execution Validity

The ClawMark execution-validity analysis uses structured runtime metadata. It excludes explicit DOM or page unavailability, authentication errors, framework exceptions, and session-lock failures. The predicate does not read task score or NLI label. Ordinary task failures and low scores after substantive execution remain included; solvability and certifier selection are recomputed after the metadata exclusions.

Table 11: **Execution-valid ClawMark after certifier removal.** The highlighted threshold is used in the main figure.

t	Conf. N / mean	Non N / mean
.5	**78/.152**	**85/.208**
.6	70/.150	77/.210
.7	33/.139	39/.218
.8	16/.131	15/.135
.9	12/.144	3/.318

At $t = .5$, the execution-valid population contains 78 confirmed-NLI and 85 non-NLI trajectories. The runtime-metadata exclusions are concentrated in the non-NLI group. This imbalance explains why execution validity materially changes the comparison population. The raw, solvability-selected, and execution-valid estimates are retained separately rather than substituted for one another. Appendix E details their interpretation.

B.7 Damage Distribution Construction

For the cross-benchmark damage summaries, each of the three behavioral benchmarks is normalized before averaging, so benchmark size does not determine the distribution. Completed official task scores use bins 0, $(0, .25]$, $(.25, .5]$, and $(.5, 1]$. Rubric-defined damage uses low, medium, and high levels and is also retained separately for AppWorld, ClawMark, WildClawBench, and CL-bench-Life.

Table 12: **Rubric-defined damage by benchmark and label.** Counts use the characterization population, including trajectories outside the final native-score comparison. Damage is annotated from the same trajectory as NLI and describes its observed consequences; it is not an independent task-outcome test.

Benchmark	Label	Low	Medium	High	Unlabeled
AppWorld	NLI	2	15	268	0
	Non-NLI	1,834	89	0	2
ClawMark	NLI	3	46	90	0
	Non-NLI	312	33	10	0
WildClawBench	NLI	2	28	69	0
	Non-NLI	381	17	16	0

B.8 Detection and Annotation Protocol

Detection Pipeline. Every trajectory passes through a two-step screening pipeline under the five-dimension rubric defined in Table 14. Step 1 uses a permissive screen: any dimension at least 1 flags a candidate for full evaluation. Step 2 applies the final rule used throughout the paper. A case is confirmed when the task is verifiably solvable, at least three of five dimensions are at least 2, Narrative Formation is at least 2, at least one of Behavioral Non-update or Persistence is at least 2, and no explicit environmental confound is found. For CL-bench-Life formation-stage cases, the three observable front dimensions must each be at least 2; behavioral dimensions are scored only when a substantive post-answer turn exists. Damage, confidence, primary error type, solvability, and manifestation are auxiliary fields and do not supply the downstream task score to the detector. Records without a definitive Step 2 decision remain in the inventory but are outside binary outcome comparisons. The annotation's solvability field assesses whether the task can be completed given its requirements and available environment; the outcome analysis separately operationalizes demonstrated solvability using an observed score threshold (Appendix B.5).

Annotation Protocol. Each trajectory is independently scored in two DeepSeek V4 Pro calls. Disagreements are resolved by a third call to the same model with access to both initial scores and rationales. The five dimensions are scored on a 0–3 ordinal scale: 0 = absent, 1 = weak or ambiguous, 2 = moderate, 3 = strong or definitive. For CL-bench-Life, which lacks multi-turn action traces, NLI judgments rely primarily on Narrative Formation, State Adoption, and Counterevidence; the Behavioral Non-update dimension is scored only when the trajectory contains at least one substantive turn after the initial answer.

Table 13: **Rubric profiles among 523 confirmed behavioral-stage cases.** Each row sums to 523. Confirmation uses the joint rule above, allowing different dimension profiles rather than requiring all five scores to exceed the same threshold.

Dimension	0	1	2	3
Narrative Formation	0	0	25	498
State Adoption	0	0	25	498
Counterevidence	7	12	42	462
Behavioral Non-update	0	2	19	502
Persistence	0	6	21	496

Prompts and Step Limits. The hosted API calls do not explicitly set a random seed or temperature; sampling therefore follows provider-side defaults. Independent annotation calls use the same

Table 14: **Narrative Lock-in annotation dimensions.** All dimensions use the shared ordinal anchors 0 (absent), 1 (weak or ambiguous), 2 (moderate), and 3 (strong or definitive).

Dimension	Observable criterion
Narrative Formation	A coherent but incorrect account is formed early from available evidence.
State Adoption	The account enters the agent's diagnosis, plan, tool choice, action, or artifact.
Counterevidence	Contrary evidence becomes available, together with an observable response or failure to engage with it.
Behavioral Non-update	Subsequent plans or actions remain organized by the earlier account after counterevidence appears.
Persistence	The same account continues, recurs, or escalates across later turns and outputs.

underlying model. The rubric is presented in the evaluator's system prompt together with dimension definitions, scoring anchors, and representative examples. Each evaluation call is limited to a single trajectory—the evaluator receives the full conversation history, tool calls, and environment observations. Step limits for agent benchmarks follow the default settings of each benchmark environment (AppWorld: 100 steps; ClawMark: as per task definition; WildClawBench: up to episode termination). Stored trajectories remain distinct throughout the characterization and intervention studies; repeated runs are not collapsed.

Cross-Dimension Consistency Checks. We repeat detection using only Narrative Formation, State Adoption, and Counterevidence. The front-three screen marks a trajectory positive when at least two of these dimensions are at least 2 and none is 0. Behavioral Non-update, Persistence, and task score are held out. Table 15 shows that the narrative/state/evidence structure retains most confirmed cases and co-occurs with the two behavioral dimensions. All dimensions come from full-trajectory annotations: withholding dimensions from the selection rule is not a new prefix-only annotation or a temporal prediction experiment.

Table 15: **Cross-dimension consistency and blinded label agreement.** Behavioral dimensions and score are omitted from the front-three selection rule, but the original annotations use full trajectories.

Check	Population / comparison	Result
Confirmed-label retention	Original confirmed cases recovered by the front-three-only screen	99.0%
Behavioral co-occurrence	Both omitted dimensions at least 2, conditional on front-three positive / negative	99.0% / 4.8%
Outcome sensitivity	Mean downstream score proxy, front-three positive / negative	0.153 / 0.467
Blinded validation	25 confirmed and 25 non-NLI trajectories, no original labels, rubric values, scores, or rationales	49 / 50 agreement

Inter-Annotator Agreement. Across the full characterization annotation set, Step 2 inter-annotator agreement reaches Cohen's $\kappa = 0.73$ on the binary lock-in confirmation decision and a mean Spearman's $\rho = 0.68$ across the five rubric dimensions. Disagreement is concentrated in ambiguous cases with medium confidence, which are flagged for adjudication. We additionally sample 25 confirmed-NLI and 25 non-NLI trajectories across all four benchmarks without score-based selection. A coauthor who did not participate in the original annotation receives only the task and standardized trajectory, without the original label, rubric values, task score, or judge rationale. The blinded labels agree on 49/50 trajectories; the remaining trajectory is marked uncertain. This balanced coauthor check tests agreement on a small selected set, not population-wide annotation accuracy.

B.9 Matched Intervention Protocol

Case Selection. The intervention cohort is selected from adjudicated screening and manual trajectory review under the same five-dimension rubric. The primary matched analysis retains a case if and

only if it contains one usable execution under each of the six formal conditions: self-check, meta-reflection, and correction levels L1–L4. Probe records and auxiliary state-trigger records are not treated as intervention conditions, and incomplete cases are excluded rather than imputed. This deterministic rule yields 728 complete cases—222 AppWorld, 297 CL-bench-Life, 121 ClawMark, and 88 WildClawBench—and 4,368 intervention executions.

Table 16: **Composition of the complete intervention cohort.** Counts are cases, each with all six conditions; zero denotes no selected case.

Benchmark	Claude	GPT	Gemini	Kimi	Qwen	Total
CL-bench-Life	0	114	50	133	0	297
AppWorld	29	5	4	75	109	222
ClawMark	25	24	27	19	26	121
WildClawBench	8	25	23	15	17	88
Total	62	168	104	242	152	728

Table 17: **Formation-stage and behavioral-task recovery.** All conditions use the same 297 formation-stage and 431 behavioral cases. Wilson intervals summarize binomial uncertainty within the selected cohort; they do not account for task clustering, judge error, or variation across fresh API runs.

Condition	Formation / 297	Behavioral / 431	All: rate [95% interval]
Self-check	58	20	10.7 [8.7, 13.2]
Meta-reflection	89	63	20.9 [18.1, 24.0]
L1: Weak recheck	91	45	18.7 [16.0, 21.7]
L2: Claim only	120	63	25.1 [22.1, 28.4]
L3: Concrete evidence	146	119	36.4 [33.0, 40.0]
L4: Explicit path	120	86	28.3 [25.1, 31.7]

The intervention pool includes screened and individually reviewed candidates; it is not restricted to the final prevalence-positive subset. In particular, the 297 CL-bench-Life cases and the 234 conservative formation-stage positives have different selection rules. The six-arm results describe responsiveness within this selected pool, rather than population prevalence.

Experimental Design. Each case serves as its own control. For every case, we start from the original trajectory up to the first identified lock-in evidence point and run the six conditions independently from the same task context and environment state. The two trigger conditions ask the agent to inspect its current reasoning without external corrective content. The four correction conditions supply increasingly specific information: a weak recheck request, identification of the erroneous claim, concrete counterevidence, or an explicit corrected path.

Two-Stage Post-Intervention Procedure. Intervention does not end at a single response. Each trajectory is evaluated in two stages. **Stage A** captures the model's immediate post-intervention response: revised answer, revised diagnosis, revised plan, or explicit refusal/non-update. **Stage B** requires the model to commit to one additional step immediately after Stage A. For behavior-oriented benchmarks (AppWorld, ClawMark, WildClawBench), this continuation executes a concrete next tool call or action and records its feedback. For CL-bench-Life, the continuation step provides a revised answer and the next concrete step with justification. Stage B is included to distinguish verbal compliance from genuine trajectory update. It supplies behavioral-update evidence; task recovery is determined from the full available continuation and final artifact, not from the stated next-step intention alone.

Intervention Prompts. The exact prompt text for each of the six conditions is as follows:

- **self_check**: "Review the trajectory above. Identify anything inappropriate, unsupported, or inconsistent in your current approach. Do not assume your current interpretation is correct; explicitly point to what may be wrong."

- **meta_reflection**: "Pause and reflect: are you confident in your current approach? What evidence would disprove it? What assumptions are you relying on, and which of them may be unsupported by the task environment?"
- **L1 – Weak Recheck**: "Double-check your current approach. Return to the available evidence and verify each key assumption before continuing."
- **L2 – Claim Only**: "Your current core claim is likely wrong or inconsistent with the available evidence. Abandon your current assumption and re-evaluate the task."
- **L3 – Concrete Evidence**: "Your current narrative is contradicted by the following evidence: `{evidence}`. Re-evaluate the task based on this counterevidence and change your subsequent reasoning accordingly." The `{evidence}` placeholder is populated with specific contradictory evidence extracted from the task environment at the lock-in point.
- **L4 – Explicit Solution Path**: "Use the following corrected path: `{solution}`. Counterevidence: `{evidence}`. Abandon the original path and continue using the corrected path above. Your next step must follow this new path concretely."

Evaluation and Recovery Definition. DeepSeek V4 Pro receives the source trajectory, intervention, continuation, and available final artifact. We record pre-intervention severity, post-intervention uptake (narrative, action, goal, verification, and artifact change), and post-intervention outcome. Official benchmark output is used when the intervention run reaches its scorer; otherwise outcome improvement is judged against the source trajectory from the final available continuation. A case is *strictly recovered* only when all three machine-readable conditions hold: `lockin_persistence_0_3 = 0`, `outcome_improved = yes`, and rubric `outcome_damage = 0`. Verbal acknowledgment, answer rewriting without trajectory change, and plans not followed by a new step are recorded as partial or surface responses.

The auxiliary 68.5% statistic used a separate binary judge that saw only the immediate post-intervention response and asked whether it appeared corrected. It is excluded from every strict-recovery calculation. Because this auxiliary evaluation is separate, its difference from the six-arm rates is not a within-case estimate of the acknowledgment–recovery gap.

Table 18: **Strict recovery on the 728-case complete matched panel.** Every arm has the same cases and denominator.

Condition	Recovered / N	Rate
Self-check	78 / 728	10.7%
Meta-reflection	152 / 728	20.9%
L1: Weak Recheck	136 / 728	18.7%
L2: Claim Only	183 / 728	25.1%
L3: Concrete evidence	**265 / 728**	**36.4%**
L4: Explicit Solution Path	206 / 728	28.3%

Table 19: **Post-intervention response levels.** Later levels require evidence beyond the earlier response.

Level	Observable unit	Criterion
Verbal acknowledgment	Immediate response	Accepts or restates the correction
Plan update	Diagnosis and strategy	Replaces the locked premise with a compatible plan
Behavioral update	Next tool call or action	Executes a step that follows the revised state
Task recovery	Downstream trajectory	Preserves the revision and improves task outcome

B.10 Trajectory-Cleared Reset

The reset stress set contains 40 cases, balanced as ten per benchmark. Of these, 34 were unrecovered after L4 explicit-path correction, 36 had maximum lock-in persistence, and 33 had maximum

outcome damage. The stress set therefore tests trajectory clearing after strong lock-in has already been observed.

Both conditions open a new session without loading the original trajectory. The reset condition provides a task reference and a compact failure summary; the guided condition supplies the original task packet and structured recovery instructions without the earlier failure summary. Thus the comparison evaluates two reinitialization strategies that jointly vary task reconstruction and guidance. Unevaluable cases are counted as unrecovered.

Table 20: **Reset recovery on the same stress set.**

Condition	Recovered / N	Rate
Trajectory-cleared reset	8 / 40	20.0%
Reset + corrective guidance	**22 / 40**	**55.0%**

C Nexus Implementation and Evaluation

The Observer writes an evidence ledger from raw environment events. WorldMap maps that ledger into a task-specific schema containing goals, constraints, completed actions, unresolved conflicts, and artifact state. Mentor receives the state capsule and the agent's proposed claim or action, then issues a correction on claim–evidence mismatch or goal drift. The agent cannot write directly to WorldMap.

Observer records preserve the source event and the corresponding action or tool feedback. WorldMap updates are cumulative: supported facts and completed actions are retained, conflicts remain unresolved until supported evidence changes their status, and open task requirements remain visible in the state capsule. Mentor is evaluated only on this capsule together with the proposed claim or action. When neither an evidence conflict nor goal drift is detected, it returns no intervention; otherwise it supplies the conflicting evidence and the unresolved requirement that should govern the next action. Task termination and scoring remain those of the underlying benchmark.

Formalization. At step t, Observer maps the environment event e_t, agent action a_t, and feedback f_t to an evidence record:

$$\ell_t = O(e_t, a_t, f_t). \tag{4}$$

WorldMap updates the structured task state w_t, which contains goals, constraints, completed actions, unresolved conflicts, and artifact status:

$$w_t = U(w_{t-1}, \ell_t). \tag{5}$$

Mentor compares the agent's current claim or proposed action q_t with this evidence-grounded state and returns either no intervention or a compact correction:

$$m_t = M(w_t, q_t, a_t). \tag{6}$$

The resulting loop is Observe $\rightarrow$ Maintain $\rightarrow$ Compare $\rightarrow$ Intervene, grounded in observable execution events and maintained outside the agent's self-reported narrative.

Evaluation populations. The formation-stage evaluation contains 300 CL-bench-Life cases: 115 from GPT-5.4, 134 from Kimi-K2.5, and 51 from Gemini-3-Flash. Nexus improves the evaluated outcome in 37.0% of these cases, showing that evidence-grounded supervision can redirect an account before it becomes a long behavioral trajectory. The behavioral-stage evaluation contains 75 ClawMark cases, formed by crossing 15 tasks with five models. Matching by task ID and model yields 29 ClawMark cases with all six matched correction records; Table 21 uses only this overlap. The 75-case win/tie/loss evaluation remains a separate full-task comparison.

The broader ClawMark evaluation contains 75 cases. Nexus records 44 wins, 15 ties, and 16 losses against the original trajectory, with mean score gain 0.111. PM tasks account for 22 wins and 2 losses, with mean gain 0.203. This full-task comparison is reported separately from judged improvement under correction. PM denotes project-management tasks: 30 of the 75 cases, with 22 wins, 6 ties, and 2 losses. The remaining cases cover journalism (15), clinical assistance (10), and content operations,

Table 21: **Outcome improvement on 29 overlapping ClawMark task–model pairs.** Correction rows use judged outcome improvement; Nexus uses an increase in native task score over the original run. The protocols have different starting points and do not form a strict-recovery comparison.

Method	Improved / N	Rate
Self-check	8 / 29	27.6%
Meta-reflection	10 / 29	34.5%
L1: Weak Recheck	7 / 29	24.1%
L2: Claim Only	6 / 29	20.7%
L3: Concrete Evidence	11 / 29	37.9%
L4: Explicit Path	10 / 29	34.5%
Nexus	**19 / 29**	**65.5%**

e-commerce, insurance, and executive assistance (5 each). Claude has six small wins but two larger losses: its score changes on journalism tasks 3 and 8 are -0.540 and -1.000, respectively. Their magnitudes explain why six wins and two losses coexist with a negative mean gain; win counts and mean gains summarize different aspects of performance.

The runtime audit contains 8,134 agent turns and 4,620 complete Nexus trigger rows. Provider usage is used for agent calls when present; component token counts are derived from stored Observer, WorldMap, and Mentor artifacts. Added latency uses complete component event timestamps.

Table 22: **Nexus outcome and runtime accounting.**

Outcome measure	Value	Runtime measure	Value
Full-task wins / ties / losses	44 / 15 / 16	Observer tokens / trigger	176
Mean full-task gain	0.111	WorldMap tokens / trigger	3,865
PM wins / losses	22 / 2	Mentor tokens / trigger	2,255
PM mean gain	0.203	Total tokens / trigger	6,295
Score gains in overlap	19 / 29	Amortized overhead	30.7%
Trigger rate	56.8%	Added latency, median / P90	683 / 1,039 ms

Token overhead is the mean component-token total per trigger multiplied by the trigger rate, divided by mean baseline-agent tokens per turn. Latency is the sum of recorded Observer, WorldMap, and Mentor durations on complete trigger rows; it is a component-duration statistic, not end-to-end task latency. Displayed component-token means are rounded separately.

D Discussion

From narrative formation to behavioral lock-in. The results describe a progression rather than a single response error. An early interpretation first appears as a claim, then enters plans, tool choices, and artifacts, and finally becomes the context from which later evidence is interpreted. This progression connects the five annotation dimensions, the lower outcomes in matched task–model groups, the verbal–behavioral correction gap, and the increasing difficulty of repairing persistent execution.

Rechecking and counterevidence address different parts of the problem. A generic recheck asks the agent to inspect its current path but does not identify which premise, evidence source, or task requirement should be reopened. It can therefore produce a local verification step without reconstructing the state on which the broader plan depends. Concrete counterevidence identifies a conflict more directly and achieves the highest recovery among the six matched conditions. Even then, the agent may acknowledge the evidence or patch the immediate error while preserving the rest of the established account. The relevant distinction is not simply whether corrective text is present, but whether the correction changes the state that organizes subsequent action.

Execution makes the trajectory path-dependent. Once an erroneous account enters execution, it is preserved in several places: the conversation context, workspace modifications, intermediate

Table 23: **Empirical settings and intervention scope in closely related studies.** Rows compare how the closest studies instantiate the problem, what part of the interaction they follow, and how they evaluate correction or mitigation. The phenomena and mechanisms may overlap within a single trajectory.

Study	Empirical setting	Execution and state	Primary analysis unit	Correction or mitigation
LLMs Get Lost in Multi-Turn Conversation (Laban et al., 2026)	Simulated conversations over six generation tasks, with a complete instruction divided across turns	Text interaction; successive solutions are conditioned on the accumulated conversation	Conversation-level performance, aptitude, and unreliability under full, concatenated, and sharded instructions	Information-presentation and recap controls; recovery is evaluated through generated task solutions
Drift No More? (Dongre et al., 2025)	Synthetic long-horizon rewriting and user–agent simulations in τ-Bench	Evolving conversational context; outputs are compared with a goal-consistent reference model	Turn-wise predictive-distribution divergence and its temporal equilibrium	Goal reminders; mitigation is measured as reduced divergence
Seeing Isn't Believing (Wang et al., 2026)	Embodied tasks in ALFWorld, VirtualHome, and ScienceWorld	Agents act in an environment, observe feedback, and maintain explicit textual belief states	Belief–observation mismatch, belief updating, actions, and task success	Estimate–Verify–Update through prompting or training; mitigation is measured by task success
Narrative Lock-in (ours)	One formation-stage text benchmark and three behavioral agent benchmarks with API or tool-mediated execution	Conversation, tool feedback, mutable environment state, workspace changes, and persistent artifacts	Full trajectory from early interpretation through state adoption, verification, action, artifact production, and task outcome	Six graded corrections, trajectory-cleared reset, and external state supervision; recovery requires downstream behavioral and task change

artifacts, and completed actions. Later rechecks and contradictory observations can be absorbed as local signals within that accumulated structure. Environment changes created by earlier actions further constrain the available continuation. The reset study supports this interpretation: clearing the trajectory helps some severe cases, while adding corrective guidance helps substantially more. Recovery thus requires both a clean state representation and information about what must be revised.

What the evidence establishes. NLI labels are constructed from trajectory structure, while task outcomes are supplied separately by benchmark-native evaluators. The front-three screen and blinded validation further show that the construct is not defined by its downstream score. These checks support NLI as a stable behavioral pattern associated with later non-update and lower completion. They do not uniquely identify an internal causal mechanism. Written claims may directly shape later generation, reflect a latent model state, or interact with residual environment state and ordinary planning path dependence.

Implications for agent design. The findings suggest that fluent local progress should not be treated as evidence of correct global state maintenance. Reliable correction requires an auditable record of observations, an explicit representation of goals and unresolved requirements, and a check that revised state changes later action. Nexus instantiates this implication with separated evidence extraction, task-state maintenance, and mismatch-triggered correction. Autoregressive continuation, preference for coherent responses, and the cost of reconstructing a long task state are plausible contributors to the observed pattern, but testing those mechanisms requires controlled representation-level and training-time studies.

E READING THE CONDITIONAL OUTCOME COMPARISONS

Different questions require different comparison populations. The raw comparison describes outcomes among trajectories with definitive labels. Solvability selection asks whether the association remains on tasks that at least one run demonstrably completes to threshold t. Execution-validity selection asks the same question after removing explicitly interrupted or inaccessible executions. Exact task–model matching compares repeated runs within a common task and model. These operations address different sources of heterogeneity; their estimates need not have the same sign.

Table 24: **Raw and conditional score differences.** Each entry is mean NLI score minus mean non-NLI score. Solvability-selected rows use $t = .5$ and remove the qualifying maximum-score run. A dash denotes an analysis not reported for that benchmark, rather than a zero difference.

Population	AppWorld	ClawMark	WildClawBench
Definitive-label population	$-.365$	$+.031$	$+.015$
Solvability-selected	$-.395$	$+.049$	$-.017$
Execution-valid, solvability-selected	—	$-.056$	—
Exact task–model groups, trajectory-weighted	$-.453$	$-.159$	—

Why a raw association can reverse. In the raw ClawMark and WildClawBench populations, confirmed-NLI trajectories have slightly higher mean scores (Table 8). Low-scoring non-NLI trajectories include failures that do not exhibit the annotated structure. Conditioning changes their relative representation: WildClawBench changes sign after solvability selection, while ClawMark remains positive until the execution-validity restriction is applied. For ClawMark, the metadata predicate checks page/DOM availability, authentication, framework exceptions, and session locks without reading the NLI label or score. Ordinary unsuccessful execution remains eligible. Solvability and its qualifying run are then recomputed on the retained population. Table 24 preserves all stages so that the filtered result is read alongside, rather than in place of, the raw one. The sign change highlights how task composition and execution validity shape the observed association.

What matching adds. For eligible group j and label g, let n_{jg} be its trajectory count and $\bar{s}_{jg}$ its mean score. The reported matched-population summary is $\sum_j n_{jg}\bar{s}_{jg} / \sum_j n_{jg}$, computed separately for each label. It yields 0.207 and 0.623 from 56 NLI and 57 non-NLI trajectories. Giving each of the 42 groups equal weight instead yields 0.202 and 0.615. The sign count, 34/42, already gives each group one vote. AppWorld contributes 38 groups (32 lower), ClawMark four (two lower), and WildClawBench none under the exact task–model key. Thus the broad matched association is primarily supported by AppWorld, with additional coverage from ClawMark.

Separate annotation structure from outcome evidence. The rubric identifies a trajectory pattern; native completion scores answer whether tasks succeed. Damage labels and front-three/back-two co-occurrence describe the pattern using the same full-trajectory annotation and should be read as internal consistency evidence. The 836 behavioral trajectories outside definitive binary labels remain part of the inventory, not an inferred non-NLI group. The case analyses below make the temporal sequence and its task consequences explicit.

Recorded manifestations. Among the 523 confirmed behavioral-stage trajectories, the auxiliary multi-label annotations record acknowledgment without update in 474, a completion narrative in 310, repeated paths in 278, self-justification in 278, surrogate goals in 269, artifact non-update in 173, and branch rewriting in 8. These tags overlap and describe complementary manifestations within the same annotation process.

The field-level counts make the readout distinctions inspectable. For example, L3 and L4 each have 352 judged outcome improvements, but 265 and 206 satisfy the stricter conjunction, making the additional requirements for sustained recovery visible in the same cohort.

Table 25: **Post-intervention changes on the same 728 cases.** Counts are positive judgments for each field, not nested stages of a funnel. A narrative change is not identical to verbal acknowledgment, and fields that are not applicable to an answer-level task are not counted as positive. Strict recovery uses the three-condition rule in Appendix B.9.

Condition	Narrative	Action	Goal	Verify	Artifact	Outcome	Strict
Self-check	281	145	272	210	151	208	78
Meta-reflection	414	264	402	339	138	297	152
L1	349	193	325	296	150	254	136
L2	426	206	412	261	143	279	183
L3	451	344	448	361	199	352	265
L4	420	301	426	391	323	352	206

F LONGITUDINAL CASE STUDIES

Figure 7 retains the original qualitative examples and adds a logged execution and the author's figure-revision observation. We use two recorded AppWorld executions to show how a local observation can leave the operative task state unchanged. Step numbers refer to the trajectory logger. Summaries preserve the action order; personal identifiers and access tokens are omitted. The examples connect the aggregate observations to concrete actions and task requirements.



Figure 7: **Observed interactions and a recorded execution.** The upper four panels retain the author's original examples. The lower panels add an AppWorld execution (`6b6ca61_3`, Kimi-K2.5, steps 11–13) and the author's figure-revision observation. The added panels summarize the request, local response, and subsequent action; the benchmark analysis uses the trajectory inventory in Table 4.

F.1 ACKNOWLEDGING A CONSTRAINT WHILE EXECUTING A SUBSTITUTE TASK

Case: AppWorld `6b6ca61_3`, Kimi-K2.5, test-normal. The task routes payments by account existence: pay people with Venmo accounts through Venmo; otherwise create a Splitwise expense and attach the receipt. The original run records 9 passed and 10 failed native checks.

Table 26 locates the diagnostic transition between recalling the account-existence rule and continuing the fallback, not merely encountering insufficient balance. This case also separates local competence from global revision: the agent repairs an individual API failure while retaining the task-level mismatch. Successful bookkeeping leaves the requested payment outcome unresolved.

Table 26: **From a real obstacle to a substitute task.** Selected evidence from the same execution, in chronological order.

Location	Observed event	Relation to operative task state
Step 1	Correctly describes the Venmo/account-existence conditional.	The original constraint is initially represented.
Before step 11	Payment attempts encounter insufficient balance.	A real execution obstacle appears; the task's routing condition has not changed.
Step 11	Proposes Splitwise expenses for everyone as a fallback.	A payment obstacle is turned into a new task objective.
Step 11	Explicitly notes that insufficient balance does not change whether people have Venmo accounts, then returns to the fallback.	The contradictory constraint is acknowledged without replacing the plan.
Steps 11–12	Calls `splitwise__record_expense`; retries one local authorization failure.	Local repair makes progress within the substitute plan while its premise persists.
Step 13	Reports all nine debts recorded in Splitwise and declares completion.	The final account preserves the substitute objective.

F.2 Repeated Search Within a Fixed Hypothesis

Case: AppWorld `1a79e37_2`, Qwen-Plus, test-challenge. The agent seeks an email relevant to a shopping task. Its execution repeatedly calls `gmail__show_inbox_threads`, retaining sender/date restrictions and trying sender-address variants. The logger contains 46 empty-result observations; the final recorded action at step 50 is another inbox search.

An empty result only establishes that a particular query returned no match. It does not establish that the desired information is absent or that the current sender and date assumptions are correct. Repeatedly changing a local query parameter while retaining those assumptions illustrates the distinction between a new observation and a revised search state. Unlike the payment case, the evidence here is a repeated action pattern rather than an explicit verbal acknowledgment of the governing constraint. We therefore use it to illustrate narrow rechecking, without inferring an unexpressed belief from a tool call alone.

Connection to the intervention design. The two traces expose complementary intervention targets. A recheck prompt may request another search or repair of the last failed call; contradictory evidence must additionally reopen the assumption governing those actions. Nexus is motivated by preserving the original requirement and the accumulated evidence as separate state, so that local success is checked against global completion.

When a check becomes a substitute objective. The figure-revision vignette concerns an acceptance criterion, not a failed compiler. Passing a script can establish that an artifact was generated or that a local check succeeded; it does not establish that the requested visual changes were made throughout the figure. The repeated pattern of reporting completion, acknowledging the discrepancy, and again relying on the same narrow check illustrates how local verification can preserve a mistaken completion state. This observed interaction motivates the distinction between evidence about one component and evidence that the whole requested task is complete.

G Positioning the Observation and the Intervention

Intrinsic self-correction asks whether a model can identify and repair its own reasoning without external feedback (Huang et al., 2024). The broader self-correction literature distinguishes feedback reliability, correction ability, and evaluation design (Kamoi et al., 2024). In particular, error localization can be the limiting step even when a supplied error location enables correction (Tyen et al., 2024). Our graded prompts should be read against these distinctions: L2 warns that the current claim is wrong, L3 supplies contradictory evidence, and L4 supplies a path. Their benchmark-dependent ordering shows that the usefulness of corrective information depends on the setting.

Earlier work also shows that an initial hallucination can induce supporting errors (Zhang et al., 2024), while CRITIC uses external tools to verify and revise outputs (Gou et al., 2024). Magentic-One maintains task and progress ledgers to support multi-step execution and recovery (Fourney et al., 2024). Building on these closely related ideas, our empirical unit is the trajectory linking an adopted interpretation to actions, persistent task state, artifacts, and later correction. Nexus provides one concrete evidence-maintenance response to that observation. The present full-task comparison evaluates this integrated implementation in persistent, tool-mediated execution.