SKILL-AS-CONTROLLER: A NEW PARADIGM FOR SKILL UTILIZATION IN LONG-HORIZON AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Skills are crucial for LLM agents performing long-horizon tasks. However, when used only as prompt context, their effect on action selection remains implicit and suboptimal. Our analysis reveals a *skill utilization gap*: agents often violate skill rules even when relevant skills are provided, while the same model can often detect these violations afterward. We therefore propose **Skill-as-Controller**, a new paradigm that uses skills to explicitly control the action distribution. At each action step, Skill-as-Controller evaluates candidate actions based on the current state and skill rules, producing skill–action consistency scores. These scores are then combined with the base action distribution through a KL-regularized policy update, yielding a closed-form skill-controlled policy without model parameter updates. Experiments on three long-horizon benchmarks with diverse LLM backbones show consistent improvements in task success and interaction efficiency. On ALFWorld, Skill-as-Controller enables Qwen-3.5-4B to achieve a 92.5% success rate, exceeding the baseline by 58.9 points, while reducing the average number of interaction steps from 24.93 to 12.40. Further analysis shows that Skill-as-Controller reduces common failures, such as state loops and prerequisite violations, while the consistency scores make the effects of skills easier to inspect. These results establish Skill-as-Controller as an effective, efficient, and interpretable paradigm for skill utilization in long-horizon agents.

1 INTRODUCTION

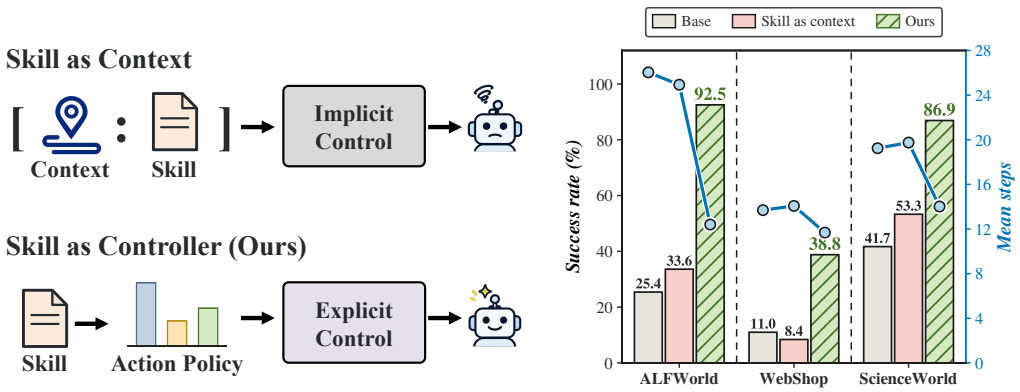
Skills provide reusable rules and guidance that help LLM agents solve long-horizon tasks (Chen et al., 2024; Wang et al., 2025). Recent work has shown that skills can substantially improve agent performance, motivating growing efforts to improve skill quality (Wang et al., 2026a; Yang et al., 2026b) and integrate skill knowledge into agents (Lu et al., 2026; Wang et al., 2026b; Yang et al., 2026a). Despite their success, an equally important question has received much less attention: *once a useful skill is available, how should an agent use it to guide its actions?*

Skills are typically included in the prompt as in-context guidance for the agent (Nottingham et al., 2024; Chen et al., 2024; Wang et al., 2025; Fu et al., 2024; Zhao et al., 2024). We refer to this common skill utilization paradigm as *Skill-as-Context*. Under this paradigm, skills provide contextual guidance for the agent but do not directly determine its actions. The model must interpret the provided skill, identify the rules relevant to the current state, and decide how to apply them when selecting the next action. As a result, the influence of skills on action selection remains implicit and largely depends on the model's reasoning over the prompt. Therefore, simply providing a relevant skill does not guarantee that the agent will follow its rules.

Our analysis further confirms this limitation and reveals a clear *skill utilization gap* under Skill-as-Context. Even when the relevant skill is provided in the prompt, agents still often select actions that violate its rules. Interestingly, the same model can often detect these violations when evaluating the actions afterward. In other words, the model may fail to follow a skill when generating an action, but can still recognize that the action violates the skill. It therefore motivates a different way of using skills: *we can use skill to explicitly evaluate candidate actions and provide direct guidance for action control.*

To this end, we propose **Skill-as-Controller**, a new skill utilization paradigm that uses skills as explicit signals for action selection rather than only as prompt context. Instead of relying on im-

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.



(a) ***Skill as Controller vs. Skill as Context*** **(b)** ***Higher Success & Fewer Steps***

Figure 1: **Comparison of skill utilization paradigms and execution performance.** **(a)** While Skill-as-Context relies on implicit context reasoning where skills often fail to constrain actions, Skill-as-Controller explicitly modulates policy distributions to govern action selection. **(b)** Evaluation across ALFWorld, WebShop, and ScienceWorld using Qwen-3.5-4B. Bars represent task success rates (%, left axis), and lines indicate mean interaction steps (right axis). Explicit policy control substantially elevates task success while compressing trajectory steps across all environments.

plicit skill following during action generation, Skill-as-Controller evaluates how well each candidate action agrees with the relevant skill, and turns this evaluation ability into explicit action control. Specifically, at each action step, skill-based evaluators assign a skill-action consistency score to each candidate based on the current state and skill rules. These scores indicate which actions better follow the skill and which are more likely to violate it. We then integrate the consistency scores with the base action distribution through a KL-regularized policy update (Peters et al., 2010; Abdolmaleki et al., 2018; Li et al., 2026). This update increases the probability of skill-consistent actions while preserving the preferences of the base policy, yielding a closed-form skill-controlled policy. In this way, Skill-as-Controller uses the model's skill evaluation ability to directly guide action selection.

Skill-as-Controller provides an effective, efficient, and interpretable way to use skills for long-horizon agents. We evaluate it on three long-horizon benchmarks using closed-source models, open-source models, and models fine-tuned with SFT or RL. Across these settings, Skill-as-Controller consistently improves task success. For example, on ALFWorld, Qwen-3.5-4B achieves a 92.5% success rate, outperforming Skill-as-Context by 58.9 points and reaching performance comparable to frontier closed-source models. It also improves task efficiency, reducing the average number of interaction steps for Qwen-3.5-4B from 24.93 to 12.40. Further analysis shows that Skill-as-Controller reduces common failure modes, including state looping and prerequisite violations, helping agents follow more reliable action paths. Finally, the explicit skill-action consistency scores make skill use more transparent by showing how skill rules affect candidate actions and the final action distribution.

Our contributions are summarized as follows:

- We identify a clear skill utilization gap in long-horizon agents, showing that providing skills in the prompt context does not ensure skill-compliant actions.
- We propose **Skill-as-Controller**, a new paradigm that scores candidate actions using skill rules and combines these scores with the base policy to explicitly guide action selection.
- We evaluate our method on three long-horizon benchmarks and diverse LLMs, showing improved task success and efficiency, fewer failure modes, and more transparent skill use.

2 SKILL UTILIZATION GAP

Skill-as-Context. Let S denote a natural-language skill, x_t the agent context at step t, and $\mathcal{A}_t$ the candidate action set. Let $\pi_0(a \mid x)$ denote the action distribution induced by the frozen base model

Table 1: Diagnostic analysis of skill utilization. (a) Action-Level Skill Rule Compliance across ALFWorld (ALF), WebShop (Web), and ScienceWorld (Sci), judged by GPT-5.5. (b) Detection of the same 125 skill violations collected from ALFWorld trajectories of Qwen-3.5-35B-A3B under Skill-as-Context. Direct Evaluation uses the original skill, while Rubric Evaluation uses action-evaluation rubrics derived from the same skill.

(a) Action-Level Skill Rule Compliance

Method	ALF	Web	Sci
Base	0.407	0.266	0.498
Context	0.498	0.268	0.570
Ours	**0.554**	**0.320**	**0.635**

(b) Violation Detection

Evaluation	Detected	Rate
Direct Evaluation	53 / 125	42.4%
Rubric Evaluation	**79 / 125**	**63.2%**

given context x. Under Skill-as-Context, the skill is appended to the agent context, $x_t^S = [x_t; S]$, and the action distribution becomes

$$\pi_{\text{ctx}}(a \mid x_t, S) = \pi_0(a \mid x_t^S). \tag{1}$$

In this paradigm, the skill affects the action distribution only through the model's contextual reasoning. There is no explicit signal that measures whether a candidate action follows the skill rules. The effect of the skill on actions therefore remains implicit in the model.

Limited Skill Rule Compliance. We first examine whether actions produced under Skill-as-Context follow the provided skill. We use GPT-5.5 as an independent judge to evaluate each executed action against the skill rules that apply to the current state. As shown in Table 1(a), action-level skill rule compliance under Skill-as-Context remains limited across all three benchmarks. Even when the relevant skill is provided, many actions still conflict with its rules. This reveals a gap between providing a skill in context and following it in behavior, which we call the **skill utilization gap**.

Explicit Action Evaluation. We next ask whether these violations necessarily mean that the model cannot identify that an action conflicts with the skill. We collect 125 violating actions from ALFWorld trajectories generated by Qwen-3.5-35B-A3B under Skill-as-Context, where each violation is identified by the GPT-5.5 judge described above. We then ask the same Qwen-3.5-35B-A3B model that generated these actions to independently judge whether each action follows the original skill. It identifies 53 of the 125 violations (42.4%). Thus, some violations that are not avoided during action generation can still be recognized when the same model evaluates the action separately.

We further derive action-evaluation rubrics directly from the same skill, where each rubric specifies how a skill rule should evaluate a candidate action in the current state. Using the same Qwen-3.5-35B-A3B model to apply these rubrics to the same 125 actions increases the number of detected violations from 53 to 79, corresponding to a detection rate of 63.2% (Table 1(b)). Rubric-based evaluation therefore identifies 26 additional violations over direct evaluation with the original skill. This result shows that organizing skill rules around explicit action evaluation can provide a stronger action-level signal than directly judging actions against the original skill.

Implication for Skill Utilization. Together, these findings motivate a more explicit use of skills. Rather than relying only on prompt context, skills can be converted into evaluators that score candidate actions according to the skill rules. These scores can then provide an explicit signal for controlling the action distribution. Section 3 develops this idea as **Skill-as-Controller**.

3 METHOD

3.1 OVERVIEW: SKILL-AS-CONTROLLER

We propose **Skill-as-Controller**, which uses skills to explicitly control agent actions (Figure 2). Let p_0 denote the frozen agent LLM and π_0 its base action distribution. We construct skill-based evaluators to score candidate actions, aggregate their scores, and combine them with π_0 through a KL-regularized update to obtain the skill-controlled policy π_S (§3.2, §3.3). Among multiple candidate evaluator sets, we select $\mathcal{C}_S^*$ on training tasks before held-out evaluation (§3.4).

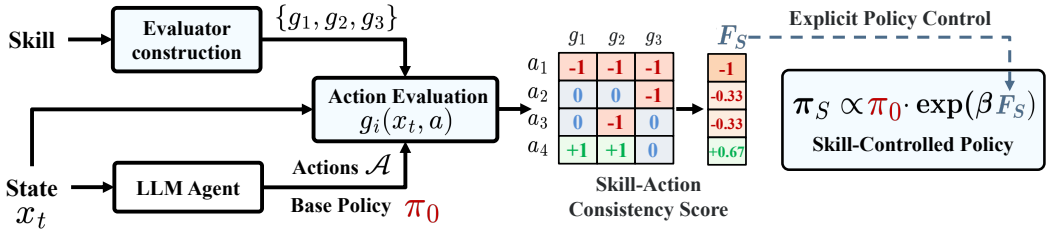


Figure 2: **Overview of Skill-as-Controller.** Given a skill, evaluators $\{g_1, g_2, g_3\}$ are constructed to assess candidate actions. At state x_t, the agent provides candidate actions $\mathcal{A}$ alongside the base policy π_0. The action evaluation module scores each action against skill rules to produce rule evaluations $g_i(x_t, a) \in \{-1, 0, +1\}$, which are aggregated into the skill-action consistency score F_S. Through explicit policy control, these scores adjust the base policy into the skill-controlled policy $\pi_S \propto \pi_0 \cdot \exp(\beta F_S)$, directly prioritizing skill-compliant actions over violating ones.

3.2 Evaluator Construction and Action Evaluation

We first construct an evaluator set from the given natural-language skill and then use the evaluators to score candidate actions in the current state.

Offline Evaluator Construction. Given a natural-language skill $\mathcal{S}$, we first decompose it into K skill rules, $\mathcal{S} = (r_1, r_2, \ldots, r_K)$, and generate an action-evaluation rubric for each rule. Each rubric specifies how the corresponding rule should evaluate a candidate action in the current state. An evaluator g_k is formed by combining the rubric for rule r_k with a scoring procedure, where the frozen agent LLM p_0 applies the rubric to the candidate action. Together, the K evaluators form a candidate evaluator set:

$$\mathcal{C}_\mathcal{S} = (g_1, g_2, \ldots, g_K). \tag{2}$$

The evaluator set is generated before task execution and remains fixed during inference.

Online Skill-Action Consistency Scoring. At decision step t, each evaluator g_k in $\mathcal{C}_\mathcal{S}$ scores each candidate action $a \in \mathcal{A}_t$ given the current context x_t:

$$g_k(x_t, a) \in \{-1, 0, +1\}, \qquad a \in \mathcal{A}_t, \tag{3}$$

where $+1$, -1, and 0 indicate that rule r_k encourages, discourages, or is neutral toward action a. If rule r_k is inapplicable in the current state, the evaluator returns 0. All evaluators are applied to each candidate action, making each rule's contribution to the skill-action consistency score explicit.

3.3 Explicit Policy Control

Aggregating Skill-Action Consistency Scores. We average the evaluator scores for each candidate action to obtain its skill-action consistency score:

$$F_\mathcal{S}(x_t, a) = \frac{1}{K} \sum_{k=1}^{K} g_k(x_t, a) \in [-1, 1]. \tag{4}$$

A positive score indicates that more rules encourage the action than discourage it, while a negative score indicates the opposite. This score is used for the subsequent policy control.

Controlling the Action Distribution. The skill-action consistency score $F_\mathcal{S}(x_t, a)$ measures how well an action follows the skill rules, while the base policy π_0 assigns action probabilities given the task context. We therefore seek to increase the expected skill-action consistency score while limiting deviation from the base policy. Thanks to prior work on KL-regularized policy optimization (Peters et al., 2010; Abdolmaleki et al., 2018; Li et al., 2026), we formulate this trade-off as the following KL-regularized objective:

$$\max_{\pi} \; \mathbb{E}_{a \sim \pi(\cdot|x_t)}[\beta F_\mathcal{S}(x_t, a)] - D_{\mathrm{KL}}\big(\pi(\cdot \mid x_t) \,\|\, \pi_0(\cdot \mid x_t)\big). \tag{5}$$

where $\beta \geq 0$ controls the strength of skill-based control. The optimal policy has the following closed form; its derivation is provided in Appendix B:

$$\pi_{\mathcal{S}}(a \mid x_t) = \frac{\pi_0(a \mid x_t) \exp(\beta F_{\mathcal{S}}(x_t, a))}{\sum_{a' \in \mathcal{A}_t} \pi_0(a' \mid x_t) \exp(\beta F_{\mathcal{S}}(x_t, a'))}. \tag{6}$$

The new policy increases the relative odds of actions with higher skill-action consistency scores.

3.4 Outcome-Guided Evaluator Set Selection

Different evaluator sets for the same skill can lead to different actions. We therefore compare candidate evaluator sets on training tasks and select the set with the highest task success rate.

Given a skill $\mathcal{S}$, we generate M candidate evaluator sets, $\mathcal{B}_{\mathcal{S}} = \{\mathcal{C}_{\mathcal{S}}^{(1)}, \ldots, \mathcal{C}_{\mathcal{S}}^{(M)}\}$, by generating different action-evaluation rubrics. We evaluate each candidate set on training tasks and select

$$\mathcal{C}_{\mathcal{S}}^{*} = \arg\max_{\mathcal{C} \in \mathcal{B}_{\mathcal{S}}} J_{\text{train}}(\mathcal{C}), \tag{7}$$

where $J_{\text{train}}(\mathcal{C})$ is the average task success rate when Skill-as-Controller uses evaluator set $\mathcal{C}$. The selected evaluator set is fixed before evaluation on held-out tasks.

4 Experiments

4.1 Experimental Setup

Benchmarks and Metrics. We evaluate on three long-horizon interactive environments: (1) **ALFWorld** (Shridhar et al., 2021), using the official `valid_unseen` split (134 tasks) across six household task types with a 30-step limit; (2) **WebShop** (Yao et al., 2022), using 500 held-out test goals from WebShop-small with a 30-step limit; and (3) **ScienceWorld** (Wang et al., 2022), using 336 official test variations across eight task families with a 50-step limit. We report task success rate as the primary metric, together with environment scores for WebShop and ScienceWorld and the mean number of interaction steps.

Backbones and Baselines. We evaluate closed-source models (GPT-5.5 and Gemini-3.6-Flash), open-source models (Qwen-3.5-4B, Qwen-3.5-35B-A3B, Llama-3.3-70B, and GPT-OSS-120B), and task-fine-tuned SkillRL models (Xia et al., 2026) (SkillRL-SFT and SkillRL-RL). We use the released SkillRL checkpoints without further training. For each backbone, we compare **Base** without skill guidance, **Skill-as-Context** with the skill provided in the prompt, and **Skill-as-Controller** (Ours) with explicit control through skill-action consistency scores.

Skills and Shared Configuration. We derive our skills from publicly available skill data from SkillOpt, D2Skill, and UCOB (Yang et al., 2026b; Tu et al., 2026b;a). Before evaluation, we manually simplify these skills to retain reusable procedural rules and remove task-specific answers and shortcut-like content, without using held-out evaluation outcomes. The same skills are shared across all backbones and used unchanged by Skill-as-Context and Skill-as-Controller. ALFWorld uses six task-family skills with 43 rules in total, while WebShop and ScienceWorld use 7 and 11 rules, respectively. For each backbone, the same model generates and applies the action-evaluation rubrics. We set $\beta = 2$, 1, and 2 for ALFWorld, WebShop, and ScienceWorld, respectively. For each benchmark, evaluator-set selection uses only its corresponding training set, which is fully separated from the held-out evaluation set; the selected evaluator set is then fixed for evaluation.

Action Selection and Evaluation Protocol. For open-source models, we obtain action probabilities from length-normalized action log-probabilities; for closed-source models, we convert model-reported confidence scores into an action distribution. The agent greedily selects the highest-probability action at each step. ALFWorld uses all environment-provided admissible actions, while WebShop and ScienceWorld use benchmark-specific candidate sets constructed from task information and model proposals. The same candidate construction and action-selection protocol are used across all compared methods.

Table 2: Performance on ALFWorld, WebShop, and ScienceWorld. Succ. denotes task success rate; Score denotes environment score; Steps denotes the mean number of interaction steps. Ours denotes Skill-as-Controller. Bold indicates the best result within each model group. * denotes closed-source models whose action distributions are obtained from model-reported confidence scores for candidate actions. We use the task-specific checkpoints released by SkillRL for ALFWorld and WebShop. ScienceWorld results for SkillRL are marked as – because the authors did not provide checkpoints for this benchmark.

		ALFWorld		**WebShop**			**ScienceWorld**		
Model	**Method**	Succ.↑	Steps↓	Succ.↑	Score↑	Steps↓	Succ.↑	Score↑	Steps↓
Closed-Source Models									
GPT-5.5*	Base	85.8	12.65	50.8	56.7	10.62	91.4	97.4	8.66
	Skill-as-Context	85.8	12.03	57.0	67.4	7.94	93.8	97.7	7.68
	Ours	**90.3**	**11.88**	**62.4**	**73.7**	**7.83**	**94.0**	**97.9**	**7.42**
Gemini 3.6 Flash*	Base	85.1	12.17	57.8	65.2	9.22	89.9	96.8	10.55
	Skill-as-Context	85.8	11.19	60.6	69.4	8.69	89.3	97.2	10.28
	Ours	**88.1**	**10.45**	**63.6**	**72.0**	**8.31**	**94.3**	**98.2**	**7.61**
Open-Source Models									
Qwen 3.5 4B	Base	25.4	26.04	11.0	15.6	13.69	41.7	25.4	19.24
	Skill-as-Context	33.6	24.93	8.4	13.0	14.07	53.3	43.5	19.75
	Ours	**92.5**	**12.40**	**38.8**	**43.6**	**11.68**	**86.9**	**95.9**	**14.02**
Qwen 3.5 35B-A3B	Base	42.5	22.43	22.0	35.5	11.03	70.2	58.4	12.51
	Skill-as-Context	49.3	20.44	26.8	39.7	10.67	61.0	55.3	15.32
	Ours	**90.3**	**12.61**	**63.0**	**72.4**	**9.87**	**91.4**	**97.1**	**10.95**
Llama 3.3 70B	Base	41.8	20.18	31.6	42.7	12.13	53.0	22.7	13.05
	Skill-as-Context	50.7	16.51	38.8	47.9	11.85	61.9	49.9	18.19
	Ours	**85.1**	**14.74**	**61.0**	**67.6**	**9.47**	**92.3**	**96.9**	**9.68**
GPT-OSS-120B	Base	31.3	24.74	45.6	49.2	11.09	59.8	41.9	14.44
	Skill-as-Context	51.5	20.67	50.2	54.7	10.38	78.9	95.6	10.98
	Ours	**76.1**	**17.24**	**68.2**	**75.6**	**9.00**	**90.2**	**97.4**	**10.12**
Trained Models									
SkillRL-SFT 7B	Base	57.5	15.58	31.2	47.7	9.93	–	–	–
	Skill-as-Context	62.7	16.89	50.2	60.1	**9.41**	–	–	–
	Ours	**80.6**	**14.19**	**58.0**	**64.6**	9.68	–	–	–
SkillRL-RL 7B	Base	61.9	15.80	57.8	59.2	11.75	–	–	–
	Skill-as-Context	68.7	14.62	62.4	65.6	9.56	–	–	–
	Ours	**92.5**	**11.72**	**67.6**	**75.1**	**8.04**	–	–	–

4.2 Main Results

Table 2 summarizes primary evaluation results across benchmarks, model scales, and skill utilization paradigms. Overall, Skill-as-Controller demonstrates clear improvements in both task completion and execution quality. We highlight three key observations:

Performance Across Benchmarks and Backbones. Skill-as-Controller brings consistent performance gains across varying model families and task environments. On open-source models, it substantially increases success rates, for example, lifting Qwen-3.5-4B from 25.4% to 92.5% on ALFWorld, and improving Llama-3.3-70B from 41.8% to 85.1%. Similar improvements are observed on WebShop and ScienceWorld. Moreover, our method maintains robust gains on task-fine-tuned checkpoints (e.g., reaching 58.0% on WebShop for SkillRL-SFT and 92.5% on ALFWorld for SkillRL-RL), showing strong adaptability across diverse base model priors.

Efficiency. Beyond task success, Skill-as-Controller executes actions more directly, substantially reducing interaction overhead. On ALFWorld, the average steps for Qwen-3.5-4B decrease from

Table 3: Step-wise decomposition of the skill utilization gap on ALFWorld using Qwen-3.5-35B-A3B. ✓ and ✗ denote whether each component is handled explicitly or left to implicit contextual reasoning. **+ Step-Inst.**: prompting with step-wise rule instructions; **+ Act-Score Inst.**: prompting with candidate action scoring instructions. Δ indicates the marginal gain over the preceding row.

Method	Rule Extraction	Action Evaluation	Policy Control	Success (%) ↑	Δ (pp)
Base	–	–	–	42.5	–
Skill-as-Context	✗	✗	✗	49.3	+6.8
+ Step-Inst.	✓	✗	✗	58.2	+8.9
+ Act-Score Inst.	✓	✓	✗	72.4	+14.2
Skill-as-Controller	✓	✓	✓	**90.3**	**+17.9**

26.04 to 12.40, and for SkillRL-RL from 15.80 to 11.72, with corresponding step reductions on WebShop. This trajectory compression demonstrates that explicit policy control based on skill-action consistency scores effectively prevents execution deadlocks and eliminates trial-and-error detours.

Decoupling Competence from Model Scale. Explicit guidance largely bridges the performance gap across model parameter scales. With Skill-as-Controller, the compact 4B-parameter model (Qwen-3.5-4B) reaches a 92.5% success rate on ALFWorld and an environment score of 95.9 on ScienceWorld, rivaling or outperforming larger unguided baselines, including GPT-OSS-120B (31.3%) and unguided closed-source models. This finding suggests that failure modes in procedural tasks often arise from ungrounded rule compliance rather than limitations in core reasoning capacity, which analytical control can effectively resolve.

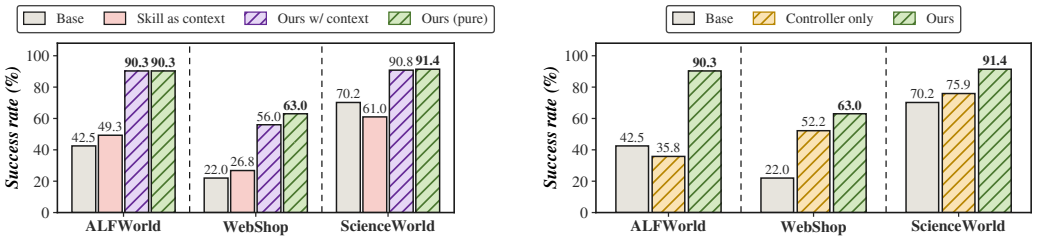
4.3 Decomposing the Skill Utilization Gap

To diagnose where Skill-as-Context fails and why Skill-as-Controller succeeds, we decompose their utilization gap through two intermediate variants on Alfworld (Table 3): (1) **+ Step-Inst.** provides only the skill rules relevant to the current step. (2) **+ Act-Score Inst.** further instructs the model to score candidate actions according to these rules. Selecting relevant rules improves performance, while explicitly scoring actions leads to a larger gain, suggesting that determining how the rules apply to candidate actions is the greater challenge. Skill-as-Controller addresses both challenges and directly adjusts action probabilities using the resulting scores, achieving the highest task success rate among the compared variants.

4.4 The Roles of Skill Context and the Base Policy

Does Adding Skills to the Context Always Help? We investigate whether adding skills to the prompt context remains necessary or beneficial when our external policy control is applied. Across all three benchmarks, we compare our method with and without skill text in the prompt, alongside Base and Skill-as-Context (Figure 3a). Adding skills to prompt context leaves success unchanged on ALFWorld and reduces it on WebShop and ScienceWorld, suggesting that the static skill text may confuse the agent and distract it from tracking dynamic observations. Our method avoids this potential interference by using skills to adjust action probabilities based on the current state without adding skills to the prompt context.

Indispensability of the Base Policy Prior. We examine whether the controller can perform well without the base policy. To test this, we evaluate a *Controller-Only* variant that removes π_0 and selects actions using only skill-action consistency scores, $\pi_{\text{ctrl}}(a \mid x_t) = \text{softmax}_{a \in \mathcal{A}_t}(F_{\mathcal{S}}(x_t, a))$ (Figure 3b). Removing the base policy substantially reduces task success compared with the full method across all three benchmarks, demonstrating that the base policy prior is indispensable in distinguishing candidate actions because discrete rule scores may yield the same skill-action consistency score for different actions.



(a) In-Context Skill Conditioning

(b) Necessity of Base Policy Priors

Figure 3: Ablation of contextual skills and foundation priors across three benchmarks. **(a)** Adding skill text to prompt context in our method leaves success unchanged on ALFWorld and reduces it on WebShop and ScienceWorld. **(b)** Removing the base policy reduces task success.

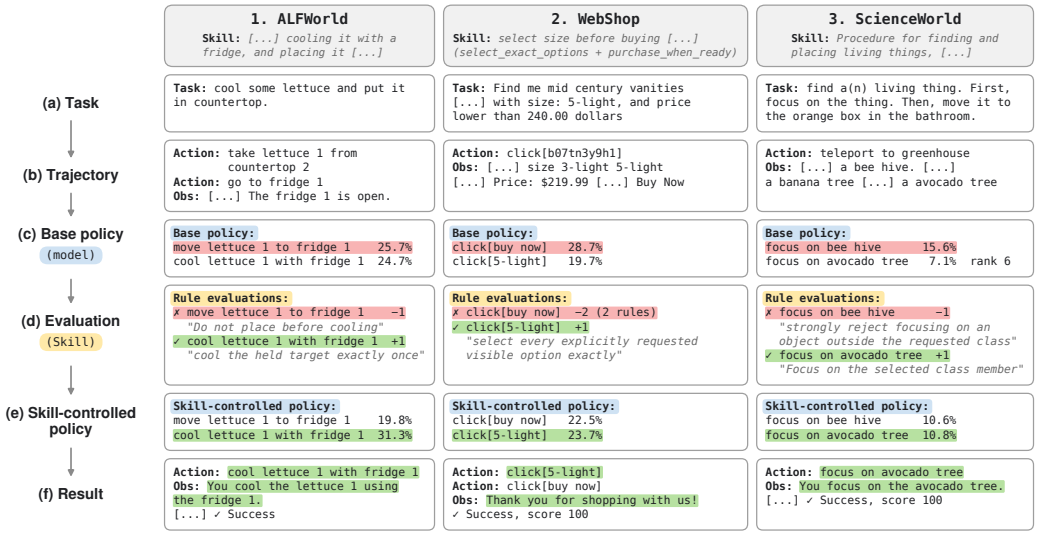


Figure 4: **Qualitative step-by-step case studies across benchmarks.** For each environment, we show: (a) task goal, (b) context observation, (c) base policy probabilities where the model favors incorrect or premature actions, (d) rule-level consistency evaluations, (e) modulated policy distributions, and (f) final execution results. In all cases, explicit evaluation penalizes invalid candidates (e.g., placing before cooling, checking out without options, or focusing on non-target objects) and promotes compliant actions, leading to successful task completion.

4.5 Quantitative and Qualitative Error Analysis

We analyze three trajectory-level error types on ALFWorld: State Looping, Prerequisite Violation, and Goal Drift (Table 4(a)). Skill-as-Controller substantially reduces the dominant errors under Skill-as-Context, indicating more reliable action execution.

Figure 4 provides cases across the three benchmarks, showing how skill-based action evaluation suppresses violating choices and shifts probability mass toward compliant alternatives.

4.6 Impact of Outcome-Guided Evaluator Selection

We compare outcome-guided evaluator selection with two alternatives on ALFWorld: merging all candidate evaluators and randomly selecting one candidate evaluator set. As shown in Table 4b, outcome-guided selection performs best, indicating that evaluator quality varies across candidates and that training-task outcomes provide a useful signal for selecting a more effective evaluator set.

Table 4: Behavioral error analysis and evaluator selection on ALFWorld. **(a)** Trajectory-level error incidence on the official ALFWorld `valid_seen` split (140 tasks) (**Loop**: state cycling; **Prereq**: prerequisite violation; **Drift**: goal drift). **(b)** Comparison of evaluator-set selection strategies. All Candidates uses all generated evaluator sets together. Random reports the average over five randomly selected evaluator sets.

(a) ALFWorld error taxonomy.

Method	Loop ↓	Prereq ↓	Drift ↓
Base	88.6%	33.6%	**1.4%**
Skill-as-Context	81.4%	35.0%	2.1%
Ours	**7.1%**	**17.9%**	**1.4%**

(b) Evaluator-set selection.

Selection	Succ. ↑
All Candidates	79.1
Random (5 runs)	86.6
Outcome-Guided (Ours)	**90.3**

5 Related Work

Skill Optimization and Utilization. Agent skills encode reusable procedural knowledge for solving tasks (Chen et al., 2024; Wang et al., 2025; Nottingham et al., 2024). Early work such as Voyager showed that skills can be accumulated from experience and reused across tasks (Wang et al., 2024). Recent work has explored constructing and refining skills, including building skill libraries from agent experience (Xia et al., 2026; Tu et al., 2026b; Nottingham et al., 2024; Wang et al., 2026c) and improving skill content through trajectory feedback and validation (Wang et al., 2026a; Yang et al., 2026b). Skills are also used to guide model training by transferring procedural knowledge into model parameters (Lu et al., 2026; Wang et al., 2026b; Yang et al., 2026a; Tu et al., 2026a; Wu et al., 2026). Overall, existing studies mainly focus on how skills are constructed, improved, or internalized. However, how skills should guide agent decision-making has received much less attention. At inference time, a common approach is to provide the full skill as prompt context and rely on the LLM to determine which rules are relevant and how they should affect the current action. We focus on skill utilization and propose a new paradigm that explicitly maps skill rules to state-dependent skill-action consistency scores.

Test-Time Policy Updating. Policy updating modifies an agent's action distribution using additional action scores or value estimates (Ahn et al., 2023). KL-regularized and relative-entropy policy optimization provide a general foundation for such controlled policy updates, where the policy is encouraged toward higher-scoring or higher-value actions while limiting its deviation from a base policy (Todorov, 2006; Peters et al., 2010; Rawlik et al., 2012; Abdolmaleki et al., 2018; Peng et al., 2019; Nair et al., 2021; Tomar et al., 2022; Hu et al., 2016). Recent work has further explored policy control and adaptation at test time, using behavioral attributes, reward-bearing interaction experience, or learned adaptation policies to guide agent decisions (Mujtaba et al., 2026; Li et al., 2026; Lou et al., 2026). Existing methods mainly derive policy guidance from learned evaluators, rewards, or past experience. Natural-language skills provide a different form of procedural guidance, but their use as an explicit policy control signal remains less explored. Our work derives state-dependent skill-action consistency scores from natural-language skills and uses these scores to explicitly adjust the policy of a frozen LLM agent.

6 Conclusion

We study how skills should be used to guide long-horizon agents. Our analysis reveals a skill utilization gap under Skill-as-Context: agents may violate skill rules even when the relevant skill is provided, while some violations can still be recognized through separate action evaluation. Motivated by this observation, we propose Skill-as-Controller, which converts skill rules into skill-action consistency scores and uses them to explicitly control the action distribution through a KL-regularized policy update. Across ALFWorld, WebShop, and ScienceWorld and diverse LLM backbones, Skill-as-Controller improves task success and generally reduces interaction steps without updating model parameters. Further analysis shows improved skill rule compliance, fewer common execution errors, and directly inspectable effects of skill rules on action selection. These results highlight skill utilization as an important dimension of building more reliable long-horizon agents.

AI USE STATEMENT

In this work, we used generative AI tools for the writing and polishing of the manuscript. These tools were not used to develop the central research idea and experiment. We have reviewed the AI-assisted content and carefully checked its correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

All authors have read and agree to abide by the ICLR Code of Ethics. This work does not involve interventions with human participants or the collection of personally identifiable information. We use publicly available benchmarks under their applicable licenses and terms of use. Potential risks and our mitigations are summarized below:

- **Privacy & Security.** We do not collect or release personal data. Qualitative examples are drawn from benchmark tasks and agent trajectories.
- **Bias & Fairness.** We report results on multiple benchmarks and provide experimental settings to facilitate external auditing. We acknowledge possible biases in the benchmarks and LLM-based evaluators and encourage further evaluation across broader tasks and domains.
- **Legal Compliance.** We comply with the applicable licenses and terms of use of third-party code, models, and benchmarks, and cite their sources.
- **Research Integrity.** We describe the evaluator construction and selection procedures and evaluation protocols in Sections 3 and 4.1. Algorithms and prompt templates are provided in Appendices A and D.

REPRODUCIBILITY STATEMENT

We describe the method in Section 3 and the benchmarks, models, and evaluation protocols in Section 4.1. Appendix A provides the algorithms for inference and evaluator construction and selection. Appendix B derives the policy update, and Appendix C examines the effect of the control strength β. Appendix D provides prompt templates for action generation and base policy action scoring.

REFERENCES

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=S1ANxQW0b.

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil J. Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Mengyuan Yan, and Andy Zeng. Do as i can, not as i say: Grounding language in robotic affordances. In Karen Liu, Dana Kulic, and Jeff Ichnowski (eds.), *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 287–318. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/ichter23a.html.

Minghao Chen, Yihang Li, Yanting Yang, Shiyu Yu, Binbin Lin, and Xiaofei He. AutoManual: Constructing instruction manuals by LLM agents via interactive environmental learning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 589–631. Curran Associates, Inc., 2024. doi: 10.52202/079017-0019. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/0142921fad7ef9192bd87229cdafa9d4-Paper-Conference.pdf.

Yao Fu, Dong-Ki Kim, Jaekyeom Kim, Sungryull Sohn, Lajanugen Logeswaran, Kyunghoon Bae, and Honglak Lee. AutoGuide: Automated generation and selection of context-aware guidelines for large language model agents. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 119919–119948. Curran Associates, Inc., 2024. doi: 10.52202/079017-3811. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/d8efbb5dd415974eb095c3f06bff1f48-Paper-Conference.pdf.

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. Harnessing deep neural networks with logic rules. In Katrin Erk and Noah A. Smith (eds.), *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2410–2420, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1228. URL https://aclanthology.org/P16-1228/.

Yibo Li, Zijie Lin, Ailin Deng, Xuan Zhang, Yufei He, Shuo Ji, Tri Cao, and Bryan Hooi. Just-in-time reinforcement learning: Continual learning in LLM agents without gradient updates. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=pLvye0zHUC.

Zhanzhi Lou, Hui Chen, Yibo Li, Qian Wang, and Bryan Hooi. Learning to learn-at-test-time: Language agents with learnable adaptation policies, 2026. URL https://arxiv.org/abs/2604.00830.

Zhengxi Lu, Zhiyuan Yao, Jinyang Wu, Chengcheng Han, Qi Gu, Xunliang Cai, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. SKILL0: In-context agentic reinforcement learning for skill internalization, 2026. URL https://arxiv.org/abs/2604.02268.

Dena Mujtaba, Brian Hu, Anthony Hoogs, and Arslan Basharat. Aligning machiavellian agents: Behavior steering via test-time policy shaping. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(44):37738–37746, March 2026. ISSN 2159-5399. doi: 10.1609/aaai.v40i44.41109. URL http://dx.doi.org/10.1609/aaai.v40i44.41109.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. AWAC: Accelerating online reinforcement learning with offline datasets, 2021. URL https://arxiv.org/abs/2006.09359.

Kolby Nottingham, Bodhisattwa Prasad Majumder, Bhavana Dalvi Mishra, Sameer Singh, Peter Clark, and Roy Fox. Skill set optimization: Reinforcing language model behavior via transferable skills. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 38409–38425. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/nottingham24a.html.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning, 2019. URL https://arxiv.org/abs/1910.00177.

Jan Peters, Katharina Mulling, and Yasemin Altun. Relative entropy policy search. *Proceedings of the AAAI Conference on Artificial Intelligence*, 24(1):1607–1612, July 2010. ISSN 2159-5399. doi: 10.1609/aaai.v24i1.7727. URL http://dx.doi.org/10.1609/aaai.v24i1.7727.

Konrad Rawlik, Marc Toussaint, and Sethu Vijayakumar. On stochastic optimal control and reinforcement learning by approximate inference. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2012. doi: 10.15607/RSS.2012.VIII.045.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Cote, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=0IOX0YcCdTn.

Emanuel Todorov. Linearly-solvable markov decision problems. In B. Schölkopf, J. Platt, and T. Hoffman (eds.), *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006. URL https://proceedings.neurips.cc/paper_files/paper/2006/file/d806ca13ca3449af72a1ea5aedbed26a-Paper.pdf.

Manan Tomar, Lior Shani, Yonathan Efroni, and Mohammad Ghavamzadeh. Mirror descent policy optimization. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=aBO5SvgSt1.

Songjun Tu, Chengdong Xu, Qichao Zhang, Yiwen Ma, Yaocheng Zhang, Linjing Li, Dong Li, Xiangyuan Lan, and Dongbin Zhao. UCOB: Learning to utilize and evolve agentic skills via credit-aware on-policy bidirectional self-distillation, 2026a. URL https://arxiv.org/abs/2606.29502.

Songjun Tu, Chengdong Xu, Qichao Zhang, Yaocheng Zhang, Xiangyuan Lan, Linjing Li, Dong Li, and Dongbin Zhao. Dynamic dual-granularity skill bank for agentic RL, 2026b. URL https://arxiv.org/abs/2603.28716.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=ehfRiF0R3a.

Hanyu Wang, Yifan Lan, Bochuan Cao, Lu Lin, and Jinghui Chen. SkillGrad: Optimizing agent skills like gradient descent, 2026a. URL https://arxiv.org/abs/2605.27760.

Hao Wang, Guozhi Wang, Han Xiao, Yufeng Zhou, Yue Pan, Jichao Wang, Ke Xu, Yafei Wen, Xiaohu Ruan, Xiaoxin Chen, and Honggang Qi. Skill-SD: Skill-conditioned self-distillation for multi-turn LLM agents, 2026b. URL https://arxiv.org/abs/2604.10674.

Jiongxiao Wang, Qiaojing Yan, Yawei Wang, Yijun Tian, Soumya Smruti Mishra, Zhichao Xu, Megha Gandhi, Panpan Xu, and Lin Lee Cheong. Reinforcement learning for self-improving agent with skill library. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1529–1550, San Diego, California, United States, July 2026c. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.69. URL https://aclanthology.org/2026.acl-long.69/.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. ScienceWorld: Is your agent smarter than a 5th grader? In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11279–11298, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.775. URL https://aclanthology.org/2022.emnlp-main.775/.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 63897–63911. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/wang25bx.html.

Jinyang Wu, Shuo Yang, Zhengxi Lu, Fan Zhang, Yuhao Shen, Lang Feng, Haoran Luo, Zheng Lian, Shuai Zhang, Zhengqi Wen, and Jianhua Tao. SEED: Self-evolving on-policy distillation for agentic reinforcement learning, 2026. URL https://arxiv.org/abs/2607.14777.

Peng Xia, Jianwen Chen, Hanyang Wang, Jiaqi Liu, Kaide Zeng, Yu Wang, Siwei Han, Yiyang Zhou, Xujiang Zhao, Haifeng Chen, Zeyu Zheng, Cihang Xie, and Huaxiu Yao. SkillRL: Evolving agents via recursive skill-augmented reinforcement learning, 2026. URL https://arxiv.org/abs/2602.08234.

Shuo Yang, Jinyang Wu, Zhengxi Lu, Yuhao Shen, Fan Zhang, Lang Feng, Shuai Zhang, Haoran Luo, Zheng Lian, Zhengqi Wen, and Jianhua Tao. OPID: On-policy skill distillation for agentic reinforcement learning, 2026a. URL https://arxiv.org/abs/2606.26790.

Yifan Yang, Ziyang Gong, Weiquan Huang, Qihao Yang, Ziwei Zhou, Zisu Huang, Yan Li, Xuemei Gao, Qi Dai, Bei Liu, Kai Qiu, Yuqing Yang, Dongdong Chen, Xue Yang, and Chong Luo. SkillOpt: Executive strategy for self-evolving agent skills, 2026b. URL https://arxiv.org/abs/2605.23904.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 20744–20757. Curran Associates, Inc., 2022. doi: 10.52202/068431-1508. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/82ad13ec01f9fe44c01cb91814fd7b8c-Paper-Conference.pdf.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19632–19642, March 2024. ISSN 2159-5399. doi: 10.1609/aaai.v38i17.29936. URL http://dx.doi.org/10.1609/aaai.v38i17.29936.

A Algorithmic Description

Algorithm 1 describes Skill-as-Controller inference with a fixed evaluator set and LLM. Algorithm 2 describes evaluator construction and selection using training outcomes. Candidate actions follow the benchmark-specific interfaces in Section 4.1.

Algorithm 1 Skill-as-Controller Inference

Require: Frozen LLM p_0; skill $\mathcal{S}$; fixed evaluator set $\mathcal{C}_\mathcal{S} = (g_1, \ldots, g_K)$; environment $\mathcal{E}$; task q; step limit T; control strength $\beta \geq 0$.
Ensure: Episode trajectory τ.

Initialize $\mathcal{E}$ for task q; receive public observation o_0; $\tau \leftarrow (o_0)$
for $t = 0, \ldots, T-1$ **do**
  Build x_t from q, o_t, and the public interaction history in τ
  Obtain candidate actions $\mathcal{A}_t$
  Compute the base action distribution $\pi_0(\cdot \mid x_t)$ using the protocol in Section 4.1, without adding $\mathcal{S}$ to x_t
  for all $a \in \mathcal{A}_t$ **do**
    Apply each evaluator $g_k \in \mathcal{C}_\mathcal{S}$ to obtain $g_k(x_t, a)$
    $F_\mathcal{S}(x_t, a) \leftarrow \frac{1}{K} \sum_{k=1}^{K} g_k(x_t, a)$
  end for
  Compute $\pi_\mathcal{S}(\cdot \mid x_t)$ using Eq. 6
  $a_t \leftarrow \arg\max_{a \in \mathcal{A}_t} \pi_\mathcal{S}(a \mid x_t)$
  Execute a_t in $\mathcal{E}$; receive the next public observation o_{t+1}
  Append (a_t, o_{t+1}) to τ
  if the episode has terminated **then**
    break
  end if
end for
return τ

Algorithm 2 Evaluator Construction and Selection

Require: Skill $\mathcal{S} = (r_1, \ldots, r_K)$ with $K \geq 1$; frozen agent LLM p_0; candidate count $M \geq 1$; training tasks $\mathcal{D}_{\text{train}}$; environment $\mathcal{E}$; step limit T; control strength $\beta \geq 0$.
Ensure: Selected evaluator set $\mathcal{C}^*_\mathcal{S}$.

▷ **Candidate evaluator construction**
Generate M sets of action-evaluation rubrics for the same skill rules $\{r_1, \ldots, r_K\}$ using p_0
For each rubric set, combine each rubric with the scoring procedure implemented by p_0 to form an evaluator set $\mathcal{C}^{(m)}_\mathcal{S} = (g^{(m)}_1, \ldots, g^{(m)}_K)$
Collect the candidate evaluator sets as $\mathcal{B}_\mathcal{S} = \{\mathcal{C}^{(1)}_\mathcal{S}, \ldots, \mathcal{C}^{(M)}_\mathcal{S}\}$
▷ **Outcome-guided selection**
for all $\mathcal{C} \in \mathcal{B}_\mathcal{S}$ **do**
  Run Algorithm 1 on each task in $\mathcal{D}_{\text{train}}$ using evaluator set $\mathcal{C}$
  $J_{\text{train}}(\mathcal{C}) \leftarrow$ mean task success over training episodes
end for
$\mathcal{C}^*_\mathcal{S} \leftarrow \arg\max_{\mathcal{C} \in \mathcal{B}_\mathcal{S}} J_{\text{train}}(\mathcal{C})$
Fix $\mathcal{C}^*_\mathcal{S}$ before held-out evaluation
return $\mathcal{C}^*_\mathcal{S}$

B Derivation of Explicit Policy Control

The update follows the standard KL-regularized formulation (Todorov, 2006; Peters et al., 2010; Rawlik et al., 2012; Abdolmaleki et al., 2018; Peng et al., 2019; Nair et al., 2021; Tomar et al., 2022; Li et al., 2026). At a fixed decision step, abbreviate $\pi_0(a \mid x_t)$ as $\pi_0(a)$ and $F_\mathcal{S}(x_t, a)$ as $F(a)$. We treat the skill-action consistency scores $F(a)$ as fixed, finite values at this step. The derivation does not require them to be estimates of environment rewards or returns. Let $\mathcal{A}^+_t = \{a \in \mathcal{A}_t : \pi_0(a) > 0\}$ be the support of the base policy. For the softmax base policy used with open-source models (Section 4.1), $\mathcal{A}^+_t = \mathcal{A}_t$ when all action logits are finite. If the base policy assigns zero probability

to an action, assigning it positive probability would give an infinite KL divergence. We therefore optimize over $\mathcal{A}_t^+$ and assign zero probability to actions outside this set. All sums below are over the finite, nonempty set $\mathcal{A}_t^+$.

For $\beta > 0$, dividing the objective in Equation 5 by β leaves its maximizer unchanged. The equivalent optimization problem is

$$\max_{\pi \in \Delta(\mathcal{A}_t^+)} J(\pi), \qquad J(\pi) = \sum_a \pi(a) F(a) - \frac{1}{\beta} \sum_a \pi(a) \log \frac{\pi(a)}{\pi_0(a)}, \tag{8}$$

where $\Delta(\mathcal{A}_t^+)$ is the probability simplex and terms with $\pi(a) = 0$ are defined by continuity. Introducing a Lagrange multiplier λ for $\sum_a \pi(a) = 1$ gives

$$\mathcal{L}(\pi, \lambda) = \sum_a \pi(a) F(a) - \frac{1}{\beta} \sum_a \pi(a) \log \frac{\pi(a)}{\pi_0(a)} + \lambda \left(\sum_a \pi(a) - 1 \right). \tag{9}$$

At an interior stationary point, the partial derivative with respect to each action probability is zero:

$$\frac{\partial \mathcal{L}}{\partial \pi(a)} = F(a) - \frac{1}{\beta} \left(\log \frac{\pi(a)}{\pi_0(a)} + 1 \right) + \lambda = 0. \tag{10}$$

Rearranging yields

$$\log \frac{\pi(a)}{\pi_0(a)} = \beta F(a) + \beta\lambda - 1, \qquad \pi(a) = \pi_0(a) \exp(\beta F(a)) \exp(\beta\lambda - 1). \tag{11}$$

The factor $\exp(\beta\lambda - 1)$ is independent of a. Enforcing $\sum_a \pi(a) = 1$ therefore gives

$$\pi^*(a) = \frac{\pi_0(a) \exp(\beta F(a))}{\sum_{a'} \pi_0(a') \exp(\beta F(a'))}, \tag{12}$$

which recovers Equation 6. The solution is strictly positive on $\mathcal{A}_t^+$ and normalized, so it satisfies all constraints. Moreover, J is strictly concave on the probability simplex: its linear score term does not affect curvature, and the negative KL term has Hessian $-\operatorname{diag}(1/(\beta\pi(a)))$ in the interior. Thus, this feasible stationary point is the unique global maximizer.

For $\beta = 0$, the original objective in Equation 5 reduces to $-D_{\mathrm{KL}}(\pi \| \pi_0)$, whose unique maximizer is π_0. The closed-form update gives the same result at $\beta = 0$.

For any two actions $a, b \in \mathcal{A}_t^+$, the update also implies

$$\log \frac{\pi^*(a)}{\pi^*(b)} - \log \frac{\pi_0(a)}{\pi_0(b)} = \beta \big(F(a) - F(b) \big). \tag{13}$$

Hence, for $\beta > 0$, actions with higher skill-action consistency scores gain relative odds. If all actions receive the same score, then $\pi^* = \pi_0$. The optimality result concerns the stated regularized objective for a fixed skill-action consistency score function; it does not establish optimality for environment returns.

C Ablation for β

In the policy update objective, the hyperparameter $\beta \geq 0$ controls the strength of skill guidance relative to the base policy prior. When $\beta = 0$, the objective minimizes the KL divergence alone, reducing to the unguided base policy π_0. As β increases, skill-action consistency scores exert stronger influence on the action distribution.

Figure 5 illustrates the sensitivity of task success and interaction efficiency to β on ALFWorld. Without skill control ($\beta = 0$), the agent achieves a 42.5% success rate with an average of 22.43 interaction steps. Introducing explicit policy control ($\beta > 0$) brings immediate gains: at $\beta = 1$, the success rate increases to 83.6% while the average number of interaction steps decreases to 13.89. Performance reaches 90.3% at $\beta = 2$ with 12.61 mean steps, and remains stable at $\beta = 4$ with 89.6% success and 12.78 mean steps. These results show that SKILL-AS-CONTROLLER is robust to the choice of β across $\beta \in (0, 4]$.

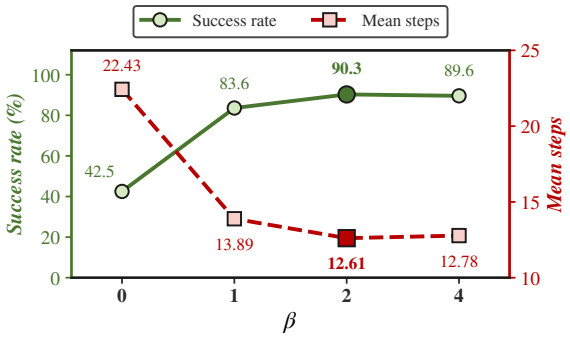


Figure 5: **Sensitivity analysis of the control strength β.** Task success rate (%, green circles, left axis) and mean interaction steps (red squares, right axis) across different values of β on ALFWorld using Qwen-3.5-35B-A3B. Explicit policy control ($\beta > 0$) substantially improves success and reduces execution steps, showing robust performance across $\beta \in (0, 4]$.

D PROMPT TEMPLATES

Listings 1–3 show the prompt for generating an action on each benchmark, and Listings 4 and 5 show the prompts used to compute base policy scores for candidate actions. Text in braces is filled in at run time, and the `Relevant Skill` section is included only when a Skill is used.

Listing 1: Prompt for ALFWorld.

```
You are an expert agent operating in the ALFRED embodied environment.
Your task is to: {goal}

Prior to this step, you have already taken {t} step(s).
Below are the most recent 6 observations and actions:
- Action: {action}
  Result: {observation}
...

Your current observation is:
{observation}

Relevant Skill:
- [{rule_id}] {rule_text}
...

Your admissible actions of the current situation are:
[{action_1}, {action_2}, ..., {action_n}].

Now it is your turn to take one action.
First reason step-by-step inside <think>...</think>. Then choose one action
inside <action>...</action>.

Rules:
- Output exactly one <think>...</think> block and one
  <action>...</action> block, with no other text.
- Keep the reasoning concise, preferably under 40 words.
- The <action> content must exactly match one admissible action from the list.
- Do not repeat the task, history, observation, action list, Skill, or Rubric.
- Stop immediately after </action>.
```

Listing 2: Prompt for WebShop.

```
You are an expert agent operating in the WebShop e-commerce environment.
Your task is to: {goal}

Prior to this step, you have already taken {t} step(s).
Below are the most recent 4 observations and actions:
- Action: {action}
  Result: {observation}
...

Your current observation is:
{observation}

Relevant Skill:
- [{rule_id}] {rule_text}
...

Your admissible actions of the current situation are:
[{action_1}, {action_2}, ..., {action_n}].

Now it is your turn to take one action.
First reason step-by-step inside <think>...</think>. Then choose one action
inside <action>...</action>.

Rules:
- Output exactly one <think>...</think> block and one
  <action>...</action> block, with no other text.
- Keep the reasoning concise, preferably under 40 words.
- The <action> content must exactly match one listed click action. When
  `search[<your query>]` is listed, it may instead be replaced by one concise
  query grounded in the public shopping goal.
- Do not repeat the task, history, observation, action list, Skill, or Rubric.
- Stop immediately after </action>.
```

Listing 3: Prompt for ScienceWorld.

```
You are an expert agent operating in the ScienceWorld science laboratory
environment.
Your task is to: {goal}

Prior to this step, you have already taken {t} step(s).
Below are the most recent 8 observations and actions:
- Action: {action}
  Result: {observation}
...

Your current observation is:
{observation}

Public task-state summary (derived only from task text, observations, and
action history):
- Current stage: {stage}
- Exact task targets: {targets}
- Explicit focus order: {focus_order}
- Completed focus checkpoints: {completed_focuses}
- Next required focus target: {next_focus}
- ...

Relevant Skill:
- [{rule_id}] {rule_text}
...

Your admissible actions of the current situation are:
[{action_1}, {action_2}, ..., {action_n}].

Now it is your turn to take one action.
First reason step-by-step inside <think>...</think>. Then choose one action
inside <action>...</action>.

Rules:
- Output exactly one <think>...</think> block and one
  <action>...</action> block, with no other text.
- Keep the reasoning concise, preferably under 40 words.
- The <action> content must exactly match one admissible action from the list.
- Do not repeat the task, history, observation, action list, Skill, or Rubric.
- Stop immediately after </action>.

- ScienceWorld's action list is parser-generated; hidden or inaccessible
  objects can still make a listed action fail.
- Do not repeat an action after an unknown/impossible/no-progress result.
- Treat `focus on X` as a high-risk commitment: X must be an exact task target
  that is currently visible, at the stage requested by the task.
```

Listing 4: Scoring prompt for ALFWorld and ScienceWorld. Each candidate action is scored as the assistant response, and the resulting action probabilities are reweighted by the skill-action consistency scores. On ScienceWorld, the public task-state summary of Listing 3 follows the trajectory.

```
[System]
You are a careful decision policy for a text-based agent. Never invent an
action outside the admissible set.

[User]
Choose the next environment action.

Task:
{goal}

Current observation:
{observation}

Recent trajectory:
- Action: {action}
  Result: {observation}
...

Admissible actions:
0. {action_0}
1. {action_1}
...

Your completion after `Next action:` must be exactly one admissible action and
nothing else.

Next action:

[Assistant, scored]
{candidate_action}
```

Listing 5: Scoring prompt for WebShop, used as in Listing 4. When the search box is available, the same prompt also generates one search query.

```
[System]
You are a careful WebShop decision policy. Never invent a click target and
never use hidden product identifiers.

[User]
Choose the next WebShop action.

Shopping goal:
{goal}

Current page:
{observation}

Recent trajectory:
- Action: {action}
  Result: {observation}
...

Available actions:
0. {action_0}
1. {action_1}
...

Rules:
- Output exactly one available action and nothing else.
- Click actions must match the list exactly.
- If `search[<your query>]` is available, replace `<your query>` with one
  concise query grounded only in the public shopping goal. Do not copy the
  literal placeholder.

Next action:

[Assistant, scored]
{candidate_action}
```

E SKILLS

Listings 6–14 give the Skills used in our experiments. Each Skill is a list of rules; when a prompt includes a Skill (Appendix D), its rules appear in this form, one rule per line. ALFWorld uses one Skill per task family, WebShop uses a single Skill, and ScienceWorld uses one Skill for the matter-change tasks and one for the classification tasks.

Listing 6: ALFWorld Skill for `pick_and_place_simple`.

```
- [search_or_open] When the target is not held or visible, open the current closed
  receptacle; otherwise search a not-yet-checked location.
- [take_exact_target] Take an admissible instance of exactly the requested target type
  immediately when visible.
- [recover_distractor] If a distractor is held, return to its source and put it down once
  before resuming the task.
- [navigate_destination] While holding the exact target, navigate to a receptacle whose type
  matches the requested destination; open it if closed.
- [place_exact_target] At the requested destination, place the held target there.
- [avoid_regression_and_loops] Do not manipulate distractors, drop the target at a wrong
  receptacle, close containers, repeat recent actions, or waste actions on
  help/look/inventory.
```

Listing 7: ALFWorld Skill for `look_at_obj_in_light`.

```
- [search_or_open] Search unvisited locations and open a closed current receptacle until the
  target and required lamp are found.
- [take_exact_target] Take exactly the requested target object when its take action is
  available.
- [recover_distractor] If a distractor is held, return it to its source before continuing.
- [navigate_to_lamp] After taking the target, keep holding it and go to the remembered
  location of the requested lamp.
- [toggle_lamp_once] While holding the target at the requested lamp, use that lamp once; do
  not toggle it early or turn it off.
- [avoid_regression_and_loops] Do not take distractors, put down the target, examine
  irrelevant objects, or repeat recent actions.
```

Listing 8: ALFWorld Skill for `pick_clean_then_place_in_recep`.

```
- [search_or_open] Search unchecked locations and open a closed current receptacle until the
  exact target is visible.
- [take_exact_target] Take exactly the requested target immediately when visible.
- [recover_distractor] If a distractor is held, return it to its source before resuming.
- [navigate_required_station] While holding the unclean target, go to the sink basin.
- [apply_required_transformation] At the sink basin, clean the held target exactly once.
- [navigate_destination] Only after cleaning, keep holding the target and go to a matching
  destination receptacle; open it if closed.
- [place_exact_target] Place the cleaned target in/on the requested destination.
- [avoid_regression_and_loops] Do not place before cleaning, transform a distractor, close
  containers, or repeat recent actions.
```

Listing 9: ALFWorld Skill for `pick_heat_then_place_in_recep`.

```
- [search_or_open] Search unchecked locations and open a closed current receptacle until the
  exact target is visible.
- [take_exact_target] Take exactly the requested target immediately when visible.
- [recover_distractor] If a distractor is held, return it to its source before resuming.
- [navigate_required_station] While holding the unheated target, go to the microwave.
- [apply_required_transformation] At the microwave, heat the held target exactly once.
- [navigate_destination] Only after heating, keep holding the target and go to a matching
  destination receptacle; open it if closed.
- [place_exact_target] Place the heated target in/on the requested destination.
- [avoid_regression_and_loops] Do not place before heating, transform a distractor, close
  containers, or repeat recent actions.
```

Listing 10: ALFWorld Skill for `pick_cool_then_place_in_recep`.

```
- [search_or_open] Search unchecked locations and open a closed current receptacle until the
  exact target is visible.
- [take_exact_target] Take exactly the requested target immediately when visible.
- [recover_distractor] If a distractor is held, return it to its source before resuming.
- [navigate_required_station] While holding the uncooled target, go to the fridge.
- [apply_required_transformation] At the fridge, cool the held target exactly once.
- [navigate_destination] Only after cooling, keep holding the target and go to a matching
  destination receptacle; open it if closed.
- [place_exact_target] Place the cooled target in/on the requested destination.
- [avoid_regression_and_loops] Do not place before cooling, transform a distractor, close
  containers, or repeat recent actions.
```

Listing 11: ALFWorld Skill for `pick_two_obj_and_place`.

```
- [search_or_open] Search and open receptacles for an undelivered target; after the first
  delivery, revisit remembered target-source locations before broad search.
- [take_exact_target] Take one undelivered instance of exactly the requested target type.
- [recover_distractor] If a distractor is held, return it to its source before resuming.
- [navigate_destination] While holding an undelivered target, go to a matching destination
  receptacle and open it if needed.
- [place_exact_target] Place the held target at the requested destination.
- [track_two_deliveries] After the first delivery, continue for a distinct second instance
  and never take back an instance already delivered.
- [avoid_regression_and_loops] Do not take a delivered object back, manipulate distractors,
  put a target at the wrong place, or repeat recent actions.
```

Listing 12: WebShop Skill.

```
- [focused_retrieval] Use an executed search query to retrieve plausible candidates while
  preserving the requested product category and retrieval-useful hard constraints; do not
  require mechanical inclusion of every request phrase.
- [feedback_driven_refinement] When a search or candidate is observably unproductive, use
  that feedback to change the next query or candidate instead of repeating an unchanged
  ineffective action.
- [plausible_candidate_selection] Open a candidate whose visible result-page evidence is
  compatible with the requested category and does not already contradict a hard requirement.
- [evidence_based_attribute_handling] Use available public product evidence to evaluate
  requested non-option attributes, reject explicit contradictions, and never treat the
  original query text or absent evidence as proof that an attribute is satisfied.
- [exact_option_alignment] Select each explicitly requested visible option exactly and
  preserve earlier correct selections before purchase.
- [price_compliance] Before purchase, use the applicable visible product price to enforce
  the request's price ceiling, including any price change caused by option selection.
- [ready_purchase] Complete the purchase only after the selected candidate has no visible
  hard contradiction and all requested visible options are aligned.
```

Listing 13: ScienceWorld Skill for the matter-change tasks (`boil`, `melt`, `freeze`, and `change-the-state-of-matter-of`).

```
- [exact_target_only] The target is the exact substance named in the task. Strongly reject
  focusing on air, the agent, furniture, food, paint, or any other distractor when it is not
  the named target.
- [find_target_location] If the named target is not visible, prefer purposeful navigation to
  a likely storage location such as the kitchen, fridge, freezer, cupboard, or workshop
  instead of unrelated interaction.
- [focus_before_change] When the exact named target becomes visible, focus on that target
  before manipulating it, as required by the task.
- [prepare_apparatus] After focusing, place the exact target or its containing vessel on an
  appropriate controllable source, then activate that source. Prefer a stove or oven for
  heating and a freezer for cooling. Reject moving unrelated objects or tools onto the
  target, and reject using a lighter on the stove or container when a controllable source is
  available.
- [correct_temperature_direction] For boiling or melting, heat the target; for freezing,
  cool it. Once the target is on an active correct source, keep it there and prefer wait,
  wait1, inspection, or thermometer checks until the requested phase is publicly observed.
  Reject removing or rearranging the apparatus during this process.
- [avoid_no_progress] Reject repeated failed actions, repeated observations with no new
  purpose, and manipulation of unrelated objects.
```

Listing 14: ScienceWorld Skill for the classification tasks (`find-animal`, `find-living-thing`, `find-non-living-thing`, and `find-plant`).

```
- [class_match] Choose an object that unambiguously belongs to the requested class: animal,
  plant, living thing, or non-living thing. Reject ambiguous food products and unrelated
  objects when a clearer instance is available.
- [find_candidate] If no clear class member is visible, navigate to likely locations such as
  the greenhouse, outside, kitchen, or living room.
- [focus_exact_candidate] Focus on the selected class member before moving it, and strongly
  reject focusing on an object outside the requested class.
- [move_to_named_box] After focusing, pick up or move the selected object to the exact box
  and location named by the task.
- [avoid_no_progress] Reject repeated observations, unrelated object manipulation, and
  moving the destination box instead of the selected object.
```