TOWARDS ROBUST LONG-LIVED AGENTS: BENCHMARKING EXPERIENCE LIFECYCLES IN STATEFUL ENVIRONMENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-lived agents must discover how an unfamiliar environment works and keep the resulting experience useful as conditions change. Existing benchmarks have expanded from historical information retrieval to interactive experience reuse and adaptation, yet maintaining experience under costly exploration and unannounced changes remains underexplored. We introduce ReScopeBench, a benchmark connecting experience acquisition, reuse, invalidation, and revision across 18 interactive environments. Our experience-first construction organizes 16-task streams around structural experience about resource relationships and rule experience about local operating conditions. Agents pursue varied goals under explicit exploration budgets, while silent changes invalidate selected dependencies and preserve others. Paired evaluations with and without retained experience measure benefits, disruption, and recovery through objective outcome scoring. Experiments with seven models show that experience acquired through interaction improves subsequent task performance, but silent environmental changes erode these gains and recovery varies substantially across models. Trace analyses further reveal a gap between retaining valid experience and applying it effectively. ReScopeBench makes maintaining actionable experience an explicit target for long-lived agent evaluation.

1 INTRODUCTION

Recent agents tackle computer-use tasks (Seed, 2025; Gonzalez-Pumariega et al., 2026), sustained creative workflows (Lin et al., 2026), and recursive self-improvement (Zhang et al., 2026a; Iacob et al., 2026). Sustained operation also requires agents to discover how an unfamiliar environment works. Task goals and tool interfaces need not reveal relevant relationships or local rules. Experience acquired through interaction can reduce repeated discovery across goals (Wang et al., 2024; 2025), yet become misleading when conditions change. Reliable operation therefore depends on a connected *experience lifecycle*: acquisition, reuse, invalidation, and revision.

Existing benchmarks address different parts of this lifecycle. LoCoMo and LongMemEval study reasoning over supplied histories, including factual updates (Maharana et al., 2024; Wu et al., 2025). LifelongAgentBench and MemoryArena extend evaluation to interactive experience reuse (Zheng et al., 2025; He et al., 2026); SkillFlow and SkillEvolBench examine procedural skill acquisition and reuse (Zhang et al., 2026b; Lei et al., 2026). AgingBench and EvoArena further investigate degradation, revision, and adaptation (Zhu et al., 2026; Xu et al., 2026). These advances go beyond retrieval, but differ in how agents acquire experience and encounter evidence that it needs updating.

A further challenge arises when discovering and rechecking experience is costly, and changes are unannounced. Reuse saves resources, but obsolete experience can misguide actions; exhaustive rechecking consumes resources needed for execution. Agents must decide what remains reliable, what requires investigation, and what to revise. Table 1 highlights this underexplored combination of *budgeted exploration*, *silent environment change*, and *autonomous revision*. Here, exploration consumes environment resources beyond model tokens; changes provide no boundary announcement; and revision requires agent-initiated investigation rather than a supplied update. An agent may acquire useful experience yet fail to recognize when its applicability conditions no longer hold.

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.

Table 1: Comparison of experience-evaluation settings. Marks indicate explicit benchmark coverage, not capabilities of accompanying methods. ReScopeBench combines interactive acquisition with budgeted exploration, silent change, and autonomous revision.

Benchmark	Acquisition	Budgeted Exploration	Transfer	Invalidation	Silent Env. Change	Autonomous Revision
LoCoMo (2024)	Provided	✗	✗	✗	✗	✗
LongMemEval (2025)	Provided	✗	✗	✓	✗	✗
MemBench (2025)	Provided	✗	✗	✓	✗	✗
MemoryAgentBench (2025)	Provided	✗	✓	✓	✗	✗
LifelongAgentBench (2025)	Interactive	✗	✓	✗	✗	✗
MemoryArena (2026)	Interactive	✗	✓	✗	✗	✗
SkillFlow (2026b)	Interactive	✗	✓	✗	✗	✗
SkillEvolBench (2026)	Interactive	✗	✓	✗	✗	✗
AgingBench (2026)	Mixed	✗	✓	✓	✗	✗
EvoArena (2026)	Mixed	✗	✓	✓	✗	✗
ReScopeBench	Interactive	✓	✓	✓	✓	✓

Conversely, detecting a conflict does not ensure that revised experience will guide subsequent decisions. Can agents maintain useful experience without being told when or how it should change?

We introduce ReScopeBench, which connects these requirements within experience-linked task streams. Agents receive goals and tool interfaces, but must discover relevant relationships and rules under limited interaction budgets. Different goals reuse shared dependencies before *silent changes* invalidate some while preserving others. Subsequent tasks test whether agents investigate conflicts, revise affected experience, and apply the revision beyond the task that first exposed a problem.

We distinguish *structural experience* about how resources are organized, connected, and accessed from *rule experience* about how local conditions affect actions. Knowing where a record resides differs from knowing whether it is eligible for processing; changed eligibility need not invalidate its location. This distinction enables analysis of *selective revision*: correcting outdated conclusions while retaining useful knowledge.

Our *experience-first* construction jointly synthesizes tasks and changes around reusable dependencies. Reference experience remains private; agents retain only their own interaction-derived experience. Task-specific graders assess delivered outcomes rather than prescribed action sequences.

Experiments with seven models show that experience acquired through interaction improves later task performance, but these benefits are not reliably maintained after environmental changes. Silent changes erode accumulated advantages, and recovery varies substantially across models; a loss of advantage does not necessarily mean that retained experience is worse than the without-experience control. Trace analyses expose further bottlenecks: fewer interactions need not imply effective reuse, and correct structural or rule experience can remain unused. These findings motivate evaluating how agents maintain actionable experience throughout its lifecycle, rather than treating retention, reuse, and revision as interchangeable evidence of learning.

Our contributions can be summarized as follows:

- We introduce ReScopeBench to study experience maintenance, connecting budgeted discovery and cross-task reuse with autonomous revision under silent changes.
- We construct 288 tasks across 18 environments, organized into 16-task streams with shared structural and rule dependencies, controlled changes, and objective outcome scoring.
- We reveal model differences in experience benefits and recovery, and distinguish acquiring correct experience from using it effectively through behavioral and content analyses.

2 Related Work

2.1 Experience-Based Agent Adaptation

Agents retain experience in several forms to improve subsequent decisions. Reflexion and ExpeL turn execution feedback and trajectories into verbal reflections or reusable insights (Shinn et al., 2023; Zhao et al., 2024). Voyager stores executable skills, while Agent Workflow Memory abstracts recurring action sequences into workflows (Wang et al., 2024; 2025). Dynamic Cheatsheet continually revises a test-time memory as new evidence arrives (Suzgun et al., 2026). These approaches differ in representation and update mechanism, but share the goal of using past interaction to guide subsequent behavior. Rather than proposing another experience-update method, ReScopeBench provides a common task-stream protocol for evaluating how agents acquire, reuse, and revise environment experience.

2.2 Benchmarking Agent Memory and Experience

Experience benchmarks differ in how evidence is obtained and how its later usefulness is tested. LoCoMo, LongMemEval, and MemBench evaluate reasoning over supplied histories, with the latter two also covering updates to remembered information (Maharana et al., 2024; Wu et al., 2025; Tan et al., 2025). MemoryAgentBench additionally tests incremental learning and selective forgetting (Hu et al., 2025). Interactive benchmarks such as LifelongAgentBench and MemoryArena evaluate experience use across tasks (Zheng et al., 2025; He et al., 2026); skill-oriented suites examine skill acquisition, refinement, and transfer (Zhang et al., 2026b; Lei et al., 2026; Gao et al., 2026). Closer to our setting, AgingBench studies memory degradation and revision, and EvoArena evaluates adaptation across evolving conditions (Zhu et al., 2026; Xu et al., 2026). AgingBench supplies change notices and directed environment checks, while EvoArena studies evolving tasks and preferences. Our focus is the conjunction of costly discovery, unannounced changes to external operating conditions, and agent-initiated repair of affected experience. ReScopeBench connects budgeted environment discovery, cross-task reuse, and autonomous revision after silent local changes within streams that share structural and rule experience.

3 ReScopeBench

3.1 Benchmark Overview

ReScopeBench evaluates agents pursuing different goals in initially unfamiliar environments. Agents receive task goals, public tools, and exploration budgets, but not complete descriptions of resource relationships or operating rules. Structural experience identifies where and how to act; rule experience determines which actions are appropriate. Both may be represented in memory, skills, or retained history.

Figure 1 links experience-first construction (a) with the evaluated lifecycle (b). We prepare executable environments, identify discoverable experience, jointly synthesize tasks and changes, and verify feasibility and scoring. In the resulting streams, agents acquire and reuse their own experience, then investigate and revise it after silent local changes. Reference experience, solution trajectories, and graders remain private. Construction provides opportunities for learning without prescribing successful acquisition or a monotonic score curve.

3.2 Benchmark Construction

Following Figure 1(a), we construct complete task streams around reusable structural and rule experience. Each stage connects the experience to evidence an agent can discover, decisions it can inform, and outcomes that can be verified.

3.2.1 Environment Preparation

We begin with domain discovery: identifying entities, workflows, and observable operations from application scenarios and, for professional software, public documentation and operational examples. These sources inform an executable specification of entity relationships, tool interfaces, pre-

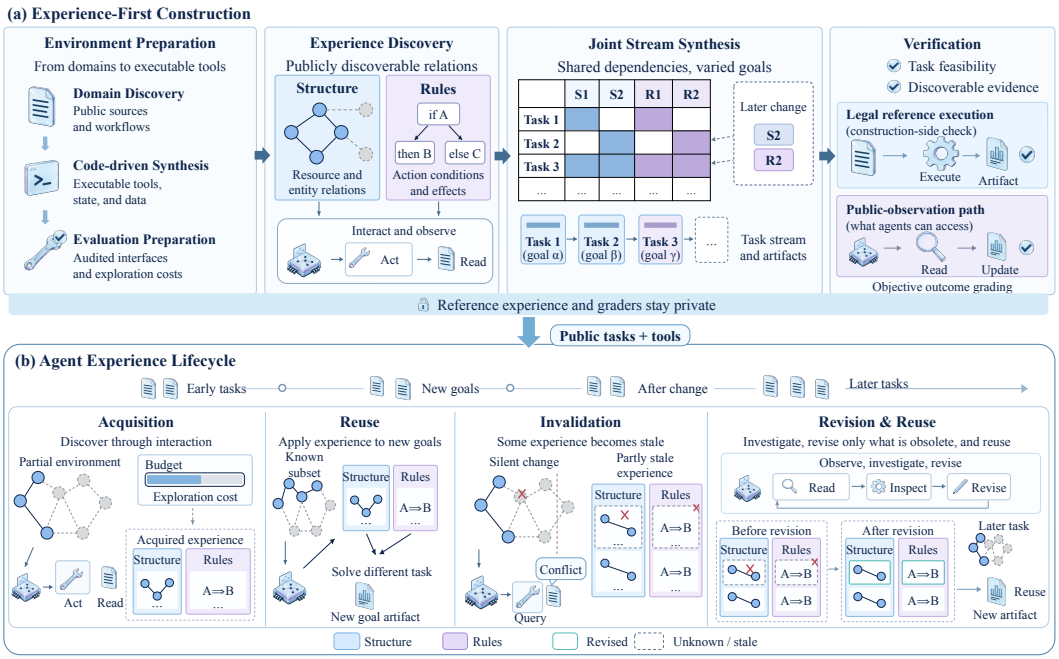


Figure 1: **ReScopeBench overview.** (a) Experience-first construction links executable environments, discoverable experience, and verified task streams. (b) Agents acquire and reuse experience, then investigate and revise it after silent local changes.

conditions, state transitions, and failure behavior. A code-driven implementation then produces tools, a state engine, and reproducible initial data. Records contain substantive information consumed by the tools, with consistent relationships, quantities, and timestamps; source-grounded and synthetic content are distinguished. Outputs depend on inputs and current state rather than fixed responses. The environments are bounded functional simulations, not replicas of proprietary systems or internal algorithms.

We prepare an agent-facing interface that supports discovery without providing a solution. Tool descriptions specify invocation semantics but omit task-specific answers and pre-solved relationships. Exploration consumes environment-specific resources, such as energy, query allowances, or service operations. Charging, exhaustion, and replenishment rules are defined before reference paths are constructed, making discovery and rechecking compete with task execution for the same resources.

3.2.2 Experience Discovery

Given an environment, we combine environment analysis with directed tool interaction to identify reusable structural relationships and local rules. Exploratory walks target unexplored relationships, conditions, and action outcomes. Each candidate experience records supporting observations, applicability conditions, and its consequences for decisions. It must affect a verifiable business outcome or the cost of reaching it across different task instances, rather than merely restating general advice or caching an answer.

We organize the candidates into a private dependency graph linking experience, publicly reachable evidence, actions, and deliverable facts. This graph identifies where experience can be acquired, which later goals can reuse it, and which dependencies a local change would invalidate. It guides stream construction; evaluated agents must discover and represent their own experience.

3.2.3 Joint Stream Synthesis

We jointly plan tasks around the shared experience dependencies. Earlier tasks expose relevant evidence; later tasks reuse the dependencies under different goals, objects, or combinations of conditions. New evidence is interleaved with reuse rather than concentrating all discovery in the first

task. Within an environment, tasks retain comparable goal types, deliverable counts, and scoring criteria so that later performance is not improved simply by making the goals easier.

Reference executions traverse this plan through legal agent-facing tools and produce actual business artifacts. These artifacts determine both the public requests and the private task-specific graders. Requests describe desired outcomes and user constraints without revealing hidden bindings, intermediate answers, or action sequences. Budget amounts are chosen after measuring informed execution, discovery, correction, and alternative-path costs. They must permit informed completion and local repair while making repeated exhaustive exploration unaffordable.

Silent local changes. We insert changes at shared dependencies used before a predefined task boundary, preserving the remaining relationships and rules. Changes affect structural bindings, operating rules, permissions, resource availability, or tool capabilities and required compositions. For tool-related changes, the public interface remains fixed: a previously effective path stops producing a valid outcome, while an alternative selection or composition of existing tools remains feasible. Neither tasks nor boundary messages announce the change or identify the new regime. Subsequent tasks revisit both changed and preserved dependencies, creating opportunities for selective revision and reuse, as illustrated in Figure 1(b).

3.2.4 VERIFICATION

We distinguish *legal reference execution* from a *public-observation path*, as shown in Figure 1(a). Full-stream replay checks executable actions, state transitions, task boundaries, and budget feasibility. Separately, discovery checks establish that the required experience can be obtained from ordinary observations without access to source code, hidden state, or reference answers. Interface audits examine leakage and inexpensive shortcuts, including alternatives to exhaustive search. Reference feasibility alone does not establish that an agent can discover the solution.

Change verification compares behavior using valid experience before the event and stale experience afterward. From the same post-change state and stale experience, a recovery path must diagnose the conflict through ordinary, cost-bearing interaction, revise the affected experience, and reuse it on a later task. Supplying a replacement answer does not satisfy this check; preserved dependencies must remain usable.

Finally, deterministic graders check completed artifacts and meaningful business milestones rather than a prescribed route or self-reported success. They accept equivalent valid outcomes and predefine the treatment of missing outputs, conflicts, and unwanted side effects. Representative model probes diagnose construction failures before tasks, budgets, tools, and graders are frozen together. These checks establish a controlled learning opportunity; natural acquisition, reuse, and recovery remain evaluation outcomes.

3.3 BENCHMARK STATISTICS

The benchmark contains 18 environments, each with a 16-task stream, totaling 288 task instances. Figure 2 groups them into nine application domains, spanning IT and data systems, healthcare, industry and robotics, business and finance, and consumer services. Tasks 1–8 provide acquisition and reuse opportunities; a silent change precedes task 9, followed by revision and reuse opportunities through task 16. These stages specify environmental conditions, not assumed learning outcomes.

The portfolio contains six rule changes, three structural changes, three permission changes, and two each involving resources, tool paths, or mixed mechanisms. Each environment is counted once by its primary implemented change. Tool-path changes alter the effective selection or composition of existing tools; mixed changes combine mechanisms.

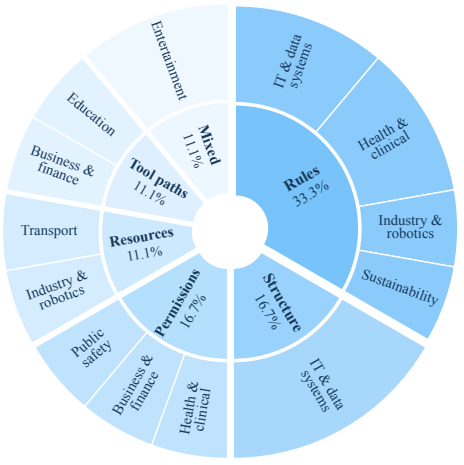


Figure 2: Environment coverage by change type (inner ring) and domain (outer ring).

Counts exclude construction probes and repeated model runs.

3.4 Evaluation Protocol and Metrics

Paired evaluation. For each model and environment, we run two streams from the same initial state. The *with-experience* condition retains naturally accumulated history, memory, and skills. The *without-experience* condition clears these assets, conversational state, and non-business cross-task caches before each task, while permitting within-task learning. Both conditions retain their own evolving business state and environment-managed observation and capability records. Task order, prompts, tools, budgets, change schedules, model configuration, context policy, graders, and termination rules are fixed within each pair. At task t, retained experience can only come from earlier interaction, not private references or graders. This comparison measures longitudinal experience-retention benefits, not an isolated intervention at identical intermediate states.

Baseline and experience benefits. Let $N_t, R_t \in [0, 100]$ denote final-artifact scores with and without experience, and $\Delta_t = N_t - R_t$. A stream has T tasks, first post-change task τ, and late-window length w, with $3 \le \tau < T$ and $1 \le w \le T - \tau$. Baseline performance (BP), online gain (OG), and positive-gain rate (PGR) are

$$\mathrm{BP} = \frac{1}{T}\sum_{t=1}^{T} R_t, \qquad \mathrm{OG} = \frac{1}{\tau - 2}\sum_{t=2}^{\tau-1} \Delta_t, \qquad \mathrm{PGR} = \frac{100}{\tau - 2}\sum_{t=2}^{\tau-1} \mathbf{1}[\Delta_t > 0]. \tag{1}$$

OG measures benefit magnitude and PGR its coverage. Both exclude the initial task, which has no incoming experience. PGR uses unrounded scores; ties and negative differences remain in the denominator. It is not a measure of repeated-run reliability.

Change impact. With $g = \Delta_{\tau-1}$, $h = \Delta_\tau$, and $L = g - h$, gain loss (GL) and experience harm (EH) are

$$\mathrm{GL} = L, \qquad \mathrm{EH} = -h. \tag{2}$$

GL measures lost advantage, whereas positive EH indicates underperformance relative to the without-experience control. Both retain their signs: negative EH means an advantage remains. Small GL need not imply robustness if little advantage was acquired. Boundary tasks and intermediate states differ, so these contrasts alone do not establish causal harm from stale experience.

Recovery. Recovery level (RL) measures late recovery of the lost gain, and recovery efficiency (RE) averages subsequent progress. With $c(x) = \min(1, \max(0, x))$,

$$\mathrm{RL} = 100\, c\left(\frac{\frac{1}{w}\sum_{t=T-w+1}^{T} \Delta_t - h}{L}\right), \qquad \mathrm{RE} = \frac{100}{T - \tau}\sum_{t=\tau+1}^{T} c\left(\frac{\Delta_t - h}{L}\right). \tag{3}$$

Both require $g > 0$ and $L > 0$; otherwise they are N/A. The post-change advantage h defines zero progress and the pre-change advantage g full recovery. An eligible stream with no improvement above h scores zero. RL clips after averaging, whereas RE clips each task before averaging and captures both early progress and persistence, not elapsed time alone.

Aggregation and coverage. We use $T = 16$, $\tau = 9$, and $w = 4$, fixed across models and environments. Metrics are computed per environment and then macro-averaged; recovery ratios are never computed from averaged curves. Higher BP, OG, PGR, RL, and RE and lower GL and EH are preferred, subject to the qualifications above. BP, PGR, RL, and RE range from 0 to 100; OG, GL, and EH are signed score-point differences. Recovery covers 9/18, 18/18, 11/18, and 17/18 environments for GPT-5.4, GPT-5.5, GPT-5.6 Luna, and GPT-6 Astra, respectively; Section 4.2 additionally compares their common eligible subset. Incomplete runs are neither zeros nor mathematically ineligible cases; completed failures retain their prescribed scores. Each condition has one run per environment, so task positions are not independent replicates. Metric definitions were refined after preliminary runs without changing outcomes or evaluation windows. We report no weighted overall score.

Table 2: Main experience-lifecycle results. Arrows indicate preferred directions; metric definitions and conditional recovery coverage are given in Section 3.4.

Family	Model	Base	Experience Benefits		Change Impact		Recovery	
		BP ↑	OG ↑	PGR ↑	GL ↓	EH ↓	RL ↑	RE ↑
MiniMax	MiniMax-M3	11.29	28.37	65.31	30.32	−16.07	68.05	58.88
DeepSeek	DeepSeek-V4-Pro-0813	24.45	47.18	87.01	50.23	−17.73	86.34	74.30
GPT	GPT-5.4	16.09	13.98	47.62	16.08	−8.19	81.86	70.23
	GPT-5.5	22.74	46.21	95.24	63.72	−9.58	88.01	75.57
	GPT-5.6 Terra	22.38	33.57	81.51	50.00	−4.41	70.22	60.60
	GPT-5.6 Luna	14.31	23.60	69.05	29.44	−9.03	35.51	33.06
	GPT-6 Astra	24.44	47.36	95.24	59.19	−8.72	87.99	76.63

4 Experiments

4.1 Experimental Setup

We evaluate seven API configurations: MiniMax-M3 (Lai et al., 2026); DeepSeek-V4-Pro-0813 (DeepSeek-AI, 2026); GPT-5.4, GPT-5.5, GPT-5.6 Terra, and GPT-5.6 Luna, with family background in OpenAI (2026); and GPT-6 Astra. Names identify evaluated deployments; citations refer to available family reports. Run manifests record inference settings. All models follow Section 3.4; construction probes are excluded. Table 2 reports model-level aggregates, whose environment sets can differ. Analysis subsets are specified separately.

4.2 Main Results

Table 2 reveals three patterns. Aggregates describe each model's evaluated environments, rather than a matched-environment ranking across all models.

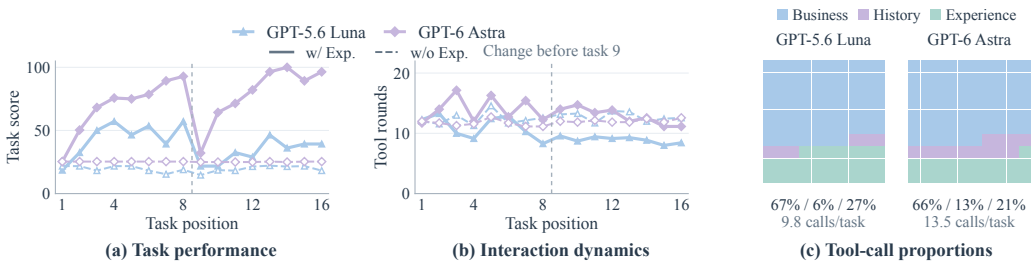
(1) Experience benefits extend across model families. All seven models obtain positive online gains, ranging from 13.98 to 47.36 points. DeepSeek-V4-Pro-0813, GPT-5.5, and GPT-6 Astra obtain gains near 47 points, while MiniMax-M3 gains 28.37 despite a baseline score of 11.29. Benefit magnitude and coverage differ: DeepSeek-V4-Pro-0813 has an OG comparable to GPT-6 Astra (47.18 versus 47.36), but lower PGR (87.01% versus 95.24%). These summaries distinguish the size of improvements from how broadly they occur.

(2) Disruption erodes benefits without necessarily reversing them. Every model exhibits positive mean GL, from 16.08 to 63.72 points, yet negative mean EH. Thus, substantial loss of acquired advantage can coexist with a remaining benefit over the without-experience control. For example, GPT-5.6 Terra loses 50.00 points of advantage but retains 4.41 immediately after change. Smaller GL alone is not evidence of robustness, since models enter the change with different acquired benefits.

(3) Acquisition and recovery expose different weaknesses. GPT-5.6 Luna obtains greater pre-change gains than GPT-5.4, but lower conditional recovery (RL: 35.51 versus 81.86; RE: 33.06 versus 70.23). Eligibility differs, so these are descriptive contrasts. On the same nine eligible environments, GPT-5.5 exceeds GPT-5.4 in RL (93.06 versus 81.86) and RE (78.31 versus 70.23). Recovery should therefore be interpreted jointly with acquired benefits and eligibility, not as an unconditional measure of agent quality.

4.3 Analysis

We analyze GPT-5.4, GPT-5.5, GPT-5.6 Luna, and GPT-6 Astra. Curves use seven common complete environments: Cinema, Cloud Warehouse, Filesystem, Medical, Oracle Clinical, Railway, and Supply Finance, selected by completion rather than performance. Boundary comparisons cover all 18 environments. Each configuration has one paired run.



(a) Task performance (b) Interaction dynamics (c) Tool-call proportions

Figure 3: Experience utility on seven complete environments: (a) task scores, (b) foreground tool rounds, and (c) tool-use proportions with experience over T02–T16.

4.3.1 Experience Utility and Interaction Behavior

To examine how experience changes task performance and interaction behavior, we compare GPT-5.6 Luna and GPT-6 Astra on the seven common complete environments. Figure 3 reports environment-macro-averaged scores and foreground tool rounds. We classify executed calls by tool name into business operations, history retrieval, and experience access or management; proportions pool calls over T02–T16. Counts exclude background consolidation and transport retries.

With experience, GPT-6 Astra improves from 67.32 during T02–T05 to 95.54 during T13–T16, while mean tool rounds decrease from 14.86 to 11.71. GPT-5.6 Luna also reduces its rounds, from 11.21 to 8.64, but its score decreases from 46.52 to 40.27. Fewer interactions alone therefore do not demonstrate more effective learning. Business operations account for similar shares (65.75% and 67.31%), whereas GPT-6 Astra devotes a larger share to history retrieval (13.39% versus 5.80%) and a smaller share to experience tools (20.86% versus 26.89%). Together, these observations indicate that experience utility depends on how retrieved information supports execution, not simply how often experience is accessed. The tool-category associations do not establish causal effects.

4.3.2 Disruption and Recovery under Environmental Change

To analyze how different changes disrupt experience, we group all 18 environments by their six declared change types and compute each model's mean with-experience score drop from T08 to T09 (Figure 4a). This raw drop differs from control-adjusted GL. We then track the with-minus-without-experience advantage on the seven common complete environments to compare recovery (Figure 4b).

Disruption varies across change types and models. Mixed changes produce drops of 71.00 and 83.50 points for GPT-5.6 Luna and GPT-6 Astra, while GPT-5.5 loses 75.00 under both structural and permission changes. Smaller drops need not imply robustness: under rule changes, GPT-5.4 loses only 8.33 points, but its pre-change score is already 20.83, compared with 95.83 for GPT-6 Astra. Recovery further separates the models. GPT-6 Astra's late advantage reaches 70.27 during T13–T16, exceeding 67.50 at T08; GPT-5.6 Luna reaches only 19.38, versus 38.21 before change. Thus, establishing useful experience and rebuilding its utility are distinct capabilities. With only two to six environments per category and domains coupled to change types, these comparisons are descriptive, not isolated estimates of change-type difficulty.

4.3.3 From Retained Experience to Action

To examine how structural and rule experience enter execution, we conduct a single-pass LLM-assisted trace audit in Filesystem and Medical at T08, T12, and T16: 96 requests and 192 type-specific components across four models. We compare pre-task memory and skill assets, including their references, with current private reference bindings, then inspect tool arguments and responses to label adoption. Components are classified by content, not storage format; successful delivery is checked against objective outcomes. Requests vary across positions, so these are aligned case audits, not repeated measurements of identical targets.

Figure 5 separates complete, correct retained components from direct use and successful delivery. GPT-5.4 retains correct structural and rule components for 12 and 15 of its 24 requests, but directly

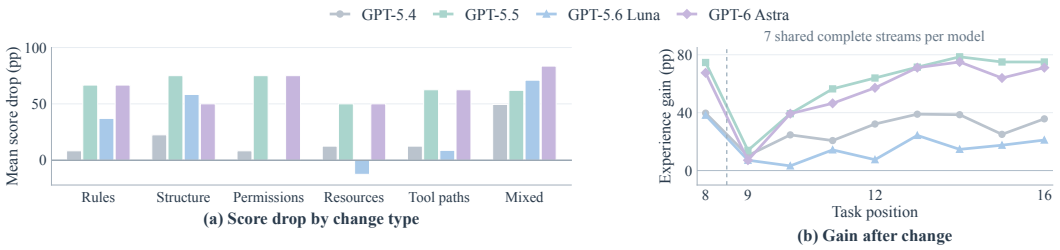


Figure 4: Disruption and recovery: (a) mean with-experience score drop across six change types in 18 environments; (b) experience advantage around and after change on seven complete environments.

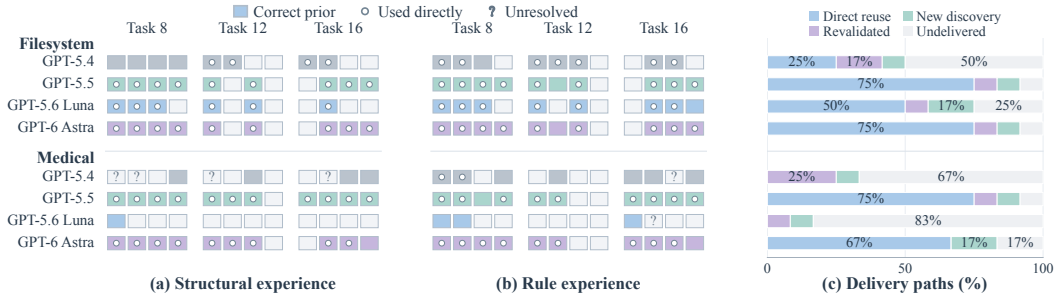


Figure 5: Retained experience and use in two environments. Filled cells mark correct structural (a) or rule (b) components; dots mark direct adoption. Panel (c) shows delivery paths, including failures.

uses only 4 and 9. Successful direct reuse accounts for 12.50% of its requests, versus 75.00% for GPT-5.5, 25.00% for GPT-5.6 Luna, and 70.83% for GPT-6 Astra. Individual cases expose different bottlenecks: GPT-5.6 Luna retains a correct rule but uses a stale route in Filesystem T16; GPT-6 Astra retains both required components for an unattempted request in Medical T16. Incomplete revision and failure to use available experience are therefore separate obstacles. A dot marks component adoption in an attempted action, not necessarily success. The audit includes failures and unattempted requests, but its selected cases and single-pass labels do not establish benchmark-wide semantic rates.

5 Limitations

The environments are bounded simulations. Single paired runs do not quantify run-to-run variability, and independently evolving worlds limit causal interpretations. Model coverage and recovery eligibility differ. The semantic audit uses selected cases and single-pass LLM-assisted labels, not a random sample or independent annotation study. GPT-5.5 participated in environment development, which may favor its behavior; broader independent validation remains necessary.

6 Conclusion

ReScopeBench evaluates the lifecycle of structural and rule experience across 18 unfamiliar environments with budgeted exploration and silent changes. Its experience-first construction links discoverable environment dependencies to downstream tasks, bringing acquisition, reuse, invalidation, and revision into a single evaluation. Seven-model evaluations reveal substantial but uneven experience benefits and recovery. Behavioral and content analyses further separate fewer interactions from effective reuse, and retained knowledge from actual use. Together, these results motivate agents that maintain the applicability of experience rather than merely accumulate it. Extending this evaluation to longer streams and repeated changes is an important next step toward testing sustained experience maintenance.

REPRODUCIBILITY STATEMENT

Section 3.2 describes benchmark construction and verification. Section 3.4 specifies the evaluation protocol, metric definitions, and aggregation rules. Section 4.3 details the data selection and annotation procedures used in the analyses.

AI USE STATEMENT

We used generative AI as a supporting tool for benchmark development, analysis, and manuscript preparation. The authors take full responsibility for the study's claims, factual accuracy, interpretation of results, and final text.

ETHICS STATEMENT

ReScopeBench uses bounded simulated environments. Its results do not establish the safety or reliability of autonomous deployment in real-world or safety-critical settings, which requires separate validation.

REFERENCES

DeepSeek-AI. Deepseek-v4: Towards highly efficient million-token context intelligence. *CoRR*, abs/2606.19348, 2026. doi: 10.48550/ARXIV.2606.19348. URL https://doi.org/10.48550/arXiv.2606.19348.

Xingze Gao, Chuanrui Hu, Hongda Chen, Pengfei Yao, Zhao Wang, Yi Bai, Zhengwei Wu, Yunyun Han, Xiaofeng Cong, Jie Gui, Yafeng Deng, and Teng Li. Evoagentbench: Benchmarking agent self-evolution via ability transfer. *CoRR*, abs/2607.05202, 2026. doi: 10.48550/ARXIV.2607.05202. URL https://doi.org/10.48550/arXiv.2607.05202.

Gonzalo Gonzalez-Pumariega, Vincent Tu, Chih-Lun Lee, Jiachen Yang, Ang Li, and Xin Eric Wang. Scaling agents for computer use. *Trans. Mach. Learn. Res.*, 2026, 2026. URL https://openreview.net/forum?id=eve4jBYa8D.

Zexue He, Yu Wang, Churan Zhi, Yuanzhe Hu, Tzu-Ping Chen, Lang Yin, Ze Chen, Tong Arthur Wu, Siru Ouyang, Zihan Wang, Jiaxin Pei, Julian J. McAuley, Yejin Choi, and Alex Pentland. Memoryarena: Benchmarking agent memory in interdependent multi-session agentic tasks. *CoRR*, abs/2602.16313, 2026. doi: 10.48550/ARXIV.2602.16313. URL https://doi.org/10.48550/arXiv.2602.16313.

Yuanzhe Hu, Yu Wang, and Julian J. McAuley. Evaluating memory in LLM agents via incremental multi-turn interactions. *CoRR*, abs/2507.05257, 2025. doi: 10.48550/ARXIV.2507.05257. URL https://doi.org/10.48550/arXiv.2507.05257.

Alex Iacob, Andrej Jovanovic, William F. Shen, Daniel Burkhardt, Meghdad Kurmanji, Nurbek Tastan, Lorenzo Sani, Niccolò Alberto Elia Venanzi, Ambroise Odonnat, Zeyu Cao, Bill Marino, Xinchi Qiu, and Nicholas D. Lane. The red queen gödel machine: Co-evolving agents and their evaluators. *CoRR*, abs/2606.26294, 2026. doi: 10.48550/ARXIV.2606.26294. URL https://doi.org/10.48550/arXiv.2606.26294.

Xunhao Lai, Weiqi Xu, Yufeng Yang, Qiaorui Chen, Yang Xu, Lunbin Zeng, Xiaolong Li, Haohai Sun, Haichao Zhu, Vito Zhang, Jinkai Hu, Jiayao Li, Rui Gao, Zekun Li, Songquan Zhu, Jingkai Zhou, and Pengyu Zhao. Minimax sparse attention. *CoRR*, abs/2606.13392, 2026. doi: 10.48550/ARXIV.2606.13392. URL https://doi.org/10.48550/arXiv.2606.13392.

Yingtie Lei, Zhongwei Wan, Jiankun Zhang, Samiul Alam, Zixuan Zhong, Peizhou Huang, Xin Wang, Jingxuan Zhang, Donghao Zhou, Yunta Hsieh, Zhihao Dou, Hui Shen, Yan Xu, Dimitrios Dimitriadis, Tuo Zhang, and Mi Zhang. Skillevolbench: Benchmarking the evolution from episodic experience to procedural skills. *CoRR*, abs/2605.24117, 2026. doi: 10.48550/ARXIV.2605.24117. URL https://doi.org/10.48550/arXiv.2605.24117.

Yunlong Lin, Zixu Lin, Zhaohu Xing, Biqiang Li, Chenxin Li, Haonan Wang, Haitao Wu, Hengyu Liu, Jianghai Chen, Kaituo Feng, Kaixin Li, Shawn Chen, Shijue Huang, Sixiang Chen, Tsung-Yi Ho, Wenxuan Huang, Xiangyan Liu, Xiaomeng Hu, Xuanhua He, Yan Sun, Yunqing Zhao, Zhiqin Yang, Zehan Wang, Zhengyang Tang, Tianyu Pang, and Xiangyu Yue. Jarvishub: An open harness for canvas-native multimodal creative agents. *CoRR*, abs/2607.23588, 2026. doi: 10.48550/ARXIV.2607.23588. URL https://doi.org/10.48550/arXiv.2607.23588.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of LLM agents. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 13851–13870. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.747. URL https://doi.org/10.18653/v1/2024.acl-long.747.

OpenAI. Openai GPT-5 system card. *CoRR*, abs/2601.03267, 2026. doi: 10.48550/ARXIV.2601.03267. URL https://doi.org/10.48550/arXiv.2601.03267.

ByteDance Seed. UI-TARS-2 technical report: Advancing GUI agent with multi-turn reinforcement learning. *CoRR*, abs/2509.02544, 2025. doi: 10.48550/ARXIV.2509.02544. URL https://doi.org/10.48550/arXiv.2509.02544.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Mirac Suzgun, Mert Yüksekgönül, Federico Bianchi, Dan Jurafsky, and James Zou. Dynamic cheatsheet: Test-time learning with adaptive memory. In Vera Demberg, Kentaro Inui, and Lluís Marquez (eds.), *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2026 - Volume 1: Long Papers, Rabat, Morocco, March 24-29, 2026*, pp. 7080–7106. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.EACL-LONG.333. URL https://doi.org/10.18653/v1/2026.eacl-long.333.

Haoran Tan, Zeyu Zhang, Chen Ma, Xu Chen, Quanyu Dai, and Zhenhua Dong. Membench: Towards more comprehensive evaluation on the memory of llm-based agents. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, volume ACL 2025 of *Findings of ACL*, pp. 19336–19352. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.FINDINGS-ACL.989. URL https://doi.org/10.18653/v1/2025.findings-acl.989.

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. *Trans. Mach. Learn. Res.*, 2024, 2024. URL https://openreview.net/forum?id=ehfRiF0R3a.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/wang25bx.html.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. Longmemeval: Benchmarking chat assistants on long-term interactive memory. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=pZiyCaVuti.

Jundong Xu, Qingchuan Li, Jiaying Wu, Yihuai Lan, Shuyue Stella Li, Huichi Zhou, Bowen Jiang, Lei Wang, Jun Wang, Anh Tuan Luu, Caiming Xiong, Hae Won Park, Bryan Hooi, and Zhiyuan Hu. Evoarena: Tracking memory evolution for robust LLM agents in dynamic environments. *CoRR*, abs/2606.13681, 2026. doi: 10.48550/ARXIV.2606.13681. URL https://doi.org/10.48550/arXiv.2606.13681.

Jenny Zhang, Bingchen Zhao, Wannan Yang, Jakob N. Foerster, Jeff Clune, Minqi Jiang, Sam Devlin, and Tatiana Shavrina. Hyperagents. *CoRR*, abs/2603.19461, 2026a. doi: 10.48550/ARXIV.2603.19461. URL https://doi.org/10.48550/arXiv.2603.19461.

Ziao Zhang, Kou Shi, Shiting Huang, Avery Nie, Yu Zeng, Yiming Zhao, Zhen Fang, Qishen Su, Haibo Qiu, Wei Yang, Qingnan Ren, Shun Zou, Wenxuan Huang, Lin Chen, Zehui Chen, and Feng Zhao. Skillflow:benchmarking lifelong skill discovery and evolution for autonomous agents. *CoRR*, abs/2604.17308, 2026b. doi: 10.48550/ARXIV.2604.17308. URL https://doi.org/10.48550/arXiv.2604.17308.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: LLM agents are experiential learners. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024, February 20-27, 2024, Vancouver, Canada*, pp. 19632–19642. AAAI Press, 2024. doi: 10.1609/AAAI.V38I17.29936. URL https://doi.org/10.1609/aaai.v38i17.29936.

Junhao Zheng, Xidi Cai, Qiuke Li, Duzhen Zhang, Zhong-Zhi Li, Yingying Zhang, Le Song, and Qianli Ma. Lifelongagentbench: Evaluating LLM agents as lifelong learners. *CoRR*, abs/2505.11942, 2025. doi: 10.48550/ARXIV.2505.11942. URL https://doi.org/10.48550/arXiv.2505.11942.

Jianing Zhu, Yeonju Ro, John T. Robertson, Kevin Wang, Junbo Li, Haris Vikalo, Aditya Akella, and Zhangyang Wang. Your agents are aging too: Agent lifespan engineering for deployed systems. *CoRR*, abs/2605.26302, 2026. doi: 10.48550/ARXIV.2605.26302. URL https://doi.org/10.48550/arXiv.2605.26302.