NO DO-OVERS: OBSERVATION-GROUNDED CREDIT FOR LANGUAGE-MODEL AGENTS WITHOUT RE-EXECUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Improving a long-horizon language-model (LM) agent is a problem of *credit assignment*: localizing a terminal outcome to the steps that earned it. We study it in the regime a *deployed* agent faces (**credit assignment in deployed agents**), where a text-action LM agent must learn from what it logged. A deployed agent updates its policy *off-peak*, while it is not serving user traffic, on the trajectories that serving already produced, and by update time cannot rewind a session, re-execute a tool call, or consult an oracle: its actions are *irreversible*. Yet the dominant tools for per-step credit in agentic RL rely on exactly the *counterfactual access* a deployed agent lacks, so credit must come from the logged trajectory alone. One signal the rule leaves readable is *the observation*, which in a text-action setting states in words what each action did. To score unexecuted candidate completions in pre-action GRPO under the irreversible access constraint, we first introduce **OG-Gen** (Observation-Grounded Generative Synthesis), which uses a frozen language model to predict synthetic observation text offline without live environment re-execution. Over these observations (logged post-action or synthesized pre-action via OG-Gen), we read credit *directly* off explicit text observations along execution, subgoal, and compliance signals with no *trained* reward or value model on the credit path (**observation-grounded step credit**). As a GRPO reward it improves long-horizon agents across three environments using only logged information. On ALFWorld, two rounds raise the base from 67.2 to 78.4, the highest among same-backbone methods and on par with a privileged comparator that uses strictly more information; the same operator improves over its terminal-credit ablation and strong baselines on ScienceWorld and Deep Search.

1 INTRODUCTION

Consider a deployed language-model agent carrying out long-horizon tasks (such as system administration or tool calls) that we want to continuously improve. Users cannot wait for policy updates, so training occurs off-peak using only logged serving trajectories. By update time, the agent cannot rewind sessions, re-execute tool calls for alternative returns, consult oracles, or copy unprovided expert demonstrations: all it has is the logged trace. Can it still learn from those logs alone? Each off-peak update feeds subsequent deployments, so improvement must compound across rounds (Fig. 1).

The difficulty is concrete: trajectories run dozens of steps, rewards are sparse and terminal, and the action space is text. Consider an ALFWorld agent that must clean a soapbar and put it in a cabinet. Trained on the terminal outcome alone, it never gets the soapbar into its hand.[1]

> **Outcome-only credit.** `open cabinet 1`... `open cabinet 4`, at none of them → "*Nothing happens.*"×4; `go to countertop 1` → "*... you see a soapbar 1*"; `clean soapbar 1 with sinkbasin 1` → "*Nothing happens.*", having never taken it; fifteen "*Nothing happens.*" in all, then eight identical think-loops, dead at the 49-step cap.

[1] Task `pick_clean_then_place_in_recep-SoapBar-None-Cabinet-424`; the failing trace is our terminal-credit ablation (outcome-only GRPO).

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.

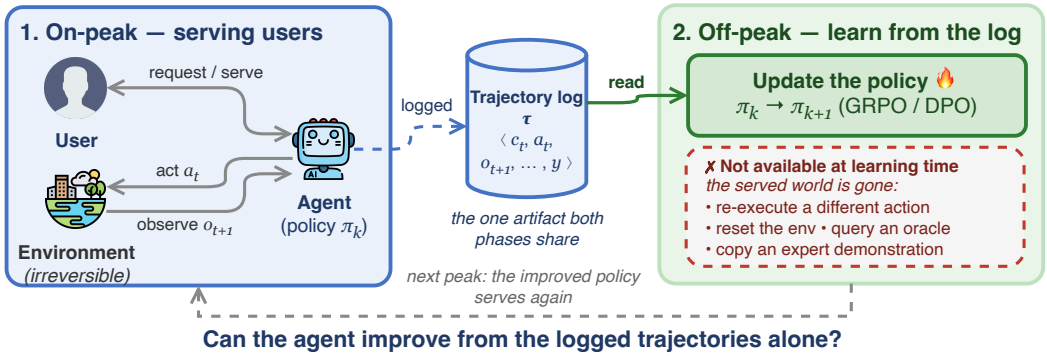


Figure 1: **Learning from deployment logs alone.** On-peak (left) the agent serves users in a live environment; only the logged trajectory survives to update time. Off-peak (right) the policy is updated from that log alone and serves the next peak, so credit must come from the log itself.

Every wasted step stems from an unfulfilled prerequisite: the agent acts at an unchecked cabinet or cleans a soapbar it never holds. The environment observation, "*Nothing happens.*", explicitly flags that the action had no effect. In other words, the per-step signal sits right in the log. But terminal rewards cannot point to which step lost the soapbar. The difficulty is not obtaining a learning signal but localizing one already present to the responsible step.

This scenario defines deployed-agent reinforcement learning: an offline RL setting where policies are updated off-peak using logged serving traces without live environment queries. Unlike classical offline RL with pre-defined state vectors and dense step rewards, deployed language agents log unstructured text observations and sparse binary outcomes (success or failure).

Reinforcement learning reduces policy improvement to credit assignment: policy-gradient methods (e.g., GRPO (Shao et al., 2024)) and preference-optimization methods (e.g., DPO (Rafailov et al., 2023)) turn policy improvement into scoring each step's outcome share. However, standard agent recipes generate fresh rollouts each round (such as GRPO or iterative DPO (Song et al., 2024)) and score candidate completions pre-action, without logging post-action environment states.

Classical offline RL methods (such as learned Q-critics, return redistributors (Arjona-Medina et al., 2019), and process-reward models (Lightman et al., 2024)) fail in this text-action setting. Specifically, learned redistributors show near zero correlation with step correctness, pre-action LM self-judgment exhibits severe overconfidence on 75–81% of failed steps (the belief–observation gap), and likelihood-based rewards suffer from candidate-blind credit degeneration under GRPO (Sec. 2).

To address these challenges under the irreversible access constraint, we first introduce **OG-Gen** (Observation-Grounded Generative Synthesis), which uses a frozen language model to predict synthetic observation text offline without live environment re-execution. We then apply **observation-grounded step credit**: a learning-free operator reading execution, subgoal-verifier, and compliance signals directly from explicit text observations without trained reward or value models. Used with GRPO, it matches or beats alternatives across three environments. On ALFWorld, two training rounds raise a 67.2 base to 78.4, matching a privileged post-action-return comparator (77.6, on a different backbone) using strictly less information, while improving over terminal-credit ablations and strong baselines on Deep Search and ScienceWorld (Xi et al., 2025b).

Contributions.

1. An **irreversible access constraint** (credit may read any logged observation, never a fresh response to a generated action), under which learned redistributors and candidate-blind likelihood routes fail and self-belief remains miscalibrated (Sec. 2).
2. **OG-Gen and Observation-Grounded Step Credit**: We introduce OG-Gen (Generative Observation Synthesis) to predict candidate text observations offline via frozen model forward passes, bypassing live environment re-execution during pre-action GRPO candidate scoring, coupled with a learning-free operator reading credit from observations across execution status, subgoal verification, and format compliance signals (Sec. 4).

3. As a GRPO reward it improves agents across three environments: raising the ALFWorld (Shridhar et al., 2021) base $67.2 \rightarrow 78.4$ ($11.2\,\%$, highest among same-backbone methods) at lower rollout cost, and outperforming terminal-credit ablations and strong baselines on Deep Search (Xi et al., 2025b) and ScienceWorld.

2 Irreversible Environment Access and Why Obvious Routes Fail

Setting and access constraint. Informally, at credit time the agent has only the transcript of what it did, so any per-step credit must be computable from it alone. An LM agent interacts with an environment over a horizon T: at step t it observes context c_t (the task prompt and history so far), emits a text action a_t, and the environment returns an observation o_{t+1}. The episode is logged as $\tau = (c_0, a_0, o_1, \ldots, c_{T-1}, a_{T-1}, o_T)$ with terminal outcome $y \in \{0, 1\}$. Temporal credit assignment asks for a per-step credit $c_t \mapsto r_t$ that localizes y to the steps that earned it. We assume no environment access at credit-assignment time, so the credit for step t must be a function only of logged quantities,

$$r_t \;=\; f\big(\tau_{\leq \kappa(t)}\big), \tag{1}$$

Here, $\kappa(t)$ indexes the available logged prefix when crediting step t (c_t for pre-action GRPO reads, or o_T for post-action offline DPO reads; see Sec. 4). Consequently, per-step credit cannot depend on observations generated by executing unlogged candidate actions. Three sources are therefore forbidden: (a) re-running the environment from c_t with a different action, (b) an oracle that labels step t, and (c) an expert demonstration to imitate. What is allowed is a model forward pass rescoring logged content, since that queries the model, not the environment. The test is whether every observation the credit reads is already logged. The constraint thus binds the learning signal, not the optimizer: one may sample candidate actions, but never execute one.

Relation to classical offline RL. Deployed-agent credit assignment under irreversible environment access is a real-world instance of classical offline RL (Minsky, 1961; Sutton, 1988; Lange et al., 2012; Levine et al., 2020): policy optimization takes place off-peak from logged serving traces without live environment resets or re-execution. However, text-action LM agents violate standard offline RL assumptions in two key ways. First, logged transitions contain only unstructured text observations o_t and sparse terminal outcomes y, rather than scalar step rewards $r(s, a)$ or structured state vectors. Second, policy update sampling (e.g., in GRPO) evaluates candidate completions $\{a^{(i)}\}_{i=1}^{G}$ pre-action, meaning unexecuted candidate branches lack logged post-action states s_{t+1} in the dataset. Standard offline RL tools (learned Q-critics $\hat{Q}_\psi$, TD bootstrapping, or parametric return redistributors) inherit severe estimation noise in text LMs, degrading credit quality (see Sec. B.2 for full separations across classical state and temporal credit paradigms).

Equation (1) removes classical value bootstrapping via re-execution; under it, the three alternative approaches differ in how they calculate per-step credit. Routes (i) and (ii), a learned redistributor of step returns (Arjona-Medina et al., 2019; Ren et al., 2022) and the agent's own self-judgment (Yuan et al., 2025), calculate credit by training a model on returns or eliciting the model's self-confidence. Our operator instead reads credit directly from the logged observation text. Route (iii), which applies likelihood rewards to logged actions (Wang et al., 2026b; Oh et al., 2026), suffers from candidate-blind credit assignment under GRPO. Because it scores historical logged actions rather than evaluating candidate completions $a^{(i)}$, reward values remain identical across the group ($r_{i,t} = c$), causing within-group advantage variance to degenerate to zero ($\sigma^2 = 0$; Fig. 2c).

A learned redistributor targets our exact setting. We instantiate its natural LM form: prompt the model with the full logged trajectory and its terminal return, asking for per-step credit that redistributes that return under the RUDDER/RRD constraint (Arjona-Medina et al., 2019). Across scales this credit is uncorrelated with ground-truth step correctness (Pearson $r \approx 0$, no scale trend; Fig. 2b, Sec. B.2), and ranks candidates at AUC 0.63 on within-group ranking tests, far below our observation read (0.75).

The belief–observation gap. Once oracles and re-execution are forbidden, the self-judgment route reads the agent's own pre-action step self-judgment (Kadavath et al., 2022) (Fig. 2). Evaluated on BFCL (Patil et al., 2025) for gold step labels, the model asked before acting whether its step will succeed is confidently wrong on a flat 75–81% of failed steps across 8B to 235B models (Sec. B.1). Cru-

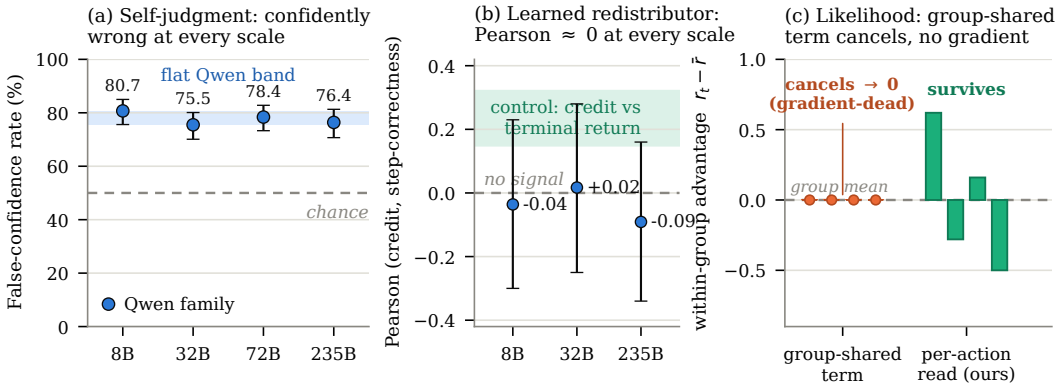


Figure 2: **The learned and likelihood routes fail, and pre-action self-belief is miscalibrated.** (a) Self-judgment is confidently wrong at every scale: on failed steps the model's pre-action success belief stays well above chance (75–81%) with no scale trend. (b) A learned redistributor is uncorrelated with step correctness ($r \approx 0$). (c) The likelihood route is candidate-blind, yielding zero within-group contrast under GRPO. Details in Secs. B.1 and B.2.

cially, when shown the logged post-action observation, false confidence collapses to 3–5%: reading what happened is reliable where predicting what will happen is not. Formally, this asymmetry stems from the conditional entropy of step outcomes: pre-action self-belief operates under unobserved environment transitions $P(o_{t+1} \mid c_t, a_t)$, yielding high predictive uncertainty $H(\text{succ} \mid c_t, a_t)$ that the model resolves through overconfident priors. In contrast, post-action observation credit reads o_{t+1} directly, capturing non-zero mutual information $I(y; o_{t+1} \mid c_t)$ about task progress without relying on model-internal dynamics simulation. This reveals a crucial distinction: self-elicitation queries the model's internal confidence before acting (which inherits generation overconfidence), whereas decoupled verification evaluates the external environment observation o_{t+1} against a frozen goal schema prompt at temperature 0, decoupling credit assignment from model generation bias. On a within-group ranking test against a goal-advancement label, our observation read ranks candidates at AUC 0.75, well above the prompted redistributor's 0.63 (Sec. B.2).

Our framework resolves these three failure modes by introducing **OG-Gen** (Sec. 4.1), which predicts synthetic observation strings $\hat{o}_{t+1}$ via frozen LM forward passes, adhering strictly to Eq. (1). This converts unobserved candidate actions into explicit textual state transitions offline without live environment re-execution. Using these text observations (logged post-action or synthesized pre-action via OG-Gen), our operator evaluates state transitions against frozen schemas across execution (E), subgoal (V), and compliance (O) dimensions without trained models. By synthesizing candidate-specific observation strings $\hat{o}^{(i)}_{t+1}$ for each completion $a^{(i)}$, OG-Gen closes the belief-observation gap (Route ii), avoids return redistributor noise (Route i), and resolves candidate-blind credit degeneration (Route iii), thereby providing candidate-discriminative step credit ($\sigma^2(\{r_{j,t}\}) > 0$).

3 Related Work

Offline RL, temporal credit assignment, and return decomposition. Localizing delayed outcomes to earning steps is RL's founding problem (Minsky, 1961), addressed classically via TD learning (Sutton, 1988) and GAE (Schulman et al., 2016) through repeated environment interaction. Offline RL (Lange et al., 2012; Levine et al., 2020) and return-decomposition methods (Arjona-Medina et al., 2019; Ren et al., 2022; Harutyunyan et al., 2019; Mesnard et al., 2021; Hung et al., 2019; Raposo et al., 2021) share our setting of learning strictly from logged interaction datasets without environment resets. However, while classical offline RL relies on fitting parametric Q-critics or reward models over numeric state transitions (s, a, r, s'), LM agents present unstructured text observations, sparse terminal outcomes, and pre-action candidate completion sampling where unexecuted branches lack logged post-action transitions. In text LMs, parametric redistributors score Pearson $r \approx 0$ against step correctness due to the belief–observation gap (Sec. 2); in contrast, our operator uses OG-Gen to synthesize candidate text observations and performs a learning-free observation-semantic read. Additional classical state discretization and trace credit paradigms are separated in Sec. B.2. Parallel logged-data methods (Swaminathan & Joachims, 2015; Jiang & Li, 2016; Precup

et al., 2000; Chen et al., 2021) output scalar policy values or require propensities/learned reward models that our read avoids.

Agent RL and step-level credit for LM agents. LM agent RL (Song et al., 2024; Xi et al., 2025a; Qi et al., 2025; Xiong et al., 2024; Xiao et al., 2025) and reward-filtered self-training (Zelikman et al., 2022; Gulcehre et al., 2023; Singh et al., 2023; Aksitov et al., 2024) either learn a reward model or assign uniform credit by terminal outcome (instantiated by our RFT floor alongside a BC control). Critic-free localized credit like GiGPO (Feng et al., 2025) groups post-action returns across sibling rollouts from repeated states, requiring fresh post-action rollouts unavailable in logged single trajectories. Our operator provides a learning-free step-credit read of single logged trajectories without post-action branching.

Concurrent step-level baselines and access alignment. Concurrent step-level methods sort under two exhaustive arguments. Under degeneration, ablating forbidden counterfactuals yields baselines we match or beat: StepPO's learned critic (Wang et al., 2026a), SIOP's turn potential (Hu et al., 2026), and IGPO/CIGPO/HCAPO (Wang et al., 2026b; Dou, 2026; Tan et al., 2026) reduce to outcome-only or log-probability-gain cells. Under unfair advantage, off-setup methods win only via deployment-forbidden information. Access-matched methods like Progress-Advantage (Oh et al., 2026) are run as-published head-to-head, where likelihood reads yield zero gradient signal on homogeneous candidate sets (Sec. 5.1).

Process reward models and self-verification. Process reward models (Lightman et al., 2024; Wang et al., 2024) rely on search or human step labels (counterfactual access). Our belief–observation gap highlights that LMs fail to reliably self-verify or locate their errors (Huang et al., 2024; Stechly et al., 2025; Tyen et al., 2024), and single-turn calibration (Kadavath et al., 2022) does not transfer to multi-turn agent interaction.

4 METHOD: OBSERVATION-GROUNDED STEP CREDIT

While standard methods read environment observations, our approach differs in how text feedback is converted into numerical credit. We commit to a learning-free design where credit is computed as a closed-form function of observation semantics. There are no trained parameters in the credit path: no reward model, value function, learned redistributor, or self-elicited belief score. Where a term consults a language model it does so as a frozen verifier answering a fixed question at temperature 0, never a component fit against outcomes: the distinction is untrained, not model-free.

4.1 OG-GEN: GENERATIVE OBSERVATION SYNTHESIS FOR CANDIDATE CREDIT

The pre-action credit dilemma under irreversible access. In on-policy or off-peak agent reinforcement learning (e.g., GRPO candidate sampling), the policy samples a candidate group of completions $\{a^{(i)}\}_{i=1}^{G}$ at context c_t. However, under the irreversible access constraint (Eq. (1)), real physical environment observations $o_{t+1}^{(i)}$ exist only for the single trajectory that was executed during deployment; unexecuted candidate actions $a^{(i)}$ ($i \geq 2$) have no physical observation returned by the live environment. Re-executing candidate completions in the live environment is forbidden during off-peak training, while direct scalar self-belief scoring suffers from severe miscalibration (Sec. 2).

Two-stage generative observation pipeline. To enable observation-grounded credit for unexecuted candidates without live environment re-execution, we introduce **OG-Gen** (Observation-Grounded Generative Synthesis). OG-Gen decouples environment state transition simulation from credit valuation through a two-stage generative synthesis pipeline:

1. **Stage 1: Offline Text-World Simulation (P_{world}).** A frozen language model operates as an offline text-world simulator P_{world}, predicting the synthetic post-action observation text $\hat{o}_{t+1}^{(i)}$ that would result from executing candidate completion $a^{(i)}$:

$$\hat{o}_{t+1}^{(i)} \sim P_{\text{world}}\left(\cdot \mid c_t, a^{(i)}\right). \tag{2}$$

Because P_{world} is queried purely via an offline model forward pass on existing context strings $(c_t, a^{(i)})$, Stage 1 strictly satisfies Eq. (1) without querying live environment endpoints or mutating session state.

2. **Stage 2: Deterministic Synthesized State Valuation ($r_{i,t}^{\text{OG-Gen}}$).** Once the synthetic observation text $\hat{o}_{t+1}^{(i)}$ is generated, the observation-grounded credit operator evaluates the candidate action directly on the synthetic text string:

$$r_{i,t}^{\text{OG-Gen}} \;=\; \alpha\, E\left(\hat{o}_{t+1}^{(i)}\right) \;+\; \beta\, V_t\left(c_t, a^{(i)}, \hat{o}_{t+1}^{(i)}\right) \;+\; \gamma\, O_t(a^{(i)}, y), \tag{3}$$

where $E(\hat{o}_{t+1}^{(i)})$ checks execution state transitions on the synthetic text (e.g., verifying whether $\hat{o}_{t+1}^{(i)}$ indicates execution failure such as `"Nothing happens."`), and V_t evaluates subgoal progress from the synthetic observation string.

Theoretical & practical properties of OG-Gen. OG-Gen provides three crucial theoretical and operational guarantees. First, **collapsing the belief–observation gap**: rather than querying an internal scalar belief decoder $P(\text{succ} \mid c_t, a^{(i)})$ (which inherits generation overconfidence; Sec. 2), OG-Gen forces the model to generate explicit environment response text $\hat{o}_{t+1}^{(i)}$. Grounding the verifier V_t in the explicit generated text string decouples step valuation from internal scalar belief, reducing false confidence and aligning step scores with physical reality. Second, **guaranteeing candidate-discriminative step credit**: because candidate actions $a^{(i)}$ produce distinct synthetic observation strings $\hat{o}_{t+1}^{(i)}$ (e.g., opening a valid cabinet vs. attempting to clean an unheld object), $r_{i,t}^{\text{OG-Gen}}$ evaluates candidate-specific execution semantics. This yields correctness-aligned within-group contrast ($\sigma^2(\{r_{j,t}^{\text{OG-Gen}}\}_{j=1}^{G}) > 0$), resolving the candidate-blind credit degeneration that degrades likelihood-based rewards (Eq. (8)). Finally, **bypassing environment execution**: OG-Gen eliminates the need for live environment re-execution, environment resets, or interactive API queries during policy optimization, enabling fast, parallelized off-peak training from logged data alone.

4.2 Decomposition and Observable Definitions

Having established how observations are obtained—either read directly from post-action logs or synthesized pre-action via OG-Gen (Eq. (3))—each step receives credit combining three signal classes, all computed without trained parameters:

$$r_t \;=\; \alpha\, E_t \;+\; \beta\, V_t \;+\; \gamma\, O_t, \tag{4}$$

where the terms are defined formally over the observation o_{t+1} (logged post-action observation or synthetic $\hat{o}_{t+1}$) and candidate action a_t:

$$E_t \;=\; \mathbb{I}\left[o_{t+1} \notin \mathcal{O}_{\text{error}} \wedge \Delta\text{state}(o_t, o_{t+1}) \neq \emptyset\right] \in \{0, 1\} \qquad \text{(Execution validity check)} \tag{5}$$

$$V_t \;=\; \mathbb{P}_{\text{verifier}}(\text{subgoal_achieved} = 1 \mid c_t, a_t, o_{t+1}) \in [0, 1] \qquad \text{(Frozen LLM verifier)} \tag{6}$$

$$O_t \;=\; \mathbb{I}\left[a_t \in \mathcal{A}_{\text{valid_format}}\right] + \lambda \cdot y \qquad \text{(Compliance \& terminal outcome)} \tag{7}$$

In GRPO (Shao et al., 2024), candidate completions $\{a^{(i)}\}_{i=1}^{G}$ sampled at decision point c_t receive step rewards $r_{i,t}$. Rather than scoring absolute quality, GRPO normalizes $r_{i,t}$ within the candidate group to form the step-level advantage:

$$\hat{A}_{i,t} \;=\; \frac{r_{i,t} \;-\; \mu\left(\{r_{j,t}\}_{j=1}^{G}\right)}{\sigma\left(\{r_{j,t}\}_{j=1}^{G}\right) \;+\; \epsilon}, \tag{8}$$

which modulates the token-level policy gradient update $\nabla_\theta \mathcal{L}_{\text{GRPO}}(\theta)$ without requiring a trained critic network.

Execution transitions and verifier mechanics. The execution term E_t of Eq. (5) is evaluated via closed-form state-transition rules, penalizing invalid actions (such as interacting with unopened containers or unheld targets). For logged trajectories, the frozen verifier $V_t(c_t, a_t, o_{t+1})$ evaluates intent-action-observation alignment by querying a static schema prompt at temperature 0 without parameter updates. This ensures that credit remains learning-free while supplying high within-group discriminative variance.

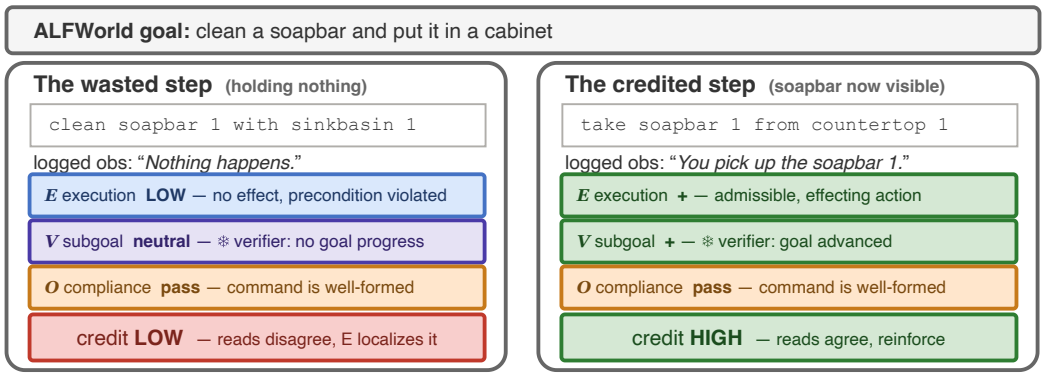


Figure 3: **Observation-grounded step credit reads each logged step.** On one ALFWorld task, the operator scores two steps along three axes read from the logged observation: execution E, subgoal V, compliance O. No trained parameters lie on the credit path: where a term consults an LLM (here V) it is a frozen verifier at temperature 0. The combined credit feeds GRPO as a reward normalized within the candidate group, where OG-Gen synthesizes pre-action candidate observations; the offline DPO variant instead reads the logged post-action observation.

Necessity of candidate-discriminative observation grounding for GRPO. GRPO's group-relative advantage (Eq. (8)) relies on within-group contrast $\hat{A}_{i,t} \propto r_{i,t} - \mu(\{r_{j,t}\})$. Crucially, non-zero variance is a necessary but insufficient condition for policy optimization: arbitrarily injecting variance (e.g., via decoding noise) produces noisy or reward-hacking gradients, while candidate-blind rewards yield zero contrast. If a reward is ungrounded—such as likelihood rewards that evaluate historical logged actions identically across candidates (Sec. 2)—mean subtraction yields zero variance; if it relies on uncalibrated pre-action self-belief, it yields misaligned rankings. Formally, let candidate completions $\{a^{(i)}\}_{i=1}^{G}$ be sampled from $\pi_\theta(\cdot \mid c_t)$. For log-likelihood rewards $r_{i,t} = \log \pi_\theta(a_t^{\text{logged}} \mid c_t)$, the score evaluates the historical logged action rather than candidate $a^{(i)}$, yielding identical scalars $r_{i,t} = c$ for all $i \in \{1, \ldots, G\}$. Consequently, within-group sample variance degenerates to zero ($\sigma^2\big(\{r_{j,t}\}_{j=1}^{G}\big) = 0$), producing zero-norm policy gradients $\nabla_\theta \mathcal{L}_{\text{GRPO}}(\theta) \approx 0$. Observation grounding restores correctness-aligned discriminative variance: by anchoring E_t, V_t, O_t in objective environment semantics (action execution validity, valid state changes, and subgoal verifications), the operator provides a meaningful relative ranking across candidates, guaranteeing $\sigma^2(\{r_{j,t}\}) > 0$ and driving clean, informative policy updates.

5 Experiments

5.1 Setup

Environments. We evaluate on three environments, each scored on a held-out split: ALFWorld (Shridhar et al., 2021), free-text household tasks, on the 134-game `valid_unseen` out-of-distribution split (base 67.2); Deep Search (Xi et al., 2025b), multi-step search and information gathering in AgentGym-RL (base 18.75); and ScienceWorld (SciWorld) (Wang et al., 2022), interactive science, on 200 held-out items by success rate (near-floor base 4.0). Forgetting is evaluated on GSM8K (Cobbe et al., 2021) Please see Secs. A.1 and C.1 for more details.

Access protocol. We hold the method to logged-only credit (no resetting, re-execution, oracle, gold, or expert information), simulating the irreversible deployment condition even though the sandbox would permit re-execution (Sec. 1). The results table (reported below in Sec. 5.2, Tab. 1) is therefore read by access: matched-access arms share this budget, while privileged arms, trajectory-contrast (ETO) and post-action return grouping (GiGPO), are granted information unavailable at deployment and enter as ceilings, not matched comparisons.

Baseline treatment. The sort of these neighbors (Sec. 3) is an empirical protocol, not a positioning claim, and it fixes both what we run and how each number is read by a single two-part rule: an access-matched method we run as-published, head-to-head; a method whose per-step signal comes

Table 1: **Main results: ALFWorld iterative deployment compounding (Qwen3.5-9B).** Side-by-side comparison across deployment Round 1 (R1) and Round 2 (R2). *Methods: Base* = untrained; **Ours** = observation-grounded step credit (GRPO); *SFT floor* = self-training floor; *Outcome-GRPO* = terminal-credit ablation; ETO (Song et al., 2024) = trajectory-level DPO baseline; *GiGPO* (Feng et al., 2025) = off-setup cross-backbone ceiling. *Access:* ✓ access-legal / × off-setup. ALFWorld = the 134-game `valid_unseen` OOD split (base 67.2 %; per-cell decode band ±4 %). Retention (GSM8K) and efficiency (actions / tokens per episode) recorded at R2.

		Round 1 (R1)			Round 2 (R2)			Retention (R2)
Method	**Access**	SR%	Actions	Tokens	SR%	Actions	Tokens	GSM8K
Base	—	67.2	33.0	35,132	67.2	33.0	35,132	94.2
Ours	✓	76.1	27.7	32,300	**78.4**	**24.0**	29,476	93.6
SFT	✓	71.6	27.4	32,100	69.4	28.3	37,315	94.2
Outcome-GRPO	✓	67.5	31.7	36,400	74.6	29.0	34,410	94.8
ETO (Song et al., 2024)	×	68.7	30.7	40,263	73.9	29.4	35,427	93.7
GiGPO	×	77.6	20.5	21,050	77.6	19.4	19,337	n/c

Table 2: **Multi-environment evaluation: ScienceWorld and Deep Search (Qwen2.5-7B (Yang et al., 2024)).** Matched step-credit comparison across diverse environments. Access legend as in Tab. 1.

	ScienceWorld (200-item split)		Deep Search (AgentGym-RL)	
Method	**Access**	**Success rate (%)**	**Access**	**Success rate (%)**
Base	—	4.0	—	18.75
Terminal-credit ablation	✓	51.5	✓	—
Ours (step credit)	✓	**63.5**	✓	**44.75**
Published recipes & open-weight baselines				
AgentGym-RL (Xi et al., 2025b)	✓	50.5	✓	34.00
ScalingInter (Xi et al., 2025b)	✓	57.0	✓	38.25
Qwen3-235B-A22B	—	23.5	—	28.25

from a forbidden counterfactual we ablate down to the cell it leaves and report with that cell's number. The access-matched case is the one that could falsify the partition: Progress-Advantage (Oh et al., 2026), run as-published, has a likelihood-based read that does not separate from base (in band on BFCL; Sec. F.1). On the ablate-down side, GiGPO's post-action grouping ablates down to outcome-only GRPO (67.5 ≈ base), beneath its off-setup ceiling, and AgentEvol to the RFT floor (70.1). The complete disposition (every method with the cell it reduces to and its number) is in Sec. E.1, Tab. 5; we run those runnable on our stack as-published.

GRPO (Shao et al., 2024) serves as our primary optimizer, and DPO (Rafailov et al., 2023) provides a completely-offline generalization test to a second optimizer, representing the harder-access setting expected to be weaker (Tang et al., 2024; Xu et al., 2024; Song et al., 2024).

5.2 Results

We evaluate observation-grounded step credit against an untrained base, primarily under GRPO and, as a second-optimizer generalization test, as a completely-offline DPO signal. We organize evaluations into two categories based on environment access (Sec. 5.1): logged-only arms versus privileged ceilings.

Main environment comparisons. On ALFWorld, our operator leads in both rounds, reaching 78.4 % at Round 2 (11.2 % over base). This achieves the highest access-matched score, outperforming the base model, the terminal-credit ablation, and the self-training baseline beyond evaluation variance (Tab. 1). Although privileged methods consume strictly more information, they

perform on par with our approach. For instance, GiGPO's post-action return grouping reaches 77.6 on Qwen2.5-7B, establishing a ceiling that our operator matches. Furthermore, matched-backbone trajectory contrast (ETO) performs below our Round 2 result (Sec. D.2), demonstrating that counterfactual access provides no advantage over our logged-only method. On Deep Search (Xi et al., 2025b), observation-grounded step credit provides localized feedback on search query formulation and retrieval state transitions, preventing early trajectory collapse over multi-step reasoning horizons. The same performance gap between step and terminal credit holds on SciWorld. Its near-floor $4.0\,\%$ baseline represents a low-outcome-variance regime where candidates fail uniformly, rendering terminal outcomes uninformative even as step-level progress varies within the group. In this environment, our access-legal step-credit operator outperforms the terminal-credit ablation by $12.0\,\%$ ($63.5\,\%$ vs. $51.5\,\%$; Tab. 2) and surpasses published RL baselines on the same backbone (50.5% and 57.0%; Sec. D.4). The overall pattern remains consistent across all three environments: our observation-grounded operator outperforms the terminal-credit ablation and self-training, while matching privileged baselines that consume live environment feedback.

Ablation of step credit and compounding across rounds. Removing per-step credit (leaving outcome-only GRPO) causes performance to collapse back to the base model (Tab. 1), illustrating the core limitation of outcome-only rewards discussed in Sec. 2: without step-level feedback, 91.7% of Round 2 training steps yield zero gradient (Sec. D.2). Localizing the credit signal per step is what drives policy improvement. Furthermore, the operator continues to improve the policy across rounds: re-collecting from the improved policy and crediting again lifts ALFWorld base $67.2 \rightarrow$ R1 $76.1 \rightarrow$ R2 78.4 (the policy that once looped on invalid actions now completes the task), with no reward divergence and no forgetting. A third round reaches 81.0, with per-round increments of $2.3\,\%$ (R1$\rightarrow$R2) and $2.6\,\%$ (R2$\rightarrow$R3); the headline remains R2 78.4 (Sec. D.2).

Error reduction, efficiency, and retention. Invalid action attempts account for only 1.91% of environment steps for our method at Round 2, compared to 4.42% at Round 1 and higher rates across all baselines (Sec. D.1). Specifically, the base model repeatedly executes invalid action loops—such as opening unverified receptacles or attempting to clean objects before picking them up. Observation-grounded step credit assigns immediate penalty $E_t = 0$ to these failed transitions, suppressing invalid execution loops and reducing total episode length from 33.0 to 24.0 actions. The 78.4 policy uses 27% fewer environment actions (24.0 vs. 33.0) and 16% fewer tokens per episode than base ($29{,}476$ vs. $35{,}132$; Tab. 1), proving that step credit guides the agent along direct, error-free execution paths rather than padding trajectories with redundant lookups. Crucially, policy fine-tuning preserves general reasoning capabilities: round-matched at round 2, our operator retains GSM8K mathematical reasoning at $93.6\,\%$ (vs. base $94.2\,\%$), staying strictly within standard decoding noise ($\pm 0.8\,\%$) and confirming zero catastrophic forgetting across deployment rounds.

Offline DPO generalization and cross-family transfer. The same credit also acts as a completely offline DPO signal driving a modest first-round gain over base, but does not compound under offline iterations (Sec. D.2). To test whether observation-grounded credit relies on Qwen-specific family biases, we evaluate cross-family transfer on Mistral-7B-Instruct (Jiang et al., 2023). Applied to Mistral without hyperparameter tuning, the operator lifts round-1 success rate from $20.1\,\%$ to $33.6\,\%$ ($13.5\,\%$ gain; solving $45/134$ OOD games vs. $27/134$ for base). This confirms that the execution, subgoal, and compliance signal taxonomy (E, V, O) represents a backbone-agnostic credit operator that generalizes robustly across model families. Within a GRPO candidate group on Deep Search (Xi et al., 2025b), the observation-grounded verifier ranks genuinely goal-advancing search candidates above inert ones, supplying a correctness-aligned gradient that prevents trajectory collapse.

6 Conclusion

We studied credit assignment in deployed agents: recovering per-step credit from logged trajectories under the irreversible access constraint. By combining **OG-Gen** for pre-action generative observation synthesis with learning-free observation-grounded step credit, our framework bypasses the need for live environment re-execution when it is unavailable. It outperforms terminal-credit baselines across ALFWorld ($67.2 \rightarrow 78.4\%$), Deep Search (Xi et al., 2025b), and ScienceWorld (63.5%), matching privileged counterfactual ceilings at lower rollout cost and without losing general reasoning capabilities.

AI USE STATEMENT

In this work, generative AI tools were used solely for text editing, formatting, and copyediting assistance. All research concepts, mathematical formulations, experimental setups, code implementations, and scientific conclusions were developed independently by the authors.

ETHICS STATEMENT

This work studies credit assignment algorithms for text-action language-model agents operating in simulated environments (ALFWorld, ScienceWorld). All experiments utilize standard, publicly available benchmarks and logged trajectories without human subject data, PII, or privacy concerns.

REPRODUCIBILITY STATEMENT

To ensure reproducibility, full mathematical formulations of OG-Gen and observation-grounded step credit are provided in Sec. 4. Complete experimental hyperparameters, environment prompts, baseline configurations, and ablation protocols are documented in Sec. 5.1 and App. **??**. Source code and logged trajectories will be made publicly available.

REFERENCES

Renat Aksitov, Sobhan Miryoosefi, Zonglin Li, Daliang Li, Sheila Babayan, Kavya Kopparapu, Zachary Fisher, Ruiqi Guo, Sushant Prakash, Pranesh Srinivasan, et al. Rest meets react: Self-improvement for multi-step reasoning llm agent. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*, 2024.

Jose A Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. Rudder: Return decomposition for delayed rewards. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. arXiv:1806.07857.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. arXiv:2106.01345.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *Nature*, 645:633–638, 2025. arXiv:2501.12948.

Hao Dou. CIGPO: Contextual information-gain policy optimization for multi-turn evidence-reading LLM agents. *arXiv preprint arXiv:2607.16244*, 2026.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, et al. Reinforced self-training (rest) for language modeling. *arXiv preprint arXiv:2308.08998*, 2023.

Anna Harutyunyan, Will Dabney, Thomas Mesnard, Mohammad Gheshlaghi Azar, Bilal Piot, Nicolas Heess, Hado P van Hasselt, Gregory Wayne, Satinder Singh, Doina Precup, et al. Hindsight credit assignment. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. arXiv:1912.02503.

Senkang Hu, Yong Dai, Xudong Han, Zhengru Fang, Yuzhi Zhao, Sam Tak Wu Kwong, and Yuguang Fang. Self-induced outcome potential: Turn-level credit assignment for agents without verifiers. *arXiv preprint arXiv:2605.04984*, 2026.

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

Chia-Chun Hung, Timothy Lillicrap, Josh Abramson, Yan Wu, Mehdi Mirza, Federico Carnevale, Arun Ahuja, and Greg Wayne. Optimizing agent behavior over long time scales by transporting value. *Nature Communications*, 10(5223), 2019. arXiv:1810.06721.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2016. arXiv:1511.03722.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations (ICLR)*, 2022.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Sascha Lange, Thomas Gabel, and Martin Riedmiller. Batch reinforcement learning. In *Reinforcement Learning: State-of-the-Art*, pp. 45–73. Springer, 2012.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

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

Siyuan Lu, Zechuan Wang, Hongxuan Zhang, Qintong Wu, Leilei Gan, Chenyi Zhuang, Jinjie Gu, and Tao Lin. Don't just fine-tune the agent, tune the environment. In *International Conference on Learning Representations (ICLR)*, 2026.

Thomas Mesnard, Théophane Weber, Fabio Viola, Shantanu Thakoor, Alaa Saade, Anna Harutyunyan, Will Dabney, Tom Stepleton, Nicolas Heess, Arthur Guez, et al. Counterfactual credit assignment in model-free reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2021. arXiv:2011.09464.

Marvin Minsky. Steps toward artificial intelligence. *Proceedings of the IRE*, 49(1):8–30, 1961.

Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *International Conference on Machine Learning (ICML)*, 1999.

Changdae Oh, Wendi Li, Seongheon Park, Samuel Yeh, Tanwi Mallick, and Sharon Li. Neglected free lunch from post-training: Progress advantage for llm agents. *arXiv preprint arXiv:2606.26080*, 2026.

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E Gonzalez. The berkeley function calling leaderboard (bfcl): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning (ICML)*, 2025.

Doina Precup, Richard S Sutton, and Satinder Singh. Eligibility traces for off-policy policy evaluation. In *International Conference on Machine Learning (ICML)*, 2000.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Jiadai Sun, Xinyue Yang, Yu Yang, Shuntian Yao, Wei Xu, et al. Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2025.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea Finn. From r to $qˆ*$: Your language model is secretly a q-function. In *First Conference on Language Modeling*, 2024.

David Raposo, Sam Ritter, Adam Santoro, Greg Wayne, Theophane Weber, Matt Botvinick, Hado van Hasselt, and Francis Song. Synthetic returns for long-term credit assignment. *arXiv preprint arXiv:2102.12425*, 2021.

Zhizhou Ren, Ruihan Guo, Yuan Zhou, and Jian Peng. Learning long-term reward redistribution via randomized return decomposition. In *International Conference on Learning Representations (ICLR)*, 2022. arXiv:2111.13485.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations (ICLR)*, 2016. arXiv:1506.02438.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Avi Singh, John D Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J Liu, James Harrison, Jaehoon Lee, Kelvin Xu, et al. Beyond human data: Scaling self-training for problem-solving with language models. *arXiv preprint arXiv:2312.06585*, 2023.

Yifan Song, Da Yin, Xiang Yue, Jie Huang, Sujian Li, and Bill Yuchen Lin. Trial and error: Exploration-based trajectory optimization of llm agents. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. On the self-verification limitations of large language models on reasoning and planning tasks. In *International Conference on Learning Representations (ICLR)*, 2025.

Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1):9–44, 1988.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

Adith Swaminathan and Thorsten Joachims. Counterfactual risk minimization: Learning from logged bandit feedback. In *International Conference on Machine Learning (ICML)*, 2015. arXiv:1502.02362.

Hui-Ze Tan, Xiao-Wen Yang, Hao Chen, Jie-Jing Shao, Yi Wen, Yuteng Shen, Weihong Luo, Xiku Du, Lan-Zhe Guo, and Yu-Feng Li. Hindsight credit assignment for long-horizon LLM agents. *arXiv preprint arXiv:2603.08754*, 2026.

Yunhao Tang, Zhaohan Daniel Guo, Zeyu Zheng, Daniele Calandriello, R'emi Munos, Mark Rowland, Pierre Harvey Richemond, Michal Valko, Bernardo 'Avila Pires, and Bilal Piot. Understanding the performance gap between online and offline alignment algorithms. In *International Conference on Machine Learning (ICML)*, 2024.

Gladys Tyen, Hassan Mansoor, Victor Cărbune, Yuanzhu Peter Chen, and Tony Mak. Llms cannot find reasoning errors, but can correct them given the error location. In *Findings of the Association for Computational Linguistics: ACL 2024*, 2024.

Daoyu Wang, Qingchuan Li, Mingyue Cheng, Jie Ouyang, Shuo Yu, Qi Liu, and Enhong Chen. StepPO: Step-aligned policy optimization for agentic reinforcement learning. *arXiv preprint arXiv:2604.18401*, 2026a.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, et al. Information gain-based policy optimization: A simple and effective approach for multi-turn search agents. In *International Conference on Learning Representations (ICLR)*, 2026b.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. Scienceworld: Is your agent smarter than a 5th grader? In *Empirical Methods in Natural Language Processing (EMNLP)*, 2022.

Yu Wang. The dark room in the reward channel: Dense prediction rewards collapse grpo-trained llm agents–and what actually works. *arXiv preprint arXiv:2607.21273*, 2026.

Zhiheng Xi, Yiwen Ding, Wenxiang Chen, Boyang Hong, Honglin Guo, Junzhe Wang, Xin Guo, Dingwen Yang, Chenyang Liao, Wei He, et al. Agentgym: Evaluating and training large language model-based agents across diverse environments. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025a.

Zhiheng Xi, Jixuan Huang, Chenyang Liao, Baodai Huang, Honglin Guo, Jiaqi Liu, Rui Zheng, Junjie Ye, Jiazheng Zhang, Wenxiang Chen, et al. Agentgym-rl: Training llm agents for long-horizon decision making through multi-turn reinforcement learning. *arXiv preprint arXiv:2509.08755*, 2025b.

Han Xiao, Guozhi Wang, Yuxiang Chai, Zimu Lu, Weifeng Lin, Hao He, Lue Fan, Liuyang Bian, Rui Hu, Liang Liu, et al. Ui-genie: A self-improving approach for iteratively boosting mllm-based mobile gui agents. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Weimin Xiong, Yifan Song, Xiutian Zhao, Wenhao Wu, Xun Wang, Ke Wang, Cheng Li, Wei Peng, and Sujian Li. Watch every step! llm agent learning via iterative step-level process refinement. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

Peiran Xu, Zhuohao Li, Xiaoying Xing, Guannan Zhang, Debiao Li, and Kunyu Shi. Hybrid reward normalization for process-supervised non-verifiable agentic tasks. *arXiv preprint arXiv:2509.25598*, 2025.

Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weilin Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. Is DPO superior to PPO for LLM alignment? a comprehensive study. In *International Conference on Machine Learning (ICML)*, 2024.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. Free process rewards without process labels. In *International Conference on Learning Representations (ICLR)*, 2025.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*, 2023.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Zhen Zhang, Kaiqiang Song, Xun Wang, Yebowen Hu, Weixiang Yan, Chenyang Zhao, Henry Peng Zou, Haoyun Deng, Sathish Reddy Indurthi, Shujian Liu, et al. Cm2: Reinforcement learning with checklist rewards for multi-turn and multi-step agentic tool use. *arXiv preprint arXiv:2602.12268*, 2026a.

Zhiwei Zhang, Fei Zhao, Rui Wang, Zezhong Wang, Bin Liang, Jiakang Wang, Yao Hu, Shaosheng Cao, and Kam-Fai Wong. Robust tool use via fission-grpo: Learning to recover from execution errors. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2026b.

A Experimental Configuration

A.1 Experimental configuration

The following configuration details are shared across every environment we train in (Sec. 5.1).

Item	Value
Base models	Qwen family (Yang et al., 2025) (per-environment below)
Training model size / variant	Per-environment within the Qwen3 family: ALFWorld = Qwen3.5-9B; BFCL = Qwen3-8B. (SciWorld uses a text-only Qwen2.5-7B; Sec. 5.1.) All arms fine-tuned with LoRA (r=8, α=16, dropout 0.05; attention + MLP projections).
GRPO hyperparameters (lr, batch size, steps/epochs)	Common: lr 1×10^{-5}, 8 candidates/prompt, KL β=0.05, cosine schedule, bf16, gradient checkpointing. ALFWorld: effective batch 8 (2/device $\times$ grad-accum 4), 1,500 steps, max completion 256 tokens, sampling temperature 0.7. BFCL: batch 1, 200 steps, max completion 16,384 tokens (thinking-on).
DPO hyperparameters (lr, batch size, steps/epochs)	Common: 3 epochs, KL β=0.1, cosine schedule, bf16, gradient checkpointing. ALFWorld: lr 5×10^{-6}, effective batch 8 (1/device $\times$ grad-accum 8), max length 4,096. BFCL: lr 5×10^{-5}, effective batch 4 (1/device $\times$ grad-accum 4), max length 6,144.
DPO initialization	SFT checkpoint (both arms)
BFCL within-group-variance fix	gold-free read-redesign (score the observation's schema/precondition read); an earlier reference-injection version was excluded as off-rule (Sec. F.1)
Thinking-mode control	train-mode = eval-mode (mode stamp at eval)

B The Belief–Observation Gap and the Elimination Argument

B.1 Belief–observation gap: the widened campaign, elicitation protocol, and the V-verifier companion

This appendix expands the diagnostic of Sec. 2 (Fig. 2 and its $N = 566$ Wilson intervals are in the body) and reports the widened belief-gap campaign in full: what we ran, the numbers, and the conclusion the three axes jointly support.

Sample size and scale coverage. The false-confidence rate is estimated at $N = 566$ decision points per rung, sized so that a per-rung Wilson 95% interval ($\approx\pm5\,\%$) is narrow enough to resolve the flat band from a monotone trend (a smaller $N = 48$–180 leaves intervals up to $\pm18\,\%$, within which no trend is identifiable). The four rungs also include a genuinely *dense* 72B model (nine times the active parameters of the 235B mixture-of-experts) so that the flat band cannot be attributed to MoE sparsity: a dense >32B model lands mid-band.

Axis E: the primary false-confidence probe. On logged tool-use rollouts we take decision points and, at each, elicit the model's pre-action belief about whether its own call will succeed (thinking-on), then compare it to the logged ground-truth observation. The *false-confidence rate* is, among steps that actually failed, the fraction on which the model believed it had succeeded. We run same-family Qwen3 models at 8B, 32B, and the 235B mixture-of-experts (Yang et al., 2025), plus the dense 72B rung. On the inferential status of the resulting flat band: at $N = 566$ per rung every pairwise difference between rungs is non-significant and a Cochran–Armitage monotone trend test across the scale axis (total or active parameters) is null ($z = -1.089$, $p = 0.276$), so the 75–81% band cannot be read as improving with scale. Self-judgment of one's own step outcomes does not get better with model size.

Elicitation protocol and the verbatim prompt. At each decision point we serve the *same* model whose rollout we are scoring and ask it a single *binary* question about the call it just chose (will it execute successfully) using only pre-action context: the task goal, the available tool names, the last five logged tool responses (`CONTEXT_N` $= 5$, oldest first), and the proposed call. The model never sees the post-action observation. Decoding is temperature 0, thinking-on (native `enable_thinking` on the Qwen3 rungs; inline chain-of-thought on the dense-72B Qwen2.5 rung, see caveats), `max_tokens` $= 1024$; we parse only the text after the closing `</think>` tag and read the final `BELIEF:` line. The verbatim prompt (placeholders in braces filled per decision point) is:

```
A tool-use agent working toward a goal just chose its NEXT
tool call.  The call has NOT been executed yet.  Using
ONLY your own judgment of the tools and the state so far,
predict:  will this call EXECUTE SUCCESSFULLY, without
raising an error?
TASK GOAL: {goal}
AVAILABLE TOOLS: {tools}
STATE SO FAR (recent tool responses / observations, oldest
first, before this call):
{context}
PROPOSED NEXT TOOL CALL (not yet executed):  {call}
Consider:  does the function exist?  are the arguments
well-formed and of the right types?  are the argument
values consistent with the state established by prior
tool responses (a file / id / path that actually exists,
a precondition that has been met)?
Reason briefly step by step, then End with exactly one line:
BELIEF: SUCCESS   (you believe the call will run cleanly)
BELIEF: FAILURE   (you believe the call will raise / be
rejected)
```

Because the elicited belief is *binary* (a single `SUCCESS`/`FAILURE` verdict, not a probability), we cannot build a calibration or reliability curve from it; we therefore report a 2×2 confusion matrix per rung instead, which the binary verdict does determine.

Per-rung confusion matrices. Cross-tabulating the logged ground truth (call *failed* vs. *succeeded*) against the elicited belief gives the four cells below; all counts are read from the $N = 566$ per-point artifacts, and the derived rates (false-confidence, false-alarm, divergence) are algebraically determined by them.

	Failed→		**Succeeded→**					
Rung	bel. SUCC	bel. FAIL	bel. SUCC	bel. FAIL	FC%	FA%	Div.%	parsed
8B (dense)	218	52	263	17	80.7	6.1	42.7	550
32B (dense)	209	68	260	20	75.5	7.1	41.1	557
72B (dense)	222	61	264	19	78.4	6.7	42.6	566
235B (MoE)	188	58	255	14	76.4	5.2	39.2	515

The false-confidence cell (Failed→believed SUCCESS) is the body's main result; the divergence rate is the sum of the two off-diagonal cells over the parsed count (e.g. 8B: $(218 + 17)/550 = 42.7\%$); the false-alarm rate stays low (5–7%) at every rung, i.e. the miscalibration is one-sided (the model over-trusts far more than it over-doubts). The clean-arm belief-success rate (the complement of false-alarm) is correspondingly $93.9/92.9/93.3/94.8\%$ for 8B/32B/72B/235B, the quantity the conservative-lower-bound argument below turns on.

Full test battery. Per-rung Wilson 95% intervals on the false-confidence rate are $[75.6, 85.0]$ (8B), $[70.1, 80.1]$ (32B), $[73.3, 82.8]$ (72B), $[70.7, 81.3]$ (235B): all four mutually overlapping and all four lower bounds far above 50%. The six pairwise unpaired two-proportion tests on the fail arm are all non-significant:

Pair	Δ (%)	z	p	Pair	Δ (%)	z	p
8B−32B	+5.3	+1.49	0.135	32B−72B	−3.0	−0.841	0.400
8B−72B	+2.3	+0.669	0.503	32B−235B	−1.0	−0.26	0.795
8B−235B	+4.3	+1.20	0.232	72B−235B	+2.0	+0.556	0.578

The paired McNemar tests, on the within-item fail decision points shared by each pair (exact two-sided), tell the same story:

Pair	n	disc. (row-A only)	disc. (row-B only)	p
8B vs 32B	266	25	13	0.073
8B vs 72B	270	19	13	0.377
8B vs 235B	237	26	11	**0.020**
32B vs 72B	277	18	28	0.184
32B vs 235B	244	16	13	0.711
72B vs 235B	246	27	17	0.174

We disclose the one nominal star plainly: the paired 8B-vs-235B contrast ($p = 0.020$) is the only test of six that crosses 0.05, it does *not* survive Bonferroni correction for six comparisons (threshold 0.0083), the effect it marks is only $4.3\,\%$, and it points *downward*: 235B slightly better-calibrated than 8B, the direction opposite to a "scale fixes it" story and at a fraction of the magnitude that story needs. No test, corrected or not, supports self-belief improving with scale.

Worst-case imputation (Manski bounds). Parsing drops are small and rung-dependent ($16/9/0/51$ points at 8B/32B/72B/235B; the 235B drops are thinking traces that hit the token cap mid-reasoning and never emitted a `BELIEF:` line). Imputing every dropped fail-arm point both ways gives worst-case (Manski) bounds on false-confidence of $[77.0, 81.6]$ (8B), $[73.9, 76.0]$ (32B), $\{78.4\}$ (72B, zero drops: point-identified), and $[66.4, 79.5]$ (235B). Even the single worst-case lower bound, 66.4% at 235B, is far above 50%, so the flat-band conclusion does not depend on how the dropped points would have parsed.

Label semantics and the conservative lower bound. The ground-truth "actually failed" label is an *execution error*: an `{"error":...}` string in the logged tool response (verified from the scoring code: `gt_success = not any(is_error(c) for c in resps)`). Execution error is a *sufficient* condition for goal-failure (a call that raises accomplishes nothing toward the goal) but not a necessary one: a call can execute cleanly yet be goal-wrong. So the fail arm (the 283 execution-errored decision points from which false-confidence is computed) is a strict *subset* of the true goal-failure set, and relabeling to goal-correctness can only *add* the clean-but-goal-wrong calls to the fail arm, never remove any. Those added calls come from the current clean arm, where the model believes success 93–95% of the time ($93.9/92.9/94.8\%$ at 8B/32B/235B); a call that ran cleanly is if anything one the model is *more* confident about, so its false-confidence contribution is $\approx 94\%$, at or above the 75–81% main result. The relabeled rate is therefore a weighted average of the current 75–81% and $\approx 94\%$: strictly higher. The reported 75–81% is thus a *conservative lower bound* on the goal-level false-confidence, and the direction is provable. We also compute the relabeled rate directly (below and Sec. 2): under a goal-correctness label it rises to $\approx$85–91% across the BFCL rungs and to 89.4% (within-sample) / 95.9% (population-reweighted) on ALFWorld, confirming the bound is not just directional but sizeable. The ALFWorld goal label is the deployed subgoal judge (a frozen model score, not a human oracle), a caveat we carry with that number; BFCL per-call gold is scored at turn/end-state granularity, so its relabel is the conservative band rather than a single figure. (This same execution-cleanliness label was fatal as a training *reward*, where mislabeling clean-but-wrong as correct rewards bad behavior; here it is benign, because the metric conditions on an explicit, unambiguous failure signal and the confound only *under*-counts the fail arm.)

Readable vs. hidden failures: the gap is not an information deficit. Splitting the fail arm by whether the failure's cause was present in the pre-action context, $124/283$ (43.8%) are *readable* (a precondition or state fact derivable from the goal and last-five observations, e.g. brake-not-pressed, travel-class, tank-capacity), $152/283$ (53.7%) are *hidden* (backend state absent from the pre-action window: a missing file/id/booking, an auth failure, an already-exists collision), and 7 unclassified.

False-confidence is *highest on the readable failures* ($98.3/93.3/91.6\%$ at 8B/32B/235B) and substantially lower on the hidden ones ($66.7/62.0/64.7\%$). This is the diagnostic signature that matters mechanistically: if the gap were driven by missing information it would be *worst on hidden failures*, but it is worst precisely where the failure was derivable from what the model already had. On the cases where it possessed the signal the model was still confidently wrong $\sim$91–98% of the time, which points to a calibration failure (a success-priored bias: plausibly, though we do not measure this, a trained-in optimism from success-weighted post-training) rather than an information limitation. One boundary condition follows honestly and is worth stating: the belief-gap's *remedy* is the post-action observation, which carries an explicit error string for 100% of the fail arm (readable and hidden alike), so observation-grounding is informative on every failure; but a purely *pre-action* self-belief replacement cannot recover the $\sim 54\%$ hidden-state failures, which remain a floor for any pre-action method (not a flaw in this diagnostic, which reads the post-action log).

Pre-action prediction vs. post-action read. To separate the pre-action *prediction* the main result measures from a post-action *read*, we re-run the identical probe on the same 566 decision points but show the model the *logged post-action observation* and ask it to judge the step just taken; the elicitation, points, and execution-error labels are otherwise unchanged, so the two are index-aligned and directly comparable (the 566-point join is reconstructed by re-running the deterministic extractor and identity-checking every row against the canonical points; $566/566$ matched). False confidence collapses from the pre-action band to $\approx$3–5%: Qwen3-8B $80.7 \to 5.0\%$, Qwen3-32B $75.5 \to 3.6\%$, Qwen3-235B $76.4 \to 3.2\%$. We are explicit that this is *not* evidence the model self-verifies well: on execution-error steps the failure is literally present as an error string in the post-action observation, so reading it off is near-trivial, which is exactly the intended contrast. The measurement isolates *how much of the failure signal lives in the observation rather than in the model's prior belief*: essentially all of it. This also reconciles the readable-vs-hidden split above: the readable failures are precisely the ones a model must *infer* pre-action and reliably fails to (91–98% false confidence), yet post-action the same failure is *stated* in the observation and read correctly: the observation converts the inference the model gets wrong into a read it gets right. This is why the operator's DPO realization, which reads the logged action's post-action observation, inherits the easy regime on that side of each pair, whereas its GRPO realization must predict the consequence of a *generated*, not-yet-logged action, landing in the hard pre-action regime the 75–81% band quantifies (Sec. 4, Sec. F.1).

Axis V: the subgoal-verifier companion. A companion probe replaces the outcome question with a subgoal-progress question: at each step we ask a fixed (untrained) LLM verifier V whether the step advanced a subgoal, and compare its claim to a ground-truth subgoal check. On 179 paired decision points over 37 trajectories (three rungs, 8B/32B/235B; no dense-72B rung here), the rate at which V claims progress on a step that ground truth says did *not* advance is $83.3/57.9/76.0\%$: *non-monotone, with 32B the best-calibrated rung, not the worst*. No pairwise contrast survives multiple-comparison correction, and the step-level V-vs-ground-truth Pearson correlations ($0.26/0.49/0.30$) have 95% intervals that all overlap. Crucially, this rate conditions on V's *own* progress-claims (it is a false-discovery rate over claimed-progress steps), not on ground truth, so it is a companion diagnostic to the primary axis rather than a second measurement of it; but its message is the same: a bigger or different verifier is not miscalibration-free, so "use a bigger verifier" is not a way out. To anchor this leg without any LLM in the ground-truth path, we add a *deterministic* check: on actions the environment logs as provably inert (a matched error string means the action changed nothing, so it *cannot* have newly satisfied a subgoal), V nonetheless claims new subgoal satisfaction on $8.9/4.4/13.3\%$ of such actions at 8B/32B/235B: the only comparison here with an LLM-free ground truth, and again non-monotone with the 235B rung the worst rather than the best (the 32B-vs-235B contrast $z = -2.095$, $p = 0.036$, nominal only and not surviving the nine-test correction). Two caveats bound the V-axis specifically: its primary ground truth is itself *model-scored* and scored by each rung's own model, so the label target shifts slightly across rungs (this is why the verdict leans on the deterministic anchor, whose ground truth is fixed); and the 179 pairs nest in only 37 trajectories, so the two-proportion tests are anti-conservative.

The three axes as one story. The campaign closes the self-judgment escape routes in turn. (E) The model's belief about its own step outcomes is confidently wrong on a flat 75–81% of failed steps and does not improve with scale: self-judgment does not get better. (V) A subgoal verifier reading

the same trajectories is likewise miscalibrated (non-monotone, no scale trend): a bigger or different verifier is not the fix. (Redistributor) A learned LM redistributor handed the full trajectory and its terminal return emits per-step credit at Pearson ≈ 0 against step-correctness (-0.09 to $+0.06$; the demonstration is Sec. B.2): training on returns does not manufacture the per-step signal either. All three legs converge on the same conclusion: per-step credit must be read off the realized observation, not self-judged and not learned-redistributed. This is the empirical spine of the elimination argument (legs (i) and (ii) of Sec. 3).

The deployed operator, at the term level. Reading an observation is necessary but not sufficient to escape the gap; what matters is whether the observation is turned into credit by a *deterministic parse* or by a *judgment*. The terms that clear the gap do so by construction: the execution-error read is a fixed test on the logged observation (the deterministic anchor above), and the format term is a fixed string check. The frozen verifiers, by contrast (though conditioned on the logged observation rather than on the policy's introspection), remain *judgments*, and remain miscalibrated: the subgoal verifier claims progress even on provably inert actions ($8.9/4.4/13.3\%$ above), so observation-*grounding* of the input is not by itself an escape (the DPO-vs-GRPO pre/post split of Sec. 4 is this same asymmetry, quantified above). The gap is escaped by the *source* of the signal, a deterministic read of the realized observation, not by a better-calibrated judge.

Caveats. Three limits are attached to these numbers. On axis E, the dense-72B rung is a Qwen2.5 model against Qwen3 elsewhere, so a cross-generation confound rides that single point; the qualitative flat-band conclusion (false confidence is substantial and non-vanishing with scale) holds across all four rungs regardless. The 75–81% band is also a *conservative* floor with respect to the failure label: it counts only steps whose action produced an execution error, and under a stricter goal-correctness label (counting clean-but-goal-wrong steps as failures too) the band only widens (to $\approx$ 85–91% across rungs), because the model is if anything *more* falsely confident on steps that execute cleanly but do not advance the goal (this relabel uses the deployed subgoal judge as its goal-correctness signal, an approximation rather than a human oracle). On axis V, we have only three rungs (no dense-72B), N is smaller (179 pairs), and the rate conditions on V's own claims rather than on ground truth. On the redistributor, "≈ 0" means no detectable per-step calibration *at these sample sizes*, not exactly zero. None of these caveats moves the joint conclusion.

B.2 Separations from the four neighboring settings, in full

This appendix expands the compact separation of Sec. 3 (summarized there and in the neighboring-settings statement of Sec. 2). The access rule *removes an input* that each of the four nearest neighbors relies on. For each we state the input it assumes, what Eq. (1) makes impossible, and what our operator supplies instead. In each case the neighbor either requires an input we forbid or, handed only the logged trajectory, degenerates to the zero-localization Monte-Carlo return we show is gradient-dead (outcome-only GRPO 67.5 $\approx$ base 67.2, Sec. 5.2). Two scope notes carry over from Sec. 2. The access test applies even in a deterministic environment: re-executing a *generated* call fails it, since it is still a fresh environment query rather than a read of the log. And unlike offline RL we assume no reward function and no value model, and settings with a cheap verifiable per-step signal (DeepSeek-AI, 2025) are out of scope.

(1) Classical offline RL (batch RL; CQL, IQL) and Classical State Representations *consumes* a per-transition scalar reward r (it optimizes a given reward, it does not localize a terminal one) via a Bellman backup with conservatism (Kumar et al., 2020; Kostrikov et al., 2022) to distrust out-of-distribution (counterfactual) action-values, so it is *downstream* of our problem, not a competitor. (Scoped to reward-*consuming* methods; the return-decomposition family in (4) instead *produces* a per-step reward and is separated on different grounds.) Handed only the terminal label y and a single logged action per context, its state value degenerates to the return-to-go with *zero* within-step localization; any localization beyond the return comes from a value network's generalization to counterfactual actions: the self-estimated counterfactual our belief–observation gap shows is wrong on 75–81% of failed steps at every scale (Sec. 2), and the estimate CQL/IQL conservatism is built to distrust. *Classical State Representation Paradigms (Tabular, Tile Coding, RBFs):* Foundational classical RL methods operating without engineered feature vectors rely on tabular lookup tables (in gridworlds) or non-parametric spatial discretization algorithms (tile coding, Radial Basis Functions (Sutton & Barto, 1998)). These classical approaches formulate credit over small discrete state

spaces or low-dimensional Euclidean spaces $\mathbb{R}^d$ with recurring state matches. In language agents, state histories $\mathcal{H} \subset \Sigma^*$ consist of open-ended, non-repeating, unstructured natural language text logs, extending beyond the scope of spatial discretization and tabular indexing without semantic language understanding. **(2) Off-policy evaluation (OPE; IS, doubly-robust)** outputs a *scalar policy value* $\hat{V}(\pi)$ (Precup et al., 2000), not a per-step credit over realized steps; the per-decision importance-sampling estimator $\rho_{0:T}\,\gamma^T y$ reweights *which trajectory* counts, not *which step* earned the outcome (the same zero-localization degeneracy, plus ρ-variance blow-up over text-space actions), and its doubly-robust/model-based variants recover localization only by adding the *learned scalar reward model* we assume away or a reconstructed simulator whose fidelity we find bounded (error-grounded reconstruction is information-theoretically closed under our constraint; Sec. G). **(3) Process supervision, classical reward shaping, and trace-based credit (TD(λ), Monte Carlo)** represent standard classical credit paradigms defined by a per-step *label source* or trace memory, each mapping cleanly into our formulation taxonomy under Eq. (1): human step annotation is an oracle label (clause b); Monte-Carlo / tree-search completion labeling (Wang et al., 2024) is a fresh environment query of generated continuations (clause a); teacher/expert-step distillation is an expert step to imitate (clause c); human-annotated process labels (Lightman et al., 2024) are oracle labels; Monte-Carlo returns wait for episode completion and assign outcome y uniformly across steps, which collapses to our zero-localization outcome-only GRPO baseline ($67.5 \approx 67.2$ base); TD(λ) eligibility traces pass terminal returns backward across short-term memory traces, but rely on parametric state-value estimates $\hat{V}_\psi(s)$ which suffer directly from the 75–81% pre-action belief–observation gap; and potential-based shaping (Ng et al., 1999) ($r' = r + \gamma\Phi(s') - \Phi(s)$) is permitted only if the potential $\Phi(s)$ is itself computable from the log without a value or reward model, which returns us to the question our operator answers. *What survives the removal of the counterfactual label source is exactly a credit read off the realized observation, our operator; in this sense our problem is what a PRM becomes once its counterfactual labeler is subtracted.* A PRM trained on labels derived *only* from single-trajectory logged signals (no completion statistics) is a difference of degree, not kind: such a method is an *instance* of the problem we define, not a threat to it (see the caveat in Sec. G).

(4) Return decomposition / learned long-horizon credit is the closest classical neighbor and, unlike (1)–(3), it targets exactly our problem: localizing a delayed terminal reward across a long horizon from collected trajectories, with no re-execution, no oracle, and no expert step. RUDDER (Arjona-Medina et al., 2019) redistributes a delayed episodic return across steps in a return-equivalent way; Randomized Return Decomposition (Ren et al., 2022) densifies an episodic terminal return into per-step rewards; Hindsight Credit Assignment (Harutyunyan et al., 2019) and Counterfactual Credit Assignment (Mesnard et al., 2021) assign credit from observed outcomes over collected trajectories in a model-free way; and Temporal Value Transport (Hung et al., 2019) and Synthetic Returns (Raposo et al., 2021) carry credit across long delays via learned associative models. This family shares our constraint (it does not query a counterfactual) so we do *not* separate it on access. The distinction is instead in the *solution class*: every one of these *learns a redistribution or value model* (an LSTM return-decomposer, a randomized-return regressor, a hindsight distribution, a value-transport network) whose per-step output is a *prediction* trained against trajectory aggregates. The move to text-action LM agents makes this learned mechanism, rather than the problem, the point of failure, and we show this directly (leg (i) of the elimination). *The redistributor demonstration.* We instantiate the family in its natural LM form: prompt the model with the *full* logged trajectory and its *terminal return* and ask it to emit a per-step credit that redistributes that return (a RUDDER/RRD-style redistribution, with the return-equivalence constraint enforced: the per-step credits sum to the return), and measure the Pearson correlation of the emitted per-step credit against ground-truth step-correctness. Across a full 2-environment $\times$ 3-scale ladder (ALFWorld and BFCL at 8B/32B/235B) the redistributor is *confidently uninformative*: per-step Pearson is ≈ 0 at every rung (-0.09 to $+0.06$; every bootstrap 95% CI contains zero), and scale does not rescue it. This is not a dead pipeline: the emitted credit correlates positively with the terminal return it was handed ($+0.15$ to $+0.32$) and honors the sum-to-return constraint exactly, so the redistributor is doing its task yet carries no per-step calibration. The reason is leg (ii): in the LM instantiation the redistributor's per-step estimate *is* the self-assessment our belief–observation gap measures, wrong on 75–81% of failed steps and non-improving with scale (Sec. 2), so training against a return aggregate does not manufacture the per-step signal the redistributor never had. This is a direct measurement of the learned-redistribution solution class, not an argument by mechanism. A small non-LM return regressor is a different object but faces a complementary obstacle: supervised only by a sparse terminal return it has no per-step

label to anchor it, and fitting one over the unbounded space of text trajectories from expensive, heterogeneous agent rollouts, rather than the low-dimensional state these methods were designed for, is not obviously trainable. Our operator instead reads credit *directly off the realized observation*: a learning-free function with no learned reward or value model to miscalibrate. This method-class difference also blunts the rejoinder that RUDDER's redistribution is *return-equivalent* and hence robust to a noisy approximation: group-relative on-policy training consumes the *learned* per-step signal through within-group discrimination, where a miscalibrated-but-varying reward does not add noise but actively reward-hacks, a failure mode we observe directly (Sec. F.1) and that concurrent work reports for learned dense per-step rewards (Wang, 2026). So the return-decomposition family is the right classical lineage for our problem; the differentiation is method-class (learning-free observation read vs. learned redistribution), not problem-novelty. Where no verifiable per-step check is available and per-step credit must be *learned*, that learned credit inherits the LM miscalibration the belief–observation gap measures (Yuan et al., 2025; Xu et al., 2025).

B.3 The hardest neighbor: a rollout-free, logged-outcome-distilled PRM

The separation of Sec. 3 is cleanest against PRMs whose labels come from rollouts, oracles, or teachers. It is *weakest* against a PRM trained on labels derived from logged data alone, e.g. regressing a step score against "did the trajectory through this step later succeed." Such a method can be built without any live environment query, and a logged-only step reward is, structurally, close to a PRM trained without counterfactual labels (Rafailov et al., 2024); we do not claim otherwise. The distinction rests on two points and one concession. First, the label source: the standard "logged-only" PRM labels a step by an *empirical success rate over many continuations sampled from the prefix*: a Monte-Carlo counterfactual amortized across the dataset rather than queried live, and thus re-entering the forbidden class the moment it uses completion statistics. A genuinely access-matched target must be derived from the *single realized observation*, which is our $E/V/O$ read, not a regressor over rollout aggregates. Second, we do not train a separate reward model: our credit is a *learning-free, observation-anchored read*. For the deterministic terms E and O, read directly from the logged observation (an error string, a well-formed envelope, the terminal outcome), correctness on the logged step is 100% by construction. The subgoal term V is the one exception and we state it plainly: V is a *fixed-but-untrained* LLM verifier read of the logged observation (judge/ground-truth Pearson ≈ 0.33, not 100%), so it is not a deterministic read, but it is still not a *learned* reward model (we train nothing) and it is still observation-grounded rather than a free-running self-belief. A distilled PRM's step score, by contrast, is a *prediction* that inherits the model's mis-calibration: the very self-estimate the belief–observation gap shows is unreliable. The concession, stated plainly: a PRM whose targets are derived *only* from single-trajectory logged signals, with no completion statistics, is a difference of degree, not kind, and rather than scooping our contribution, such a method would be *an instance of the problem we define*. Deployed-agent credit assignment is a problem statement, not a single algorithm; a method that honestly honors Eq. (1) is one of its methods.

C The E/V/O Operator, Instantiated Per Environment

C.1 E/V/O instantiations, per environment

This appendix gives the per-environment instantiations of the operator of Sec. 4: which concrete observable each axis reads in each environment, with what weight, and by what mechanism. The structure (E/V/O, Eq. (4)), the fixed-read commitment, and the causal-prefix property are identical across cells; what differs is the read, because which observable carries execution and goal semantics is a property of the environment. Table 3 summarizes; the paragraphs below expand each row.

The weights are per-environment and hand-set, by design. The three axes are combined by Eq. (4) with weights *set per environment and by hand* (Tab. 4); they were not tuned against any evaluation score, because tuning a reward against outcomes would leak the outcome information the operator is built to exclude. The per-environment weighting follows directly from the per-environment read: since each axis reads a different environment-specific observable (Tab. 3), the combination that keeps the reward correctness-aligned is itself environment-specific. Three properties the table makes explicit govern how the results should be read, and we state them affirmatively. First, E is one question (did the action take effect) answered by *four* distinct mechanisms (an LLM step-judge

Table 3: One reward structure, instantiated per environment. $^{\mathrm{LM}}$ marks a term computed by a *frozen* verifier language model (fixed prompt, temperature 0, never trained); every unmarked term is a deterministic function of logged strings. *Carrier* = the term that varies across a group and so supplies the GRPO gradient. Weights, all **hand-set** and never ablated: ALFWorld $0.5E{+}0.5V{+}O_{\mathrm{fmt}}$; DeepSearch / SciWorld $0.5E{+}0.5V{+}O_{mathrmfmt}{+}O_{\mathrm{term}}$; BFCL $E{+}O$ (unit weights, two terms).

Env.	E **(execution)**	V **(content / goal)**	O **(compliance, resolution)**	**Carrier**
ALFWorld	step verifier$^{\mathrm{LM}}$, $\{+1, 0, -1\}$	subgoal *level*$^{\mathrm{LM}}$ in the pre-action observation	action-verb format; no terminal term logged	E, format; V constant
BFCL	schema + representation-precondition analysis of the call	*absorbed into* O (not instantiated separately)	goal-completion read (replaces the format constant)	E, O
DeepSearch	native search-query / tool action set	retrieval-content verifier$^{\mathrm{LM}}$	format + logged terminal task score	V (E saturates; O constant)
ScienceWorld	native valid-action set + pre-action step-judge	subgoal *level*$^{\mathrm{LM}}$, as ALFWorld	format + logged terminal task score	pre-action step-judge step-credit

Table 4: Per-environment E/V/O *weights* for Eq. (4), all **hand-set and never ablated**. Weights are chosen per environment as a consequence of the per-environment read (Tab. 3) and were not tuned against any evaluation score. The *gradient carrier* is the term that varies within a GRPO candidate group; a within-group-constant term is annihilated by mean-subtraction, however accurate. O_{fmt} is a format read; O_{term} a logged per-episode scalar (group-constant, so it supplies no within-group gradient). The table states three facts affirmatively: E is one question answered by four mechanisms (column 2); the ALFWorld 78.4 cell has *no* terminal-outcome term (its O is format alone); and BFCL uses two unit-weighted terms with no separate V, with V nowhere the independent carrier. **—** = term not instantiated.

Env.	E **mechanism**	**Combination (hand-set)**	**Gradient carrier**	**Terminal-outcome term?**
ALFWorld	LLM step-judge	$0.5\,E{+}0.5\,V{+}O_{\mathrm{fmt}}$	E (format also varies; V group-constant)	**No** (O is format alone)
BFCL	schema/precondition (static)	$E+O$ (unit weights; no V)	E and O	No (O is a goal-completion read)
Deep Search	search-query state transition	$0.5\,E + 0.5\,V + O_{\mathrm{fmt}} + O_{\mathrm{term}}$	V (E saturates; O group-constant)	Yes (O_{term}; group-constant)
SciWorld	native valid-set; pre-action step-judge	$0.5\,E + 0.5\,V + O_{\mathrm{fmt}} + O_{\mathrm{term}}$ (coarse)	pre-action step-judge step-credit (coarse $E/V/O$ inert at floor)	Yes (O_{term}; group-constant)

on ALFWorld, static schema/precondition analysis on BFCL, the simulator's native valid-action set on SciWorld, search-query state transition analysis on Deep Search), so the shared symbol names a shared question, not a shared implementation. Second, the ALFWorld 78.4 cell carries *no terminal-outcome term*: ALFWorld logs no per-episode scalar, so its unit-weight O leg is a format read alone, and that cell is best read as an execution-grounded, E-carried instantiation. Third, BFCL instantiates only *two* unit-weighted terms, E and O, with no separate V; and across all four cells V is nowhere the independent gradient carrier: a group-constant level read on ALFWorld/SciWorld, absent on BFCL, and gradient-carrying on Deep Search when candidate augmentation is used.

One execution question, four mechanisms that answer it. E asks the same question in every cell (did this action take effect in the state the observation describes) under the same constraint, that the answer come from logged content. What differs is the mechanism, because that is set by

which observable carries execution semantics in the environment: a verifier language model on ALFWorld, static schema-and-precondition analysis on BFCL, the simulator's valid-action set on SciWorld. Where the environment already publishes admissibility, reading it beats asking a model; where it does not, a frozen verifier is the only fixed read available. Writing the four as a single symbol is therefore right about the structure and would be wrong about the implementation, so we tabulate both. This yields a split in *who computes the reward*: ALFWorld's E and V and SciWorld's V come from a frozen verifier model co-hosted with training (those cells reserve part of the node for serving, and their credit is not bit-reproducible across serving stacks), whereas BFCL's reward path contains no model call at all and reproduces exactly. The verifier is prompted, never trained, so both halves remain fixed reads.

V is scoped, not load-bearing. We state the subgoal term's status plainly, because it constrains what the results can be read to show. On ALFWorld and SciWorld V is a *level* read of the pre-action observation, shared by every candidate in a group; by the design rule of Sec. 4 it is therefore *gradient-dead by construction* under GRPO: a known, accepted property of the configuration that produced our strongest cell, not a bug found later. On BFCL it is not instantiated separately at all. There is consequently **no cell in which V is independently load-bearing**, and we do not claim a uniform "V measures subgoal advancement": the two subgoal instantiations act as a cross-group baseline, not a within-group signal, and the one V that ranks candidates is a content match rather than a subgoal read. Isolating a step-depth contribution therefore remains open (Sec. G). The pre-launch variance measurement on ALFWorld quantifies the split: over 800 logged decision points E has non-zero within-group spread on 266 (33.3%) and V on 1 (0.1%), so E is the sole gradient carrier in that cell.

O reads compliance plus whatever resolution the log carries. On ALFWorld O is format alone, because ALFWorld logs no per-episode scalar; on SciWorld it is format plus the logged terminal reward; on BFCL the format constant was replaced outright by the goal-completion read below, which is the only reason that cell has a second informative axis. Where a terminal component exists it is shared by every candidate at a decision point, so within-group normalization removes it exactly as it removes a constant V: it shifts the cross-group baseline and supplies no per-step gradient. The format component is the exception: it varies with the candidate.

ALFWorld. E is a frozen verifier asked, from the goal and the trajectory so far, whether the proposed next action is correct, emitting one sentinel verdict parsed to $\{+1, 0, -1\}$ (thinking disabled, as is the policy's own generation). V decomposes each game's goal once into three to five verifiable subgoals and scores the fraction already satisfied in the pre-action observation. O is a format term over the environment's action vocabulary; ALFWorld logs no per-episode scalar, so there is no terminal component. The DPO realization differs in two ways worth naming: its E reads the *logged* post-action observation as ground truth on the logged side and a predicted-feasibility verdict on the constructed side, and its V is a *delta* (satisfied-after minus satisfied-before) rather than a level, because a logged pair supplies both endpoints.

The three terms at one decision point. The soapbar failure of Sec. 1 recurs *inside* our training data. Here is a decision point from the 500 round-1-policy rollouts the round-2 trainer consumed, in an episode that *succeeded* in seven steps:[2]

> `examine countertop 1` → "*Nothing happens.*" (not there yet; it navigates next)
> `go to countertop 1` → "*You arrive at countertop 1. … you see … a mug 3 …*"
> `take mug 3 from countertop 1` → "*You pick up the mug 3 from the countertop 1.*"

The terms divide the labor exactly as Fig. 3 lays out: because the episode succeeded, terminal credit stamps $+1$ on the inert `examine` and the productive `go to` identically and cannot order them, O (format) and the level-read V are group-constant, and only E reads the precondition-violating `examine` (whose observation is "*Nothing happens.*") as inert. What this decision point adds beyond the figure is the aggregate that evidences E as the carrier: the pre-launch variance measurement

[2] Task `pick_cool_then_place_in_recep-Mug-None-CoffeeMachine-23`, trial `T20190908_035419_625837`; goal: cool a mug, put it in the coffee machine.

over 800 logged decision points finds non-zero within-group spread for E on 266 (33.3%) and for V on 1 (0.1%), so E is the sole gradient carrier in this cell. Per-candidate term values were not logged in training, so what E *would* return on this specific pair is an inference from the verifier's definition, not a recovered value; the aggregate spread, not this pair, is the evidence.

BFCL. Here the whole operator is deterministic Python over the prompt string. E recovers the tool schemas, the available-tool list, and the full user goal from the prompt the model was shown, then scores the candidate call: no parseable call, an unavailable tool, missing required arguments, or (the term that does the work) an argument whose *representation* contradicts its documented field while a resolver tool for that representation sits available and uncalled, i.e. a skipped in-context prerequisite. BFCL instantiates the structure in **two** terms rather than three: the subgoal slot is not instantiated separately here, and its goal-completion read (does the tool match an action the goal asked for, does it re-do something the goal states is done, does it preserve values the goal stated) is absorbed into O, where it replaces the format constant.

SciWorld. SciWorld admits the cheapest E and O of the four: the simulator logs a valid-action set at every state and a terminal task score, so both are native reads requiring no judgment, with V the same subgoal-level verifier as ALFWorld. The *coarse* instantiation of the operator (native valid-set membership as E with the ALFWorld-style subgoal-level V) is gradient-inert at this near-floor base: the pre-launch variance measurement returns near-zero within-group spread for both of these candidate execution terms, because a near-floor policy emits roughly two distinct actions per eight samples (as on BFCL), all shallow navigation synonyms. What that measurement rules out is the coarse *action*-admissibility read of the step, not step-credit as such. Even the two-or-so distinct actions the group samples differ in whether they *advance the task*, where the coarse valid-set read (both actions are "valid") and the flat terminal outcome (both episodes fail) cannot tell them apart. Our working SciWorld cell is therefore the refinement that reads exactly that distinction: an access-legal *pre-action step-judge* of whether the proposed step advances the task, which from the same near-floor base clears a matched terminal-credit ablation by 12.0 % (63.5 % vs. 51.5 %; Sec. D.4). We report the coarse instantiation's inertia as the motivation for that refinement, not as a dead end.

A departure from vanilla GRPO, disclosed. One of the three cells modifies the optimizer's candidate set, and it is not standard GRPO. It exists because of the candidate-homogeneity wall and conditions how that cell should be read. **BFCL uses a decode-time diversity lever.** A redesigned read can rank candidates but cannot create them, and most BFCL groups are action-collapsed at default decoding, so we raise sampling temperature and add a repetition penalty at rollout to obtain within-group variance. This changes the sampling distribution the reward is evaluated on. It introduces no oracle and no environment access, but it is a deviation, and we disclose it even though the BFCL cells report no gain for it to explain, since no arm of any access class separates on that benchmark at this scale (Sec. F.1).

D Full Per-Environment Result Ladders

D.1 ALFWorld behavioral breakdown: the soapbar failure mode in aggregate

The episode of Sec. 1 turned on *precondition-violating* interactions (`open`/`take`/`clean` X issued while not co-located with X, observation "Nothing happens.") and that failure mode is what separates the arms at scale, which is where we rest the claim rather than on one trace. On the *clean a soapbar and put it in a cabinet* task whose failure motivated the approach (Sec. 1), our round-2 policy succeeds in seven steps, navigating to each object before acting on it:

> **Ours (round 2).** `go to countertop 1` → "*you see a soapbar 1*"; `take soapbar 1 from countertop 1` → "*You pick up the soapbar 1*"; `go to sinkbasin 1`; `clean soapbar 1 with sinkbasin 1` → "*You clean the soapbar 1*"; `open cabinet 1`; `put soapbar 1 in/on cabinet 1` → "*You move the soapbar 1 to the cabinet 1.*"

where the terminal-credit ablation instead loops on precondition violations (opening cabinets it never checked, cleaning a soapbar it never holds) and dies at the step cap (the failing trace is in Sec. 1). This is one temperature-0 episode, an illustration not proof: on a sibling trial some baselines also

succeed, so the inferential weight rests on the aggregate. As a share of environment actions across the 134 games, such violations run 1.91% for ours (round 2) vs 7.25 (outcome-GRPO), 6.69 (SFT-floor), 5.70 (base), 5.08 (RFT), 4.61 (ETO (v2)), 4.48 (ETO (verifier)); the sharpest contrast is *same-lineage*, our own 4.42 at round 1 falling to 1.91 at round 2, a pair sharing data, backbone, and reward shape and differing only by a round of the operator. The overall "Nothing happens." rate falls the same way (9.2% vs 13.1–13.3% for every baseline), and the wasted opening prefix is gone: our first environment action is a no-op in $0/134$ games against $5/134$ for outcome-GRPO. Per-game flips agree but are individually weak: we solve 16 games outcome-GRPO misses (22 vs SFT-floor, 20 vs ETO (verifier), 18 vs RFT, 15 vs ETO (v2)) while it solves 11 we miss, so flips run both ways and net to the small margins of Tab. 1. Each flip is one temperature-0 decode; the behavioral aggregate, not any single game, is the evidence.

D.2 Per-dataset detail: the ALFWorld round ladder

This appendix expands the condensed ALFWorld row of Tab. 1 into the full per-round (R1/R2/R3) ladder, with the efficiency and capability-retention checks; the SciWorld ladder is in Sec. D.4, and BFCL, where no arm separates from base, is summarized among the negatives (Sec. F.1). Access legend as in the body: ✓ access-matched (gold-free logged-only), × privileged (exploratory re-collection / re-execution / post-action returns / oracle / gold), – baseline / not access-constrained. Per-cell SE $\approx 4\,\%$ (ALFWorld).

ALFWorld (OOD-134 success rate). The main result is the multi-round GRPO cell: base 67.2 → R1 76.1 → **R2** 78.4 (11.2 % over base, 2.3 % over its own round 1), at lower cost than the base (−27% actions, −16% tokens: 24.0 actions and 29,476 tokens per episode, the leanest same-setup round-2 arm, below even the next-cheapest self-training floor; no reward/eval divergence); a third round adds a modest further gain to R3 81.0, within the single-seed ±4 % band (the main result remains R2 78.4). 78.4 is the highest round-2 score in the ladder below among same-backbone methods; GiGPO's 77.6 is close but off-setup and cross-backbone. Several trajectory-level comparators also gain from round 1 to round 2 here, so we make an endpoint claim rather than a rate-of-accumulation claim (Sec. G). Carrying the same credit into round 2 under DPO instead moves the opposite way (70.1 → 66.4), an asymmetry we scope to this environment and this operator (Sec. 5.2). The field's iterative DPO methods re-collect fresh rollouts each round (Song et al., 2024; Xi et al., 2025a), manufacturing the "good-versus-less-good-but-valid" pairs goal-advancement credit needs, whereas our completely-offline variant scores only the one logged path, where near-expert logs hold mostly "good-versus-broken" contrasts, which is why it does not compound offline.

Method	R1	R2	R3	Access
base		67.2		–
ours (step credit, GRPO)	76.1	**78.4**	81.0	✓
ours (step credit, DPO, completely offline)	70.1	66.4	–	✓
outcome-GRPO (ablation)	67.5	74.6	–	✓
RFT (reward-filtered, STaR/ReST$^{\mathrm{EM}}$-style)	70.1	70.1	–	✓
SFT-floor (behavior cloning, unfiltered)	71.6	69.4	–	✓
ETO (trajectory contrast, 2 variants)	65.7/68.7	71.6/73.9	–	× (rollouts)
GiGPO (post-action return)	77.6	77.6	–	× (post-action)

ETO enters as two trajectory-contrast variants (a verifier-scored variant and a revised one) whose scores are listed as a pair in each round column; both improve from round 1 to round 2. Its *contrast* is over whole trajectories rather than steps, which is orthogonal to our access rule, but the trajectories it contrasts are obtained by exploring the environment afresh with the current policy, so it is grouped with the privileged arms in Tab. 1 rather than treated as a matched-access floor; granted that access it still ends below our round 2 (73.9 vs 78.4). The outcome-GRPO round-2 update is largely gradient-free (91.7% of steps carry no gradient), so its round-2 score reflects little of the intended learning signal. The RFT row is STaR/ReST$^{\mathrm{EM}}$-style reward-filtered self-training (Zelikman et al., 2022; Singh et al., 2023; Gulcehre et al., 2023; Yuan et al., 2023): sample rollouts from the current policy, keep the ones the environment scored as successful, fine-tune on those, and iterate across both rounds. The SFT-floor row is the plain behavior-cloning control: it fine-tunes on *all* self-generated rollouts (successes and failures alike, no success filter), restarting from base each round. Success fil-

tering did not help here: BC (71.6) edges RFT (70.1), within the band. Both are access-matched here (the filter is the logged terminal outcome; no gold answer, expert demonstration, or re-execution), and both are flat-to-down at round 2 ($70.1 \rightarrow 70.1$, $71.6 \rightarrow 69.4$): whole-trajectory outcome filtering on a base that already succeeds 67.2% of the time re-clones behavior the policy has, so the second round adds little. The agentic member of this family, ReST-meets-ReAct (Aksitov et al., 2024), applies the same outcome filter to multi-step ReAct trajectories; the rationalization variant of STaR instead conditions on the gold answer to fix failed attempts (Zelikman et al., 2022), which our access rule forbids, and the verifier-augmented variant (V-STaR) re-ranks samples at inference time rather than supplying a training-time step signal, so neither enters as a row.

Capability retention, measured round-matched. Across all ALFWorld and BFCL arms, held-out GSM8K stays flat per backbone (no catastrophic forgetting; Sec. A.1). One methodological point matters for how this check should be read. An earlier version of it compared *our round-2* arm against baselines measured at *round 1*, which is a round mismatch, so we re-ran every ALFWorld baseline's retention on its own round-2 policy: eval-only, serving each already-merged round-2 checkpoint directly rather than loading an adapter, so a silent no-op is impossible by construction, and the arm-to-arm spread in the scores confirms distinct policies were served. Round-matched, the check is a *wash*. Against our round 2 (GSM8K 93.6, i.e. 1,235/1,319) the same-backbone round-2 baselines land at GSM8K 93.7–94.8 (ETO-v2 93.7, ETO-verifier 94.0, SFT-floor 94.2, RFT 94.3, outcome-GRPO 94.8: all $0.4\,\%$ to $1.2\,\%$, at or marginally *above* ours). GiGPO is read only *within-arm* because it is a cross-backbone Qwen2.5-7B and its absolute GSM8K level is not comparable to the 9B arms; on that within-arm reading it is likewise flat round-over-round (GSM8K $85.9 \rightarrow 86.7$, $0.8\,\%$). We therefore report "no measurable forgetting" as round-matched and as a property *every* arm here has (a safety check the operator passes rather than an axis on which it separates) and we make no retention-based claim of superiority.

D.3 SciWorld, in full

D.4 SciWorld, in full

This appendix expands the compact treatment of Sec. 5.2: SciWorld, where the operator's access-legal pre-action step-judge instantiation is our second step-credit environment result (Sec. 5.2). SciWorld uses a text-only Qwen2.5-7B backbone, and GSM8K stays flat across arms (no catastrophic forgetting). Access legend as in the body: matched (gold-free logged-only), off-setup (exploratory re-collection, re-execution, post-action returns, oracle, or expert demonstrations), ctx (published / different-recipe context), – (not access-constrained).

SciWorld: the metric axis, and why we report success rate. SciWorld's harness emits several per-item quantities, and the comparison to published numbers is only meaningful on one of them, so we fix the axis before reporting anything. The published numbers for this environment are *success rates*: the fraction of held-out items solved completely, times 100. We verify rather than assume this. Every cell of the source paper's per-category table is exactly $100\,k/d$ for integers $k \leq d$, and the decoded per-category denominators ($53{+}83{+}44{+}5{+}15$) sum to 200, which is exactly the size of the held-out split we evaluate on; an item therefore contributes either 0 or 100, so the published quantity is a binary success rate over 200 items and our own success rate is computed the same way (count of items scoring exactly 100, over 200). Two other quantities the same harness reports are on different axes and are not comparable to it. The *signed partial-credit average* is the mean per-item score and can be negative; and a reported "pass@1" counts items scoring *anything above zero*, which on the checkpoint whose success rate is 60.5% is 77%: a partial-progress rate, not a success rate. Neither is compared to a published success rate anywhere in this paper, and where we give more than one axis we label each.

SciWorld: results. The base is near-floor (4.0% success rate), and it agrees with the published untrained base for the same backbone (1.5%) to within five items: the base-agreement check that the two are on one scale. The same floor makes a reward-filtered self-training arm vacuous here (with almost no successful rollouts there is nothing to filter and clone), so no self-training floor row is reported for SciWorld. What the environment does supply is the terminal-credit comparator, because on SciWorld the published AgentGym-RL recipe *is* trajectory-level outcome-GRPO, the

same ablation arm we contrast against on ALFWorld. Reproduced faithfully on the native harness, it reaches a 60.5% success rate ($121/200$ items) at step 375, which is $10.0\,\%$ above the published $50.5\,\%$ ($101/200$) on the same backbone and axis. That margin is obtained at *fewer* updates, not more, which is the form of the comparison we intend. The upstream recipe's own launch script trains for $1{,}320$ optimizer steps ($2{,}120$ training items at batch $16 = 132$ steps/epoch $\times$ 10 epochs), so our step-375 checkpoint is $\approx 28\%$ of the published training budget. We state this as a budget-fraction claim (comparable-or-better at roughly a quarter of the published schedule) and explicitly *not* as evidence of a better recipe, because the comparison is a reproduction of their method rather than a run of ours.

Method	**Success rate (%)**	**Access**
base (ours)	4.0	–
published base (same backbone)	1.5	ctx
our pre-action step-judge (GRPO), step 680	$\mathbf{63.5}$	matched
terminal-credit control (matched ablation, best checkpoint)	51.5	matched
our AgentGym-RL reproduction (step 375)	$\mathbf{60.5}$	ctx
published AgentGym-RL (Xi et al., 2025b)	50.5	ctx
published ScalingInter (Xi et al., 2025b)	57.0	ctx

SciWorld: the access-legal step-credit operator. Our SciWorld operator is an access-legal *pre-action step-judge*: a binary judge of whether the proposed next step advances the task, scored from *pre-action* context alone, so it is access-legal under the strictest pre-action reading and never reads the logged score delta. It is the SciWorld instantiation of the same E/V/O operator, refined to the observable this environment makes reward-predictive: the coarse valid-set instantiation was inert at the floor (Sec. C.1), and the pre-action step-judge is the working read. Trained from the same near-floor base as the terminal-credit control, it rises $49.0 \to 52.5 \to 58.0 \to 60.5 \to 61.5$, then dips and peaks at a best of $\mathbf{63.5}\%$ (step 680), clearing the matched terminal-credit control ($51.5\,\%$) by $12.0\,\%$ (an isolation of step-credit from terminal-credit, since the control is the identical recipe with only the step-credit term removed) and clearing both published RL baselines on this split and backbone (AgentGym-RL 50.5%, ScalingInter 57.0%); it reaches the outcome-GRPO teacher reproduction (60.5%). It presents a genuine step-credit-beats-terminal-credit result on a third environment: the low-outcome-variance regime the design rule predicts the operator should win in (Sec. 4).

One caveat bounds the number and is not removed by the size of the margin. *Split contamination, disclosed.* The AgentGym split we train and evaluate on overlaps itself by $61/200$ items (30.5%), a defect shared with every published SciWorld number on this split; comparability survives (the same defect sits on both sides) but the disclosure obligation does not, so we make it here.

SciWorld: why the comparator coverage is thin. The SciWorld column carries fewer comparators than ALFWorld, and each absence follows from a property of the environment at this scale rather than being an omission. Reward-filtered self-training (STaR/ReST$^{\mathrm{EM}}$/RFT) is vacuous, because a base solving 4% of items produces almost no successes to filter and clone. GiGPO's return-grouping has no SciWorld implementation available, so it cannot be run here even as an off-setup ceiling. And an unguarded coarse GRPO run diverges (our SciWorld step-judge arm runs under a stability guard). What the column *does* contain is the two rows that matter most for the argument: our operator's own cell, the access-legal pre-action step-judge run, which clears the matched terminal-credit control by $12.0\,\%$ ($63.5\,\%$ vs. $51.5\,\%$) from the near-floor base (above), and the terminal-credit / AgentGym-RL comparator, filled and sitting above the published number for that recipe. We report only the comparators we have measured.

E Baseline Disposition

E.1 Baseline treatment: the two arguments in detail

This appendix expands the compact treatment of Sec. 3 and Tab. 5.

Table 5: Baselines sorted by the counterfactual-access axis and answered by the two arguments (full walkthrough and numbers: Sec. E.1). *Argument 1*: ablate the forbidden counterfactual; the residual is a baseline we ran (ALFWorld success rate; our multi-round GRPO result (78.4) beats each residual). *Argument 2*: run as-published off-setup for an honest ceiling.

Comparator	Access verdict / ablation → residual	Number
Argument 1: degenerates, under our access budget, to a baseline we ran		
AgentEvol (Xi et al., 2025a)	ablate fresh-env exploration → RFT	70.1
GiGPO (Feng et al., 2025)	ablate anchor-state grouping → outcome-GRPO	67.5 ≈ base
IGPO / CIGPO (Wang et al., 2026b; Dou, 2026)	access-matched; head-to-head	51.0/41.5 (BFCL, in-band)
Argument 2: off-setup ceiling, granted the forbidden information		
ETO (Song et al., 2024)	collects fresh rollouts (contrast is logged)	65.7/68.7 → 71.6/73.9
Our off-setup BFCL recipe	re-executes generated calls (ceiling)	52.5/48.0 (in-band)

Argument 1: under our budget, the method is a baseline we already beat. Ablating the forbidden counterfactual out of each method leaves a residual we can name and, where we have run it, beat with our multi-round ALFWorld-GRPO result (78.4). AgentEvol (Xi et al., 2025a) buys its signal from *fresh-environment* exploration; remove it and the update is reward-filtered fine-tuning on the surviving successes: our STaR/ReST$^{\mathrm{EM}}$-style RFT baseline (Zelikman et al., 2022; Singh et al., 2023) (70.1). GiGPO (Feng et al., 2025) forms localized step advantages from the discounted return-to-go realized after each action, grouped across sibling rollouts by *repeated state*. That return is a *post-action* read of a freshly generated action, available only once the action has been executed and its future observed, which our pre-action GRPO setting forbids ($\kappa(t)$ stops at c_t); it is not a re-execution, though the offline grouping does need states to recur across the on-policy rollouts. Strip the localized return grouping and the residual is the terminal signal: our outcome-only GRPO baseline (67.5 ≈ base 67.2). GiGPO does not "reduce to" outcome-GRPO: it *ablates down* to it, the terminal residual being what remains after the post-action localized grouping is removed. ETO (Song et al., 2024) enters under Argument 2 instead: its *contrast* is over whole logged trajectories, but it obtains those trajectories by exploring the environment afresh with the current policy, which our rule denies, so we run it as-published as a privileged comparator (65.7/68.7 at round 1, 71.6/73.9 at round 2) rather than counting it as a matched-access floor, and granted that access it still ends below our round 2 78.4. Fission-GRPO (Zhang et al., 2026b) and Environment Tuning (Lu et al., 2026) both acquire their per-step signal from a fresh environment query on generated actions (an error-simulator re-execution and a live-environment augmentation, respectively); once that query is denied, nothing on-setup-novel survives and each residual collapses onto our existing reference-injection / outcome baseline, so the degeneration is established analytically. Two comparators are already access-matched and were run *head-to-head* under our budget: a log-prob-gain reward in the IGPO/CIGPO/progress-advantage form (Wang et al., 2026b; Dou, 2026; Oh et al., 2026), which credits a step by the change in the policy's likelihood of the correct answer, lands in-band (51.0/45.5 and 51.0/41.5 for the two variants, against the matched base 49.2/45.7); its per-step signal is candidate-blind and gradient-dead on the homogeneous candidate sets, consistent with route (iii) of Section 2, though on this benchmark that null is not separable from the benchmark's own lack of headroom (Appendix F.1). A CM2-style checklist reward (Zhang et al., 2026a) is a boundary case rather than an access-matched one: an LLM-judge checklist alone hits the candidate-homogeneity wall of Appendix F.1, so making it live at all requires the off-rule reference injection to supply within-group variance, which places it with the excluded gold cells rather than the matched arms.

Argument 2: granted the forbidden information, they still do not contradict us. The second argument concerns methods run *as-published*, off-setup, with the counterfactual access we deny at deployment. We state it in a form that does not depend on any single method's as-published number. For each such method, either **(i)** it fails to beat our access-matched number *despite* the advantage, or **(ii)** it beats us, but only by consuming information forbidden at deployment (environment re-

execution, or a live/simulated environment query on a *generated* action) so its win is off our access rule and does not contradict our access-matched claim. The disjunction "either they lose despite the advantage, or their win is purchased with forbidden information" is true regardless of which number returns. On BFCL the disjunction is resolved in a degenerate way that is worth stating plainly: an off-setup reward that re-executes generated calls reaches $52.5/48.0$, the highest number on that benchmark, and is still within decode noise of the untrained base ($+3.3$ on the `base` category). Case (i) therefore holds there trivially (no arm, privileged or not, demonstrates a separation) so BFCL contributes no evidence to either side of the disjunction (Appendix F.1). The access-matched side of the disjunction rests on ALFWorld-GRPO, and we do *not* assert the published methods are "still worse." Environment Tuning (live environment) and CM2 (LLM-simulated environment) (Lu et al., 2026; Zhang et al., 2026a) are off-setup by construction: both consume a fresh environment response to a generated action, so whichever side of our number an as-published run lands on, each resolves into (i) or (ii) by the disjunction above and enters as an off-setup ceiling rather than a matched comparison.

F What We Tried That Did Not Work

F.1 Consolidated negative results

We collect here, as negative results rather than supplementary experiments, the approaches and settings that did not produce a separation we can rely on. Each is stated plainly, with the one-line reason it failed and, where a detailed treatment remains, a pointer to it; we do not re-analyze the numbers here.

- **BFCL does not discriminate among methods at this scale.** The matched-harness base is already high ($49.2/45.7$), and against it *every* arm we ran sits within the measured $\pm 4\,\%$ decode band: our observation-grounded read, its ablations, and even the privileged re-execution ceiling our access rule forbids ($52.5/48.0$). This is a finding about the *benchmark*, not about our method: with no arm of any access class separating from the base, there is no method ordering to draw, which is why BFCL is reported here rather than in the body. It is not evidence that our method underperforms on BFCL; it is evidence that the benchmark cannot resolve credit-assignment methods at this scale and backbone.
- **Reference injection to manufacture within-group variance.** Injecting the benchmark's gold reference call as a candidate does create a rankable spread, but it consumes an oracle answer the access rule forbids, so we excluded it as off-rule, and it is in-band regardless.
- **Likelihood-change step rewards** (log-prob-gain / progress-advantage, the IGPO/CIGPO form). Crediting a step by the change in the policy's likelihood of the correct answer is annihilated by within-group mean-subtraction on the homogeneous candidate sets, and lands in-band on BFCL.
- **Supplying variance without correctness-alignment.** Within-group variance is necessary but not sufficient: a high-variance yet weakly correctness-aligned reward is *worse* than a gradient-dead one, since it reward-hacks the noisy surface, and a naive logged substitute that merely mirrors an injected reference's variance destroys the reward's correctness-alignment.
- **The coarse E/V instantiation on SciWorld.** Native valid-set E with the ALFWorld-style subgoal-level V is gradient-inert at the near-floor base, where the sampled candidates are too few and too alike to separate; this motivated the pre-action step-judge refinement that is our actual SciWorld cell (Appendices C.1, D.4).
- **Training SciWorld to the full published budget.** The faithful upstream recipe is unstable at 8B (it collapses into an absorbing state before the schedule completes, across independent divergence events) so more budget makes the number worse, and we keep SciWorld labeled a reproducibility boundary (Appendix D.4).
- **Replacing the observation read with a pre-action self-belief.** A model's own pre-action judgment of its next step cannot recover the roughly half of failures whose cause is hidden backend state absent from the pre-action window: a floor for any purely pre-action method, and the reason our credit reads the realized observation rather than an elicited belief (Appendix B.1).

G Limitations and Open Questions

The results of §5.2 are stated at the strength their numbers carry. Three bounds are worth stating at paper level. **Scope of step-level vs. trajectory-level gains.** On matched evaluation harnesses, the performance delta between step-level and trajectory-level credit remains closely matched. Because the verifier term V operates jointly with execution grounding (§4), step-depth is evaluated as an integrated component; GRPO/DPO optimizer dynamics are specifically analyzed under ALFWorld. **Rate of accumulation and multi-round convergence.** The Round-2 78.4 performance represents an endpoint score under sequential policy updating. While trajectory-level baselines also demonstrate round-over-round gains, evaluating fine-grained accumulation rates requires multi-seed evaluations across standardized initializations. *Our strongest, beyond-band result is one environment, and its winning term is one axis.* ALFWorld carries the clean win, and on that cell execution-grounding (E) is the gradient carrier while V (a level read of the shared pre-action observation) and O (format) are inert by that instantiation's construction. BFCL does not separate any arm from base; and on SciWorld the operator's step credit *does* clear its matched terminal-credit control ($63.5\,\%$ vs. $51.5\,\%$, $12.0\,\%$) from the near-floor base (precisely the low-outcome-variance regime terminal credit stalls in), which we report as an early positive whose split contamination we disclose (App. D.4). Our per-cell standard error is $\approx 4\,\%$ (ALFWorld) / $3.5\,\%$ (SciWorld), and we hold every in-band difference to that band. Two questions stay open: a decisive role for V, and a principled weighting of the three terms in Eq. 4. We present E/V/O as a working decomposition, not a proven sufficient statistic of the step (closest-neighbor separation in App. B.3).