EVOBELIEF: FALSIFIABLE WORLD MODELS FOR SELF-IMPROVING LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Self-evolving LLM agents typically improve how they act through accumulated experience, reflection, or skills. Yet even procedurally capable agents can fail when their beliefs about environment dynamics are incomplete, incorrect, or outdated. We study a complementary axis of self-evolution: evolving those beliefs during deployment. Such beliefs can support multiple tasks, but incorrect ones can also propagate across them. We introduce **EvoBelief**, a self-evolving world model that represents environment knowledge as falsifiable hypotheses. Unexpected outcomes and decision-relevant unknowns trigger hypothesis formation. Later observations and live controlled experiments provide independent validation before a rule can guide predictions or plans within its supported scope. Counterexamples narrow or retire rules as the environment changes. The language model remains frozen, and each rule remains linked to its evidence and the decisions it changes. Across four benchmarks spanning static dynamics, hidden drift, and scientific discovery, EvoBelief improves task performance and reuses learned rules across tasks. On ALFWorld without actor-side transition memory, it improves success over the matched Frozen WM control by 23.9 and 31.3 percentage points with Claude Haiku 4.5 and GPT-5.4, respectively. On MiniGrid, it revises stale rules and recovers after unannounced dynamics changes. In DiscoveryWorld's Reactor Lab, it discovers and validates quantitative laws, completing 83.3% of Normal-tier and 58.3% of Challenge-tier instances, versus 0% for each evaluated baseline. These results demonstrate self-improvement through testable, revisable environment knowledge rather than model-parameter updates.

1 INTRODUCTION

Existing self-evolving LLM agents mostly adapt to new or changing environments by improving *how they act*. They learn from past interaction trajectories, refine behavior through reflection, accumulate memories, or distill experience into reusable skills (Zhao et al., 2024; Shinn et al., 2023; Park et al., 2023; Wang et al., 2023). These mechanisms are effective in addressing failures arising from inadequate strategy or procedure. But an agent also acts on its prior beliefs about how the environment works, and in a new environment these beliefs can be wrong. Consider the agent in Figure 1, whose task is to synthesize a compound X. It already knows how to heat, purify, and analyze samples, yet it fails because its prior predicts that X is stable up to 120 °C, while X in fact decomposes at 100 °C. A task-specific correction, such as "use a lower temperature," can address the immediate failure but does not capture the underlying property. The agent should instead test how X's stability depends on temperature and retain the resulting evidence-supported constraint. This belief can then guide not only synthesis, but also related tasks such as shipping, storage, and analysis.

This motivates **belief evolution**, a complementary form of self-evolution that updates a persistent model of environment dynamics rather than only adapting task-specific behavior. A policy specifies how an agent acts toward a goal, $\pi(a \mid s, g)$, whereas a world model captures how the environment responds, $W(s' \mid s, a)$. When the world model is wrong, even a capable agent may choose actions based on incorrect predicted outcomes. Conversely, correcting an environmental belief can improve decisions across multiple tasks without separately updating a procedure or skill for each task. This raises a natural question: *how can an LLM agent form, test, revise, and reuse beliefs about its environment while keeping the underlying language model frozen?*

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.

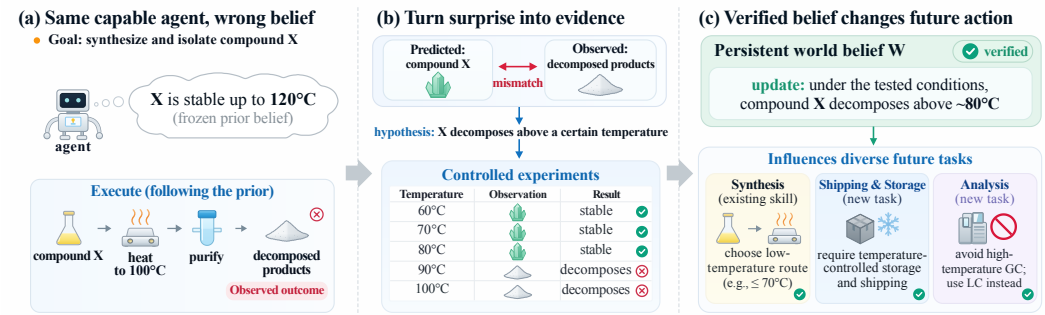

Figure 1: Illustrative example. (a) Following its frozen prior, a capable agent heats compound X to 100 °C and gets decomposition products. (b) The mismatch triggers a hypothesis, which controlled experiments test. (c) The verified belief is stored in the persistent world model W and changes later actions in the original task and in new ones, without any parameter update. Based on the updated belief, the agent avoids gas chromatography (GC), whose operating temperature exceeds the supported range, and correctly chooses to use liquid chromatography (LC) instead.

A growing line of work has begun to adapt world models for LLM agents. WALL-E and WALL-E 2.0 induce rules from collected trajectories to align an LLM world model with the environment, and keep the rules that correct the most prediction errors in the collected trajectories (Zhou et al., 2026; 2025). Imagine-then-plan approaches such as ITP-I use offline-trained world models to simulate future outcomes during planning, while keeping the world model itself fixed (Liu et al., 2026). WorldEvolver is closest to our setting, updating world-model knowledge online from prediction mismatches with a memory of past episodic transitions and semantic facts (Zhang et al., 2026b). These works demonstrate the value of environment-specific world models for agents, but leave open three challenges central to continual adaptation: **validation**, **maintenance**, and **application**. Newly inferred rules must first be validated against additional evidence before being trusted, and once established, they must be maintained as the environment changes, by being revised, narrowed, or retired when contradicted by new evidence. Finally, validated beliefs should influence future predictions and decisions, including tasks different from the one in which they were acquired.

We introduce **EvoBelief**, a self-evolving world model that addresses these challenges through an explicit cycle of hypothesis formation, validation, and revision (Figure 2). When an outcome contradicts the model's prediction, or when an unknown matters for a decision, EvoBelief proposes falsifiable hypotheses about the local environment dynamics involved. It tests these hypotheses against later observations or through controlled experiments, promotes evidence-supported hypotheses into reusable rules, and revises, narrows, or retires rules when later evidence contradicts them. Only validated rules are allowed to influence the agent's future predictions and plans.

EvoBelief maintains a two-timescale belief representation, capturing episode-local situational state (z_t) and persistent, evidence-linked beliefs about environment dynamics (W_t). The underlying LLM remains frozen, and all adaptation occurs in this external world model, separating the model's fixed parametric prior from an environment-specific belief that evolves through test-time interaction. This explicit representation also makes belief evolution *inspectable*. Each rule is linked to the evidence that supports it, the conditions under which it applies, and subsequent observations that may challenge it, which allows us to trace how a validated rule altered a prediction or plan, and evaluate how learned beliefs transfer across tasks, influence decisions, and adapt when the environment changes. Our contributions are:

- We introduce **belief evolution** as a complementary form of self-improvement for LLM agents, in which a persistent world model captures environment-specific beliefs that can evolve at test time through interaction and generalize across tasks without updating the underlying language model.
- We develop a belief lifecycle for **validation**, **maintenance**, and **application**: hypotheses are validated before being trusted, established rules can be revised or retired as evidence changes, and validated beliefs are applied to relevant future predictions and decisions.
- We evaluate EvoBelief on four benchmarks covering static and changing dynamics, and scientific discovery, showing improved task success, cross-task transfer of learned beliefs, adaptation to shifts in dynamics, and discovery of quantitative laws through active experimentation.

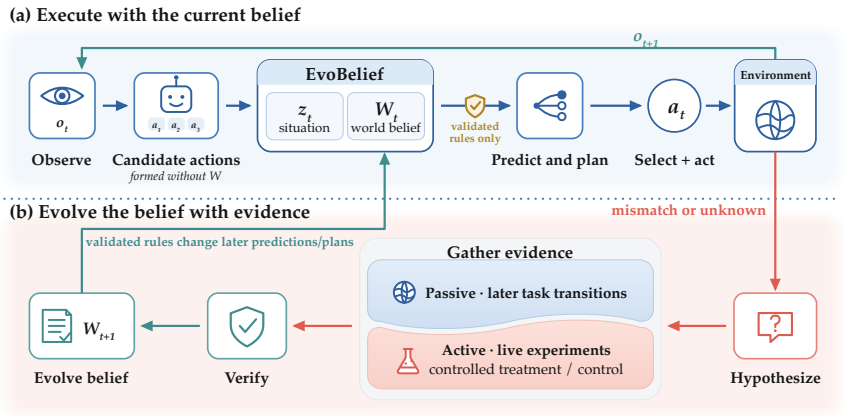

Figure 2: **Verified belief revision closes the loop from experience to later behavior.** EvoBelief uses the current z_t and W_t to predict action consequences. A grounded prediction mismatch or a decision-critical unknown opens an evidence loop that forms falsifiable hypotheses, gathers evidence, and verifies their scope. Only validated, applicable rules guide predictions and plans.

2 Related Work

Belief states and world models. In a POMDP, the belief state is a distribution over latent environment states (Kaelbling et al., 1998), and predictive state representations characterize state through predictions of future observations (Littman & Sutton, 2001). World models address the complementary question of how actions change the world, from planning with learned dynamics to recurrent and latent models (Sutton, 1990; Ha & Schmidhuber, 2018; Hafner et al., 2019; Assran et al., 2025; Wang et al., 2026). Recent language-agent work externalizes state uncertainty or maintains linguistic beliefs under partial observability (Singh et al., 2026; Murphy, 2026; Jiang et al., 2026; Liao et al., 2026), and the belief-based world model of Kumar et al. (2026) exposes a hand-specified belief over the current state to the policy but does not learn it from interaction. EvoBelief separates the two roles across timescales: z_t estimates the current situation, and W_t is a persistent belief over local environment dynamics learned from interaction. Related world-model agents plan with the model as a simulator, inject world knowledge into planning, learn environment rules from experience, or evolve transition memories (Hao et al., 2023; Qiao et al., 2024; Zhou et al., 2026; 2025; Liu et al., 2026; Fang et al., 2025; Zhang et al., 2026b); others explore curiosity-driven exploration, deliberate environment probing, theory formation, and closed-loop scientific discovery (Pathak et al., 2017; Levy et al., 2025; Song & Cai, 2026; Kabra et al., 2026; Jin et al., 2026), and interactive benchmarks now require such discovery (Wang et al., 2022; Jansen et al., 2024). EvoBelief differs in that only validated, scoped claims may change a prediction or a plan.

Experience, memory, and self-evolving agents. LLM agents improve from experience through reflections, episodic memories, reusable lessons, procedures, and skills (Shinn et al., 2023; Zhao et al., 2024; Park et al., 2023; Wang et al., 2023; Ouyang et al., 2026b;a; Chen et al., 2024; Ma et al., 2026; Shang & Yang, 2026), and a broader self-evolving literature modifies prompts, scaffolds, search procedures, or agent code while keeping the model fixed (He et al., 2026; Li et al., 2026; Zhang et al., 2026a; Robeyns et al., 2025; Lee et al., 2026; Xu et al., 2026). ProPlay learns procedural world models, HDSO optimizes skills through hypothesis-driven testing, and AutoManual builds environment instruction manuals from interaction (Ma et al., 2026; Shang & Yang, 2026; Chen et al., 2024). EvoBelief instead maintains predictive claims about environment dynamics with explicit conditions, evidence-supported scope, and counterexample-driven revision. These substrates evolve *how the agent acts*; EvoBelief evolves *what the agent believes about the deployed environment* and separates belief existence from behavioral authority, which a learned claim gains only after independent validation and can lose to later counterexamples. That lifecycle is also what continual adaptation asks for when transition dynamics change and earlier knowledge goes stale (Xie et al., 2021; Wołczyk et al., 2021; Kessler et al., 2023). Belief-calibrated optimization is closest in vocabulary (Chen et al., 2026), but its beliefs concern how an evaluator responds to edits of the agent's own scaffold rather than the deployed environment's dynamics.

3 METHOD

3.1 PROBLEM FORMULATION

At time t, an environment has latent state $s_t \in \mathcal{S}$, emits an observation $o_t \in \mathcal{O}$, and transitions after action $a_t \in \mathcal{A}$. A task is specified by goal g, and the agent observes the interaction history $h_t = (o_0, a_0, o_1, \ldots, a_{t-1}, o_t)$. The environment may contain local dynamics that are inconsistent with the frozen language model's pretrained prior, and those dynamics may change during deployment. EvoBelief keeps model parameters θ fixed and adapts a two-timescale belief substrate

$$\mathbf{b}_t = (z_t, W_t). \tag{1}$$

Here z_t is an episode-local *state estimate*: a structured account of currently observed entities, relations, quantities, and task progress. It is updated incrementally and reset at every episode boundary. It answers what the agent currently takes to be true about this situation. W_t is a persistent *environment dynamics belief*, the agent's cross-episode account of how this environment responds to actions. It survives episode resets and updates under future evidence. Thus the frozen LLM supplies a general parametric prior, while W_t is an environment-specific belief layered on top of that prior.

3.2 WORLD KNOWLEDGE AS TESTABLE HYPOTHESES

The unit of knowledge in W_t is a *hypothesis*, a falsifiable claim about how the environment responds to an action under observable conditions. We write its predictive core as

$$H = (C, A, Y), \tag{2}$$

where C is an observable condition, A is an action or procedure, and Y is the predicted observable consequence. Each hypothesis also maintains a scope, a test recipe, and an evidence record. Establishing C, executing A, and observing whether Y occurs, therefore, provides a direct test of the claim. A may be a single action, a multi-step procedure, or a measurement intervention, while Y may be a state change or a measured quantity. A hypothesis may contain both fixed entities and abstract object roles. When applied, each abstract role is bound to a concrete object in the current context, and a wider scope is earned only from evidence across varied bindings, rather than assumed in advance. Hypotheses are formed online from the agent's own interaction, starting from an empty W_0. After each action, EvoBelief compares the observed outcome with the prediction made before acting. An admissible transition may support or contradict an existing hypothesis, or trigger a new proposal. A grounded *prediction mismatch*, or a *decision-critical unknown* whose outcome matters for the next decision, can trigger the proposal of a set of *rival hypotheses*, as a single failure may have several explanations. In the compound X example, decomposition may depend on the heating temperature, the heating duration, or the vessel. EvoBelief retains these alternatives so that later evidence can discriminate among them. Importantly, proposing a hypothesis does not count as evidence for it, and a newly formed hypothesis has no behavioral authority until it is independently tested. Details on hypothesis identity, event encoding, and evidence admissibility are given in Appendix B.

3.3 TESTING HYPOTHESES: INDEPENDENT EVIDENCE AND ACTIVE EXPERIMENTS

Hypotheses are tested through two evidence routes. *Passive evidence* arises when an ordinary task transition happens to test an existing claim, as when a later task heats compound X and records whether it remains intact. *Active evidence* comes from experiments run specifically to obtain observations that the task would not otherwise provide and to discriminate among rival hypotheses. Experiments run as live actions in the environment and consume the task's interaction budget, which is why their admission is selective. EvoBelief admits an active experiment only when its outcome can affect a decision the agent must make, when no validated rule already resolves the question, and when the required setup is executable. Each test is evaluated against criteria fixed before execution and settles as *supported*, *counterexampled*, or *inconclusive*. The inconclusive outcome records that the test failed to witness the required condition or that the outcome was missing or ambiguous, rather than treating such evidence as support or contradiction. Unresolved hypotheses remain pending and may be tested in later episodes. Experiment compilation details are given in Appendix B.

3.4 From Evidence to Authority: The Rule Lifecycle

An unvalidated hypothesis cannot justify a task-directed action as established knowledge, but it can motivate an experiment. Reuse beyond a tested instance must be earned from independent evidence in episodes, bindings, and tests. Let S denote the supports of a hypothesis, and let $N_{\text{ep}}(S)$, $N_{\text{bind}}(S)$, and $N_{\text{test}}(S)$ count the number of distinct episodes, bindings, and independent tests they span. A hypothesis becomes a validated rule, committed to W_t for reuse, only when

$$\text{Commit}(H) \iff N_{\text{ep}}(S) \geq \tau_{\text{ep}} \wedge N_{\text{bind}}(S) \geq \tau_{\text{bind}} \wedge N_{\text{test}}(S) \geq \tau_{\text{test}}. \tag{3}$$

The thresholds τ_{ep}, τ_{bind}, and τ_{test} are the minimum counts required for reuse, fixed before evaluation and listed in Appendix B. Each generalized role must vary across the supporting bindings, and a hypothesis can never be committed on its formation evidence alone. A single controlled experiment counts as one independent test, regardless of how many treatment or control conditions it contains. When counterevidence arrives, EvoBelief narrows the rule's supported scope and recomputes whether the remaining evidence still satisfies Eq. 3. A rule that no longer qualifies for reuse falls back to whatever narrower scope remains supported, while repeated contradiction can retire it entirely. Because the evidence record is preserved, every change in a rule's status remains traceable to the observations that caused it. The same mechanism handles drift: stale rules lose reusable status as counterexamples accumulate, and new evidence supports updated rules under new dynamics.

3.5 Using the World Model: Gated Prediction and Planning

Validated rules tell the agent what consequences to expect from its actions. EvoBelief uses these predictions to assess candidate actions or to work out a sequence of actions toward the goal. At each decision, the agent forms a set of candidate actions A_t without access to W_t. For each candidate $a \in A_t$, the frozen predictor provides an anchor prediction

$$p_t^0(a) = F_\theta(a, g, z_t), \tag{4}$$

without learned world knowledge. They are the reference point for what the learned world model changes. The rules are retrieved from W_t by action and context relevance, and retrieval alone does not change anything. Before relying on a rule, EvoBelief checks that its condition holds and that the objects involved lie within its evidence-supported scope. This check uses the state estimate z_t when assessing candidates and the state predicted after each earlier step when working out a sequence of actions. An applicable rule $H = (C, A, Y)$ then predicts the consequence Y of taking A, and only such rule-supported changes to the anchor prediction are retained. The predicted consequences enter decisions in two ways. First, they can revise the expected outcome of a candidate action, changing its ranking, or ruling it out when a necessary precondition is violated. A rule whose condition does not hold makes no prediction, so it cannot rule out an action by itself. Second, rules can be chained into a plan toward g when the consequence of one satisfies the condition of the next, and the agent executes the plan's first step even if it is not among the candidates in A_t. If no rule applies, the agent falls back to its W-blind choice. An unresolved claim that matters for a decision may instead trigger an admissible experiment that tests it (§3.3). Whenever a validated rule changes the executed action, EvoBelief records the rule, the prediction or plan it changed, and its outcome. The success of using a rule does not count as independent support for that rule. See Appendix C for more details.

4 Experiments

4.1 Setup

We evaluate on four benchmarks spanning three deployment regimes: ScienceWorld and ALFWorld (static, largely prior-aligned dynamics), MiniGrid (hidden dynamics drift), and DiscoveryWorld (scientific discovery). Splits, budgets, and configurations are in Appendix F. We compare against ReAct, WorldEvolver, and *Frozen WM*. Frozen WM, our mechanistic control, uses the same W-blind actor and predictor as EvoBelief but lets no learned rule alter a prediction or a plan, so comparing it with EvoBelief isolates what acting on validated world knowledge adds. ALFWorld and ScienceWorld are evaluated without and with actor-side transition memory, provided identically to every method. EvoBelief runs without it on MiniGrid, and no method uses it on DiscoveryWorld. Claude Haiku 4.5 and GPT-5.4 are the backbones used. Within each comparison all methods share the benchmark interface, task order, seeds, budgets, and scoring, and each task is attempted once.

Table 1: Success rate (%) with and without actor-side transition memory, on 134 unseen ALFWorld tasks and the same 90 ScienceWorld tasks for both backbones. WorldEvolver and ITP-I are official implementations, run through the same harness, interface, and budgets as every other method.

		ALFWorld		ScienceWorld	
Setting	Method	Haiku 4.5	GPT-5.4	Haiku 4.5	GPT-5.4
No memory	ReAct	33.58	46.27	55.56	41.11
	ITP-I (Liu et al., 2026)	29.85	39.55	42.22	36.67
	WorldEvolver (Zhang et al., 2026b)	35.07	61.94	52.22	65.56
	Frozen WM	42.54	44.78	46.67	53.33
	EvoBelief	**66.42**	**76.12**	**64.44**	**71.11**
Memory	ReAct	45.52	50.00	50.00	46.67
	ITP-I (Liu et al., 2026)	35.07	40.30	47.78	42.22
	WorldEvolver (Zhang et al., 2026b)	58.96	72.39	64.44	67.78
	Frozen WM	57.46	55.22	47.78	58.89
	EvoBelief	**68.66**	**76.87**	**68.89**	**72.22**

Table 2: MiniGrid task success (%) before and after the hidden key switch. Each cell pools 60 episodes from three 40-episode streams per family; MG-C and MG-M use development seeds and MG-R uses fresh control seeds. Haiku 4.5, without transition memory. Higher is better.

	MG-C		MG-M		MG-R	
Method	Pre-drift	Post-drift	Pre-drift	Post-drift	Pre-drift	Post-drift
Frozen WM	0.00	0.00	1.67	0.00	48.33	50.00
WorldEvolver	0.00	0.00	6.67	16.67	20.00	26.67
EvoBelief	**61.67**	**95.00**	**78.33**	**81.67**	**83.33**	**78.33**

Implementation. On ALFWorld, MiniGrid, and ScienceWorld, action–effect rules are induced from typed transitions and applied to predict state changes for planning and veto, without a rule-conditioned model call. On DiscoveryWorld, quantitative and categorical laws are fitted to measurements and observations and validated by a held-out prediction frozen before the intervention. Only DiscoveryWorld's frozen predictor engine, the fixed prediction component shared with Frozen WM, makes rule-conditioned model calls, with auxiliary rules that never enter EvoBelief's belief store. ALFWorld relies on passive evidence, while the other three benchmarks also run live experiments. EvoBelief uses the same role-level model interfaces on every benchmark, and its representations, hypothesis families, and control components are fixed before evaluation (Appendix C).

4.2 Regime I: Static, Largely Prior-Aligned Environments

ScienceWorld and ALFWorld test whether belief evolution helps when most dynamics agree with the pretrained prior. Table 1, which also includes the world-model lookahead baseline ITP-I, shows that EvoBelief has the highest observed success rate in every setting shown. The gains come from local mechanics that the prior gets wrong or leaves underspecified. In the ALFWorld trace of Figure 3, a learned rule that the agent must first reach the open fridge changes the predicted consequence of cooling, supplies the missing prerequisite, and completes the task. Across the deployment, these rules reduce repeated exploration, yielding more successes with fewer decisions and model calls (Figure 4). Because W_t persists across tasks, rules learned in one task family also guide the others. Across the four ALFWorld configurations, over 80% of the decisions under a validated rule fall in a family other than the one where the rule was validated (Table 8). The generic rules (go to, take, put, open) account for this cross-family reuse, while the clean, heat, and cool rules stay within their own family and are reused across object types unseen when they were formed (Tables 8 and 9). On average, EvoBelief's margin over Frozen WM is largest without memory for both backbones and stays positive with it, since episodic recall can supply some of the same cross-episode information and the two substrates act partly as substitutes. Family breakdowns and paired tests are in Appendix G.

4.3 Regime II: Hidden Dynamics Drift

MiniGrid tests whether a learned world model stays useful when the environment changes silently during deployment. In every 40-episode stream the key that opens the door switches after episode

Task 79: put a cool mug in cabinet. Both agents hold mug 3 at countertop 1; the fridge is across the kitchen.

Validated rule in W_t (committed before this task) formed on bread (ep. 74); supported on lettuce (ep. 76)

cool OBJECT with fridge → OBJECT becomes cool,
only if OBJECT is in hand and the agent stands at the open fridge

Frozen WM (same actor; no rule may act)	EvoBelief (planner supplies the missing step)
predicts "the mug becomes cool"	predicts "no effect: not at the open fridge"
acts cool mug 3 with fridge 1	acts go to fridge 1, then cool mug 3 with fridge 1
observes "Nothing happens."	observes "mug 3 is cool"
outcome *never recovers; stops at the 50-decision cap*	outcome *task done in 10 decisions*

Figure 3: A rule-induced decision change on a real ALFWorld trace (Haiku 4.5, unseen task). Frozen WM predicts that cooling succeeds, executes the draft to no effect, and never recovers. EvoBelief uses a validated cooling rule, moves to the fridge first, and completes the task in ten decisions.

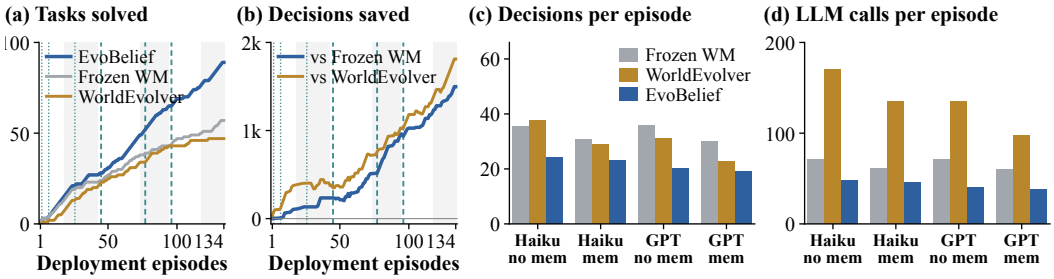

Figure 4: **Learning over deployment and efficiency on ALFWorld.** (a) Tasks solved so far over the 134 unseen tasks (Haiku 4.5, no transition memory). Shaded blocks are the six task types, and dotted and dashed lines mark when the generic and the clean, cool, and heat rules were committed. (b) Decisions saved by EvoBelief over Frozen WM and WorldEvolver. (c, d) Mean decisions and model calls per episode for both backbones, without and with memory (lower is better).

20, and MG-C and MG-M add preconditions the prior does not expect (Figure 5). EvoBelief solves most episodes before and after the switch (Table 2), while Frozen WM and WorldEvolver rarely solve MG-C or MG-M, where success requires the hidden rule. On MG-R, a less adversarial control where the prior solves about half of the episodes, EvoBelief still improves substantially, so the gain is not explained by a weak prior or a floor effect. Over the deployment (Figure 6), early episodes go to experiments on uncertain dynamics, after which validated rules carry short plans reused across episodes. When the silent key switch makes the door rule fail, the counterevidence retires it and reopens experimentation, EvoBelief validates a successor rule, and success returns to its pre-switch level. On three held-out seeds EvoBelief solves 69.6% of episodes against 1.3% for Frozen WM, and 80.8% against 7.1% when every color and room name is permuted, so the rules are grounded in observed effects rather than entity names (Table 10). Mechanism ablations (Table 12) show that both sides of the loop matter. Without active experiments the critical rules never gain independent support, and without learned rules the planner falls back to the prior, so both variants stay at the prior's level. Keeping validated rules in shadow, with the same experiments but no executed plan step, also lowers success, most on MG-C, whose solution is a multi-step chain. Lifecycle traces, the remaining ablations, and a matched decision-level trace are in Appendix G.4.

4.4 Regime III: Scientific Discovery

DiscoveryWorld tests whether EvoBelief can acquire environment laws that neither prior knowledge nor episodic recall can supply. We evaluate two themes at the Normal and Challenge tiers. Reactor Lab hides a quantitative relation between a measured crystal property and the reactor's resonance, so the target resonance must be computed from a law inferred from the agent's own measurements. Plant Nutrients hides a categorical rule for which nutrient settings make a planted seed grow. In Reactor Lab, no baseline (ReAct, WorldEvolver, or the benchmark's Hypothesizer) completes a task

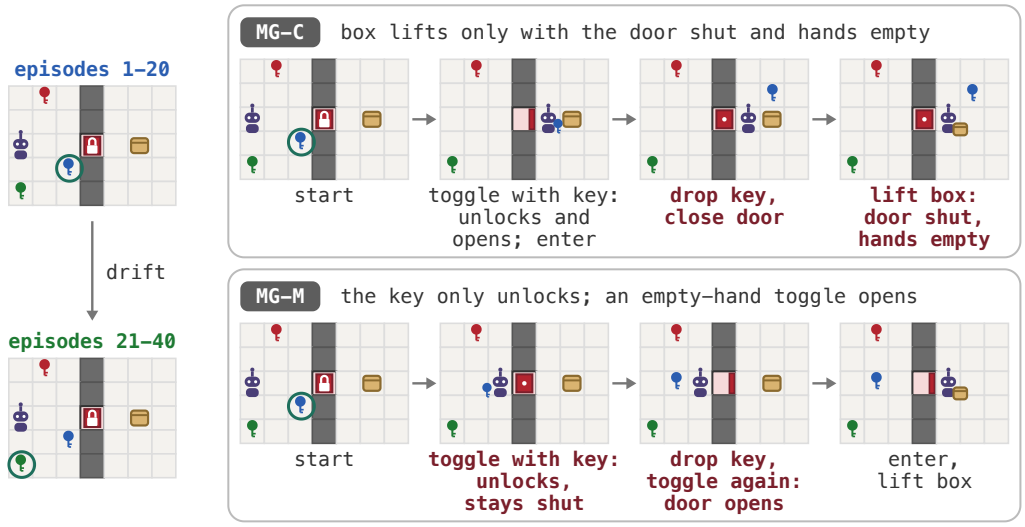

Figure 5: **The two central MiniGrid drift families.** Each episode is a two-room grid with a locked door, three keys, and a target box in the far room. **Left:** the drift shared by every family. Only the ringed key unlocks the door, blue for episodes 1–20 and green for 21–40. The switch is silent, and the red key that matches the door is a decoy. **Right:** the preconditions each family requires to complete the task of lifting the box. In MG-C the box lifts only with the door shut and hands empty. In MG-M the toggle only unlocks the door, and a second, empty-handed toggle opens it. Dark red marks steps the pretrained prior does not expect. Figure 9 in the appendix shows all three families.

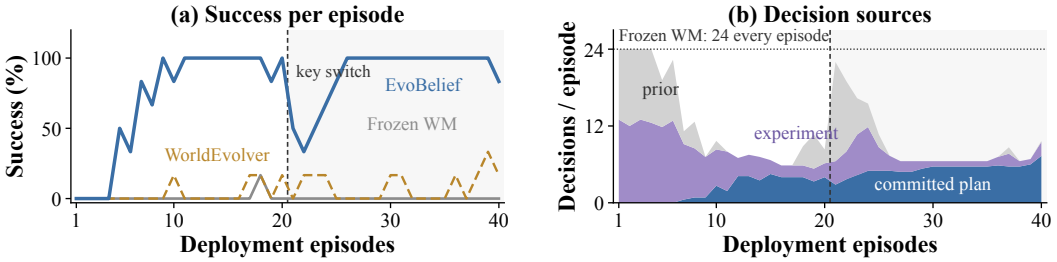

Figure 6: Adaptation over deployment on MiniGrid. (a) Success by episode. The dashed line marks the silent key switch after episode 20. (b) EvoBelief decisions per episode by source: pretrained prior, active experiment, or committed plan. Both panels average the six run streams.

instance or receives knowledge credit, whereas EvoBelief completes most Normal and a majority of Challenge task instances (Table 3). Every completed Reactor Lab task instance follows the intended lifecycle (Tables 13 and 14), in which rival quantitative laws are tested, the survivor is validated by a prediction frozen before a held-out crystal is tried, and the validated law configures the target reactor. Plant Nutrients tests the same loop with a categorical rule whose consequence is realized only through a multi-step planting-and-growth procedure. EvoBelief completes the most task instances in both tiers and is the only method that earns knowledge credit in both. Its knowledge margin over every baseline excludes zero, while its completion margin over the strongest baseline of each tier has an interval that includes zero (Table 7), so discovering the rule and completing the downstream task are related but distinct challenges. Scientific traces, the paired knowledge margins, failure cases, the Hypothesizer's output ceiling, and the development protocol are in Appendix G.5.

4.5 Ablation and Analysis

In Figure 7a, which changes only the persistent substrate, explicit falsifiable rules outperform episodic transitions, free-form semantic memory, and an episodic-belief-only variant. In Figure 7b, EvoBelief leads on State-Δ F1, exact match, and token F1 on both benchmarks, while WorldEvolver has higher feasibility accuracy on ScienceWorld but lower content scores. The routes of §3.5 differ

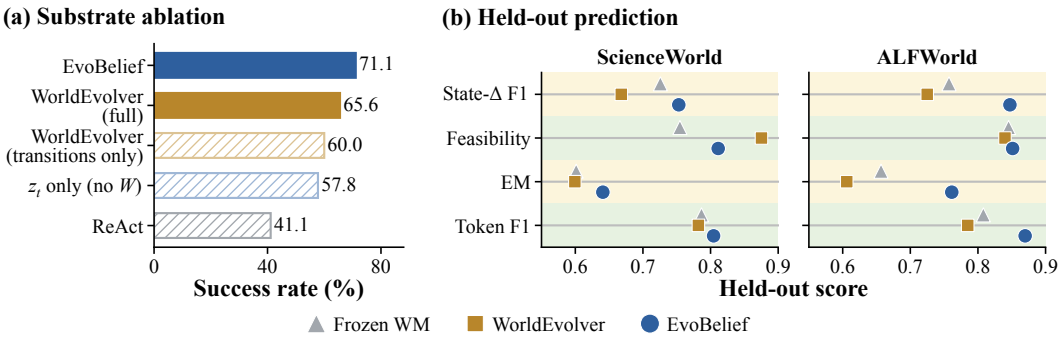

Figure 7: **Substrate ablation and held-out prediction.** (a) Success on the ScienceWorld tasks of Table 1 with GPT-5.4 and no transition memory, varying only the persistent substrate. Hatched bars store no explicit rules. (b) Structured state-change prediction on 1,500 held-out transitions per benchmark with Claude Haiku 4.5. Exact values are in Table 16.

Table 3: DiscoveryWorld results with GPT-5.4 on twelve fresh task instances per task family and difficulty tier. Completion, Knowledge, and Procedural are the benchmark's task-completion, blind knowledge, and subgoal-progress scores, in percent.

	Normal			Challenge		
Method	Completion	Knowledge	Procedural	Completion	Knowledge	Procedural
Reactor Lab						
ReAct	0.00	0.00	70.33	0.00	0.00	64.66
WorldEvolver	0.00	0.00	72.67	0.00	0.00	69.54
Hypothesizer	0.00	0.00	74.00	0.00	0.00	68.39
EvoBelief	**83.33**	**79.17**	**89.00**	**58.33**	**62.50**	**84.20**
Plant Nutrients						
ReAct	41.67	16.67	76.85	50.00	0.00	77.78
WorldEvolver	16.67	8.33	70.37	75.00	0.00	83.33
Hypothesizer	50.00	0.00	79.63	75.00	0.00	84.26
EvoBelief	**75.00**	**75.00**	**82.41**	**83.33**	**33.33**	**85.19**

by environment (Table 6). Passively acquired rules drive planning and veto across tasks on ALFWorld, validated plans dominate on MiniGrid, where retirements track the switch, and validated laws set plan parameters on DiscoveryWorld. ScienceWorld records 424 live experiment steps and 14 decisions changed by validated rules, 13 of them confirmed, so its behavior change runs mainly through experiments. Gap sizes, deployment-lifetime results, and design notes are in Appendix G.

5 Conclusion & Discussion

Agent self-improvement need not begin by changing the agent. When failures come from hidden preconditions, local dynamics, or stale rules, the missing object is an environment belief that evidence can challenge. EvoBelief keeps that belief as falsifiable hypotheses, promotes them into rules only through independent evidence, and lets counterexamples revoke that authority when the environment drifts. Validated rules guide predictions and plans, while unresolved claims can motivate live experiments. This converts learned environment dynamics into higher task success and shorter episodes than the same agent without the evolving world model, while the belief state stays open to inspection. Where episodic memory already supplies what the prior lacks, rules and memory act partly as substitutes. Because a validated rule describes the environment rather than a task, one verified correction can improve many tasks at once, including tasks the agent has never seen. Retiring rules matters as much as learning them, because the ability to withdraw authority keeps accumulated knowledge an asset rather than a liability when the environment changes. More broadly, language models make hypotheses cheap to generate but not to verify, so an agent should act on a belief only after it has survived a test it could have failed. Carrying this discipline to multimodal, embodied, and real-world software agents is a next step.

AI USE STATEMENT

Large language models assisted this work in two roles. As a writing tool, they helped with copy-editing, LaTeX formatting, and tightening the prose of the manuscript. As a coding tool, they supported the implementation and debugging of parts of the research code. Every AI-assisted passage and code path was reviewed, tested, and revised by the authors, who take full responsibility for the text, claims, code, and artifacts of this work.

ETHICS STATEMENT

This work studies language-model agents in four simulated environments: two text-based benchmarks, a grid world, and a simulated scientific-discovery benchmark. No human subjects, personal data, or physical systems are involved. We use public benchmarks, released baselines, and commercial language models with attribution to their sources. Our experiments are confined to simulators, and the reported results do not establish safety in physical or social environments.

REPRODUCIBILITY STATEMENT

Appendix B gives the method details, Appendix C the canonical prompts, the model-call inventory, and the retrieval mechanics, Appendix D the decision-step algorithm, and Appendix F the benchmarks, splits, baselines, protocol, and decoding settings. Appendix G reports the paired statistics behind every matched comparison and the mechanism counts taken from the sealed run records.

REFERENCES

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Minghao Chen, Yihang Li, Yanting Yang, Shiyu Yu, Binbin Lin, and Xiaofei He. AutoManual: Constructing instruction manuals by LLM agents via interactive environmental learning. *Advances in Neural Information Processing Systems*, 37:589–631, 2024.

Yuhan Chen, Zhihua Tian, Mahavir Dabas, Charith Peris, Rahul Gupta, Ming Jin, Feiyang Kang, Siyuan Zhang, Nan Wang, and Ruoxi Jia. Belief-calibrated optimization: An explicit world model for agentic optimization. *arXiv preprint arXiv:2609.01861*, 2026.

Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo Perez-Vicente, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and J K Terry. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. *Advances in neural information processing systems*, 36:73383–73394, 2023.

Tianqing Fang, Hongming Zhang, Zhisong Zhang, Kaixin Ma, Wenhao Yu, Haitao Mi, and Dong Yu. Webevolver: Enhancing web agent self-improvement with co-evolving world model. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 8970–8986, 2025.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. *Advances in neural information processing systems*, 31, 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. Reasoning with language model is planning with world model. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 8154–8173, 2023.

Yufei He, Juncheng Liu, Yue Liu, Yibo Li, Tri Cao, Zhiyuan Hu, Xinxing Xu, and Bryan Hooi. Evotest: Evolutionary test-time learning for self-improving agentic systems. In *International Conference on Learning Representations*, volume 2026, pp. 104166–104202, 2026.

Peter Jansen, Marc-Alexandre Côté, Tushar Khot, Erin Bransom, Bhavana Dalvi Mishra, Bodhisattwa Prasad Majumder, Oyvind Tafjord, and Peter Clark. Discoveryworld: A virtual environment for developing and evaluating automated scientific discovery agents. *Advances in neural information processing systems*, 37:10088–10116, 2024.

Haitao Jiang, Lin Ge, Hengrui Cai, and Rui Song. Pabu: Progress-aware belief update for efficient llm agents. *arXiv preprint arXiv:2602.09138*, 2026.

Jiajie Jin, Yuyang Hu, Kai Qiu, Qi Dai, Chong Luo, Guanting Dong, Xiaoxi Li, Tong Zhao, Xiaolong Ma, Gongrui Zhang, et al. Toward generalist autonomous research via hypothesis-tree refinement. *arXiv preprint arXiv:2606.11926*, 2026.

Sanchit Kabra, Nikhil Abhyankar, Saaketh Desai, Prasad Iyer, and Chandan K Reddy. Llm-autoscilab: Closed-loop scientific discovery via active experimentation with llms. *arXiv preprint arXiv:2605.24043*, 2026.

Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

Samuel Kessler, Mateusz Ostaszewski, MichałPaweł Bortkiewicz, Mateusz Żarski, Maciej Wolczyk, Jack Parker-Holder, Stephen J Roberts, Piotr Mi, et al. The effectiveness of world models for continual reinforcement learning. In *Conference on Lifelong Learning Agents*, pp. 184–204. PMLR, 2023.

Shubham Kumar, Harshit Kumar, Narendra Ahuja, and Saurabh Jha. Towards a belief-based world model for llm agents. *arXiv preprint arXiv:2609.00455*, 2026.

Yoonho Lee, Roshen Nair, Qizheng Zhang, Kangwook Lee, Omar Khattab, and Chelsea Finn. Meta-harness: End-to-end optimization of model harnesses. *arXiv preprint arXiv:2603.28052*, 2026.

Guillaume Levy, Cedric Colas, Pierre-Yves Oudeyer, Thomas Carta, and Clement Romac. Worldllm: Improving llms' world modeling using curiosity-driven theory-making. *arXiv preprint arXiv:2506.06725*, 2025.

Yibo Li, Zijie Lin, Ailin Deng, Xuan Zhang, Yufei He, Shuo Ji, Tri Cao, and Bryan Hooi. Just-in-time reinforcement learning: Continual learning in llm agents without gradient updates. *arXiv preprint arXiv:2601.18510*, 2026.

Junfeng Liao, Qizhou Wang, Jianing Zhu, Bo Du, Rui Yan, and Xiuying Chen. Belief memory: Agent memory under partial observability. *arXiv preprint arXiv:2605.05583*, 2026.

Michael Littman and Richard S Sutton. Predictive representations of state. *Advances in neural information processing systems*, 14, 2001.

Youwei Liu, Jian Wang, Hanlin Wang, Beichen Guo, and Wenjie Li. Imagine-then-plan: Agent learning from adaptive lookahead with world models. *arXiv preprint arXiv:2601.08955*, 2026.

Yijun Ma, Zehong Wang, Yiyang Li, Ziming Li, Xiaoguang Guo, Weixiang Sun, Chuxu Zhang, and Yanfang Ye. ProPlay: Procedural world models for self-evolving LLM agents. *arXiv preprint arXiv:2606.12780*, 2026.

Kevin Murphy. Agentic forecasting using sequential bayesian updating of linguistic beliefs. *arXiv preprint arXiv:2604.18576*, 2026.

Siru Ouyang, Jun Yan, Yanfei Chen, Rujun Han, Zifeng Wang, Bhavana Dalvi Mishra, Rui Meng, Chun-Liang Li, Yizhu Jiao, Kaiwen Zha, et al. Skillos: Learning skill curation for self-evolving agents. *arXiv preprint arXiv:2605.06614*, 2026a.

Siru Ouyang, Jun Yan, I Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long Le, Samira Daruki, Xiangru Tang, et al. Reasoningbank: Scaling agent self-evolving with reasoning memory. In *International Conference on Learning Representations*, volume 2026, pp. 94327–94354, 2026b.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pp. 2778–2787. PMLR, 2017.

Shuofei Qiao, Runnan Fang, Ningyu Zhang, Yuqi Zhu, Xiang Chen, Shumin Deng, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. Agent planning with world knowledge model. *Advances in Neural Information Processing Systems*, 37:114843–114871, 2024.

Maxime Robeyns, Martin Szummer, and Laurence Aitchison. A self-improving coding agent. *arXiv preprint arXiv:2504.15228*, 2025.

Fangxin Shang and Yehui Yang. Hypothesis-driven skill optimization for LLM agents. *arXiv preprint arXiv:2606.22330*, 2026.

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.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.

Joykirat Singh, Zaid Khan, Archiki Prasad, Justin Chih-Yao Chen, Akshay Nambi, Hyunji Lee, Elias Stengel-Eskin, and Mohit Bansal. Agent-brace: Decoupling beliefs from actions in long-horizon tasks via verbalized state uncertainty. *arXiv preprint arXiv:2605.11436*, 2026.

Xinyuan Song and Zekun Cai. Ask the world before acting: Budgeted environment probing for world-model calibration. *arXiv preprint arXiv:2606.31422*, 2026.

Richard S Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990*, pp. 216–224. Elsevier, 1990.

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 preprint arXiv:2305.16291*, 2023.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. Scienceworld: Is your agent smarter than a 5th grader? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11279–11298, 2022.

Ying Wang, Oumayma Bounou, Yann LeCun, and Mengye Ren. Adajepa: An adaptive latent world model. *arXiv preprint arXiv:2606.32026*, 2026.

Maciej Wołczyk, Michał Zając, Razvan Pascanu, Łukasz Kuciński, and Piotr Miłoś. Continual world: A robotic benchmark for continual reinforcement learning. *Advances in Neural Information Processing Systems*, 34:28496–28510, 2021.

Annie Xie, James Harrison, and Chelsea Finn. Deep reinforcement learning amidst continual structured non-stationarity. In *International Conference on Machine Learning*, pp. 11393–11403. PMLR, 2021.

Tianshi Xu, Huifeng Wen, and Meng Li. Adapting the interface, not the model: Runtime harness adaptation for deterministic llm agents. *arXiv preprint arXiv:2605.22166*, 2026.

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

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin gödel machine: open-ended evolution of self-improving agents. In *International Conference on Learning Representations*, volume 2026, pp. 104223–104294, 2026a.

Xuan Zhang, Wenxuan Zhang, See-Kiong Ng, and Yang Deng. Self-evolving world models for llm agent planning. *arXiv preprint arXiv:2606.30639*, 2026b.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024.

Siyu Zhou, Tianyi Zhou, Yijun Yang, Guodong Long, Deheng Ye, Jing Jiang, and Chengqi Zhang. Wall-e 2.0: World alignment by neurosymbolic learning improves world model-based llm agents, 2025. URL https://arxiv.org/abs/2504.15785.

Siyu Zhou, Tianyi Zhou, Yijun Yang, Guodong Long, Deheng Ye, Jing Jiang, and Chengqi Zhang. Wall-e: World alignment by neurosymbolic learning improves world model-based llm agents. *Advances in Neural Information Processing Systems*, 38:65352–65391, 2026.

A Limitations and Future Work

Our experiments use text-based environments, where every transition can be logged and every prediction checked. The belief lifecycle itself is not tied to text, and extending it to multimodal, embodied, and real-world software agents is a natural next step. Such settings bring stochastic effects, which call for statistical tests in place of count thresholds when committing a rule. They also involve interventions that can be costly or irreversible, where live experiments would run under explicit risk limits or human approval. When episodic memory already supplies what the prior lacks, it overlaps with explicit beliefs, so learning when to rely on each is a promising direction. Finally, because validated beliefs carry their evidence records, they could be shared across agents or distilled into model weights, turning what one agent has verified into the prior of the next.

B Method Details

This appendix collects implementation-level mechanisms that §3.2–§3.3 reference but do not require for the main argument.

Hypothesis identity and the evidence ledger. Hypothesis identity is goal-free: a canonical signature over (C, A, Y) merges rediscoveries of the same claim across different tasks and evidence polarities, so evidence gathered under one goal accrues to the same hypothesis when it resurfaces under another. The per-hypothesis evidence record of §3.2 is implemented as an append-only *evidence ledger*; entries are never rewritten, each entry retains the episode, binding, and test it came from, and every lifecycle transition cites the ledger entries that caused it. Hypotheses cover action-conditioned transitions, object properties and measurement procedures, multi-step procedures, and conditional constraints, the negative counterpart that says what an action *cannot* do while its condition is witnessed.

Evidence admissibility. Transitions whose outcome reflects the harness rather than the environment—parser rejections, interface errors, and transport failures—are excluded from hypothesis formation and from settlement. A prediction mismatch is admissible only when it is grounded in the realized environment observation itself, not in the agent's own summary of it.

Experience events and the anchor reference. After executing a_t and observing o_{t+1}, EvoBelief builds a typed experience event $e_t = \mathrm{Encode}(o_t, a_t, o_{t+1}, \hat{o}_{t+1}, z_t)$. EvoBelief has one frozen predictor, not two: the anchor forecast comes from the frozen model, the rule-conditioned forecast is computed by code from the applicable rules on ALFWorld, MiniGrid, and ScienceWorld and by the engine's rule-conditioned call on DiscoveryWorld only, and mismatch detection reuses a forecast already made for action selection. Raw observations supply the evidence a hypothesis can cite, while z_t helps the agent notice what might be worth hypothesizing but never validates it. A settled hypothesis is not re-tested unless a new outcome contradicts it, and a successful trajectory may additionally be distilled into a candidate procedure.

Passive and active evidence routes. The two routes of §3.3 settle the same evidence records and drive the same lifecycle. Passive evidence costs nothing because the transition was going to happen anyway, whereas an active experiment spends task steps, and its model calls are metered and reported with the rest of the run. Experiments are multi-step because informative states are often not one action away. Testing the temperature hypothesis for compound X requires first *witnessing the condition* by verifying that the sample is actually at the intended temperature in the vessel, then holding it for the intended duration, then reading whether it remains intact. A test whose condition was never witnessed settles as inconclusive rather than as evidence, since a sample that never reached the intended temperature proves nothing about the temperature hypothesis on its own and recording it as a counterexample would corrupt the record.

Discriminating tests and counting. Rival hypotheses are settled against the same realized evidence, each against the prediction it froze before the test ran, so one measurement can support one rival and counterexample the others. An outcome that every rival predicts discriminates nothing and supports no conditional claim. All readouts of one experimental design count as a single independent test, so a multi-step experiment cannot inflate the support counts of the commitment gate in

§3.4. Per-episode budgets bound hypothesis formation, the number of experiments, and the steps they may spend, and a hypothesis formed in episode k can be validated only by an independent later occurrence.

Experiment admission. Beyond the admission conditions of §3.3, an experiment is admitted only when at least two plausible outcomes would lead to different next actions and no public readout already answers the question. Admitted experiments are executed in the order in which their questions become decision-relevant, and an experiment whose question is resolved by other evidence is closed without running.

Commitment thresholds. Every instantiation fixes the thresholds of Eq. 3 before evaluation and never tunes them on a target. Reuse across episodes requires supports on at least two distinct bindings in at least two distinct episodes, at least one of which is independent of any decision the hypothesis itself influenced. Where a task instance is a single long episode, as in DiscoveryWorld, a hypothesis is committed by one pre-registered held-out test whose prediction was frozen before the test ran.

Actor-side and predictor-side retrieval. Retrieval means two separate things in our experiments. *Actor-side transition memory* is the "w/ transition memory" setting in every results table, available to every method equally, and is ordinary episodic recall of past transitions with no validation attached (Appendix C.4). *Predictor-side rule retrieval* exists only in EvoBelief and is deliberately recall-heavy, because retrieval carries no authority. The applicability gate decides which retrieved rule, if any, may alter a prediction or a plan, so a wrongly retrieved rule is inert text rather than a changed expectation (Appendix C.3).

Experiment compilation and measurement. An *experiment compiler* turns a chosen hypothesis and a binding into an executable test derived from the hypothesis itself: setup steps that establish its condition, the action, and readouts with explicit support and falsification branches. Whether the recorded observations satisfy the pre-declared criteria is judged by a measurement role independent of proposal and selection: it returns true, false, or unknown, must ground its verdict in the observation text, and an instrument failure is recorded as a fault, never as evidence about the world.

C Language-Model Calls, Prompts, and Retrieval

This appendix lists the language-model calls an EvoBelief agent makes, gives one canonical prompt per role, and states how rules are retrieved. The four benchmarks share the same role-level prompting principles and output contracts. Benchmark-specific adapters change only the observable environment fields, the action vocabulary, and the structured schemas required by each public interface; they do not change what a role is asked to do. The purpose is to make one claim checkable: the language model proposes actions, predicts consequences, and extracts typed observations, but it never forms, tests, promotes, retires, retrieves, or plans with a rule that reaches behavior. Those steps are ordinary code, and the actor never sees rule text. All prompts below are quoted verbatim from the frozen code of the sealed runs, except that the output-field rules and the worked example of the hypothesis proposer are omitted where marked; user messages are described by their fields because they are JSON renderings of the current observation.

Table 4 summarizes how each benchmark instantiates the framework of Section 3; the implementation paragraph of §4.1 gives the same division in prose. In DiscoveryWorld's frozen predictor engine, the rule-conditioned prediction is

$$p_t^W(a) = F_\theta(a, g, z_t, R_t^{\mathrm{E}}(a)), \tag{5}$$

where $R_t^{\mathrm{E}}(a)$ contains the engine's own applicable rules, distinct from the validated rules $R_t(a)$ of Algorithm 1; only rule-supported changes are retained. This call is separate from the law layer's direct consequence prediction and law-derived execution plans.

C.1 Roles and Calls

Table 5 lists the calls per decision by role. Three facts hold on every benchmark. The actor prompt contains no rule, prediction, or plan; it receives the goal, the observation, the action interface, its

Table 4: The two instantiations. The last column indicates use of the frozen engine's rule-conditioned model call; calls per decision by role are in Table 5.

Benchmark	Hypothesis source	Behavioral route	Experiment route	Rule-conditioned call
ALFWorld	Typed transitions; code induction	Rule-based planning; precondition veto	Passive evidence only	None
MiniGrid	Typed transitions; code induction	Validated plan steps replace the draft	Live experiments scheduled by code	None
ScienceWorld	Public observations and typed transitions; code induction	Rule-based control, observation obligations, and veto	Live experiments admitted by code	None
DiscoveryWorld	Laws fitted by code from measurements and observations	Code-projected consequences; law-derived execution plans	Compiled live experiments	Conditional

own history, and, in the transition-memory setting, the shared memory block of Section C.4. The anchor prediction $p_t^0(a)$ and the rule-conditioned prediction $p_t^W(a)$ are the same predictor prompt, with the rule field empty or filled:

$$p_t^0(a) = \text{predictor, rules omitted}, \quad p_t^W(a) = \text{predictor, applicable rules supplied.}$$

And the model's hypothesis-proposal, distillation, and binding calls feed the frozen predictor engine's own rule store, which is consumed only on DiscoveryWorld.

Table 5: **Language-model calls per decision, by role.** N is the number of W-blind alternative candidates (at most 4 on ScienceWorld, 2 on DiscoveryWorld). Per-episode calls (a state seed at episode start, hypothesis proposal after a mismatch, trajectory distillation after a success, role binding when an experiment of the engine's own is compiled) are described in the text. Call counts are taken from one configuration per benchmark, Haiku 4.5 on ALFWorld and MiniGrid and GPT-5.4 on ScienceWorld and DiscoveryWorld; the roles and calls per decision do not depend on the backbone.

Benchmark	Actor	Alternatives	Anchor	Rule-conditioned	Analysis
ALFWorld	1	code	1 (batched)	none	code
MiniGrid	1	code	1 per candidate (5)	none	1
ScienceWorld	1	1	$1 + N$	none	1
DiscoveryWorld	1	1	$1 + N$	up to $1 + N$	1

Benchmark adapters. ALFWorld makes exactly two calls per decision, the actor draft and one batched anchor-prediction call over the whole candidate slate (6,172 logical calls over 3,086 environment steps in the sealed Haiku run with transition memory, a ratio of 2.000 in every episode); its candidates are instantiated from the public action grammar by code, and its typed public state is projected by code rather than by a model. MiniGrid issues one anchor-prediction call per candidate and one evidence-analysis call per transition, about 8.5 calls per decision in a sealed episode. ScienceWorld adds one alternatives call, about six calls in a routine decision. DiscoveryWorld uses the benchmark's own ReAct actor prompt unchanged and averages 5.02 logical calls per executed environment action; it is the one benchmark on which the frozen engine's rule-conditioned prediction is requested, and the call is skipped for a candidate when no retrieved rule applies to it. The shared frozen engine additionally runs a bounded number of sandboxed tests of its own typed rules on ScienceWorld and ALFWorld, in a fresh simulator replayed to the current state: 91 and 81 executions in the two ScienceWorld EvoBelief cells against 94 and 33 in the matched Frozen WM cells, and 1,392 to 1,473 per ALFWorld Frozen WM cell, where the EvoBelief traces carry no field for them. The frozen engine's auxiliary tests run in replayed simulator copies, consume no live task steps, and contribute no evidence to EvoBelief's belief store. On MiniGrid the engine's fork provider is absent and every experiment is a live action of the generic controller (892 to 1,277 per 240-episode cell), and on DiscoveryWorld no fork ever executed. The Knowledge score on DiscoveryWorld is computed after the episode by the benchmark's judge prompt, which never feeds back into the agent. Decoding settings are matched within each benchmark, except for the Hypothesizer's higher output limit: temperature 0 everywhere, with per-role output ceilings fixed in the frozen configuration (actor 256

tokens on ALFWorld, 768 on MiniGrid, 160 on ScienceWorld, and 3,072 on DiscoveryWorld; predictor 768, or 2,048 for the batched ALFWorld call; evidence analysis 768 on MiniGrid, 1,536 on ScienceWorld, and 8,192 on DiscoveryWorld; hypothesis proposal 1,536) and a 60,000-character input ceiling per role.

C.2 Canonical Prompts by Role

Actor. The actor chooses one executable action toward the goal from observable information only.

Actor prompt

You are an embodied task agent. Use only the public goal, current public observation, realized history, and any explicitly labeled method-visible memory. Never assume hidden evaluator state. Return strict JSON.

Input, in order: the action interface (the grammar or the currently available actions), the optional memory block, the goal, the current observation, and the last 24 rows of the agent's own history as `Observation:` and `Action:` lines. Output: one JSON object `{"action": "..."}`. On ScienceWorld the same instruction is issued in ReAct form ("Use only observable information. Never assume hidden state or use reward/scorer internals.") and answered in two lines, `Thought:` and `Action:`, with a 160-token budget; on DiscoveryWorld the benchmark's distributed ReAct prompt is used unchanged, pinned by its digest. When the EvoBelief layer replaces the draft, the executed action enters this history as an ordinary `Action:` row with its realized observation; the actor is never told that its draft was overridden.

Candidate alternatives. Where alternatives are proposed by the model (ScienceWorld and DiscoveryWorld), the call is W-blind by construction.

Candidate-alternatives prompt

You generate a small finite set of alternative task actions. You have no world-model rules and must not assume hidden state. Return exactly JSON `{candidates:[{action:"...", reason:"..."}]}`. Use only the supplied goal, current observation, grounded state claims, recent executed history, `baseline_action`, and action grammar. Propose at most the supplied limit, only actions distinct from `baseline_action`. Maximize useful candidate coverage: include distinct grounded action templates when appropriate, and for a multi-argument grammar action consider distinct grounded argument assignments or orientations instead of paraphrasing one choice. Prioritize an explicit missing goal prerequisite, recovery from a blocked/repeated action, or an evidence-grounded terminal action. Copy every concrete referent from the supplied surfaces. Do not invent an entity, action, observation, scientific fact, rule, or valid-action oracle. An empty list is valid.

Input: `goal`, `observation`, `state_claims`, the last six executed steps, `baseline_action` (the actor's draft), `action_grammar`, `limit`. On ALFWorld and MiniGrid the alternatives are instantiated from the public action grammar over already-observed entities by code.

Predictor. The predictor evaluates exactly the supplied action and never proposes or selects one.

Predictor prompt

Evaluate exactly the supplied `draft_action`; do not propose, edit, select, authorize, or recommend an action. Learned world-model rules may affect prediction only. Return exactly JSON `{expected_outcome:"...",prediction_rule_ids:[...],action_rule_ids:[], critical_unknown:null|"...",predicted_observable_change:null|"...", action_feasibility:"feasible|blocked|unknown", goal_alignment:"advances|neutral|conflicts|unknown"}`. `prediction_rule_ids` may cite only supplied prediction-authority rules that materially support this prediction. `action_rule_ids` MUST always be empty: W has no action-proposal or action-authority surface. Use only observable state and supplied rules; never use hidden state, score, reward, or a valid-action oracle.

Input: `goal`, `observation`, `state` (the typed claims of z_t), `unknowns`, `draft_action`, and `validated_rules`, which is empty for $p_t^0(a)$ and holds the rules that passed the applicability

check for $p_t^W(a)$. Output contract: expected outcome, observable change, feasibility, goal effect, a critical unknown if any, and the rules cited. On ALFWorld the same contract is requested once for the whole candidate slate, with one entry per candidate and the observable change expressed as a typed delta over the benchmark's public schema.

Transition and evidence analysis. After every executed transition the model updates the typed state estimate z_t from observation spans and reports whether the anchor prediction matched. Its verdict is recorded; whether it counts as evidence for any rule is decided by code.

Transition and evidence analysis prompt

After a realized transition, use only the supplied `observation_spans` and return exactly JSON with `{state_update:{added_claims:[{text,span_id}], superseded_claims:[{new_text,span_id,supersedes:[claim_id]}], maintenance_pruned_claims:[claim_id]}, prediction_verdict:"match|mismatch|unscorable", evidence_span_id:null|"OBS_NNN"}`. Every non-null span ID must be copied from `observation_spans`. Do not retype quotes. Validate each state-update list independently: when no row is grounded, return an empty list. Never infer hidden state, reward, score, completion, action validity, or unobserved effects.

Input: the prior claims, the observation before, the executed action, the observation after, the prediction, and the observation spans with identifiers. A repair variant re-requests only the JSON framing of a malformed reply. At episode start a seed call of the same form returns the initial claims and unknowns (at most 32 and 12) from the first observation. On ALFWorld the typed public state is projected by code and this role is not called.

Input: the prediction, the observations before and after, the executed action, the current claims, and the formation channel. When the trigger is a newly observed transition rather than a mismatch, the first sentence reads "A previously unrecorded action transition was directly observed. The prediction may have matched using model prior, but no supplied W rule explained it. Record a reusable, testable observable law from this event." A companion distillation call after a successful episode selects reusable property, procedure, or measurement knowledge from the executed trace under the same copy-exactly discipline; its candidates enter the engine's store as proposed hypotheses with no authority.

Experiment binding. When the engine compiles an experiment, the model binds each abstract role of the hypothesis to one currently observed entity and cites the claims and quotes that justify the binding; it does not judge the hypothesis, and a formation defect such as an unfalsifiable or unbindable hypothesis is reported by code before any call.

Experiment binding prompt

Bind every supplied ROLE to one current observable referent for an isolated scientific experiment. Return exactly JSON `{bindings:[{role_id,referent,claim_ids:[...], observation_quotes:[...]}]}`. Use each ROLE exactly once. Every claim ID must be supplied and every quote copied verbatim from the observation. Respect `slot_kinds`, `distinct_from`, `requirements`, and source action affordances. Source surfaces are provenance examples, not required current names. Do not invent an entity, property, or hidden state.

Input: the role specifications, requirements, ordered interventions, current claims, current observation, and action grammar. EvoBelief's own experiments on ScienceWorld, MiniGrid, and DiscoveryWorld are compiled and bound by code from public entities and do not use this call.

Knowledge judge (DiscoveryWorld). The benchmark's own scoring prompt, used unchanged after the episode: it receives the task description, the agent's knowledge, and one critical question, and answers `1` or `0` with an explanation; the benchmark asks two critical questions per task instance, one call each. For EvoBelief the agent's knowledge is the deterministic rendering of the store, in which a statement is listed as established only if its prediction was frozen before the test, the realized evidence settled it as passed, and no counterexample stands. The rendered knowledge never names an arm, and every prompt is checked for that before dispatch.

C.3 Rule Retrieval

Retrieval is deterministic on every benchmark and carries no authority: a retrieved rule changes a prediction or a plan only after the applicability check of §3.5. The keys are the same everywhere, an exact match on the action template, a binding of the rule's roles to entities in the current public state, the rule's guard evaluated on that state, and its earned scope. The adapters differ in how many rules are considered. On ScienceWorld the store scans every rule and keeps those whose action template and arity match the candidate, whose slots resolve to public entities within the earned scope, and whose guards have a solution over the current public atoms; a rule that fails contributes nothing and touches no counter, so querying several candidates cannot fabricate support. On ALFWorld the planner is offered every validated rule branch, and applicability is decided inside the search by instantiating the branch's template over goal-relevant entities and evaluating its guard on the projected state; a validated precondition can also veto the draft. On MiniGrid the belief store offers the planner the best three admissible guard conjunctions per claim, ordered by authority tier, net evidence, reliability, and generality, and a rule applies only where its conjunction unifies with the public facts. On DiscoveryWorld the law store scans every entry through four audited gates, formed at or before the current task instance, committed, granted by its held-out test, and scope-matched to the public world, and the arbiter intervenes only when the resulting objective prefers a different candidate from the fixed slate by more than the objective's own floor, keeping the actor's action on every failure, ambiguity, or absence of authority. The frozen engine's own rule-conditioned prediction on DiscoveryWorld ranks its typed rules by exact action match or lexical overlap under a fixed limit, and code decides applicability before the call.

C.4 Actor-Side Transition Memory

The setting "with transition memory" installs one shared factor on the actor of every method. Before the actor is asked for an action, the factor retrieves at most four realized public transitions from earlier episodes, ranked by lexical overlap between the current goal and public state and the stored goal and pre-state, with a per-item limit of 1,200 characters and a total budget of 4,800. Only transitions of completed earlier episodes are retrievable, every realized transition is stored regardless of its outcome, no outcome label enters the store, and the candidate action never enters the query. Each item is rendered as its goal, the observation before, the action, the observation after, and the public delta, under the header `Method-visible memory: (Cross-episode public transition memory (previous episodes only):` on ALFWorld), spliced into the actor prompt between the action interface and the history. The factor reads no rule, no store, and no arm name, and a run refuses to start if its budget differs from the frozen one. On MiniGrid the EvoBelief runs use the setting without memory, and the memory rows of the MiniGrid tables belong to the baselines.

D Decision-Step Algorithm

Algorithm 1 summarizes one decision step of §3.5. Candidates are formed without access to W_t; each receives an anchor prediction and an effective prediction $\hat{p}_t(a)$ that retains only rule-supported changes to $p_t^0(a)$; where a rule-conditioned model output (Eq. 5) is available, the changes it proposes are retained only where a validated rule supports them. An experiment may be admitted at the current step when a decision-relevant claim is unresolved, its setup is executable, and budget remains (§3.3); admission freezes the test criteria and does not by itself give the experiment priority. SELECT follows the fixed admission, retention, and priority rules of the instantiation: it returns a candidate action, the first step of a validated plan, or the next step of an admitted experiment, together with the basis d_t for that choice, and it preserves the W-blind choice when no permitted intervention applies. PLAN checks rule conditions and scope at every predicted intermediate state, and its first step need not lie in A_t. The decision record d_t is formed before execution and preserves the predictions and evidence provenance needed to distinguish independent tests from rule-caused confirmations. After the environment step, ABSORB updates the experiment's progress from the executed transition, settles tests only when their outcome can be assessed, and recomputes the lifecycle of §3.4; unfinished tests remain pending. Every route spends the same task budget. The implementations of CONSEQUENCE, PLAN, and SELECT, including the use of rule-conditioned model calls, are specified in §4.1 and Appendix C.

Algorithm 1 One EvoBelief decision step

Require: goal g, frozen model F_θ, history h_t, state estimate z_t, world belief W_t, pending experiment E_t (possibly empty), remaining environment-action budget $B_t > 0$

$A_t \leftarrow \textsc{Candidates}(g, h_t, z_t)$ ▷ formed without access to W_t
for all $a \in A_t$ **do**
  $p_t^0(a) \leftarrow F_\theta(a, g, z_t)$ ▷ anchor prediction
  $R_t(a) \leftarrow \textsc{Applicable}(W_t, z_t, a)$ ▷ validated rules whose condition holds and whose scope covers the binding
  $\hat{p}_t(a) \leftarrow \textsc{Consequence}(z_t, a, g, p_t^0(a), R_t(a))$ ▷ rule-supported changes only; $p_t^0(a)$ if $R_t(a) = \varnothing$
end for
$E_t \leftarrow \textsc{Admit}(W_t, z_t, g, B_t, E_t)$ ▷ open or keep an experiment: decision-relevant, executable, within budget; criteria frozen
$\pi_t \leftarrow \textsc{Plan}(W_t, z_t, g)$ ▷ intermediate states predicted by validated rules; empty if no rule applies
$(a_t, d_t) \leftarrow \textsc{Select}(A_t, \{p_t^0(a)\}, \{\hat{p}_t(a)\}, \pi_t, E_t, z_t, g)$ ▷ a candidate, the first step of π_t, or the next step of E_t, with its basis
$o_{t+1} \leftarrow \textsc{Env}(a_t)$; $B_{t+1} \leftarrow B_t - 1$ ▷ every route spends the task budget
$(z_{t+1}, e_t) \leftarrow \textsc{Encode}(z_t, a_t, o_{t+1})$
$(W_{t+1}, E_{t+1}) \leftarrow \textsc{Absorb}(W_t, E_t, e_t, d_t)$ ▷ form; advance or settle tests; commit, narrow, demote, retire
record d_t and the realized outcome o_{t+1}

E Further Worked Examples

Figure 8 walks through three further cases of the loop of §3: paint mixing, a hidden melting threshold, and circuit contact points.

F Experimental Setup Details

Benchmarks. The static, prior-aligned regime uses **ScienceWorld** (Wang et al., 2022), a text-based science laboratory whose mechanics are stationary but partially hidden (90 episodes per method), and **ALFWorld** (Shridhar et al., 2020), a household suite testing beliefs about object state, affordances, and procedures (the 134 unseen test tasks, each run once per method). The dynamic regime uses **MiniGrid** (Chevalier-Boisvert et al., 2023), where the agent must identify the key that actually opens a locked door, cross into the far room, and lift a target box within 24 steps (Figures 5 and 9 describe the families). A seed fixes the sequence of layouts of a stream: key, agent, and box locations are re-sampled every episode, with the first three episodes on the three fixed layouts of the original benchmark. The campaign runs MG-C and MG-M on three development seeds, on three held-out seeds fixed before the runs, on the same held-out seeds with every color and room name permuted in the observations, and on a third seed set reserved for the mechanism ablations, giving 240 matched episodes per method per set; MG-R runs on three further fresh seeds, 120 matched episodes per method. The discovery regime uses **DiscoveryWorld** (Jansen et al., 2024) on two themes. In Reactor Lab, each reactor activates only when tuned to within ± 2 Hz of a resonance frequency governed by a hidden parametric law of a crystal property measurable through instruments, so the held-out resonance of a new reactor cannot be retrieved from memory and must be computed from a fitted law. In Plant Nutrients, a field's soil controller sets nutrient levels, a planted seed grows only under the hidden nutrient rule, and the Challenge tier hides that rule in a logical combination of two nutrients. Both themes run at their Normal and Challenge tiers on GPT-5.4: twelve fresh task instances per tier, frozen before any run, four methods on the same task instances, one attempt each (provider-side failures were re-run under the pre-declared transport policy), 320 environment actions and 1,650 model calls per task instance, and no transition memory. On Reactor Lab, EvoBelief's experiments draw on a science budget of 208 of the 320 actions. Completion and the subgoal-based procedural score are the benchmark's own, and Knowledge is the benchmark's LLM-judged score over two sealed critical questions, judged blind to the method with one judge configuration shared by both themes. The benchmark's own Hypothesizer agent is included as a baseline; its rows in Table 3 come from a separately frozen re-run whose only change is a raised

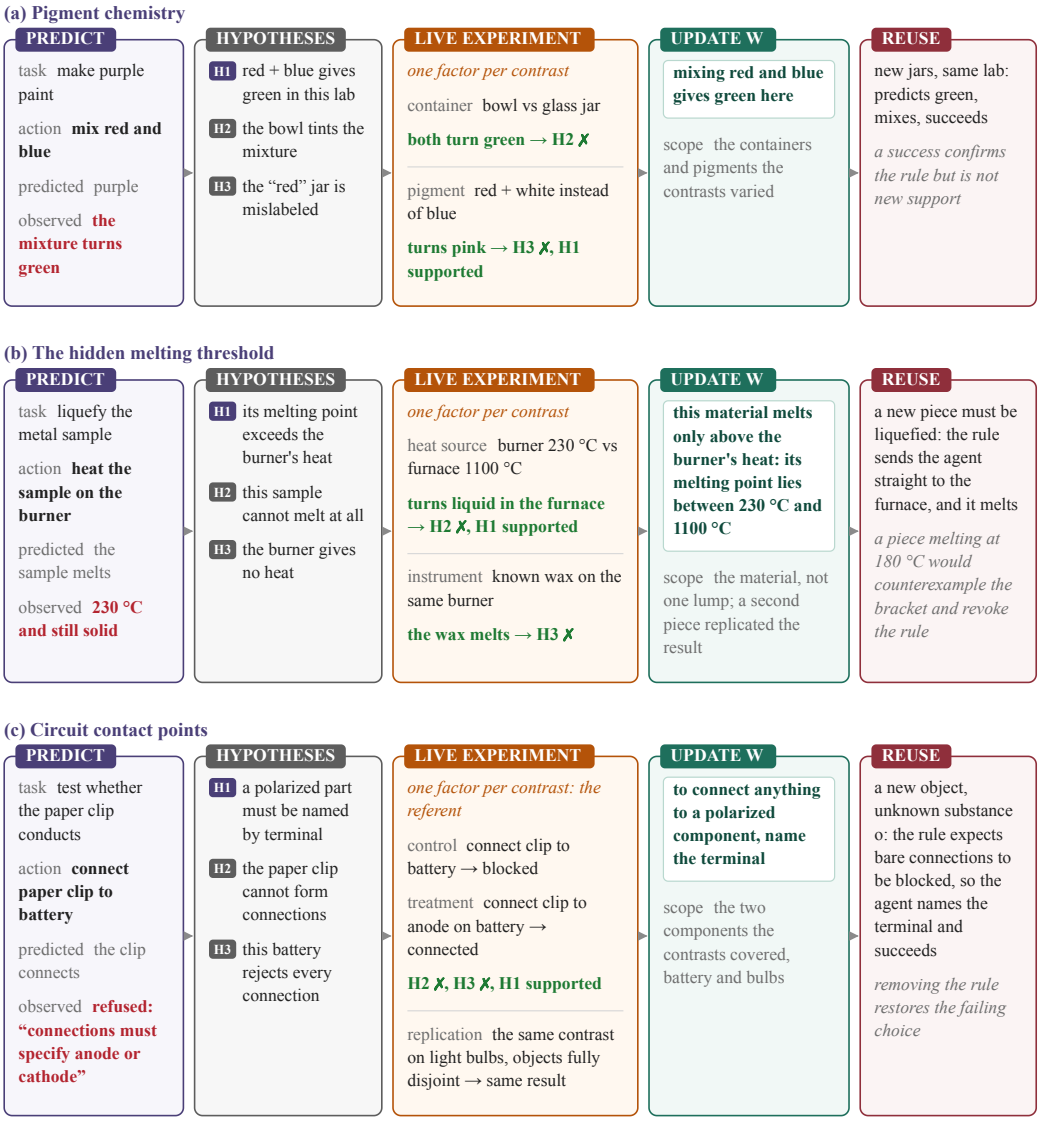

Figure 8: **Worked examples of the EvoBelief loop.** (a) Paint mixing: a mismatch spawns rival hypotheses, two single-factor contrasts settle them, and the rule is reused in a later task with the scope those contrasts earned. (b) A hidden melting threshold: two heat sources bracket the melting point, a positive control clears the burner, and the rule covers the material under the tested conditions. (c) Circuit contact points: a refused action yields a terminal-naming rule, replicated on disjoint objects, that later wires a new object correctly on the first attempt.

output ceiling (Appendix G.5). A frozen-belief diagnostic and the Plant Nutrients protocol and mechanism are reported in Appendix G.5.

Actor settings. ALFWorld and ScienceWorld run under two actor settings, without and with *transition memory*; on MiniGrid the EvoBelief runs use the setting without it and Table 11 gives the baselines with it; no method uses it on DiscoveryWorld. Transition memory and EvoBelief's predictor-side rule retrieval are specified in Appendices C.4 and C.3.

Baselines. **ReAct** keeps no persistent cross-episode belief (Yao et al., 2022). **ITP-I** consults imagine-then-plan lookahead with a learned world model (Liu et al., 2026). **WorldEvolver** evolves a world model online over stored transitions and semantic memory, the strongest published self-

evolving world-model baseline (Zhang et al., 2026b). WorldEvolver and ITP-I are the official released implementations, run through the same harness, interface, and budgets as every other method. WALL-E 2.0's published ALFWorld result uses a different rule-learning protocol and backbone and is not directly comparable to our single-pass benchmark evaluation. **Frozen WM** is the same W-blind actor with the same anchor predictor and selector, but its predictions and choices follow the pretrained prior alone: no learned rule may alter a prediction, a ranking, or a plan. It does not run EvoBelief's belief store, so the comparison isolates what consuming validated knowledge adds over the prior-driven agent. Frozen WM and EvoBelief are independent live runs paired by task, not replays. For rule-guided interventions we record the rule, the prediction or plan it changed, and the realized outcome. The paired full-run comparisons evaluate the complete EvoBelief system, including its experiments.

Protocol and fairness. WorldEvolver's original evaluation reports best-of-L success over $L{=}5$ trials per task, so a task counts as solved when any one of the five trials succeeds (Zhang et al., 2026b). In the main benchmark comparisons, each task is attempted once. All methods, including the baselines we rerun, share one harness with the same task set, task order, seeds, action interface, context limits, retry policy, and scoring. Both backbones run the identical protocol, and success is task completion as scored by the benchmark. On ScienceWorld with GPT-5.4, EvoBelief's run without transition memory was preregistered, executed blind, and sealed first. The Frozen WM and WorldEvolver comparators, and then all three runs with transition memory, were authorized afterwards and executed blind on the same 90 tasks, each group unblinded once.

Metrics and controls. MiniGrid is split at the hidden changepoint into pre-drift and post-drift halves. Mechanism diagnostics count rules formed, tested, supported, counterexampled, and retired, and *decision flips*, decisions where the anchor prediction alone would have selected differently and a validated rule's authorized change is the recorded cause. Held-out structured state-change prediction uses an identical frozen prediction interface for every method. The frozen predictor engine that supplies the anchor prediction, shared with the Frozen WM control, also runs sandboxed tests of its own typed rules on ScienceWorld and ALFWorld; these never substitute a task action, their outcomes never enter EvoBelief's store (Appendix C.1).

G Additional Results

G.1 Mechanism Summary

Table 6 collects, in one vocabulary, what the belief machinery did on each benchmark, and §4.5 summarizes how the routes of §3.5 carry different loads by regime. The counts are reported in the unit each implementation records, so the columns are comparable in kind rather than in magnitude: an ALFWorld hypothesis is a rule branch over object roles, a MiniGrid hypothesis is a guarded rule instance that is re-validated after every epoch, a ScienceWorld hypothesis is a typed rule, and a DiscoveryWorld hypothesis is a candidate law.

G.2 Paired Tests and Confidence Intervals

Because every comparison is paired, Table 7 reports, for each matched comparison, the exact two-sided McNemar test on the discordant pairs and a percentile bootstrap interval for the difference in success rate, resampling paired episodes ten thousand times. The DiscoveryWorld rows are the sealed campaign's own instance-paired analysis.

G.3 Regime I: Qualitative Traces

ScienceWorld: a circuit rule learned from a refused action. Figure 8(c) traces this case, from the refused `connect paper clip to battery` through the controlled contrast and its replication on light bulbs to the first-attempt wiring of a brand-new object. On the same task families, WorldEvolver retained a broad prose heuristic that connections should specify terminals, which never became an executable, scoped rule; it kept proposing failing `use` and `move` actions.

Table 6: **What the belief machinery did, by benchmark.** One setting per benchmark: ALFWorld, Haiku 4.5 without transition memory (134 tasks); MiniGrid, the six development streams (240 episodes); ScienceWorld, GPT-5.4 without transition memory (90 tasks); DiscoveryWorld Reactor Lab and Plant Nutrients, GPT-5.4 (24 task instances each). Each row counts the unit named in its label, and the rows are not stages of one funnel. ALFWorld: 10 validated rules over 43 validation events. MiniGrid: 220 distinct validated claims over 2,185 validated rule versions, and counterexamples are negative witness records. ScienceWorld: 54 experiments opened, 39 completed, yielding 573 support records, while the 4,031 passive supports come from ordinary task transitions. Supports are counted per occurrence, so a claim observed repeatedly in one context adds many records but little independent evidence, which is why validated claims are far fewer than support records. Confirmed: the predicted effect held in the next observation, over the decisions a rule changed on ALFWorld, all plan steps on MiniGrid, the scorable rule predictions on ScienceWorld, law-derived plans on Reactor Lab, and positive consequence verifications on Plant Nutrients. On DiscoveryWorld, the counterexample row counts rival laws falsified. ScienceWorld's procedural safeguards for task milestones and irreversible steps, fixed before evaluation (§4.1), are not belief actions and are not counted. A dash means the implementation does not record the quantity.

	ALFWorld	MiniGrid	ScienceWorld	DW Reactor	DW Plant
Hypotheses formed (claims)	12	639	236	669	324
Passive supports (records)	230	2,156	4,031	–	–
Live experiments (runs)	0	458	54	96	51
Validated claims (distinct)	10	220	36	18	38
Counterexamples (records)	9	10,645	21	296	12
Retirements (rule versions)	2	845	70	–	15
Decisions under validated rules	707	886	14	209	1,129
unit	actions	plan steps	actions	actions	actions
Experiment steps (live actions)	0	1,051	424	–	–
Confirmed	265/324	830/886	13/14	17/18	36
unit	decisions	plan steps	predictions	plans	verifications

ScienceWorld: a material-property rule learned from a prediction mismatch. The prior expects a rubber object to be rigid; the observation shows it bending. The mismatch spawns a flexibility hypothesis that gathers four independent supports across episodes. In a later flexibility-classification task, the validated rule flips two executed decisions, and the paired runs on the same episodes separate cleanly: score 100 with rule consumption versus 71 for Frozen WM without it.

Regime I details. On ALFWorld the gain concentrates on the task families whose procedures the prior misstates, above all cooling, heating, and two-object placement, where a wrong belief about what an action does, or about the order in which two actions must be taken, cannot be repaired by recall alone. Over the deployment (Figure 4), the curves begin to separate after the first rules are committed. The generic household dynamics, that reaching a receptacle reveals its contents, that taking requires an empty hand, that placing moves an object, are committed within the first task type and consumed in every later one, so each new type begins with most of its dynamics already trusted; the one process rule a type adds, cooling, heating, or cleaning, is committed within its first few tasks and reused on the rest. The cumulative gap therefore widens with deployment, fastest on the process types where the prior misstates the procedure. A trusted rule removes the exploratory and repeated actions the prior spends on rediscovering a precondition it has already met, so EvoBelief solves more tasks with fewer decisions and fewer model calls, whereas WorldEvolver pays for its world model with far more calls than the agents that consult none. Because the benchmark enumerates its task types in blocks, later episodes are also harder for the prior: over the second half of the deployment, where the enumeration reaches the process types, Frozen WM solves fewer tasks than in the first half while EvoBelief solves more. This is why the comparison is drawn on matched tasks: the widening is measured against the same agent on the same task, not against an easier past. Of the 1,115 decisions a validated rule changed across the four settings, 907 had their predicted effect fully confirmed in the next observation. The store classified 199 outcomes as inconclusive: the go-to rule predicted arrival at a receptacle and a description of its contents, the agent arrived, and the receptacle turned out to be closed, so the second atom was unobservable. In the remaining 9, the environment refused the action, and the store recorded the refusal as evidence against the rule's condition or as an inconclusive refusal. None of the 1,115 was classified as an effect mismatch.

Table 7: **Paired tests for the matched comparisons.** Success rates in percent; Δ is EvoBelief minus the baseline in percentage points with its 95% percentile bootstrap interval over paired episodes; Ours and Base are the two success rates, Disc. counts the episodes only EvoBelief solved against those only the baseline solved; p is the exact two-sided McNemar test on those pairs. ALFWorld pairs the 134 unseen tasks, WorldEvolver being the authors' release; ScienceWorld pairs its 90 tasks; MiniGrid pairs episodes by stream and position, 240 per seed set, and the post-versus-pre rows compare EvoBelief's second half with its own first half, pairing episode $i + 20$ with episode i of the same stream; MiniGrid episodes within a stream share an evolving belief state, and the episode-level tests and intervals do not account for this dependence; DiscoveryWorld pairs the 12 fresh task instances per tier of each theme, with the Hypothesizer at its raised output ceiling.

Benchmark, setting	Baseline	n	Ours	Base	Δ	95% CI	Disc.	p
ALFWorld, Haiku, no memory	Frozen WM	134	66.42	42.54	23.88	16.42, 31.34	34 / 2	<0.001
	WorldEvolver	134	66.42	35.07	31.34	21.64, 41.04	51 / 9	<0.001
ALFWorld, Haiku, memory	Frozen WM	134	68.66	57.46	11.19	2.99, 20.15	25 / 10	0.017
	WorldEvolver	134	68.66	58.96	9.70	1.49, 17.91	24 / 11	0.041
ALFWorld, GPT-5.4, no memory	Frozen WM	134	76.12	44.78	31.34	23.13, 40.30	45 / 3	<0.001
	WorldEvolver	134	76.12	61.94	14.18	5.22, 23.13	29 / 10	0.003
ALFWorld, GPT-5.4, memory	Frozen WM	134	76.87	55.22	21.64	13.43, 29.85	33 / 4	<0.001
	WorldEvolver	134	76.87	72.39	4.48	−2.99, 12.69	18 / 12	0.362
ScienceWorld, Haiku, memory	Frozen WM	90	68.89	47.78	21.11	7.78, 34.44	29 / 10	0.003
	WorldEvolver	90	68.89	64.44	4.44	−7.78, 16.67	18 / 14	0.597
ScienceWorld, GPT-5.4, no memory	Frozen WM	90	71.11	53.33	17.78	4.44, 31.11	28 / 12	0.017
	WorldEvolver	90	71.11	65.56	5.56	−4.44, 15.56	14 / 9	0.405
ScienceWorld, GPT-5.4, memory	Frozen WM	90	72.22	58.89	13.33	3.33, 23.33	17 / 5	0.017
	WorldEvolver	90	72.22	67.78	4.44	−6.67, 15.56	16 / 12	0.572
MiniGrid, development seeds	Frozen WM	240	79.17	0.42	78.75	73.33, 83.75	189 / 0	<0.001
	WorldEvolver	240	79.17	5.83	73.33	67.50, 79.17	178 / 2	<0.001
MiniGrid, held-out seeds	Frozen WM	240	69.58	1.25	68.33	62.08, 74.17	165 / 1	<0.001
MiniGrid, permuted labels	Frozen WM	240	80.83	7.08	73.75	68.33, 79.17	177 / 0	<0.001
MiniGrid, development, post vs pre	pre-switch	120	88.33	70.00	18.33	10.83, 25.83	24 / 2	<0.001
MiniGrid, held-out, post vs pre	pre-switch	120	75.83	63.33	12.50	3.33, 22.50	25 / 10	0.017
MiniGrid, permuted, post vs pre	pre-switch	120	87.50	74.17	13.33	5.83, 21.67	21 / 5	0.002
DiscoveryWorld Reactor, Normal	ReAct	12	83.33	0.00	83.33	58.33, 100.00	10 / 0	0.002
	WorldEvolver	12	83.33	0.00	83.33	58.33, 100.00	10 / 0	0.002
	Hypothesizer	12	83.33	0.00	83.33	58.33, 100.00	10 / 0	0.002
DiscoveryWorld Reactor, Challenge	ReAct	12	58.33	0.00	58.33	33.33, 83.33	7 / 0	0.016
	WorldEvolver	12	58.33	0.00	58.33	33.33, 83.33	7 / 0	0.016
	Hypothesizer	12	58.33	0.00	58.33	33.33, 83.33	7 / 0	0.016
DiscoveryWorld Plant, Normal	ReAct	12	75.00	41.67	33.33	0.00, 66.67	5 / 1	0.219
	WorldEvolver	12	75.00	16.67	58.33	16.67, 91.67	8 / 1	0.039
	Hypothesizer	12	75.00	50.00	25.00	−16.67, 66.67	5 / 2	0.453
DiscoveryWorld Plant, Challenge	ReAct	12	83.33	50.00	33.33	8.33, 58.33	4 / 0	0.125
	WorldEvolver	12	83.33	75.00	8.33	−16.67, 33.33	2 / 1	1.000
	Hypothesizer	12	83.33	75.00	8.33	−16.67, 33.33	2 / 1	1.000

ALFWorld: rules travel across tasks. The belief persists across the 134 unseen tasks, so a rule validated early is available to every later task. Table 8 counts, for every setting, where decisions under a validated rule were made relative to the task in which the rule earned its authority, and Table 9 lists the rules of one setting. Uses in the task where the rule was validated are rare by construction, since validation needs independent evidence from at least two tasks.

G.4 Regime II: Drift Details

Regime II details. The two families MG-C and MG-M were designed so that the solution requires one step the pretrained prior does not expect, and the 24-step budget leaves no room to discover it by chance. The same prior-governed agent is competent where the environment mostly agrees with its prior and helpless where one hidden rule contradicts it, so the near-zero rows of MG-C and MG-M in Table 2 measure the hidden rules, not a weak baseline. In the first episode of the development stream on MG-C (seed 256), ReAct picks up the color-matched red key, toggles the door three times without effect, cycles through the other keys, opens the door with the blue key on its fourteenth step, enters the far room, and then fails three lifts in a row because the door it opened is still open; it never closes it. On MG-M the same agent unlocks the door on its fifteenth step, sees it closed but unlocked, and spends the remaining steps trying to walk through it. The frozen predictor does not help here: it predicts with the same prior the actor drafts from, so it expects the decoy key to work just as the actor does and never overrules the draft, and all prior-governed methods share a

Table 8: **Where committed rules are used on ALFWorld.** Rule-use decisions: decisions whose executed action a validated rule licensed. Later task: a task after the one in which the rule was first validated. Other family: the task family differs from that of the task in which the rule was validated. Type unseen at formation: the executed action binds at least one role to an object type absent from every binding that formed the rule (instance numbers ignored). Changed: decisions under a validated rule whose executed action differs from the controller's draft after normalizing the surface forms of put. Confirmed: the transition was not rejected and every atom the rule predicted appears in the next observation. Percentages are of decisions under a validated rule, and of changed decisions for the last column.

Setting	Rules	Rule-use decisions	Later task	Other family	Type unseen at formation	Changed	Confirmed
Haiku, no memory	10	707	99.86	82.46	88.12	324	81.79
Haiku, memory	8	753	99.87	85.79	90.17	250	78.40
GPT, no memory	8	774	99.87	85.66	91.34	304	83.22
GPT, memory	8	789	99.75	86.69	90.87	237	81.43

Table 9: **The rules of one setting, GPT-5.4 without transition memory.** Validation task: the index of the task in the deployment order at which the rule first earned authority, with that task's family. Tasks used: distinct later tasks with a decision taken under the rule. Bindings: distinct object and receptacle instances the rule was bound to. The two go-to rules differ in their effect: one predicts that reaching a receptacle reveals its contents, the other only the new location. Clean, heat, and cool rules can only be used in their own family.

Rule	Validation task	Tasks used	Families	Bindings	Changed	Confirmed
go to (contents)	1, look at in light	115	6	26	221	170
take	2, look at in light	109	6	83	20	20
put	7, look at in light	91	5	63	4	4
open	24, pick and place	63	5	8	8	8
go to (location)	21, pick and place	33	5	10	33	33
clean	44, clean then place	25	1	16	4	4
heat	95, heat then place	18	1	9	11	11
cool	76, cool then place	13	1	9	3	3

Table 10: **MiniGrid on held-out seeds (%).** Haiku 4.5, actor without transition memory. Three seeds on both families, fixed before any run and never used while building the method, 240 episodes per method per block; the permuted block renames every color and room in the observations of both methods. Higher is better.

Seeds	Method	Pre-switch	Post-switch	All
Held-out	Frozen WM	2.50	0.00	1.25
	EvoBelief	**63.33**	**75.83**	**69.58**
Held-out, permuted labels	Frozen WM	10.00	4.17	7.08
	EvoBelief	**74.17**	**87.50**	**80.83**

single success. With transition memory the baselines recover part of MG-M by recalling the second toggle (Table 11) but not MG-C, whose lift condition is a conjunction rarely witnessed by accident. MG-R is the control for this design: it keeps the decoy key and the silent switch, so the prior stops at about half of the episodes, whereas EvoBelief reaches a similar level on all three families, so the learned rules erase a difficulty gap that the prior cannot. On MG-R EvoBelief loses nothing before the switch, the two agents are level only in the first episodes after it, and EvoBelief pulls furthest ahead in the late post-switch episodes once the successor rule is committed, so the gain is not an artifact of a floor. On the development streams of Figure 6, the last dozen episodes of every stream are at or above their pre-switch level, and Frozen WM uses the full budget of 24 decisions in every episode and makes 178 model calls per episode against 98 for EvoBelief. Every method runs under the same harness, budget, seeds, and step limit. Development seeds were used while building the generic instantiation; the held-out seeds, their label permutation, the mechanism seeds, and the MG-R seeds were fixed before those runs.

Table 11: **MiniGrid baselines with transition memory (%).** Development seeds, 240 episodes per method, the same streams as Table 2. Recall of earlier transitions helps on MG-M, whose second toggle is witnessed by accident, and not on MG-C, whose lift condition is a conjunction. EvoBelief without transition memory (Table 2) exceeds every row.

Method	MG-C	MG-M	Pre-switch	Post-switch	All
ReAct	1.67	45.00	21.67	25.00	23.33
WorldEvolver	0.00	50.00	26.67	23.33	25.00
Frozen WM	3.33	35.00	21.67	16.67	19.17

Table 12: **What the MiniGrid gain needs (%).** Each block compares variants of EvoBelief on one seed set, 240 episodes per row. The planner without learned rules never leaves the start state of its search and defers to the prior on every decision; without active experiments, rules never gather independent support; in the shadow variant rules are learned and validated but no plan step is executed; without the commitment gate a rule may act on its first supported instance. Higher is better.

Seeds	Variant	Pre-switch	Post-switch	All
Development	Planner without learned rules	0.83	0.00	0.42
	No active experiments	0.83	0.00	0.42
	EvoBelief	**70.00**	**88.33**	**79.17**
Mechanism	Rules never executed (shadow)	50.00	64.17	57.08
	No commitment gate	66.67	70.00	68.33
	EvoBelief	**61.67**	**77.50**	**69.58**

Experiments double as exploration on MiniGrid. In the shadow variant of Table 12 the belief forms, tests, and validates rules exactly as in the full method and the planner computes the same plans, but no plan step is executed: every executed action comes from the prior or from an experiment. In episode 7 of MG-C seed 797 twelve of its fourteen decisions are experiment steps, which pick up the working key, open the door, close it again from the far room, and leave the agent one drop away from the box; the prior's draft finishes the task. In episode 16 the full method spends three experiment steps and then executes a four-step committed plan, entering the far room, closing the door, dropping the key, and lifting the box, while the shadow variant needs twelve decisions, eleven of them experiment steps, and succeeds only because the prior's last draft coincides with the plan step it was not allowed to take. The value of executing committed rules is therefore concentrated on MG-C, whose solution is a chain no single experiment completes, and it also shortens episodes and lowers cost. Acting on rules at their first supported instance, without the independent confirmation the commitment gate demands, leaves success and the rate of failed authorized steps close to the full method's on this benchmark, whose public transitions are deterministic; we report it as a null result rather than as support for the gate.

One rule's life, across the silent changepoint. The stream on MG-C seed 749 shows the lifecycle doing what drift demands. In episode 5 an active experiment validates a first door rule, that toggling opens the door while the blue key is held; the same episode refutes it, because the rule ignored the door's own state, and it is retired within three decisions. In episode 6 an experiment validates its successor, which conditions on the door being locked and on where the other keys lie, and that rule carries every plan through episode 20. The switch makes it fail at the first decision of episodes 21 and 22; two refutations in distinct episodes open a new evidence epoch for the claim, releasing the pre-switch evidence, and success resumes in episode 24 on experiment and prior steps. In episode 26 an active experiment validates the post-switch successor, that toggling opens the door while the green key is held, which then carries the remaining plans. No grant is ever archival: authority is always backed by currently surviving evidence, so the first generation does not survive its own counterexamples and the successor earns commitment with freshly built supports. On MG-M the same machinery commits the two-stage door as two rules, that the working key unlocks the door and that an empty-handed toggle then opens it, and relearns the first of them after the switch.

One decision flip, end to end. Our sharpest single data point is a matched trace (Figure 10), taken from an earlier development run of the prediction-revision route on the same environment, where a rule changes the predicted consequence of a candidate rather than a plan. After 137 decisions in lockstep, the Frozen WM agent selects `drop`, while EvoBelief's committed key rule changes the

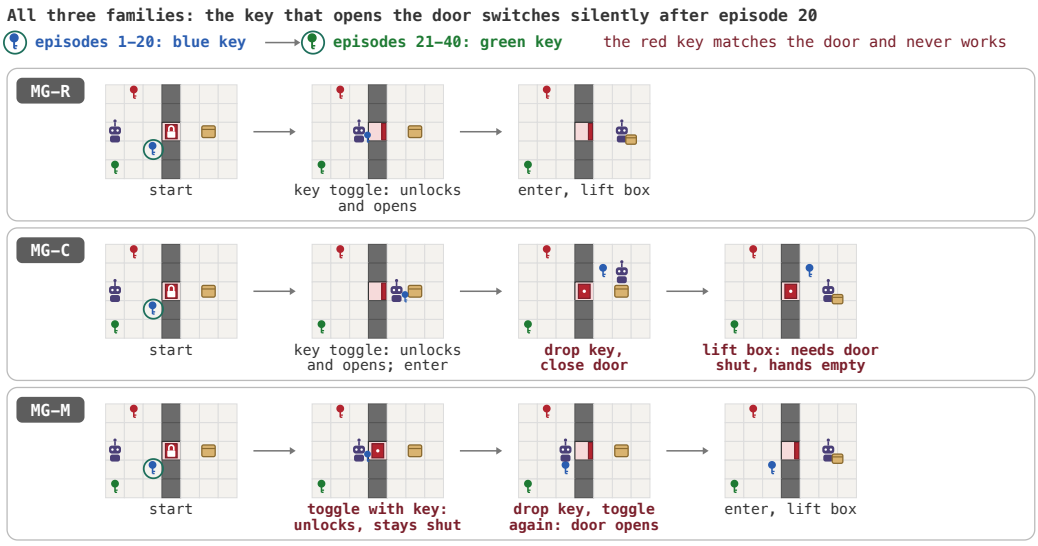

Figure 9: **What each MiniGrid family requires.** One row per family, each frame a state of the two-room grid: the robot is the agent, a key drawn next to it is the key it carries, the box drawn next to it is the lifted target. A padlock marks a locked door, a plain door is shut but unlocked, and a door drawn aside is open. In every family the red key matches the door and never works, the blue key opens the door for the first twenty episodes, and after episode 20 the green key does instead; the ring marks the key that opens the door and the switch is never announced. In MG-R (R for rebind) one toggle with the working key unlocks and opens the door, and nothing more is asked. In MG-C (C for conjunctive) the door behaves the same, but the box lifts only with the door shut and the agent's hands empty, so after entering the agent must drop the key and close the door it just opened. In MG-M (M for multi-stage) toggling with the working key only unlocks the door, which stays shut until a second toggle with empty hands. The switch is the only change within a stream; the family rules hold in both halves. The steps in dark red are the ones the pretrained prior does not expect.

predicted consequence of `toggle red door`, the ranking flips, and the next observation shows the door open. Every link from evidence to the confirming observation is recorded, and even the success the rule itself caused is not allowed to count as evidence for it. This is the prediction-to-action gap closed on one concrete decision.

G.5 Regime III: Discovery Details

Table 13 counts what happened on the twenty-four fresh Reactor Lab task instances, Table 14 lists the validated law of every task instance, and the record below shows one rule in full. A law handed to the agent does not substitute for the loop: a frozen belief preloaded with a calibration law of the right kind, formed on disjoint seeds and run without experiments on four Normal and six Challenge task instances, names the variable and earns half of the knowledge credit but never the coefficients, and completes nothing.

One rule in full. The sealed record of the rule validated on Normal seed 9, trimmed of hashes and file paths. The rule itself is what W stores; the validation is the prospective test that granted it authority; the use is the plan the actor executed under it, on a crystal the test never touched.

Rule record: Reactor Lab, Normal seed 9

```
rule HVA6B0D2A654F9   family FUNCTIONAL_RELATION   lifecycle committed_reusable
  relation   resonance_frequence = 104.997462 * channel_5 + 107.251721
  scope      subject: quantum crystal; input: channel_5 read by the spectrometer;
             output: resonance_frequence published by the crystal reactor holding it
  evidence   3 independent supports, 0 counterexamples
```

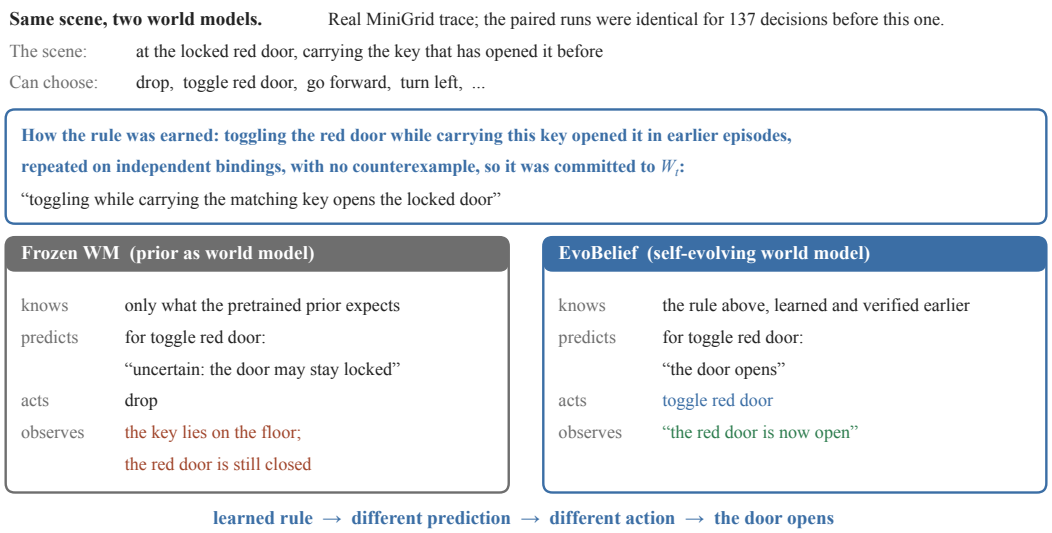

Figure 10: **One decision flip on MiniGrid, from an earlier run of the prediction-revision route.** The paired runs are identical for 137 decisions; then the agent that consults its committed key rule toggles the door, and the agent that follows the prior drops the key.

Table 13: **What EvoBelief did on the fresh task instances.** Counts from the sealed science records of the twelve task instances per tier; the baselines have no such record by construction. A hypothesis is a candidate law with a prediction frozen before any experiment. One executed experiment settles every rival whose frozen prediction its realized outcome contradicts, so fewer experiments falsify more rivals, and a law is validated only by a held-out prediction the reactor confirms, never by fit quality alone. Model calls and environment actions are totals over the tier.

	Normal	Challenge
Hypotheses frozen	236	433
Experiments executed	48	48
Rivals falsified	157	139
Laws validated	10	8
Task instances with a validated law	10	8
Task instances whose law-derived plan passed	10	7
Task instances completed	10	7
Actions taken under a validated law	115	94
Model calls, EvoBelief	2744	4988
Model calls, ReAct	19559	18790
Model calls, WorldEvolver	6108	6949
Environment actions, EvoBelief	2026	2737
Environment actions, ReAct	3839	3540
Environment actions, WorldEvolver	3051	3472

```
validation, a prospective test
  fit before the test     f = 105.0 * channel_5 + 107.0   (held-out crystal excluded)
  held-out crystal        channel_5 = 30.5, frozen prediction 3309.5 Hz
  intervention            reactor dialed to 3310 Hz; observed state: activated
  verdict                 settled_pass, authority granted

use, a plan derived from the rule
  target crystal          channel_5 = 54.17, frozen prediction 5794.96 Hz
  intervention            reactor dialed to 5795 Hz; observed state: activated
  outcome                 consumption_pass; the task is then submitted
```

Plant Nutrients: protocol. The theme's hidden rule is categorical. A first version of the Plant Nutrients instantiation was run on six fresh task instances under a pre-declared performance gate and failed it: it completed no task instance, although its rules had recovered the hidden nutrient rule on every Normal task instance and a table consistent with the hidden combination on every Challenge

Table 14: **The law validated on each fresh task instance.** f is the reactor resonance in Hz; c_5, d, t, and q are the crystal's spectrometer channel 5, density, temperature, and quantum gap. x is the held-out crystal's reading, Predicted the prediction frozen before the reactor was dialed, and Dialed the frequency at which the reactor then activated. Rivals is the number of rival laws falsified on that task instance, Plan the outcome of the plan derived from the law for the target crystal (a passed plan completes the task), and Knowledge the judge's score on that task instance. Normal seeds 13 and 18 and Challenge seeds 9, 14, and 19 hide the variable in a radiation reading with random background; on Challenge seed 20 no rival survived within budget, and on seed 17 the law was validated but its plan never settled.

Seed	Validated law	x	Predicted	Dialed	Rivals	Plan	Knowledge
Normal							
s9	$f = 104.99\,c_5 + 107.25$	30.50	3309.50	3310.00	18	pass	1.00
s10	$f = 99.01\,d + 102.24$	61.31	6172.69	6173.00	14	pass	1.00
s11	$f = 105.00\,t + 95.96$	17.81	1966.05	1966.05	18	pass	1.00
s12	$f = 105.01\,q + 90.40$	49.72	5311.60	5312.00	16	pass	1.00
s13	none (radiation seed)				7	–	0.00
s14	$f = 102.99\,c_5 + 91.92$	39.09	4118.27	4118.27	12	pass	1.00
s15	$f = 108.00\,d + 99.69$	34.31	3805.48	3805.48	12	pass	1.00
s16	$f = 106.01\,t + 102.64$	46.75	5058.50	5059.00	18	pass	1.00
s17	$f = 96.00\,q + 90.93$	69.78	6789.88	6790.00	10	pass	0.50
s18	none (radiation seed)				9	–	0.00
s19	$f = 105.99\,c_5 + 98.23$	16.28	1823.68	1824.00	12	pass	1.00
s26	$f = 105.99\,t + 100.10$	48.04	5192.24	5192.24	11	pass	1.00
Challenge							
s9	none (radiation seed)				4	–	0.00
s10	$f = 14.00\,t^2 + 32.99\,t + 568.99$	11.50	2800.00	2800.00	17	pass	1.00
s11	$f = 17.00\,d^2 + 25.85\,d + 510.48$	15.61	5058.63	5059.00	21	pass	1.00
s12	$f = 17.01\,q^2 + 20.72\,q + 783.45$	5.49	1409.67	1410.00	15	pass	1.00
s13	$f = 17.99\,c_5^2 + 29.04\,c_5 + 645.76$	12.84	3985.97	3986.00	18	pass	1.00
s14	none (radiation seed)				7	–	0.00
s15	$f = 19.00\,t^2 + 29.88\,t + 212.78$	18.57	7321.15	7321.15	15	pass	1.00
s16	$f = 17.99\,d^2 + 33.16\,d + 41.08$	11.85	2960.66	2961.00	5	pass	0.50
s17	$f = 12.89\,q^2 + 23.02\,q + 790.49$	7.12	1608.01	1608.55	9	unsettled	0.50
s18	$f = 14.99\,c_5^2 + 24.03\,c_5 + 581.88$	17.27	5470.28	5470.27	18	pass	1.00
s19	none (radiation seed)				3	–	0.00
s20	none				7	–	0.50

task instance. The failure had three named causes. A correct rule was retired by a counterexample that was not one, because growth also requires a seed in the tile and seeds inside tiles are not published; the validated rule set the soil controller but did not carry the rest of the planting chain, which the actor took without the rule; and the knowledge surface described what appeared on the soil rather than what a planted seed would do, which the judge did not credit. The version reported in Table 3 is a revised implementation that answers those three causes. A counterexample by absence is admitted only after the plan has publicly created the causal opportunity, that is, after the deposit, the refill, and a pre-declared number of maturation ticks; a validated rule controls the whole plan, from configuring the controller through binding, acquiring and depositing the seed, waiting for maturation, and observing; and the knowledge surface renders the causal chain and is scoped to the task instance. Development used only task instances already consumed by the pilot. The code was then frozen and run once on the twelve Normal and twelve Challenge task instances of a fresh set that had been enumerated blind before the pilot ran and never executed by any method. One repair, the identification of the actuator on a previously used development Challenge instance, was landed after the development checks passed and was re-validated offline and on one previously used development instance; the pilot is never pooled with the reported result.

Plant Nutrients: mechanism. The DW Plant column of Table 6 counts what happened on the twenty-four fresh task instances. Of the nineteen completed task instances, seventeen completed through a validated task plan; one completed through a discriminating plan whose task plan fell short, and one completed with no plan. The retirements are informative: fifteen committed rules were retired by counterexamples, nine of them after the plan itself had created the opportunity, and

Table 15: **The Hypothesizer at its original 3,072-token output ceiling.** Same metrics and task instances as Table 3, in percent; the Challenge runs ended on output truncation, and their means are over the task instances sealed before the failure.

Family	Tier	Sealed	Completion	Knowledge	Procedural
Reactor Lab	Normal	12 of 12	0.00	0.00	74.33
Reactor Lab	Challenge	2 of 12	0.00	0.00	74.14
Plant Nutrients	Normal	12 of 12	66.67	0.00	82.41
Plant Nutrients	Challenge	1 of 12	0.00	0.00	66.67

thirty-five absence observations that would have counted as counterexamples in the first version were deferred because the opportunity had not yet been created. Knowledge margins over the baselines, in percentage points with paired bootstrap intervals: on Normal, 58.3 [33.3, 83.3] over ReAct, 66.7 [25.0, 100.0] over WorldEvolver, and 75.0 [50.0, 100.0] over the Hypothesizer; on Challenge, 33.3 [8.3, 58.3] over ReAct, WorldEvolver, and the Hypothesizer. Completion contrasts are in Table 7.

One Plant Nutrients rule in full. The rule validated on Normal seed 9, summarized from the sealed record.

Rule record: Plant Nutrients, Normal seed 9

```
rule HC2312AAC19FAD   family CATEGORICAL_ASSOCIATION   lifecycle standing
  relation   thorium = high -> mushroom ; thorium = low -> none ; thorium = zero -> none
  evidence   2 tests executed: 1 support for this rule, 1 rival falsified,
             0 counterexamples

use, a plan derived from the rule
  setting    thorium = high on the field's soil nutrient controller
  chain      configure, plant, refill, wait for maturation, observe
  outcome    2 plantings, 2 observed growths, 0 absences; requirement
             satisfied; the task is then submitted
```

The Hypothesizer baseline at two output ceilings. The Hypothesizer consolidates its knowledge into one long structured reply per step. Under the output ceiling frozen for the original campaigns, 3,072 tokens, that reply was truncated on both Challenge tiers, three bounded resends failed the same way, and the pre-declared attempt policy classes a failure at a frozen input or output admission as a method failure that is not retried: the Reactor Lab run sealed two Challenge task instances and the Plant Nutrients run one. Table 15 reports those original runs. For a complete baseline we re-ran the Hypothesizer whole-arm under a separately frozen identity whose only change is that ceiling, raised to 8,192 tokens, on the same task instances by hash, one attempt per cohort, blind until every cohort had sealed. Prompt, policy, backbone, budgets, populations, and judge were unchanged, and no other method was re-run. Under the raised ceiling eleven consolidation replies exceeded the old ceiling and none approached the new one, so each of them would have been a truncation before. Those re-runs are the Hypothesizer rows of Table 3. The two Normal re-runs also show the run-to-run variation of the backbone: the Plant Nutrients completion count moved from eight to six on the same twelve task instances with the same prompt.

G.6 Analysis Notes

Held-out prediction evaluation. Table 16 scores 1,500 transitions per benchmark from episodes disjoint from every episode on which the belief was formed (ScienceWorld: 90 development episodes; ALFWorld: the 140 seen-split development episodes, with test transitions drawn from the unseen split), each scored once, with Haiku 4.5 as the backbone for every row, one frozen prediction prompt, and greedy decoding. State-Δ F1 is the set F1 between predicted and realized canonical state-change atoms, averaged over transitions, with a correct prediction of no change scoring one; Feasibility is agreement on whether the action executes; EM and Token F1 compare one deterministic rendering of the predicted and the realized state record. Frozen WM is the controller without W; the WorldEvolver rows run the official implementation with its semantic memory alone and with both memories. The ALFWorld block is a post-hoc rescoring of sealed structured predictions under the ScienceWorld metric definitions, which were fixed before any ALFWorld aggregate under them was computed. Under a paired bootstrap over transitions (10,000 resamples), EvoBelief's lead over every other row on State-Δ F1, EM, and Token F1 has a 95% interval excluding zero on both

Table 16: Held-out structured state-change prediction on ScienceWorld and ALFWorld, 1,500 transitions per benchmark, disjoint from belief formation. State-Δ F1 measures canonical state changes; Feasibility measures action executability; EM and Token F1 compare structured state. Haiku 4.5.

	Model	State-Δ F1 ↑	Feasibility ↑	EM ↑	Token F1 ↑
ScienceWorld	Frozen WM (no evolving W)	0.7258	0.7547	0.6013	0.7866
	WorldEvolver (semantic-only)	0.7061	0.7827	0.6027	0.7814
	WorldEvolver (full)	0.6680	**0.8753**	0.5993	0.7820
	EvoBelief	**0.7530**	0.8113	**0.6407**	**0.8045**
ALFWorld	Frozen WM (no evolving W)	0.7572	0.8453	0.6567	0.8080
	WorldEvolver (semantic-only)	0.6862	0.8500	0.5733	0.7607
	WorldEvolver (full)	0.7250	0.8400	0.6060	0.7849
	EvoBelief	**0.8473**	**0.8513**	**0.7613**	**0.8698**

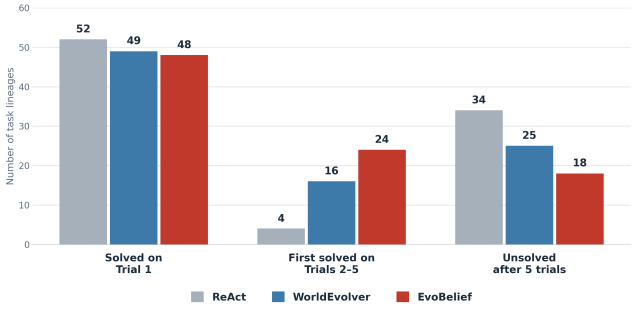

Figure 11: First-success distribution over 90 ScienceWorld task lineages (5 trials each); W persists across trials and resets between lineages. All methods solve a similar number of lineages on the first attempt; EvoBelief converts later trials into 24 additional first-solves, versus 16 for WorldEvolver and 4 for ReAct.

benchmarks; on Feasibility, WorldEvolver-full's lead on ScienceWorld excludes zero, while every ALFWorld difference has an interval that includes zero.

G.7 Learning over the Deployment Lifetime

Each lineage is the same task instance attempted five times in a row with the same seed, actor, and budgets. Within a lineage, W accumulates from trial to trial, and it is reset before the next lineage begins. Memory baselines accumulate and reset on the same schedule, and later-trial recovery is evaluated relative to them under that schedule.