DOES AN AGENT KNOW HOW FAR IT HAS GONE? TRACING TASK PROGRESS REPRESENTATIONS DURING LONG-HORIZON TOOL USE

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-horizon LLM agents must maintain an evolving task state that reflects what has been resolved and what remains to be done as new evidence arrives. We study this internal state in agents as task progress representation. Prior work largely examines intermediate execution through explicit outputs or observable behaviors, but leaves unclear whether such a representation exists and evolves during execution, a question requiring an external progress reference and controls for confounds such as task identity and interaction depth. By leveraging externally verifiable task constraints and matched comparisons, we identify residual directions that reflect verified task progress and thus provide evidence for an internal task progress representation, specifically at two execution points: *entry* (the first token after the latest observation) and *decision* (the first answer token after reasoning). We trace how this representation evolves through execution: when new tool observations yield larger verified progress gains, the internal progress signal advances further and is largely preserved through reasoning. We further analyze cases that deviate from this pattern and find them associated with behavioral failures such as poor evidence assimilation, motivating two targeted interventions: Evidence Assimilation Steering (EAS) and Evidence Review Steering (ERS). Building on these findings, we propose Probe-Gated Steering (PGS), which uses deviations in the task progress representation together with behavioral labels to train probes that detect likely failures online and selectively apply EAS or ERS. Extensive experiments validate the effectiveness of PGS in improving agent performance.

1 INTRODUCTION

Long-horizon LLM agents must maintain an evolving understanding of the task. In tool-use planning, agents gather evidence over extended interactions, integrate new observations with what is already known, and continually decide whether to search further, revise plans, or commit final answers. Reliable planning therefore requires agents to track task progress throughout execution. We refer to the state supporting this capability as a task progress representation.

Many evaluations of long-horizon agents emphasize endpoint outcomes such as task success or final state correctness (Trivedi et al., 2024; Zhou et al., 2024; Yao et al., 2024). Recent work has begun to examine intermediate execution via explicit progress summaries (Chai et al., 2026), verbalized calibration under tool feedback (Xuan et al., 2026), and behaviorally defined step-level drift (Sui et al., 2026). These studies characterize intermediate execution using explicit reports from the model or behaviorally defined quantities. It remains unclear whether agents maintain an internal task progress representation that continues to reflect what has actually been resolved as new observations are integrated and reasoning unfolds during execution.

Identifying the task progress representation requires both an external reference for actual task progress and a way to disentangle the internal progress signal from correlated factors such as task identity and execution stage. In constrained planning environments, we address both challenges by deriving an external progress reference from mechanically verifiable task constraints and using matched cross-rollout comparisons to control these confounds. We examine residual states at two points spanning the turn from new evidence to action selection: *entry*, the first generated token after

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.

the latest observation, and *decision*, the first answer token after reasoning. At each point, we identify progress-sensitive directions in the residual stream that reflect task progress under matched comparisons, suggesting that the model internally represents how far the task has progressed. We further identify a direction shared across *entry* and *decision*, providing a common coordinate for tracing how the representation evolves between them. Along this coordinate, we uncover a consistent progress flow: when a new tool observation yields a larger verified progress gain, the internal progress signal advances further at the next *entry* and is largely preserved through reasoning to *decision*.

We further examine cases that deviate from this dominant progress flow and find that such deviations reveal difficulties in how agents use available evidence to guide subsequent reasoning and actions. This motivates two targeted activation interventions: *Evidence Assimilation Steering* (EAS) at *entry* and *Evidence Review Steering* (ERS) at *decision*. EAS helps the model better integrate new observations during reasoning, while ERS encourages the model to reconsider available evidence before committing to an action, improving the downstream behavior.

Building on these findings, we propose *Probe-Gated Steering* (PGS), which leverages the task progress representation together with behavioral labels to construct supervision for training two probes at *entry* and *decision*. These probes identify states indicative of potential execution failures online and selectively apply EAS or ERS intervention, improving evidence assimilation and supporting more informed plan revision and action selection. Across models, benchmarks, and protocols, PGS consistently improves planning and constraint satisfaction metrics while increasing the rate of valid plan delivery, indicating gains in both planning quality and execution robustness. Ablation studies also show that both EAS and ERS contribute to these gains and highlight the importance of targeted intervention at appropriate execution points.

In short, our main contributions are summarized as follows:

- We identify the task progress representation at entry and decision points and a shared direction along which the representation follows a progress flow as new observations contribute evidence toward task constraints.
- We link deviations from the dominant progress flow to execution difficulties and show that targeted activation steering can mitigate them: EAS improves evidence assimilation at entry, while ERS improves action selection at decision.
- We introduce PGS, which uses separate probes to identify cases warranting intervention and selectively applies EAS or ERS on-the-fly, yielding consistent gains across models and planning benchmarks.

2 Related Work

2.1 Progress in Agent Execution

Recent work has increasingly moved beyond final task outcomes to characterize progress and intermediate states during agent execution. Progra (Chai et al., 2026) explicitly trains progress awareness through interaction summaries and future-action planning; StructAgent (Wu et al., 2026) maintains a verifier-backed task state for progress tracking; others study the reliability of self-reported task progress throughout execution (Wang et al., 2026), verbalized calibration (Xuan et al., 2026), and tool necessity (Sun et al., 2026). Many of these works characterize intermediate execution through explicit or behaviorally defined signals. Whether the model itself maintains a task progress representation that tracks externally verified progress remains less studied.

2.2 Activation Steering

Activation steering provides an inference-time means of controlling model behavior through interventions on internal activations. The classic Contrastive Activation Addition (CAA) method (Rimsky et al., 2024) derives steering directions from positive and negative residual-stream contrasts and applies them during generation. Recent work applies activation steering to reasoning behaviors (Venhoff et al., 2025), chain-of-thought compression (Azizi et al., 2026), and reasoning biases (Valentino et al., 2025). Heyman & Vandeputte (2026) studies adaptive activation steering by learning token-specific intervention strengths. In agentic settings, TACT (Sui et al., 2026) identifies

activation associated with overthinking and overacting, and steers states toward calibrated behavior in coding agents. In contrast, our work studies the activation representation underlying task progress tracking in long-horizon agents.

3 Analysis Setup

3.1 Agent Execution and Representation Anchors

Our analysis spans two typical reasoning models, Qwen3-14B (Qwen Team, 2025) and Ministral-3-14B-Reasoning (Mistral AI, 2026), and two constrained planning benchmarks, DeepPlanning-Travel (Zhang et al., 2026a) and TravelPlanner (Xie et al., 2024). We retain native benchmark prompting with minimal adaptation, collect multiple natural rollouts per task, and use a simple and consistent protocol with a single action per turn. At each turn, the model receives the accumulated interaction history and current task input or tool observation, reasons over this context, and produces a formal answer containing either the next tool call or the final answer. To explore the model's representations around this process, for a given task, rollout, turn, and layer, let $h^E, h^D \in \mathbb{R}^d$ denote the residual-stream states at two points, *entry* and *decision*, respectively. The exact boundary is determined by each model's generation format, as detailed in Appendix A.

3.2 Externally Verified Task Progress

For each task, we construct a checklist $\mathcal{C} = \{c_1, \ldots, c_M\}$ by decomposing the benchmark's structured task constraints and canonical target fields into mechanically verifiable atomic evidence items. At a specific turn, let $q_j \in \{0, 1\}$ indicate whether item j is supported by sufficient tool evidence in the interaction history. We define *externally verified task progress* as $p = \frac{1}{M}\sum_{j=1}^{M} q_j$, which measures externally available task evidence rather than the model's internal understanding of progress. Larger values indicate that more verified evidence relevant to resolving task constraints has accumulated. Therefore, p serves as an external ordinal reference for the task progress representation.

4 Identifying Task Progress Representation

We ask whether the model's residual stream at *entry* and *decision* contains a representation of how far the task has progressed, using the externally verified progress from Sect. 3 as an ordinal reference. We first identify a progress-sensitive direction at each point separately, and then integrate a shared direction reflecting task progress at both.

4.1 Progress-Sensitive Directions at Entry and Decision

To isolate task progress from task identity and trajectory position, we compare natural states from different rollouts of the same task at the same turn. For a matched state pair (a, b), we designate a as the higher-progress state when $p_a - p_b \geq \tau_p$, with $\tau_p = 0.10$ in the main analysis. Matching on task and turn controls task-specific requirements and interaction depth, and partly mitigates confounding from total observation volume, while preserving natural variation in execution across rollouts.

Following the standard contrastive activation construction (Rimsky et al., 2024), we form the higher-minus-lower residual difference at a selected layer for each matched pair $n = (a_n, b_n)$:

$$d_n^r = h_{a_n}^r - h_{b_n}^r, \qquad r \in \{E := \textit{entry}, D := \textit{decision}\}. \tag{1}$$

For *entry* and *decision* separately, we compute the corresponding progress-sensitive direction as:

$$v^r = \mathrm{norm}\left(\sum_{n \in \mathcal{P}_{\mathrm{train}}} \frac{d_n^r}{\|d_n^r\|_2}\right), \qquad r \in \{E, D\}, \tag{2}$$

where $\mathrm{norm}(\cdot)$ denotes L2 normalization. Normalizing each pair difference before aggregation can prevent larger activation differences from receiving greater weight.

Then, we evaluate whether the resulting direction preserves the verified progress ordering on held-out pairs:

$$\mathrm{PairAcc}(v^r) = \Pr_{n \in \mathcal{P}_{\mathrm{eval}}}\left[(d_n^r)^\top v^r > 0\right]. \tag{3}$$

Table 1: PairAcc for progress-sensitive directions constructed separately at *entry* and *decision* and for a shared direction fitted from both *entry* and *decision* contrasts. Same action: *decision* pairs matched by non-terminal action. Shuffle: training pair progress labels randomized before fitting.

Model	Benchmark	Separate direction		Shared direction v^P			
		Entry v^E	*Decision* v^D	*Entry*	*Decision*	Same action	Shuffle
Qwen3-14B	DeepPlanning-Travel	0.752	0.733	0.766	0.751	0.687	0.497
	TravelPlanner	0.862	0.823	0.852	0.835	0.807	0.502
Ministral-3-14B -Reasoning	DeepPlanning-Travel	0.790	0.810	0.747	0.850	0.700	0.477
	TravelPlanner	0.827	0.773	0.757	0.803	0.783	0.473

with chance performance at 0.5. Directions and layer choices are derived on training pairs and frozen before evaluation, as detailed in Appendices B and D.3.

As shown in Table 1, the directions fitted independently at *entry* and *decision* consistently capture the relative ordering of verified task progress across models and planning benchmarks. In the primary Qwen3-14B on DeepPlanning-Travel setting, v_E and v_D achieve PairAcc of 0.752 and 0.733, respectively, with similarly strong results in the other settings. Because each comparison holds task identity and turn fixed, the observed signal cannot be explained simply by the task itself or by interaction depth. Fig. 1 further shows that this signal is stable across the examined middle-to-late layers, rather than being confined to a single layer. Together, these results indicate that the model internally contains a task progress representation at both *entry* and *decision*, suggesting that the model maintains an internal notion of task progress during long-horizon planning.

4.2 A Shared Direction across Entry and Decision

The results above indicate that task progress is represented in the agent's residual stream at both *entry* and *decision* during long-horizon tool use. To trace how this representation changes between the two points, we next ask whether a single direction can reflect the verified progress ordering at both points, revealing a common progress-sensitive component. We construct this shared progress direction by jointly pooling the *entry* and *decision* contrasts:

$$v^P = \mathrm{norm}\left(\sum_{n\in\mathcal{P}_{\mathrm{train}}}\left(\frac{d_n^E}{\|d_n^E\|_2}+\frac{d_n^D}{\|d_n^D\|_2}\right)\right). \tag{4}$$

The resulting $v^P$is evaluated unchanged at both points using the same PairAcc. Table 1 illustrates that the shared direction v^P achieves PairAcc broadly comparable to that of the independently fitted directions at both *entry* and *decision*, and even higher in several settings. For Qwen3-14B on DeepPlanning-Travel, v^P reaches 0.766 at *entry* and 0.751 at *decision*. Under the same action control at *decision*, PairAcc remains above chance, suggesting that v^P is not driven solely by action identity. Moreover, randomly reversing the training pair orientations before fitting returns PairAcc to chance level, showing that the signal depends on verified progress ordering. Fig. 1 further shows that v^P maintains stable PairAcc across the middle-to-late layers, closely tracking the separate directions. Together, these results identify a progress-sensitive direction shared across *entry* and *decision*, providing a common coordinate for tracing task progress dynamics.

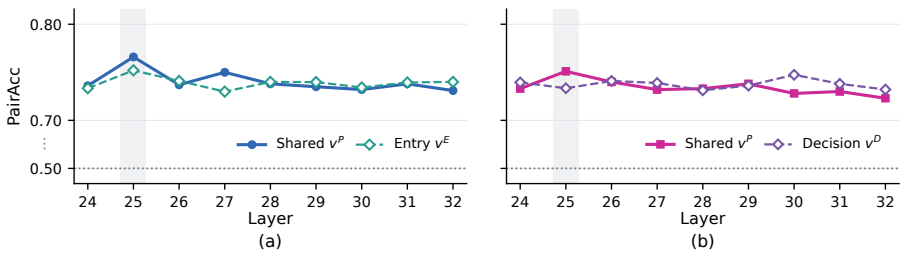


Figure 1: Layer-wise PairAcc for the separate *entry* and *decision* directions and the shared direction, measured with Qwen3-14B on DeepPlanning-Travel.

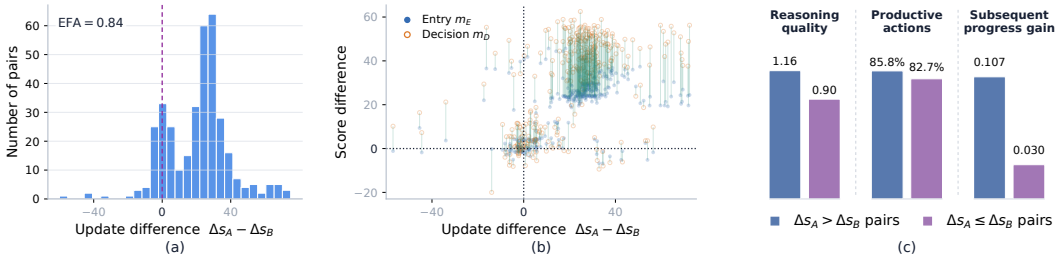


Figure 2: Task progress representation dynamics and subsequent performance for Qwen3-14B on DeepPlanning-Travel. For matched pairs with $\Delta p_A > \Delta p_B$, **(a)** shows the distribution of $\Delta s_A - \Delta s_B$, **(b)** shows the *entry* and *decision* score differences, and **(c)** compares mean reasoning quality, productive action rate, and mean subsequent gain $p_{t+2} - p_{t+1}$ between the two groups.

5 Tracing Task Progress Representation Flow

The directions identified in Sect. 4.1 provide separate, largely static views of the model's internal task progress representation at *entry* and *decision*. With the common progress coordinate established in Sect. 4.2, we can now trace how the representation evolves through execution.

5.1 Task Progress Representation Advances with Verified Progress

We first explore whether interactions that establish more verified progress are accompanied by larger forward updates from *decision* to the next *entry*. To track these updates, we compute scores for *entry* and *decision* states along the shared progress direction as $s^r = (h^r)^\top v^P$, where $r \in \{E, D\}$. For a natural transition from *decision* at turn t to the next *entry* after the tool observation, we define

$$\Delta p = p_{t+1} - p_t, \qquad \Delta s = s^E_{t+1} - s^D_t. \tag{5}$$

Here, $t+1$ indexes the next turn after the tool observation. Δp measures the gain in externally verified task progress supported by the new tool observation, while Δs captures the corresponding update in the model's internal signal of task progress, as read out along the shared coordinate.

We assess the ordinal agreement between Δp and Δs across matched interactions, quantifying how consistently internal updates vary with verified progress gains. Each pair consists of two interactions from different rollouts of the same task at the same turn, with the same tool family and equal p_t. Both interactions start from exactly the same verified progress p_t, so the comparison focuses on what each interaction adds from a common progress baseline rather than on an initial progress gap. Then, we label each pair (A, B) so that $\Delta p_A > \Delta p_B$ and define External Flow Accuracy (EFA) as the fraction of pairs whose internal updates preserve this ordering:

$$\mathrm{EFA} = \Pr(\Delta s_A > \Delta s_B \,|\, \Delta p_A > \Delta p_B)\,. \tag{6}$$

Our analysis yields an EFA of 0.84 for Qwen3-14B on DeepPlanning-Travel, with a task-cluster bootstrap 95% confidence interval of $[0.79, 0.90]$, whose lower bound remains far above the chance level of 0.5, indicating robustness across clusters. In Fig. 2(a), the internal update margins $\Delta s_A - \Delta s_B$ are predominantly positive, with a pronounced peak well above zero, showing a clear tendency that interactions with larger gains exhibit more positive internal updates. Fig. 2(c) further shows that this alignment is associated with better performance at the next turn, including higher mean reasoning quality and a slightly higher productive action rate. We use GPT-5.6 Sol to score reasoning quality and action productivity, with details in Appendix B.3. The mean subsequent gain in p is also larger for the $\Delta s_A > \Delta s_B$ group, reaching 0.107 compared with 0.030 for the $\Delta s_A \le \Delta s_B$ group, a more than threefold difference. This pattern also holds when changing either the benchmark or the model, with EFA reaching 0.73 for Qwen3-14B on TravelPlanner and 0.76 for Ministral-3-14B-Reasoning on DeepPlanning-Travel. The results reveal a systematic progress flow of the task progress representation, with internal updates tracking gains toward task resolution rather than merely reflecting static task states.

5.2 Reasoning Largely Maintains Task Progress Representation

From *entry* to *decision*, no new tool evidence arrives and p remains unchanged, so the same verified progress ordering provides a reference at both points. For $p_A > p_B$, we define

$$m_E = s_A^E - s_B^E, \qquad m_D = s_A^D - s_B^D. \tag{7}$$

Conditioning on $m_E > 0$ ensures a correct progress ordering before reasoning. The maintenance rate $\Pr(m_D > 0 \mid m_E > 0)$ therefore measures how often this ordering is preserved at *decision*, indicating whether the task progress representation remains aligned after reasoning.

For Qwen3-14B on DeepPlanning-Travel, 84.6% of pairs correctly ordered at *entry* remain correctly ordered at *decision*, indicating that reasoning largely preserves progress distinctions already reflected in the representation. Fig. 2(b) illustrates this preservation at turn $t+1$: positive *entry* score differences generally remain positive at *decision*. Many of these pairs also exhibit larger *decision* score differences, suggesting that the retained progress ordering can become more pronounced after reasoning. We also find that some initially incorrect orderings are corrected, with 44.5% of pairs not correctly ordered at *entry* becoming correctly ordered at *decision*.

This preservation also holds across settings, with maintenance rates of 91.7% for Qwen3-14B on TravelPlanner and 76.0% for Ministral-3-14B-Reasoning on DeepPlanning-Travel. Together, these results show that this progress flow is largely sustained during reasoning, as information about the progress already established remains present in the model's task progress representation.

6 From Deviations to Targeted Steering

Sect. 5 characterizes the dominant progress flow with EFA while also revealing occasional deviations that may signal states hindering task execution at *entry* and *decision*. To investigate this possibility, we manually inspect generated outputs for *entry* cases ($\mathcal{D}_E : \Delta p_A > \Delta p_B,\ \Delta s_A \le \Delta s_B$) and *decision* cases ($\mathcal{D}_D : m_E > 0,\ m_D \le 0$). These observations suggest that cases with such deviations exhibit more pronounced shortcomings in evidence assimilation and action selection than those without. We therefore develop targeted activation steering to address these issues.

6.1 Evidence Assimilation Steering at Entry

Inspecting the reasoning generated for *entry* cases in $\mathcal{D}_E$, we observe missed implications of new tool evidence, incomplete updates to what has been established or remains unresolved, and insufficient plan adjustment. This points to evidence assimilation as a natural repair target, motivating us to construct *Evidence Assimilation Steering* (EAS), an activation direction toward representations for better integration of new evidence.

We collect rollouts from the training tasks and assess the evidence-assimilation quality of each turn's reasoning using GPT-5.6 Sol under a fixed rubric, as in Appendix B.4. High-quality reasoning forms positive set $\mathcal{G}$, while reasoning from $\mathcal{D}_E$ with poor evidence assimilation forms $\mathcal{P}$. This contrast is intended to favor states that are better at interpreting new evidence in light of task requirements and at using it to update the task assessment and plan. We average the residual states $h_{i,k}$ of each selected sample i over its generated reasoning tokens R_i and apply the standard contrastive construction:

$$\bar{h}_i = \frac{1}{|R_i|} \sum_{k \in R_i} h_{i,k}, \qquad v^{\mathrm{EAS}} = \mathrm{norm}\left(\frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} \bar{h}_i - \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \bar{h}_i \right). \tag{8}$$

We apply the update $\widetilde{h}_{i,k} = h_{i,k} + \alpha_E v^{\mathrm{EAS}}$ to the first three reasoning tokens starting at *entry*, then continue generation without further steering. Steering the model along this direction aims to improve the integration of new evidence into reasoning and planning.

Fig. 3(a) shows that, across the tested layers, EAS yields high-quality reasoning in 38%–66% of cases with poor baseline evidence assimilation, while degrading 4%–14% of initially high-quality cases. Using the action criterion in Sect. 5.1, we find that 48%–72% of unproductive actions become productive, while 2%–12% of productive actions deteriorate, as shown in Fig. 3(b). The notable recovery and limited harm suggest that brief steering at *entry* can improve evidence assimilation and subsequent action selection while largely preserving already effective behavior.

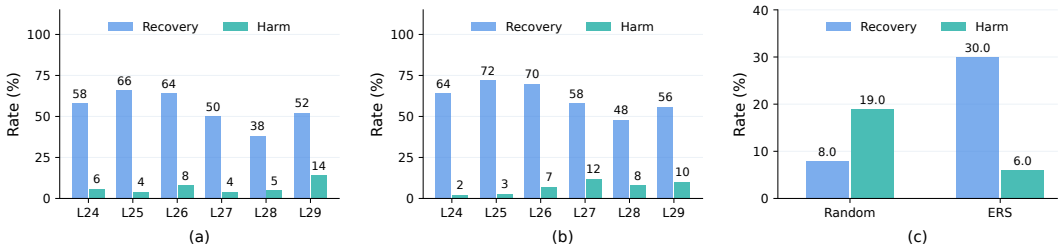


Figure 3: Effects of EAS and ERS on reasoning and action quality for Qwen3-14B on DeepPlanning-Travel. On held-out tasks, Recovery measures improvement on $\mathcal{D}_E/\mathcal{D}_D$ failures for EAS/ERS, while Harm measures degradation on non-deviation cases. (a) reasoning change under EAS, (b) action change under EAS, and (c) ERS recovery and harm against a random direction.

6.2 Evidence Review Steering at Decision

We also find unproductive actions among the generations for the $\mathcal{D}_D$ cases. At this boundary, the reasoning has already been generated, so the model can only rely on it and the available evidence to produce the formal answer, i.e., the tool action. We ask whether local steering can help the model use this information more effectively for action selection while keeping the reasoning text fixed.

We insert a brief reflection instruction at the beginning of the formal answer, immediately after the completed reasoning. The instruction asks the model to reflect on its completed reasoning and the latest observation against the task requirements, and then generate an appropriate next tool call. From $\mathcal{D}_D$ training cases, we collect answers generated with and without the instruction into $\mathcal{A}_{\text{ref}}$ and $\mathcal{A}_{\text{nat}}$, respectively. For each sample i, we average residual states over the full formal answer, including any instruction, to obtain $\bar{h}_i$. The *Evidence Review Steering* (ERS) activation direction is:

$$v^{\text{ERS}} = \text{norm}\left(\frac{1}{|\mathcal{A}_{\text{ref}}|}\sum_{i\in\mathcal{A}_{\text{ref}}}\bar{h}_i - \frac{1}{|\mathcal{A}_{\text{nat}}|}\sum_{i\in\mathcal{A}_{\text{nat}}}\bar{h}_i\right). \tag{9}$$

ERS is designed to encourage the model to reflect before committing to a tool action, creating an opportunity to revise an otherwise unproductive choice. We then apply ERS without the reflection instruction at *decision* to $\mathcal{D}_D$ evaluation samples. Fig. 3(c) shows that ERS recovers 30% of initially unproductive actions and only degrades 6% of productive actions from normal cases. These results support using reflection as a reference for activation steering toward more productive tool use.

7 Online Detection and Selective Steering

Building on the offline findings in Sect. 6, we propose *Probe-Gated Steering* (PGS), a method for improving long-horizon planning performance. PGS uses separate linear probes f_E and f_D to selectively apply EAS at *entry* and ERS at *decision*, respectively.

To train probes that identify failure-prone states informed by progress flow deviation, we construct supervision separately from the matched pairs in Sect. 5 and the behavioral annotations in Sect. 6. For the *entry* probe, $b_i = 1$ denotes poor evidence assimilation in reasoning and $b_i = 0$ good evidence assimilation. For the *decision* probe, $b_i = 1$ denotes an unproductive action and $b_i = 0$ a productive tool call. r_i denotes the fraction of deviation pairs among all matched pairs containing case i. To balance supervision reliability and coverage, we only retain high-consistency cases: a positive example $y_i = 1$ with weight $\omega_i = r_i$ requires $r_i \geq 0.8$ and $b_i = 1$, while a negative example $y_i = 0$ with weight $\omega_i = 1 - r_i$ requires $r_i \leq 0.2$ and $b_i = 0$. Let h_i be the residual state of retained case i at the selected layer and $p_i = f_\theta(h_i)$ the predicted probability. We train each probe independently on its retained set $\mathcal{D}$ by minimizing the regularized weighted binary cross-entropy:

$$\min_\theta -\sum_{i\in\mathcal{D}}\omega_i\Big[y_i\log p_i + (1-y_i)\log(1-p_i)\Big] + \frac{\lambda}{2}\|\theta\|_2^2, \tag{10}$$

These criteria treat cases satisfying neither condition as ambiguous and exclude them, while naturally accommodating the uncommon cases appearing in both pair types. This supervision grounds

Table 2: Pass@2 results (%) on DeepPlanning-Travel and TravelPlanner under Native and ReAct protocols. Pairwise best scores are bold. "M." and "m." denote macro and micro rates, respectively.

Protocol	Model	Method	DeepPlanning-Travel				TravelPlanner					
			DR	CS	PS	CP	DR	CS (m.)	CS (M.)	HC (m.)	HC (M.)	Final
Native	Qwen3-14B	Base	97.5	37.2	8.3	22.8	96.0	57.8	3.0	14.9	8.0	1.0
		+PGS	**99.2**	**42.0**	**12.5**	**27.2**	**98.0**	**59.4**	**5.0**	**19.2**	**11.0**	**2.0**
	Qwen3-32B	Base	99.2	41.8	13.3	27.6	97.0	68.3	10.0	24.0	13.0	3.0
		+PGS	**100.0**	**44.7**	**15.8**	**30.3**	**99.0**	**70.3**	**11.0**	**26.0**	**15.0**	**5.0**
	Ministral-3-14B	Base	82.5	29.6	6.7	18.1	87.0	51.9	1.0	14.4	7.0	0.0
		+PGS	**86.7**	**36.0**	**11.7**	**23.9**	**91.0**	**57.6**	**2.0**	**17.3**	**8.0**	**2.0**
	Apriel-1.6-15B	Base	71.7	25.8	8.3	17.1	76.0	48.4	1.0	15.4	6.0	1.0
		+PGS	**77.5**	**33.9**	**12.5**	**23.2**	**83.0**	**55.0**	**2.0**	**16.8**	**7.0**	**2.0**
ReAct	Qwen3-14B	Base	99.2	40.9	11.7	26.3	97.0	62.1	3.0	20.2	8.0	2.0
		+PGS	**100.0**	**43.1**	**15.0**	**29.1**	**98.0**	**65.3**	**5.0**	**22.6**	**11.0**	**3.0**
	Ministral-3-14B	Base	84.2	32.9	10.8	21.9	86.0	62.3	4.0	22.1	10.0	2.0
		+PGS	**88.3**	**39.8**	**14.2**	**27.0**	**89.0**	**64.8**	**6.0**	**24.5**	**13.0**	**4.0**

case selection and weighting in task progress representations, helping probes identify potential behavioral failures associated with internal deviations and provide an online signal for intervention.

Let f_E and f_D denote the linear sigmoid probes at *entry* and *decision*, with respective thresholds τ_E and τ_D fixed before evaluation. During online execution, PGS first generates and consumes the *entry* token, then evaluates $f_E(h^E)$ on the resulting residual state. If $f_E(h^E) > \tau_E$, PGS discards that token and regenerates from the preceding prefix, applying EAS at the selected layer to the first K tokens of the new continuation. The same procedure applies at *decision*, where $f_D(h^D) > \tau_D$ triggers rollback and regeneration with ERS. At either point, if the threshold is not exceeded, the token is retained and generation continues.

8 Experiments and Results

8.1 Experimental Setup

We evaluate PGS on two long-horizon planning benchmarks, DeepPlanning-Travel (Zhang et al., 2026a) and TravelPlanner (Xie et al., 2024), where agents use tools and construct travel plans satisfying constraints. We employ four common reasoning models, namely Qwen3-14B (Qwen Team, 2025), Qwen3-32B (Qwen Team, 2025), Ministral-3-14B-Reasoning-2512 (Mistral AI, 2026), and Apriel-1.6-15B-Thinker (Radhakrishna et al., 2025). Each model is assessed with and without PGS under one or both of two protocols: the native execution protocol (referred to as *Native*) in Sect. 3 and the standard ReAct (Yao et al., 2023). All models are run with the thinking mode enabled.

DeepPlanning-Travel contains 120 tasks without a predefined split, so we construct probes and steering directions on one half, evaluate on the other half, and aggregate results across both. For TravelPlanner, we use its validation set as the training portion and evaluate on 100 randomly sampled test tasks. We measure Delivery Rate (DR) for both benchmarks, which is the proportion of tasks yielding an evaluable final plan. For DeepPlanning-Travel, we also report Commonsense Score (CS) for plan feasibility, Personalized Score (PS) for user requirement satisfaction, and Composite Score (CP) as their mean. For TravelPlanner, we additionally assess pass rate for hard constraints (HC). Final Pass Rate (Final) measures the proportion of fully correct plans. Given the difficulty of long-horizon tasks, we report pass@2 results using two runs per task, taking all metrics from the higher-scoring trajectory. Following official recommendations, we set temperatures to 0.6 for Qwen3 and Apriel-1.6 and 1.0 for Ministral-3, with top-p $= 0.95$ for Qwen3 and Ministral-3. Within each setting, PGS and the unsteered baseline share the same prompts and tools. Probe thresholds are fixed before evaluation. Further details are provided in Appendix D.

8.2 Main Results

Table 2 shows that PGS is consistently effective in improving agent performance across models, benchmarks, and execution protocols. For example, on DeepPlanning-Travel, PGS improves Qwen3-14B's CP from 22.8 to 27.2 under Native and from 26.3 to 29.1 under ReAct, with similar gains for Ministral-3-14B-Reasoning. On TravelPlanner, PGS raises Final Pass Rates by improving

constraint satisfaction. These results suggest that PGS helps agents better incorporate new observations into reasoning and select more productive actions during long-horizon planning.

More importantly, PGS enhances DR, especially for models with weaker baseline completion ability, indicating improved robustness in long-horizon execution. We observe that PGS helps alleviate unstable execution states over long horizons, such as repeated malformed tool calls, cyclic generations for Apriel-1.6, and occasional missing `[/THINK]` markers for Ministral-3. These failures would otherwise prevent agents from delivering valid final plans.

Several observations are also worth noting. First, Apriel-1.6 features relatively unstable generation alongside a substantially longer reasoning process, which results in lower DR and CS. However, it still achieves a relatively competitive PS, possibly because longer reasoning sometimes enables more careful planning. Second, the full case accuracy remains zero on DeepPlanning-Travel across settings, suggesting that the model's intrinsic capability remains the bottleneck, despite the benefits of steering. One possible reason is that DeepPlanning-Travel is uniformly demanding, with multiple tightly coupled constraints, where any single mistake causes failure. In contrast, TravelPlanner features a mix of difficulty levels, including simpler tasks that are easier to solve perfectly.

8.3 Ablation Study

We first examine the contributions of EAS and ERS. Table 3 illustrates that removing either "*entry* probe + EAS" or "*decision* probe + ERS" reduces performance compared to the full PGS. This indicates that both interventions contribute to the overall improvement. For example, removing "*entry* probe + EAS" decreases CP from 27.2 to 24.4. The larger degradation suggests that, for Qwen3-14B on DeepPlanning-Travel, improving evidence assimilation during reasoning has a stronger impact on downstream planning quality.

Table 3: Ablation study on DeepPlanning-Travel with Qwen3-14B under the Native protocol. All results are reported with pass@2. We also present results on the Chinese version of DeepPlanning-Travel.

Method	DR	CS	PS	CP
Baseline	97.5	37.2	8.3	22.8
PGS	99.2	42.0	12.5	27.2
w/o *entry* probe + EAS	97.5	39.6	9.2	24.4
w/o *decision* probe + ERS	99.2	40.9	10.8	25.9
Random trigger	96.7	38.8	9.2	24.0
Decision probe + direct ERS instruct	90.8	31.0	7.5	19.2
Baseline (Chinese)	95.8	38.2	8.3	23.3
PGS (Chinese)	98.3	42.7	11.7	27.2

We further investigate whether the effectiveness of PGS depends on selecting appropriate intervention points. We replace probe-guided intervention with random triggering, where EAS or ERS is applied at randomly selected reasoning rounds. Random triggering provides only marginal improvements over the baseline and remains far below full PGS, suggesting that PGS benefits from both steering and more appropriate intervention points. We test whether ERS can be replaced by directly providing its source review instruction at the same *decision* intervention points. However, inserting a review instruction after reasoning completion and before answer generation can be a rigid intervention, as it underperforms ERS, achieving only 90.8 DR and 19.2 CP. ERS instead steers the model's states toward evidence review and reconsideration without such an interruption. Finally, we also evaluate PGS on the official Chinese version of DeepPlanning-Travel. PGS achieves comparable improvements, increasing CP from 23.3 to 27.2, showing that the learned steering mechanism is not strongly dependent on a specific language format. See Appendices C and D.5 for case analyses.

9 Conclusion and Discussion

In this paper, we investigate whether long-horizon LLM agents maintain an internal representation of task progress during tool-use planning. We identify the task progress representation that evolves with task execution at *entry* and *decision* points: greater progress gains are associated with forward updates, while reasoning largely preserves this representation. We also find that deviations from this progress flow are associated with ineffective behaviors, which can be mitigated by targeted activation steering. Therefore, we introduce PGS to selectively improve agent execution. Our results reveal that task progress is encoded in the internal states of long-horizon agents, providing a new perspective for understanding agent execution beyond final outcomes. Future work will explore broader agent settings and ways to comprehensively understand long-horizon agent execution.

AI USE STATEMENT

In this work, we used generative AI tools in a limited and supportive manner. Specifically, we used such tools to assist with language polishing and improving the clarity of some manuscript text, while the research ideas, technical content, and scientific writing were developed and verified by the authors. We also used generative AI tools as auxiliary support during literature exploration, such as helping retrieve and organize relevant information. We did not use generative AI tools to generate experimental results, determine research conclusions, design evaluation protocols, or replace human judgment in any scientific decision-making process.

Additionally, we used publicly accessible large language model APIs for model-based evaluation and annotation as described in the paper. These uses were limited to the specified evaluation procedures, and the generated outputs were reviewed and verified by the authors. All AI-assisted content, including text, analyses, and artifacts, was carefully checked by the authors.

ETHICS STATEMENT

The datasets and benchmarks in this work are publicly available and are used in accordance with their licenses and terms of use. For model-based evaluation and annotation, we use publicly accessible large language model APIs in accordance with their service policies. Our study does not involve collecting new human data, personal information, or sensitive user interactions. All experiments are conducted in controlled planning environments. Activation steering is used only as a research intervention to improve agent behavior and is not intended for deployment in real-world user-facing systems. We acknowledge that the studied behaviors and representations may inherit biases, limitations, or unintended properties from the underlying models, benchmarks, and evaluation models. Our analysis primarily characterizes internal representations and their association with externally verified task progress. While we use activation interventions to evaluate the practical utility of identified representations, these interventions are not intended to establish a complete causal account of agent behavior. Therefore, the results should be interpreted within the scope of the studied settings.

REPRODUCIBILITY STATEMENT

To facilitate reproduction, we provide complete descriptions of the experimental setup and analysis methodology in the paper and appendix, including model configurations, benchmark settings, prompts, activation extraction procedures, steering methods, and evaluation protocols. We document the essential procedures and choices required to reproduce the reported analyses and results. All datasets and evaluation environments are publicly available.

REFERENCES

Seyedarmin Azizi, Erfan Baghaei Potraghloo, Souvik Kundu, and Massoud Pedram. Activation steering for chain-of-thought compression. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics, ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 36676–36687. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.FINDINGS-ACL.1828. URL https://doi.org/10.18653/v1/2026.findings-acl.1828.

Huacan Chai, Zijie Cao, Maolin Ran, Yingxuan Yang, Jianghao Lin, Xin Peng, Hairui Wang, Renjie Ding, Ziyu Wan, Muning Wen, Weiwen Liu, Weinan Zhang, Fei Huang, and Ying Wen. Progra: Progress-aware reinforcement learning for multi-turn function calling. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics, ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 6519–6535. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.FINDINGS-ACL.325. URL https://doi.org/10.18653/v1/2026.findings-acl.325.

Haoyang Chen, Yi Liu, Jianzhi Shao, Xiaozhou Xu, Zhe Sun, and Wei Hu. Polished but unresolved: Identifying late-stage pressure states in long-horizon tool-use agents. *arXiv preprint arXiv:2609.00823*, 2026.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: large language models can self-correct with tool-interactive critiquing. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=Sx038qxjek.

Geert Heyman and Frederik Vandeputte. Steer like the LLM: Activation steering that mimics prompting. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=06Nk3dJDMq.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/91edff07232fb1b55a505a9e9f6c0ff3-Abstract-Conference.html.

Mistral AI. Ministral 3. *CoRR*, abs/2601.08584, 2026. doi: 10.48550/ARXIV.2601.08584. URL https://doi.org/10.48550/arXiv.2601.08584.

Qwen Team. Qwen3 technical report. *CoRR*, abs/2505.09388, 2025. doi: 10.48550/ARXIV.2505.09388. URL https://doi.org/10.48550/arXiv.2505.09388.

Shruthan Radhakrishna, Aman Tiwari, Aanjaneya Shukla, Masoud Hashemi, Rishabh Maheshwary, Shiva Krishna Reddy Malay, Jash Mehta, Pulkit Pattnaik, Saloni Mittal, Khalil Slimi, Kelechi Ogueji, Akintunde Oladipo, Soham Parikh, Oluwanifemi Bamgbose, Toby Liang, Ahmed Masry, Khyati Mahajan, Sai Rajeswar Mudumba, Vikas Yadav, Sathwik Tejaswi Madhusudhan, Torsten Scholak, Sagar Davasam, Srinivas Sunkara, and Nicohlas Chapados. Apriel-1.5-15b-thinker. *CoRR*, abs/2510.01141, 2025. doi: 10.48550/ARXIV.2510.01141. URL https://doi.org/10.48550/arXiv.2510.01141.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. Steering llama 2 via contrastive activation addition. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 15504–15522. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.828. URL https://doi.org/10.18653/v1/2024.acl-long.828.

Yijia Shao, Vinay Samuel, Yucheng Jiang, John Yang, and Diyi Yang. Collaborative gym: A framework for enabling and evaluating human-agent collaboration. *CoRR*, abs/2412.15701, 2024. doi: 10.48550/ARXIV.2412.15701. URL https://doi.org/10.48550/arXiv.2412.15701.

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

Yuan Sui, Yulin Chen, Yibo Li, Xue Jiang, Yufei He, Yihong Dong, Xiaoxin He, Tianyu Gao, and Bryan Hooi. TACT: mitigating overthinking and overacting in coding agents via activation steering. *CoRR*, abs/2605.05980, 2026. doi: 10.48550/ARXIV.2605.05980. URL https://doi.org/10.48550/arXiv.2605.05980.

Chung-En Sun, Linbo Liu, Ge Yan, Zimo Wang, and Tsui-Wei Weng. LLM agents already know when to call tools - even without reasoning. *CoRR*, abs/2605.09252, 2026. doi: 10.48550/ARXIV.2605.09252. URL https://doi.org/10.48550/arXiv.2605.09252.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. Appworld: A controllable world of apps and people for benchmarking interactive coding agents. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 16022–16076. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.850. URL https://doi.org/10.18653/v1/2024.acl-long.850.

Marco Valentino, Geonhee Kim, Dhairya Dalal, Zhixue Zhao, and André Freitas. Mitigating content effects on reasoning in language models through fine-grained activation steering. *CoRR*, abs/2505.12189, 2025. doi: 10.48550/ARXIV.2505.12189. URL https://doi.org/10.48550/arXiv.2505.12189.

Constantin Venhoff, Iván Arcuschin, Philip Torr, Arthur Conmy, and Neel Nanda. Understanding reasoning in thinking language models via steering vectors. *CoRR*, abs/2506.18167, 2025. doi: 10.48550/ARXIV.2506.18167. URL https://doi.org/10.48550/arXiv.2506.18167.

Boyang Wang, Yunhan Wang, and Yalun Wu. The unreliable progress bar: Can llm agents reliably report task progress throughout execution? *arXiv preprint arXiv:2609.08589*, 2026.

Yaxuan Wang, Quan Liu, Zhenting Wang, Zichao Li, Wei Wei, Yang Liu, and Yujia Bao. Promptbridge: Cross-model prompt transfer for large language models. *CoRR*, abs/2512.01420, 2025. doi: 10.48550/ARXIV.2512.01420. URL https://doi.org/10.48550/arXiv.2512.01420.

Wenyi Wu, Sibo Zhu, Kun Zhou, Aayush Salvi, Zixuan Song, and Biwei Huang. Structagent: Harness long-horizon digital agents with unified causal structure. *CoRR*, abs/2607.11388, 2026. doi: 10.48550/ARXIV.2607.11388. URL https://doi.org/10.48550/arXiv.2607.11388.

Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and Yu Su. Travelplanner: A benchmark for real-world planning with language agents. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54590–54613. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/xie24j.html.

Weihao Xuan, Qingcheng Zeng, Heli Qi, Yunze Xiao, Junjue Wang, and Naoto Yokoya. The confidence dichotomy: Analyzing and mitigating miscalibration in tool-use agents. 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), ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 11325–11349. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.ACL-LONG.520. URL https://doi.org/10.18653/v1/2026.acl-long.520.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *CoRR*, abs/2406.12045, 2024. doi: 10.48550/ARXIV.2406.12045. URL https://doi.org/10.48550/arXiv.2406.12045.

Yinger Zhang, Shutong Jiang, Renhao Li, Jianhong Tu, Yang Su, Lianghao Deng, Xudong Guo, Chenxu Lv, and Junyang Lin. Deepplanning: Benchmarking long-horizon agentic planning with verifiable constraints. 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), ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 7377–7407. Association for Computational Linguistics, 2026a. doi: 10.18653/V1/2026.ACL-LONG.335. URL https://doi.org/10.18653/v1/2026.acl-long.335.

Yunfei Zhang, Boyu Feng, Changhua Pei, Zexin Wang, Zhihuang Peng, Xinlong Liu, Hengyue Jiang, Difeng Ma, Jiayi Zhang, Yongzhou Yao, Yanan Zhao, Fei Sun, Yintong Huo, Zhaoyang Liu, Jingjing Li, Gaogang Xie, and Dan Pei. Longrca bench: Diagnosing responsible roles and root causes in long-horizon agent failures. *CoRR*, abs/2608.15242, 2026b. doi: 10.48550/ARXIV.2608.15242. URL https://doi.org/10.48550/arXiv.2608.15242.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=oKn9c6ytLx.

A Details for Analysis Setup

A.1 Model-Specific Boundary Adaptation

We use each checkpoint's native chat template and output syntax rather than forcing all models into a unified format. Across models, however, the benchmark prompt and protocol remain consistent as a whole. The model-specific adaptations below concern only details such as message stopping and parsing boundaries. They never alter a tool observation, rewrite tool arguments or plan content, repair a violated task constraint, or change a checker output.

Qwen3 uses explicit tags <think> and </think>. A simple routine tool turn has the following generated structure:

```
<think> reasoning ...  </think>
(optional natural language response may appear here)
<tool_call>{"name":  ..., "arguments":  {...}}</tool_call>
```

and a terminal turn, as required by the system prompt, replaces the tool block with a complete <plan>...</plan> block. Of course, whether there is natural language output after </think> depends on the model's state at the time and the prompt. For Qwen3, the *entry* is the first generated token of each assistant turn. The *decision* is the first generated token immediately following the unique </think> token.

Ministral-3-14B-Reasoning-2512 is run with its officially recommended reasoning chat template. Moreover, according to official recommendations, historical tool calls, observations, and previous reasoning should all be retained in the context. We append the official system prompt provided by Mistral to our custom system prompt as recommended by the model authors. Its native tool-calling syntax is:

```
[THINK] reasoning ... [/THINK]
[TOOL_CALLS]function_name[ARGS]{...}.
```

It is worth noting that this model has a non-negligible chance of omitting the [/THINK] boundary during generation. We did not apply any special handling; we only emphasized this in the system prompt with a single sentence: "Every assistant generation must strictly follow the complete structure below: first complete the reasoning within [THINK]...[/THINK] and explicitly output the closing tag [/THINK], and only then begin the formal answer."

Apriel-1.6-15B-Thinker separates its reasoning from its formal response using [BEGIN FINAL RESPONSE], e.g.,

```
reasoning ...  [BEGIN FINAL RESPONSE]
<plan> complete itinerary ...  </plan>.
```

However, its single-turn reasoning is itself relatively long, typically exceeding 7000 tokens, and it also needs to be retained in the context. Therefore, the accumulated context length of this model is much larger than that of other models.

A.2 Constructing Externally Verified Task Progress

For each task, we decompose the benchmark's structured constraints and canonical target fields into an equally weighted checklist $\mathcal{C} = \{c_1, \ldots, c_M\}$. For DeepPlanning-Travel, for example, the benchmark provides structured constraints and canonical target fields in its `meta_info.hard_constraints`, which specify both the constraint type and the expected canonical answer. For example, a transport constraint provides the outbound and inbound train numbers, a hotel constraint provides the target hotel name, and a restaurant constraint provides the target restaurant name. For example, a task may include metadata of the following form:

```
"query": "I'm planning a trip from Harbin to Dalian. There
are four of us traveling, and we'd like to take the train
round-trip -- please just pick a suitable train for us. ...
I'd like to find a hotel that has been newly renovated after
2025 ... we'll need to book two rooms ... for dining, I'd
like to have one meal arranged near 'Xinghai Bay Boardwalk'.
Ideally, it would be at a restaurant offering private room
service ...",
"meta_info": {
  "hard_constraints": {
    "train_seat_status": {
      "outbound_train_no": "G710",
      "inbound_train_no": "G725"
    },
    "hotel_newest_decoration": {
      "hotel_name":
        "Hanting Hotel Dalian Xian Road Commercial Street"
    },
    "restaurant_specific_tag_nearby": {
      "restaurant_name": "Yuanben Tang Restaurant",
      "required_tag": "Private Room"
    }
  }
  ...
}
```

We therefore construct the atomic checklist mechanically, treating each transport direction, hotel, restaurant, and individual attraction as one item c_j. For each c_j, we check whether the currently visible history contains a successful tool observation whose tool type, query arguments, and returned canonical entity match the item. At *entry* t, $q_j^{(t)} = 1$ only when such a successful tool observation already present in the visible history supports item c_j. Missing, failed, or ambiguous evidence leaves it unresolved. Composite constraints are handled fail-closed: for example, a budget item is resolved only when all required prices are available and the resulting cost satisfies the budget. Once resolved, an item remains resolved. We define externally verified task progress as

$$p^{(t)} = \frac{1}{M} \sum_{j=1}^{M} q_j^{(t)}. \tag{11}$$

This deterministic, API-free score measures how much task-relevant evidence the agent has already obtained, and thus reflects objective task progress, rather than the quality of its final plan.

For instance, a task with five checklist items may start at $p = 0$, rise to 0.4 after two transport items are verified, stay at 0.4 after a failed hotel search, and then reach 1.0 once the remaining hotel, attraction, and restaurant items are verified. Progress therefore changes only when new tool evidence resolves a required item.

B Details for Progress Representation Analysis

B.1 Pair Sampling and Evaluation Splits

The pairs analyzed in Sect. 4 are built from natural, unintervened rollouts. Each pair joins two states from different rollouts of the same task at the same turn, oriented from lower to higher externally

verified progress, with the main analysis requiring $p_a - p_b \geq 0.10$. We do not require identical preceding histories: matching on task and turn fixes task-specific requirements and interaction depth, while allowing the two rollouts to accumulate different evidence through natural execution. This design avoids attributing progress-sensitive directions to task identity or mere turn length.

Simply enumerating all rollout combinations would give tasks or turns with many trajectories or eligible states disproportionate weight. We therefore randomly sample pairs within each task and rotate across turns before filling the remaining quota. Each pair is formed from two different rollouts of the same task at the same turn, and each rollout can participate in only a limited number of pairs, so that no single task or turn becomes an implicit statistical weight.

Direction construction and PairAcc evaluation use disjoint task sets and independently collected rollouts: 80 tasks are used for fitting the activation directions and selecting layers, and 40 separate tasks are reserved for evaluation. PairAcc denominators therefore reflect held-out pair counts. For example, for Qwen3-14B on DeepPlanning-Travel, the shared direction is evaluated on 1,000 *entry* and 1,000 *decision* pairs, with 150 same action pairs. Although the stronger layers typically fall around middle-to-late layers, the effect is not confined to these layers: as shown by the layer-wise results in Fig. 1 of the main text, progress ordering remains stable across a broad range of middle-to-late layers.

Moreover, for the direction construction in Sect. 4.1, we compute activation differences within these sampled pairs. This can partially cancel shared additive components associated with task-specific requirements, interaction depth, and the amount of observation information accumulated in context, while retaining rollout-specific variation. We then normalize each contrast before aggregation, preventing pairs with larger activation differences from receiving greater weight.

For r_i in Sect. 7, we aggregate the outcomes of all matched progress flow comparisons involving state i. Each comparison receives a score of 1 if the pair exhibits a progress flow deviation and 0 otherwise. We define r_i as the average of these scores, i.e., the fraction of matched comparisons involving state i that exhibit deviations. Values close to 1 indicate consistent participation in deviation pairs, whereas values close to 0 indicate consistent participation in normal-flow pairs. Most states participate exclusively in either normal-flow or deviation pairs; only a small fraction appear in both. To avoid introducing noisy supervision from such ambiguous cases, we retain only states with strong deviation consistency ($r_i \geq 0.8$) or strong normal-flow consistency ($r_i \leq 0.2$), providing strict supervision for the intervention probes.

B.2 Controls and Sensitivity

The same-action control restricts evaluation to pairs whose endpoints produce the same exact non-terminal action name, reducing the possibility that *decision* accuracy is explained by action identity alone. We identify such pairs through deterministic action-name matching, and then use GPT-5.6 Sol with manual verification to confirm the consistency of the matched actions.

The shuffle control tests whether the fitted direction depends on the verified progress ordering rather than generic variation between paired activations. For every fitting pair n, we independently draw $\epsilon_n \in \{-1, +1\}$ and randomly replace its higher-minus-lower contrasts by $\epsilon_n d_n^E$ and $\epsilon_n d_n^D$. The same sign is used at *entry* and *decision* so that their within-pair correspondence is preserved, after which a new shared direction is fitted from the shuffled contrasts. We then evaluate that direction on the original, correctly oriented evaluation pairs. Thus the shuffle leaves the task split, selected pairs, activation magnitudes, layers, and evaluation labels unchanged; it destroys only the association between a training contrast and which endpoint has greater verified progress. As shown in Table 1, directions fitted from these randomly reoriented contrasts return to approximately chance PairAcc, indicating that they do not systematically recover the verified progress ordering.

Sensitivity to τ_p. We test whether the progress-sensitive directions in Sect. 4 depend on the matching threshold τ_p. In the main analysis, $\tau_p = 0.10$ already yields clearly above-chance PairAcc, suggesting that the model separates progress states even when the externally verified progress gap is relatively small. Using a stricter threshold $\tau_p = 0.15$ for Qwen3-14B on DeepPlanning-Travel under native protocol, PairAcc remains stable and slightly improves for both separately fitted directions: *entry* increases from 0.752 to 0.760, and *decision* increases from 0.733 to 0.736. This indicates that the identified directions capture an ordinal representation of externally verified task progress. When

pairs with larger verified progress gaps are considered, the directions slightly improve in preserving the progress ordering. This is expected: larger progress differences provide a stronger contrastive signal, allowing the direction to better reflect externally verified task progress.

Pair normalization sensitivity. We use the pair-normalized construction in the main analysis because it gives each matched pair an equal contribution when estimating the progress direction. Without pair-wise normalization, pairs with larger residual displacement magnitudes implicitly receive larger weights. The outer L2 normalization only fixes the direction scale and does not affect the sign-based PairAcc metric. As a sensitivity check, we compare it with the raw construction using the same data. As shown in Table 4, the results remain consistent, with pair normalization yielding slightly higher PairAcc.

Table 4: Pair-normalization sensitivity at L25 for Qwen3-14B on DeepPlanning-Travel.

Construction	Entry	Decision
$\mathrm{norm}(\sum_n d_n^r)$	0.711	0.725
$\mathrm{norm}(\sum_n d_n^r / \|d_n^r\|_2)$	0.752	0.733

Uncertainty estimation. We estimate uncertainty in the EFA reported in Sect. 5.1 using a task-cluster bootstrap over the 351 matched external-flow pairs for Qwen3-14B on DeepPlanning-Travel. In each of 2,000 replicates, we sample tasks with replacement and carry all matched pairs from each sampled task together, thereby preserving within-task dependence. Counting zero margins ($\Delta s_A - \Delta s_B = 0$) as incorrect, we obtain an EFA of 0.8433 and a percentile-bootstrap 95% confidence interval of $[0.7851, 0.8974]$. The interval remains entirely above the chance level of 0.5. This indicates that the progress flow alignment is robust to task-level sampling variability: interactions with larger externally verified progress gains consistently produce larger forward updates in the model's internal progress representation from *decision* to the next *entry*.

B.3 Evaluation Prompts for Reasoning and Action Quality

We use GPT-5.6 Sol to score reasoning quality and action quality for the analyses in Secs. 5.1 and 6. The corresponding prompt templates are shown in Figs. 4 and 5. Reasoning quality uses three labels, "good", "mixed", and "poor". Since reasoning traces are long and their quality is often graded rather than binary, we include "mixed" as an intermediate buffer so that clear "good" and "poor" cases can be separated more reliably. Action quality uses two labels, "productive" and "unproductive", because an action is directly judged by whether it is a reasonable and helpful step toward the task.

For the subsequent progress gain in Fig. 2(c), the agent takes its next action, the environment executes the tool call, and the resulting observation is added to the visible history. We then recompute p from the newly available, mechanically verified evidence, such that $p_{t+2} - p_{t+1}$ measures the additional task progress produced by this action. For the reasoning quality results in Fig. 2(c), we map "good", "mixed", and "poor" to scores of 2, 1, and 0, respectively, and report the mean score within the $\Delta s_A > \Delta s_B$ and $\Delta s_A \leq \Delta s_B$ groups.

For Fig. 3, we use strict criteria to avoid overstating recovery. Reasoning recovery requires a transition from "mixed" or "poor" to "good", with "poor"-to-"mixed" transitions not counted; action recovery requires a transition from "unproductive" to "productive". For each recovery analysis, we randomly sample 50 cases satisfying the original failure condition, while each harm analysis uses 100 initially "good" or "productive" cases sampled from pairs that preserve the normal progress-flow ordering. We additionally manually inspect the sampled cases underlying Fig. 3 to verify the reported recovery and harm outcomes. The criteria are applied consistently across panels (a)–(c).

To further assess annotation reliability, we randomly sample 100 instances from each of the two annotation pools used in the analyses above, corresponding to reasoning-quality and action-quality evaluations, and independently re-annotate them with human judgment and Gemini-3.5 Flash. The three-way agreement among GPT-5.6 Sol, human annotators, and Gemini-3.5 Flash reaches 97% for reasoning quality and 92% for action quality, indicating high consistency between the automatic annotations and independent judgments. Disagreements are primarily concentrated in the "mixed" reasoning quality category.

For the evidence-assimilation annotations used in Sect. 6.1, we adapt the reasoning-quality prompt in Fig. 4 and place additional emphasis on how well the reasoning understands existing information and integrates the latest tool evidence. The core part of system prompt is shown in Fig. 6. The label

System prompt

```
You are a strict local reasoning reviewer for long-horizon travel planning agents.
You only evaluate the quality of the reasoning part when the Agent performs next-turn planning after receiving latest_observation, focusing on its soundness/reasonableness and semantic absorption.

# Evaluation Target
The input contains the task requirements, structured hard constraints, the visible history before latest_observation, latest_observation, Reasoning_(t+1), and optional Action_(t+1) context.

You need to judge, overall, whether Reasoning_(t+1), in light of the task requirements, hard constraints, and visible history, is good reasoning:
1. Correctly understands the context and keeps the plan task-oriented:
  - whether it is basically consistent with the task requirements, hard constraints, and relevant visible history;
  - whether its reasoning plan is on a productive track toward task completion, rather than moving away from the task goal.
2. Reasonably uses the latest observation:
  - whether it takes latest_observation into account in a sensible way;
  - whether it avoids ignoring, misreading, or overreading the observation;
  - whether its interpretation of the current situation is broadly reasonable.
3. Shows basically sound reasoning and next-step thinking:
  - whether the reasoning is coherent and logically consistent, rather than circular, contradictory, or unsupported;
  - whether it forms a basically reasonable next-step idea that helps advance the task;
  - whether the next-step idea is broadly consistent with the available context and constraints.

# Causal and Evidence Boundaries
- Action_(t+1) only helps understand what the reasoning intends to do; you must not infer the reasoning label from whether that action later succeeds, fails, or produces future observations.
- Even if Action_(t+1) appears reasonable, this must not raise the reasoning label; you must examine whether the reasoning itself provides a next-step idea that is basically consistent with that action and supported by evidence.
- Do not use facts outside the input, and do not use the final plan or the final outcome of the trajectory.
- The reasoning is not required to solve the entire task in one step; remaining unfinished items alone do not make it poor.
- Only evaluate evidence relevant to the task's hard requirements and the current basis for the next step; you are not required to enumerate or discuss unselected observation candidates that are irrelevant to the constraints.
- If the constraint-satisfying entity, key numerical values, and the basis for the next step are all correct, a single terminology misuse that does not change constraint satisfaction is usually a minor issue and cannot by itself downgrade good to mixed; mixed requires at least a substantive omission or error affecting required evidence, state updates, or the basis for the next step.
- Do not judge as good merely because the text is fluent, long, restates the observation, piles up tool calls, or the next tool appears relevant. good requires an overall demonstration of correct understanding of existing information, reasonable absorption of latest_observation, and a basically reasonable next-step idea.
# Labels
- good: Overall reasoning quality is good; it correctly understands existing information, reasonably absorbs key evidence from latest_observation, and forms a basically reasonable idea that helps solve the task. Minor omissions or lack of detail that do not affect the core judgment are allowed.
- mixed: Overall reasoning quality is moderate; there is partial correct absorption, but also important deficiencies affecting reasoning quality, such as missing key evidence, vague state updates, a weak next-step idea, or local errors; overall it cannot be called good, but it is not a fundamental misreading either.
- poor: Overall reasoning quality is poor; it ignores or severely misreads latest_observation, misinterprets existing information or constraints, integrates evidence poorly, reasons in a confused manner, fails to update key states, or has a next-step idea that clearly conflicts with evidence/constraints.

Return only one JSON object, with no Markdown, code fences, or extra fields:
{"label":"good|mixed|poor","confidence":"high|low","reason":"One to two concise pieces of evidence describing the key aspects of this reasoning quality"}
```

User prompt

```
Evaluate only the reasoning quality of cycle (t+1) below, focusing on whether it reasonably absorbs the latest observation. Do not use the future outcome of the action; return only the required JSON.

For the overall assessment, you may consider:
1. Whether it is clearly consistent with the task requirements, hard constraints, and information already established in the history;
2. Whether it reasonably interprets and integrates the key evidence in latest_observation;
3. Whether it forms a generally reasonable next-step idea, and whether the reasoning helps advance task solving.
# Input Materials
{
 "task_requirements": "<original user request>",
 "hard_constraints": "<structured benchmark constraints>",
 "relevant_history_before_latest_observation": "<visible history before the latest observation>",
 "latest_observation": "<latest tool observation at cycle t; treat as new evidence>",
 "reasoning_t_plus_1": "<assistant reasoning at cycle t+1>",
 "action_t_plus_1_context_only": "<optional action context at cycle t+1>",
 "audit_identity": "<sample provenance and audit identifiers>"
}
```

Figure 4: Prompt template used to evaluate reasoning quality.

set remains "good", "mixed", and "poor", but we use only "good" and "poor" cases to form $\mathcal{G}$ and $\mathcal{P}$ for EAS, excluding "mixed" cases to keep the contrast clean.

B.4 Behavioral Quality and Flow Deviations

We also examine behavioral quality across the progress flow patterns defined in Sect. 6. Although $\mathcal{D}_E$ and $\mathcal{D}_D$ are defined over matched pairs, the analysis here is conducted at the individual-sample level. For each set, we collect the individual cases appearing in that set and randomly sample 50 of them for annotation. We additionally sample 50 individual cases from strictly aligned progress flow pairs, excluding cases that also appear in either deviation set, to provide a reference for comparison.

Clear behavioral differences emerge across these flow patterns. Among cases from $\mathcal{D}_E$, 46.0% are labeled as poor reasoning, compared with 20.0% among aligned-flow cases, while the proportion of good reasoning decreases from 68.0% to 22.0%. Similarly, cases from $\mathcal{D}_D$ contain more unproductive actions (26.0% vs. 14.0%) and fewer productive actions (74.0% vs. 86.0%) than the aligned-flow reference. These statistics, combined with manual inspection, are consistent with the

System prompt

```
You are an expert reviewer of local actions for long-horizon travel planning agents.
You only judge whether the action/tool call actually issued in current_round is a reasonable, defensible step that is helpful for advancing the task, given the information visible at that time and the available tool capabilities. Do not evaluate whether the entire trajectory ultimately succeeds, and do not require a single action to solve the whole task at once.
# Core Criteria
- Criteria for productive: Directly satisfying requirements, collecting necessary evidence, searching for candidates, validating hypotheses, and looking for alternatives can all be judged productive. The core criterion is whether the action is helpful for advancing the task.
- Tool search parameters usually cannot express all of the user's natural-language requirements. If the action reasonably uses the relevant parameters actually supported by the tool and intends to continue checking services, prices, ratings, etc., from the returned results, it is reasonable exploration.
- Intermediate steps are not the final answer. Even if the tool call does not immediately produce the final plan, as long as it reasonably acquires information or advances constraint satisfaction under the current state, it can be judged productive.
- Unproductive is used for actions that are clearly wrong or not helpful for the task: directly conflicting with the user's explicit requirements; clearly wrong on key conditions such as city/date/star rating/transportation mode; being significantly irrelevant to the task; illegal/invalid calls; unreasonably repeating the same call after previous results have already ruled it out; clearly mismatching the direction that should be advanced at the current point; or abruptly ending when explicit task requirements remain.
- The core criterion for unproductive is that the action is not helpful for advancing the task, or contains a clear error.
- Do not speculate about the internal intentions of the model under evaluation; only judge whether there is a clear and reasonable task-progress rationale for the action.
# Evaluation Target and Its Relationship to Thinking
- The evaluation target is the tool call/action actually issued in current_round, not candidate actions discussed in the thinking.
- previous_round_thinking and current_round_thinking are only used to understand the context in which the action was formed; they are not facts that have already been executed.
- If the plan in the thinking is reasonable but the final issued action is clearly wrong or unhelpful, still judge unproductive based on the final action.
- Do not automatically judge productive merely because the thinking is long, fluent, or confident, and do not judge unproductive merely because the thinking is incomplete. Our evaluation target is the final action of the current round, not the model's reasoning process; the core question remains whether the final action in current_round was reasonable and helpful for task progress at that time.
# Tool Capability Example
Suppose the user requests "three-star and with a pool", but the hotel search tool can only filter by city, date, star rating, and brand. In this case, querying all three-star hotels so that you can continue filtering pool-equipped candidates from the returned results is reasonable exploration and should be judged productive.
Conversely, if the queried star rating, city, or date clearly conflicts with the requirements, or if the same query is mechanically repeated after it has already been proven ineffective, or if other constraints should clearly take priority in the current context but the agent queries hotels without justification, it should be judged unproductive.
# Input Boundaries
The input contains user_query, previous_round, current_round, and the explicitly structured previous_round_thinking and current_round_thinking. The two thinking fields are the reasoning_content of the execution model Qwen3-14B in the corresponding assistant turns, listed item by item by assistant message; they are used to understand the context in which the action was formed, but they are not facts that have already been executed. current_round also contains the complete semantic content of the current assistant turn and the tool observation subsequently produced, if any; previous_round is used to determine context and obvious repetition. Do not use information outside the input.
Return only one JSON object, with no Markdown, code fences, or extra fields:
{"label":"productive|unproductive","reason":"One to two concise reasons"}
```

User prompt

```
Please judge whether the final action/tool call actually issued in current_round is productive. "productive" means that, given the information visible at that time and the available tool capabilities, the action is helpful for advancing the task; "unproductive" means it is clearly wrong, violates requirements, or is not helpful for the task. Return only the required JSON.

# Input Materials
{
 "user_query": "<original user request>",
 "previous_round": "<complete previous agent round and observation>",
 "current_round": "<complete current agent round, final action/tool call, and resulting observation if present>",
 "previous_round_thinking": "<structured reasoning_content from the previous round>",
 "current_round_thinking": "<structured reasoning_content from the current round>"
}
```

Figure 5: Prompt template used to evaluate action productivity.

The adjustment part

```
You need to judge, overall, whether Reasoning_(t+1), in light of the task requirements, hard constraints, and visible history, does the following, with particular emphasis on how well it understands existing information and integrates evidence:

1. Correctly understands existing information:
   - whether it correctly tracks what has already been established, ruled out, or left unresolved in the visible history;
   - whether it correctly interprets task requirements, hard constraints, prior tool observations, and previously accepted or rejected candidate states;
   - whether it avoids contradicting or distorting facts, constraints, or candidate states already established;
   - whether it distinguishes what is already known from what is still uncertain, unverified, or unresolved.

2. Reasonably absorbs and integrates the latest evidence:
   - whether it identifies the key task-relevant evidence in latest_observation;
   - whether it interprets the meaning and implications of that evidence, rather than merely restating or quoting the observation;
   - whether it connects the new evidence to existing information and prior state, updating what is satisfied, what is ruled out, and what remains unresolved;
   - whether it handles conflicts, missing information, or ambiguity appropriately, without ignoring, overreading, or misreading the evidence;
   - whether the resulting update of the current task state is basically correct.

3. The reasoning reflects a basically reasonable next-step idea/plan:
   - whether it turns the above understanding into a basically reasonable next-step idea that overall helps advance the task;
   - whether that next-step idea is basically consistent with latest_observation, existing evidence, and hard constraints.
   - Detailed planning is not required; the focus is on whether the next step follows from a sound reading of the available evidence.
```

Figure 6: Adapted system prompt for evidence-assimilation annotations in Sect. 6.1. It follows Fig. 4 but emphasizes understanding existing information and integrating the latest tool evidence.

association between progress flow deviations and the evidence-assimilation and action-selection difficulties examined in Sect. 6.

Table 5: Behavioral quality of individual cases from each progress flow pool. Each row reports 50 randomly drawn cases; values in %. "Aligned flow" denotes cases from pairs that preserve the expected progress flow ordering, excluding any that also appear in either deviation set.

Sample source	Reasoning		Action	
	Poor	Good	Unproductive	Productive
Cases from $\mathcal{D}_E$	46.0	22.0	–	–
Cases from $\mathcal{D}_D$	–	–	26.0	74.0
Cases from aligned flow	20.0	68.0	14.0	86.0

Query

I'm planning a trip from Hangzhou to Nanchang on November 12, 2025, returning on November 18. Specifically, I have a few small requests—could you please help me arrange the itinerary? For the return journey, I'd like to take a train, preferably a direct one with the shortest possible travel time to save time. For accommodation, please choose a hotel within the price range of 240 to 290 RMB per night—my budget is roughly in that range.
By the way, I'd also like to schedule two restaurant visits during the trip. One should be near Bayi Square, preferably at a restaurant that offers a waiting area service, which would be more convenient, rating high priority. The other should be near 'Aixi Lake Forest Wetland Park'—please find me the highest-rated restaurant nearby, as I'd like to experience some local specialties.
That's basically all—I believe I've provided all the necessary information. Please go ahead and plan the itinerary for me directly without asking further preferences. Thank you! November 12, 2025 is Wednesday

Original Reasoning

Okay, the user needs to travel from Hangzhou to Nanchang and return by train on November 18, requiring a direct train with a short travel time. First, I need to determine the return-train information. According to the provided tool response, several high-speed trains were found. The earliest arrival is train G3790, departing from Nanchang East Station at 08:02 and arriving at Hangzhou West Station at 10:47, with a duration of 165 minutes and a price of CNY 224. This train satisfies the user's requirements for a direct and short journey, so I will select it for the return trip.
Next is outbound transportation from Hangzhou to Nanchang. The user did not explicitly specify the outbound mode, but since the return trip is by train, the user may also want to take a train outbound. However, the user did not specify this, so perhaps I need to confirm it further, but the rules prohibit asking questions. I may therefore need to assume that the user wants to travel outbound by plane or train. The user did not mention this, so perhaps I should check whether other tool information is available. The user may want to fly outbound, but I need to check whether flight information is available. No flight-search results have been provided, so I may need to query flights first. Based on the user's question, however, perhaps the user only specified that the return trip should be by train, while the outbound trip can be arranged independently, such as by plane or train. Since the user did not specify, perhaps I should first handle the return trip and then the outbound trip.
...(omitted)
The user also wants two restaurant visits: one near Bayi Square at a highly rated restaurant with a waiting area, and another near Aixi Lake Forest Wetland Park at the highest-rated restaurant. I need to use `recommend_restaurants` to obtain this information. First, I need the coordinates of Bayi Square, then call `recommend_restaurants` and filter for highly rated restaurants with waiting-area service. Similarly, I need the coordinates of Aixi Lake Forest Wetland Park, then call `recommend_restaurants` and select the highest-rated restaurant.
The user also needs a hotel priced between CNY 240 and CNY 290 per night. I therefore need to call `query_hotel_info` with Nanchang as the destination, check-in on 2025-11-12, check-out on 2025-11-18, and a price range of CNY 240–290. A hotel star rating or brand may also be needed, but the user did not specify either, so they may be unnecessary. I can then select a suitable hotel from the results.
...(omitted)
Combining all the information, I need to proceed as follows:
1. Determine the return-train information, which has already been handled.
2. Query an outbound flight or train; assume the user wants to fly and call `query_flight_info`.
3. Query hotels in Nanchang priced between CNY 240 and CNY 290.
...(omitted)
7. Arrange the daily itinerary with continuous timing and reasonable scheduling of attractions, restaurants, transportation, and accommodation.
I now need to call these tools in sequence and generate the itinerary after obtaining the required data.

EAS-Steered Reasoning

Okay, the user wants to travel from Hangzhou to Nanchang, departing on November 12 and returning on November 18. The return trip should use a direct train with the shortest possible travel time. The accommodation budget is CNY 240–290 per night. The itinerary must also include two restaurant visits: one at a highly rated restaurant with a waiting area near Bayi Square, and the other at the highest-rated restaurant near Aixi Lake Forest Wetland Park.

First, the return train has already been queried. Several high-speed trains were found, and the shortest travel time is 120 minutes on G296, priced at CNY 316. Could this exceed the user's budget? The stated CNY 240–290 budget applies to the hotel, so the train price may be acceptable. I may need to check whether there is a cheaper direct train, but the user emphasizes minimizing travel time, so I should select the fastest option.

Next, I need to arrange outbound transportation. The user did not specify the outbound mode, so I may need to query flights or trains. The user only specified that the return trip should be by train, so the outbound preference is uncertain. I may need to query further, but given the available tool results, I can first complete the return-trip requirement.
Then, I need to query hotels in Nanchang priced between CNY 240 and CNY 290. I should call query_hotel_info with Nanchang as the destination, check-in on November 12, and check-out on November 18.
...(omitted)
I now need to continue calling the remaining tools, beginning with a hotel search in Nanchang, followed by restaurant recommendations and then attraction and transportation planning.

Figure 7: EAS case: Query, Original Reasoning, and EAS-Steered Reasoning at the same sample. The previous tool observation lists G3790 (165 min, CNY 224) and G296 (120 min, CNY 316). Original reasoning selects G3790; EAS-steered reasoning selects the shorter G296.

C Case Analysis of Evidence Assimilation and Action Selection

We provide two qualitative recovery cases from the test split. Both cases compare natural and intervened generation at the same task state: the EAS case compares reasoning generated from the same *entry* state, while the ERS case fixes the generated reasoning and compares actions generated from the same *decision* state. Figs. 7 and 8 present the corresponding queries, latest observations, generated outputs, and judge labels.

Fig. 7 shows an EAS case where the latest train-search observation contains two direct options: G3790 with a 165-minute duration and G296 with a 120-minute duration. The user explicitly asks for the shortest travel time. In the natural reasoning, the model focuses on the earlier arrival and lower price of G3790 and selects it, thereby ignoring the shorter G296; this reasoning is labeled "poor". After EAS, the model correctly compares the durations, identifies G296 as the shortest

Query

I plan to travel from Tianjin to Zhengzhou on November 12, 2025, and return on November 17, 2025. The entire trip will be about 6 days—please help me plan a detailed itinerary. For transportation, I'd like to take trains for the entire journey. Please book first-class seats for both outbound and return trips, as they're more comfortable and allow me to relax and rest during the ride. For accommodation, I'd like to stay somewhere nice—please find me the highest-rated hotel in Zhengzhou. Since this is a vacation, comfort is important. By the way, I'd like to visit all recommended 'Natural Scenery' attractions. I've heard the natural landscapes around Zhengzhou are quite impressive, so I'd like to experience as much as possible. Also, please arrange one meal at a restaurant near the 'Xinmi Huangdi Palace Royal Hot Spring Resort Hotel'. There are four of us traveling, so we'll need two rooms. By the way, it would be great if this restaurant offers a waiting area service, so we can rest comfortably while waiting. That's pretty much everything—I believe I've provided all necessary details. Please go ahead and prepare the full itinerary for me without asking for further information. Thank you! November 12, 2025 is Wednesday

Fixed Reasoning

Okay, the user needs a six-day trip from Tianjin to Zhengzhou, traveling entirely by train, staying at the highest-rated hotel in Zhengzhou, visiting all recommended natural-scenery attractions, and dining at a restaurant with a waiting area near the Xinmi Huangdi Palace Royal Hot Spring Resort Hotel.
...(omitted)
The user also requests a meal at a restaurant with a waiting area near the Xinmi Huangdi Palace Royal Hot Spring Resort Hotel. I need to query the location of the hotel and then recommend nearby restaurants. I may first need to obtain the hotel's coordinates and then use the restaurant-recommendation tool. The user may be conflating an attraction with a hotel, because Xinmi Huangdi Palace Royal Hot Spring Resort Hotel may itself be an attraction, but the user explicitly asks for a restaurant near it. I therefore need to determine whether it is a hotel or an attraction. Based on the previous tools, the user likely wants to dine near the hotel, so I should first obtain its location and then recommend nearby restaurants.
...(omitted)
I should probably call `search_location` to obtain the coordinates of the Xinmi Huangdi Palace Royal Hot Spring Resort Hotel, then use `recommend_restaurants` to find nearby restaurants and filter for waiting-area service. Since the user has already named the hotel, I may be able to search directly by that place name, or first verify whether it is an attraction. If the hotel is itself an attraction, I may need to include a visit, but the user only asks for a nearby restaurant. I should therefore directly arrange a meal at a restaurant near the hotel.
...(omitted)
Finally, I need to generate a detailed daily itinerary covering transportation, attractions, meals, and hotel activities, with valid temporal connections and an accurate budget calculation.

Original Tool Call

```
{
  "name": "query_restaurant_details",
  "arguments": {
    "restaurant_name": "Xinmi Huangdi Palace Royal Hot Spring Resort Hotel"
  }
}
```

ERS-Steered Tool Call

```
{
  "name": "search_location",
  "arguments": {
    "place_name": "Xinmi Huangdi Palace Royal Hot Spring Resort Hotel"
  }
}
```

Figure 8: ERS case: Query, Fixed Reasoning, Original Tool Call, and ERS-Steered Tool Call at the same *decision* state. The reasoning is held fixed, and only the tool call is compared. The original tool call confuses the hotel with a restaurant, while the ERS-steered tool call locates the hotel before searching for nearby restaurants. Red "...(omitted)" marks omitted text.

direct train, and uses this corrected conclusion to support the next planning step, such as the hotel query shown in the figure. The key change introduced by EAS is therefore not fluency, but how evidence is used: the latest observation is now integrated into constraint checking and subsequent plan formation.

Fig. 8 shows an ERS case involving a request to find a restaurant near the Xinmi Huangdi Palace Royal Hot Spring Resort Hotel. In the original trajectory, the reasoning already generated before decision proposes locating the hotel before searching for nearby restaurants. However, the original tool call invokes `query_restaurant_details` with the hotel name, confusing a hotel with a restaurant and producing an unproductive step. After ERS, with the reasoning held fixed, the model instead calls `search_location` for the hotel, which is the necessary prerequisite for a later nearby-restaurant search. This repair does not add new external evidence or rewrite the reasoning; it aligns the formal action with the already generated reasoning.

Together, these cases offer concrete illustrations of two failure modes associated with progress flow deviations: poor evidence assimilation at *entry* and unproductive action selection at *decision*. They also show that EAS and ERS help repair. These examples are qualitative and mainly serve to make the associated failure modes and corresponding interventions more concrete.

D Details for Experiments

D.1 Constraint Heterogeneity in the TravelPlanner

TravelPlanner tasks vary in the number of applicable user-specific hard constraints. The benchmark therefore contains both simple tasks with a single applicable requirement and more demanding tasks

that combine several requirements. Its deterministic evaluator always checks the total budget and, when requested, additionally checks accommodation rules, cuisine coverage, room type, and transportation.

For example, a simple task asks one traveler to visit Ogdensburg for three days under a $1,500 budget and specifies no additional local requirement; its only applicable hard constraint is the budget. A more constrained task asks two travelers to visit Dallas under a $1,400 budget, requires lodging that permits parties, requests an entire home or apartment, and specifies four cuisines. It therefore has four applicable hard constraints: budget, accommodation rule, room type, and cuisine coverage.

Besides, since long-horizon agent evaluation involves substantial execution cost, prior studies (Wang et al., 2025; Shao et al., 2024; Zhang et al., 2026b) have adopted subsets sampled from TravelPlanner for efficient evaluation and analysis rather than exhaustively running the full benchmark. Following this practice, we randomly sample 100 test tasks from the official TravelPlanner test split before evaluation and keep this subset fixed across all compared methods, avoiding any performance-based task selection. For DeepPlanning-Travel, which lacks a predefined split, we adopt a two-fold evaluation protocol following recent practice (Chen et al., 2026): probes and steering directions are constructed on one fold and evaluated on the other, with results aggregated across both folds. For each task, we report all metrics from the higher-scoring of two rollouts. The selection score is CP on DeepPlanning-Travel and the fraction of satisfied constraints among all commonsense and applicable hard constraints on TravelPlanner.

D.2 Qwen3-14B Native Prompt Adaptation

We retain the original DeepPlanning-Travel system prompt and make only two protocol adaptations. First, we replace its opening with the following two-stage instruction (shown in English):

> You are a top-tier travel planning expert. Your task is to create a comprehensive, executable, and logically rigorous travel plan. All information provided by the user is complete and includes all their preferences; you must not and cannot ask the user any additional preferences or requirements.
>
> Your workflow is divided into two stages:
>
> 1. First, use tools to collect all necessary information (such as flights, routes, prices, etc.); in each turn, use only one tool and make one complete tool call.
>
> 2. Second, after sufficient information is gathered, generate the final travel plan within `<plan></plan>` tags, strictly adhering to the rules below.

Second, we insert the following instruction within the prompt:

> `[Single-Action Protocol]` Keep the original DeepPlanning tool-use behavior, but emit at most one complete `<tool_call>` per assistant turn. The runtime will execute it immediately and return the result to you; only after that may you choose the next tool or emit one complete `<plan>...</plan>`. Never batch multiple tool calls into a single turn.

Apart from enforcing one complete tool call per assistant turn, we retain the original DeepPlanning-Travel system prompt unchanged, including its tool-grounding requirements, output schema, planning rules, and in-context examples. The same acting prompt is used for the natural baseline and PGS. We adopt a single-action protocol for analysis and experiments. This protocol provides a clear and unambiguous cycle of actions and observations. During qualitative trajectory inspection, we also observed that the model's reasoning tended to be more directly relevant to the current *decision* than under the original system prompt. This may be because each turn requires resolving only a single action, rather than multiple actions at once.

D.3 Implementation Details and Hyperparameter Selection

PGS involves several intervention-level hyperparameters, including steering strengths and intervention windows. While these parameters could be tuned separately for each setting, we perform a small-scale coarse search on training rollouts to identify a reasonable operating region rather than

Table 6: Selected probe configurations and their test performance for Qwen3-14B on DeepPlanning-Travel.

Probe	C	Threshold τ	Pos. recall	Neg. recall
entry	10^{-3}	0.5	80%	83%
decision	10^{-2}	0.7	82%	89%

exhaustively optimizing every configuration. For Qwen3-14B on DeepPlanning-Travel, a representative configuration applies

$$\tilde{h} = h + 0.5 v_{\mathrm{EAS}}, \qquad \tilde{h} = h + 1.0 v_{\mathrm{ERS}},$$

with intervention windows of 7 and 1 tokens, respectively. Note that the diagnostic steering experiments in Sect. 6 use a shorter intervention window of three tokens to isolate the effect of EAS, whereas the final PGS experiments use the configuration above. We fix these strengths and windows across all PGS evaluation settings instead of tuning them separately for each setting.

Preliminary searches on Qwen3-14B and Ministral-3-14B-Reasoning show that effective configurations generally occupy a broad region, with relatively small performance differences among them. We therefore fix these intervention strengths and windows across all PGS experiments instead of tuning them separately for each setting. This avoids performing narrow hyperparameter optimization separately for each model and benchmark, which would reduce comparability across settings and substantially increase experimental cost.

Our layer-wise analysis shows a broadly consistent task progress signal across middle-to-late layers. Within this range, we use layer 25 for Qwen3-14B, layer 30 for Ministral-3-14B-Reasoning, and layer 26 for Apriel-1.6-15B-Thinker as representative choices, informed by training-set PairAcc and EFA, with Ministral's layer index referring only to its text backbone and excluding its vision layers.

We next describe the hyperparameter selection and evaluation of the probes. Since deviations are defined through matched comparisons and are not directly available online, we use them only to select and weight reliable training cases offline, and train the probes on the full residual states for single-trajectory detection at inference time. We restrict our preliminary search to Qwen3-14B on DeepPlanning-Travel, considering the regularization parameter $C \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1\}$ and the online gating threshold $\tau \in \{0.5, 0.6, 0.7, 0.8\}$ for each probe. We first select a configuration that performs well on the corresponding probe-training data. We then evaluate the selected configuration on a separate set of labeled test cases that are disjoint from those used to fit the probe. Each probe's test set contains 100 positive and 100 negative cases. The positive and negative classes follow the definitions in Sect. 7. For the *entry* probe, positive and negative cases correspond to poor and good evidence assimilation, respectively; for the *decision* probe, they correspond to unproductive and productive actions, respectively. Positive-class recall is the proportion of positive cases correctly classified as positive. Negative-class recall is the proportion of negative cases correctly classified as negative.

As shown in Table 6, the selected *entry* and *decision* probes achieve 80%/83% positive/negative-class recall and 82%/89% positive-class recall/specificity, respectively. We empirically set $\tau_E = 0.5$ for *entry* and $\tau_D = 0.7$ for *decision*, and reused these thresholds across all remaining models and settings without further threshold search. The resulting gains support the practical reuse of this shared threshold configuration and show that PGS can remain effective without setting-specific threshold optimization.

D.4 Why not directly use the review instruction at Decision?

A substantial line of prior work has shown that reflection can improve model outputs when given a dedicated generation or refinement stage. For example, Self-Refine (Madaan et al., 2023) generates feedback and then performs refinement; Reflexion (Shinn et al., 2023) uses verbal reflections to inform later trials; and CRITIC (Gou et al., 2024) introduces an explicit critique-and-revision process with external feedback. In these settings, reflection has its own generation space and turns, allowing the model to reconsider previous outputs and incorporate the revised reasoning into subsequent generations.

Prompts for ERS

1. First, I should distinguish the hard requirements that are already confirmed, ruled out, or still unresolved based on the currently available evidence. Before forming the next action, I should briefly review these states, identify the most critical evidence gaps, organize the current understanding of the task, and then select an appropriate tool call.

2. Before generating the next action, I should revisit which available observations support the current conclusions and identify which necessary information remains insufficiently supported by evidence and requires further search or integration. Based on this assessment, I should then select the next appropriate action.

3. I should review the necessary constraints that remain unsatisfied again and assess, based on the available information, which gaps are most critical and which issues no longer require further search. I need to establish a clear priority among the remaining actions and then generate the next tool call accordingly.

Figure 9: The three review-instruction variants used to construct ERS. This diversification reduces dependence on a single prompt formulation when eliciting the activation contrast underlying evidence review and reconsideration.

Our *decision* intervention occurs at a fundamentally different point. The model has already completed reasoning and is about to generate the formal answer or tool call. Injecting a review prompt at this boundary does not provide a normal additional reasoning stage; instead, it inserts an externally supplied natural-language segment into the model's native reasoning-to-answer transition, immediately before a format-sensitive action. This changes the model's generation right as it switches from free-form reasoning to structured output, which can make the intervention too rigid and less robust. Consequently, it may not reliably improve downstream outcomes and can sometimes disrupt tool-call generation or formatting. Moreover, because the preceding reasoning tokens have already been generated and cannot be revised, post-hoc correction is inherently more limited than methods that include a dedicated reflection and revision stage. The model can reconsider how to act based on the existing reasoning and available evidence, but it cannot regenerate the reasoning trajectory that led to the current *decision* state, so it remains influenced by the reasoning already generated.

This result does not undermine the construction of ERS, because the review instruction plays a fundamentally different role during direction construction and deployment. It is used only to elicit a review-related internal state, rather than as the intervention itself. By contrasting review-conditioned and natural generations, ERS captures the activation shift associated with evidence review and reconsideration, and applies this direction directly at inference without inserting additional review text. Thus, the weak end-to-end performance of direct prompting does not imply that the review-induced activation contrast is uninformative; rather, ERS uses that contrast while avoiding the side effects of explicitly inserting the review stage.

Furthermore, to improve the robustness of ERS construction, we use three semantically aligned review instructions when constructing ERS and randomly sample one for each intervention instance. As illustrated in Fig. 9, all three variants share the same underlying objective: to review the currently available evidence, distinguish resolved, ruled-out, and still-unresolved requirements, identify the most important remaining evidence gaps, and use this assessment to guide the next tool action. Their surface formulations, however, emphasize slightly different aspects of this review process, such as evidence support, unresolved constraints, and action prioritization. This prompt-level variation reduces **reliance on any single wording pattern** and encourages the resulting contrastive direction to capture the underlying *decision* state shift toward reviewing accumulated evidence, reconsidering unresolved task requirements, and selecting a more informed next action before committing to the tool call.

D.5 Case Study and Output Examples

To provide an intuitive understanding of the generated trajectories, we present concrete output examples in this subsection. All examples are directly sampled from our formal experiments. Fig. 10 illustrates the breadth of our qualitative examples across three models, two benchmarks, and both Chinese and English data: Qwen3-14B on the Chinese version of DeepPlanning-Travel, Ministral-3-14B-Reasoning on TravelPlanner, and Apriel-1.6-15B-Thinker on the English version of DeepPlanning-Travel.

The Qwen3-14B example is a complex and multi-constraint Chinese DeepPlanning-Travel itinerary for a four-day trip from Kunming to Zhengzhou. The original plan already captures several require-

Qwen3-14B on the Chinese version of DeepPlanning-Travel

Day 1:
Current City: from 昆明 to 郑州
Accommodation: 全季郑州东站省政府酒店，¥457/间/晚
07:00-09:00 | travel_intercity_public | flight MU5829，昆明长水机场 - 郑州新郑国际机场，¥333/人
09:00-09:40 | buffer | 下飞机、提取行李
09:40-10:40 | travel_city | 郑州新郑国际机场 - 全季郑州东站省政府酒店，35km，60min，¥60
10:40-11:30 | hotel | 办理入住，全季郑州东站省政府酒店
11:30-12:30 | meal | 午餐，全季郑州东站省政府酒店餐厅，¥80/人
12:30-14:00 | travel_city | 全季郑州东站省政府酒店 - 北龙湖湿地公园，15km，30min，¥30
14:00-17:00 | attraction | 北龙湖湿地公园，¥0/人
17:00-17:10 | travel_city | 北龙湖湿地公园 - 双生四季甜品融合餐厅(姚桥社区北院店)，1.5km，10min，¥0
17:10-18:20 | meal | 晚餐，双生四季甜品融合餐厅(姚桥社区北院店)，¥120/人
18:20-19:00 | travel_city | 双生四季甜品融合餐厅(姚桥社区北院店) - 全季郑州东站省政府酒店，3km，10min，¥0
19:00-24:00 | hotel | 休息，全季郑州东站省政府酒店
Day 2:
Current City: 郑州
Accommodation: 全季郑州东站省政府酒店，¥457/间/晚
08:00-08:50 | travel_city | 全季郑州东站省政府酒店 - 城隍庙，25km，40min，¥50
08:50-10:00 | attraction | 城隍庙，¥0/人
10:00-10:10 | travel_city | 城隍庙 - 二广餐馆，1.5km，10min，¥0
10:10-11:20 | meal | 午餐，二广餐馆，¥66/人
11:20-12:30 | travel_city | 二广餐馆 - 郑州博物馆，10km，20min，¥20
12:30-14:30 | attraction | 郑州博物馆，¥20/人
14:30-15:00 | travel_city | 郑州博物馆 - 全季郑州东站省政府酒店，15km，30min，¥30
15:00-18:00 | hotel | 休息，全季郑州东站省政府酒店
18:00-19:10 | meal | 晚餐，全季郑州东站省政府酒店餐厅，¥80/人
19:10-20:00 | travel_city | 全季郑州东站省政府酒店 - 郑州二七塔，8km，20min，¥20
20:00-22:00 | attraction | 郑州二七塔，¥0/人
22:00-24:00 | hotel | 休息，全季郑州东站省政府酒店
Day 3:
Current City: 郑州
Accommodation: 全季郑州东站省政府酒店，¥457/间/晚
08:00-08:50 | travel_city | 全季郑州东站省政府酒店 - 郑州黄河风景区，20km，40min，¥40
08:50-11:30 | attraction | 郑州黄河风景区，¥50/人
11:30-11:40 | travel_city | 郑州黄河风景区 - 郑州黄河大观楼，1km，10min，¥0
11:40-12:40 | meal | 午餐，郑州黄河大观楼餐厅，¥60/人
12:40-13:30 | travel_city | 郑州黄河大观楼 - 郑州高铁站，15km，30min，¥30
13:30-14:00 | buffer | 安检，候机
14:00-16:10 | travel_intercity_public | flight 8L9508，郑州新郑国际机场 - 昆明长水机场，¥506/人
Budget Summary:
Transportation: 2052 元。其中机票（333+506）*2=1682元；市内交通：30+60+30+50+20+30+20+40+30=280元
Accommodation: 1371 元。1间房，3晚，3*457=1371元
Meals: 708 元。（120+80+66+80+60）*2=708元
Attractions & Tickets: 150 元。（20+50+20）*2=150元
Total Estimated Budget: 4271 元

Ministral-3-14B-Reasoning on TravelPlanner

Day 1:
Current City: from St. Louis to Detroit
Transportation: Flight Number: F3846130, from St. Louis to Detroit, Departure Time: 10:42, Arrival Time: 13:12
Breakfast: -
Attraction: Detroit Institute of Arts, Detroit
Lunch: BMG - All Day Dining, Detroit
Dinner: Zabardast Indian Kitchen, Detroit
Accommodation: NEW Brooklyn studio get away!, Detroit
Day 2:
Current City: Detroit
Transportation: -
Breakfast: A Dong Restaurant, Detroit
Attraction: Motown Museum, Detroit; Detroit Zoo, Detroit
Lunch: Bistro Flamme Bois, Detroit
Dinner: Urban Punjab, Detroit
Accommodation: NEW Brooklyn studio get away!, Detroit
Day 3:
Current City: Detroit
Transportation: -
Breakfast: Taksim, Detroit
Attraction: Michigan Science Center, Detroit; Comerica Park, Detroit
Lunch: Mandap - Hotel Express Towers, Detroit
Dinner: Cafe 6, Detroit
Accommodation: -

Apriel-1.6-15B-Thinker on the English version of DeepPlanning-Travel.

Day 1:
Current City: from Ningbo to Suzhou
Accommodation: Four Points by Sheraton Kunshan, ¥358/room/night
06:43-09:45 | travel_intercity_public | train K336, Ningbo Station - Suzhou Station, ¥62.5/person
09:45-10:00 | buffer | Deplaning, baggage claim
10:00-11:18 | travel_city | Suzhou Station - Four Points by Sheraton Kunshan, 51.75km, 78min, ¥203
11:18-11:30 | hotel | Check-in, Four Points by Sheraton Kunshan
11:30-12:45 | travel_city | Four Points by Sheraton Kunshan - Suzhou Museum, 49.72km, 75min, ¥194
12:45-12:50 | travel_city | Suzhou Museum - Su Yaxing · Wenren Su-Style Noodle House (Museum Branch), 0.27km, 5min, ¥0
12:50-13:50 | meal | Lunch, Su Yaxing · Wenren Su-Style Noodle House (Museum Branch), ¥45/person
13:50-13:55 | travel_city | Su Yaxing · Wenren Su-Style Noodle House (Museum Branch) - Suzhou Museum, 0.27km, 5min, ¥0
13:55-15:55 | attraction | Suzhou Museum, ¥0/person
15:55-16:01 | travel_city | Suzhou Museum - Master of the Nets Garden, 3.95km, 6min, ¥17
16:01-18:01 | attraction | Master of the Nets Garden, ¥20/person
18:01-18:06 | travel_city | Master of the Nets Garden - Du Kou · Tea Restaurant (Pingjiang Road Branch), 3.56km, 5min, ¥16
18:06-19:06 | meal | Dinner, Du Kou · Tea Restaurant (Pingjiang Road Branch), ¥102/person
19:06-20:20 | travel_city | Du Kou · Tea Restaurant (Pingjiang Road Branch) - Four Points by Sheraton Kunshan, 49.29km, 74min, ¥193
20:20-24:00 | hotel | Rest, Four Points by Sheraton Kunshan
Day 2:
Current City: Suzhou
Accommodation: Four Points by Sheraton Kunshan, ¥358/room/night
08:00-09:18 | travel_city | Four Points by Sheraton Kunshan - Suzhou Station, 51.75km, 78min, ¥203
09:18-09:30 | buffer | Boarding procedures
09:39-12:06 | travel_intercity_public | train G7507, Suzhou Station - Ningbo Station, ¥196/person
12:06-12:30 | buffer | Deplaning, baggage claim
Budget Summary:
Transportation: ¥1860 RMB. Train (K336 62.5*4=250, G7507 196*4=784) + City transport (826)
Accommodation: ¥716 RMB. 2 rooms * 358 = 716
Meals: ¥588 RMB. (45*4 + 102*4) = 588
Attractions: ¥80 RMB. (20*4) = 80
Total Estimated Budget: ¥3244 RMB

Figure 10: Representative plan outputs from three models across DeepPlanning-Travel and TravelPlanner. The examples cover both Chinese and English settings and illustrate different levels of planning granularity.

ments, including a China Eastern direct outbound flight and the requested dining arrangements near Beilonghu Wetland Park and Chenghuang Temple. However, it fails to select the cheapest eligible China Eastern flight, uses a hotel that does not satisfy the combined three-star and SPA requirement, and returns one day earlier than the requested November 15 departure, resulting in an incomplete three-day itinerary. The budget summary also contains noticeable accounting inconsistencies. The Ministral-3-14B-Reasoning example is an English TravelPlanner plan for a group of six traveling from St. Louis to Detroit. It covers all three days with concrete accommodations, restaurants, and attractions, but exceeds the specified $3,200 budget and leaves local activity transitions relatively coarse. In this output format, entries marked with "–" mean the corresponding field is not applicable or not separately scheduled that day. For example, no extra intercity transportation is needed while staying in Detroit, and no accommodation is needed after the trip ends. The Apriel-1.6-15B-Thinker example is an English DeepPlanning-Travel itinerary for a two-day trip from Ningbo to Suzhou. While it satisfies most explicit user requirements, it still suffers from several global planning issues, including an overly distant hotel location, an attraction schedule conflict, and an incorrect return-day structure. Overall, these cases illustrate both the diversity of plan formats and the types of residual errors that arise in practice. They also show that high-quality planning requires coordinating multiple coupled constraints beyond simply generating a fluent itinerary.

Qwen3-14B Refined Plan after PGS

Day 1:
Current City: from 昆明 to 郑州
Accommodation: 郑州中州雅致酒店(中牟电影小镇店)，￥107/间/晚
07:00-09:00 | travel_intercity_public | flight MU5576，昆明长水机场 - 郑州新郑国际机场，￥302/人
09:00-09:40 | buffer | 下飞机、提取行李
09:40-10:40 | travel_city | 郑州新郑国际机场 - 郑州中州雅致酒店(中牟电影小镇店)，35km，60min，￥50
10:40-11:30 | hotel | 办理入住，郑州中州雅致酒店(中牟电影小镇店)
11:30-12:30 | meal | 午餐，郑州中州雅致酒店(中牟电影小镇店)餐厅，￥80/人
12:30-13:30 | travel_city | 郑州中州雅致酒店(中牟电影小镇店) - 北龙湖湿地公园，15km，30min，￥30
13:30-17:30 | attraction | 北龙湖湿地公园，￥0
17:30-18:30 | travel_city | 北龙湖湿地公园 - 双生四季甜品融合餐厅(姚桥社区北院店)，1.5km，10min，￥0
18:30-19:30 | meal | 晚餐，双生四季甜品融合餐厅(姚桥社区北院店)，￥60/人
19:30-20:30 | travel_city | 双生四季甜品融合餐厅(姚桥社区北院店) - 郑州中州雅致酒店(中牟电影小镇店)，1.5km，10min，￥0
20:30-24:00 | hotel | 休息，郑州中州雅致酒店(中牟电影小镇店)
Day 2:
Current City: 郑州
Accommodation: 郑州中州雅致酒店(中牟电影小镇店)，￥107/间/晚
08:30-09:00 | travel_city | 郑州中州雅致酒店(中牟电影小镇店) - 郑州黄河文化公园，25km，40min，￥60
09:00-12:00 | attraction | 郑州黄河文化公园，￥60/人
12:00-13:00 | travel_city | 郑州黄河文化公园 - 郑州中州雅致酒店(中牟电影小镇店)，25km，40min，￥60
13:00-14:00 | meal | 午餐，郑州中州雅致酒店(中牟电影小镇店)餐厅，￥80/人
14:00-15:00 | travel_city | 郑州中州雅致酒店(中牟电影小镇店) - 二七纪念馆，18km，30min，￥40
15:00-18:00 | attraction | 二七纪念馆，￥45/人
18:00-19:00 | travel_city | 二七纪念馆 - 瓯玉川宴会中心，3km，10min，￥0
19:00-20:00 | meal | 晚餐，瓯玉川宴会中心，￥100/人
20:00-24:00 | hotel | 休息，郑州中州雅致酒店(中牟电影小镇店)

Day 3:
Current City: 郑州
Accommodation: 郑州中州雅致酒店(中牟电影小镇店)，￥107/间/晚
08:30-09:00 | travel_city | 郑州中州雅致酒店(中牟电影小镇店) - 城隐庙，12km，30min，￥40
09:00-12:00 | attraction | 城隐庙，￥30/人
12:00-13:00 | travel_city | 城隐庙 - 二广餐馆，0.5km，10min，￥0
13:00-14:00 | meal | 午餐，二广餐馆，￥66/人
14:00-15:00 | travel_city | 二广餐馆 - 郑州黄河博物馆，10km，20min，￥30
15:00-17:00 | attraction | 郑州黄河博物馆，￥50/人
17:00-18:00 | travel_city | 郑州黄河博物馆 - 郑州中州雅致酒店(中牟电影小镇店)，15km，30min，￥40
18:00-19:00 | meal | 晚餐，郑州中州雅致酒店(中牟电影小镇店)餐厅，￥80/人
19:00-24:00 | hotel | 休息，郑州中州雅致酒店(中牟电影小镇店)
Day 4:
Current City: from 郑州 to 昆明
Accommodation: -
08:00-09:00 | travel_city | 郑州中州雅致酒店(中牟电影小镇店) - 郑州新郑国际机场，20km，40min，￥50
09:00-09:30 | buffer | 安检，候机
09:30-12:30 | travel_intercity_public | flight MU5830，郑州新郑国际机场 - 昆明长水机场，￥350/人
Budget Summary:
Transportation: 2114 元。其中机票（302+350）*2=1704元；市内交通：30+50+60+60+40+30+40+50=370元
Accommodation: 321 元。1间房，3晚，3*107=321元
Meals: 1140 元。(80+60+100+80+66+80)*2=1140元
Attractions & Tickets: 285 元。(60+45+30+50)*2=285元
Total Estimated Budget: 3850 元

Figure 11: Refined output of Qwen3-14B on the DeepPlanning-Travel instance after applying PGS. The generated trajectory better satisfies the user-specific requirements, including the outbound flight selection, accommodation preference, and complete multi-day schedule.

Fig. 11 further presents the same Qwen3-14B instance after applying PGS. The refined plan corrects the major mismatches with the user's request: it selects MU5576 as the lowest-priced eligible China Eastern direct flight, replaces the original accommodation with a three-star hotel offering SPA service, preserves the requested dining arrangements, and restores the full four-day schedule with the return trip placed on the final day. These coordinated revisions improve both personalized satisfaction and the temporal structure of the itinerary, yielding a more complete and coherent final plan. However, some fine-grained planning details still require further refinement. For example, although the return flight satisfies the direct-flight requirement, its reported price is inaccurate; meanwhile, a few entities and budget entries still exhibit grounding and accounting inconsistencies.

The comparison demonstrates that PGS can effectively identify and correct local planning deficiencies, such as missing steps and insufficiently grounded entities, leading to a more coherent and executable trajectory. By refining the original plan according to the user's requirements, PGS improves the overall quality of the generated itinerary and produces a more structured final plan.