Learning to Surpass: Training Tool-Using Agents with Anchored Feedback

John Gkountouras[1,2] **Fengjun Wang**[2] **Angelantonio Castelli**[2] **Satendra Kumar**[2]
[1]ILLC, University of Amsterdam
[2]Booking.com

Abstract

Training tool-using agents for planning is challenging when high-quality demonstrations exist only as final outcomes without the intermediate tool traces that produced them. A common response is to impute traces and apply imitation learning, but this approach is indirect and structurally limited to matching the reference rather than improving upon it. We propose anchored comparative feedback: rather than imitating a reference plan, we train agents to surpass it under a comparative LLM judge while satisfying hard constraints. The anchor provides a stable optimization target, and comparing two identically rendered plans neutralizes surface-level judge biases. On TripTailor, a tool-augmented travel planning benchmark, anchored GRPO achieves 80.1% final success rate compared to 43.7% for imitation learning on imputed traces, with gains on both constraint satisfaction and LLM-judged preference that persist across judge families, rubrics, and a held-out reward model. Gains generalize beyond TripTailor, with a fivefold improvement on the held-out TravelPlanner benchmark. These results suggest that anchored comparative feedback offers an effective approach for learning transferable tool-using planners from outcome-only supervision.

1 Introduction

Large language models increasingly act as *tool-using agents* that interleave language generation with calls to external APIs, search systems, and structured databases [Yao et al., 2022, Schick et al., 2023, Qin et al., 2023, Li et al., 2023]. This paradigm is especially compelling for planning and decision-making, where tools provide grounded state information and the model must synthesize multi-step choices into a coherent plan. Yet even when tool access is available, producing plans that are simultaneously *valid* and *high-quality* remains difficult. A core reason is that the planning objective is only *partially specifiable*: some requirements can be verified programmatically (e.g., budget feasibility, entity validity in a database), but others are inherently holistic and comparative (e.g., whether an itinerary is well-personalized or makes sensible trade-offs). This tension is explicit in constrained generation benchmarks like TripTailor [Wang et al., 2025b], which combines programmatic feasibility checks with preference-based evaluation against human reference plans. Multiple plans may satisfy all constraints yet differ substantially in subjective quality. Rule compliance can be measured; "goodness" often cannot.

A second challenge concerns what kind of supervision is available for training tool-using planners. In many practical settings, we can obtain high-quality *outcomes* (e.g., curated itineraries, final reports, resolved tickets, completed code changes), but do *not* observe the intermediate tool interactions producing them. A common response is to *impute* missing intermediate steps and train via imitation: constructing tool traces or teacher trajectories and applying supervised fine-tuning [Schick et al., 2023, Qin et al., 2023, Li et al., 2023, Xu et al., 2026]. This approach produces broad tool competence but relies on the fidelity of reconstructed trajectories and structurally limits the model to imitating the

Work done during an internship at Booking.com. Corresponding author: i.gkountouras@uva.nl

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

reference rather than improving upon it. When strong outcome-level artifacts already exist, we would like to *directly leverage* them as targets for training, rather than as templates to copy.

We propose **anchored comparative feedback** as a way to train tool-using planners from outcome-only supervision. The key idea is to treat a high-quality outcome as an *anchor* and train the agent to satisfy verifiable constraints while *improving upon* the anchor under a comparative LLM judge. Rather than asking a judge for an absolute score ("is this plan good?"), we ask it to compare the model's plan against the anchor ("is this plan better than the reference?"). This anchored formulation provides a stable optimization target: the preference reward depends on how the plan compares with the anchor, while hard constraint gates prevent trading validity for superficial judge-preferred artifacts. Anchors can come from human references or synthetic data; this choice defines what the model is encouraged to surpass. We instantiate this approach using Group Relative Policy Optimization [Shao et al., 2024] with gated rewards that zero out preference signal for constraint-violating trajectories. Prior work compares against references in single-turn generation (Section 2). Here the reference has no tool calls, so the policy has to find its own, and only plans that pass the hard constraints receive the comparison reward.

Our goal is not to claim universal benefit but to characterize when anchored feedback helps for constrained planning. We address four questions through experiments on TripTailor: how anchor-based improvement compares to trace-imputation under outcome-only supervision; how human versus synthetic anchors shape constraint satisfaction and preference; how verifiable constraints should combine with comparative preference rewards to resist judge biases; and how stable the results are across judge families, rubrics, and protocols [Zheng et al., 2023, Gu et al., 2024]. Our main contributions are:

- **Anchored feedback outperforms trajectory imitation.** When only outcome-level demonstrations exist, anchored GRPO achieves 80.1% final success versus 43.7% for supervised fine-tuning on imputed traces, with gains on both constraint satisfaction (+34 points) and LLM-judged preference (+22 points); see Section 4.1 for metric definitions.
- **Human anchors generalize better than synthetic anchors.** Both achieve strong in-distribution performance, but human-anchored training shows superior transfer to held-out evaluation rubrics, suggesting that synthetic anchors may encode teacher-specific biases.
- **Hard constraint gating prevents preference gaming.** Without anchoring, optimization concentrates on verifiable constraints at the expense of holistic quality (97.9% rationality but only 38.5% LLM wins). Anchored gated rewards achieve balanced improvements on both axes.
- **Improvements are robust and transferable.** Gains persist across judge families (GPT-4o, Claude, and two open-weights judges), evaluation paradigms (LLM judges, contrastive reward model), and rubric designs (training-aligned and orthogonal). A human evaluation on the full test set confirms the gain over the base model. Constraint-satisfaction skills transfer to related benchmarks, with a fivefold improvement on TravelPlanner.

2 Related work

Tool-using agents and trace-based supervision. A standard approach to training tool-using agents is to collect or synthesize trajectories that include tool calls and observations, then apply supervised fine-tuning. Methods differ in how traces are obtained: filtering API calls by predictive value [Schick et al., 2023], annotating solution paths [Qin et al., 2023], manually constructing API chains [Li et al., 2023], or synthesizing trajectories with teacher models [Xu et al., 2026]. This paradigm has produced broad tool competence, but it implicitly assumes access to tool traces or a reliable procedure for constructing them. Our setting departs from this assumption: we focus on *outcome-only* supervision, where high-quality final plans exist but intermediate tool interactions are unknown. Trace imputation in this setting is indirect and structurally limited to matching the reference.

Preference-based alignment and AI feedback. RLHF popularized optimizing language models using preference feedback [Ouyang et al., 2022], and RLAIF replaces human labels with AI-generated preferences, enabling scalable alignment and quality optimization [Bai et al., 2022, Lee et al., 2024]. In parallel, direct preference optimization methods provide non-RL alternatives for learning from pairwise preferences [Rafailov et al., 2023]. Our work builds on comparative feedback formulations, using a reference plan as an anchor. The judge evaluates both plans on the same rubric, and we derive

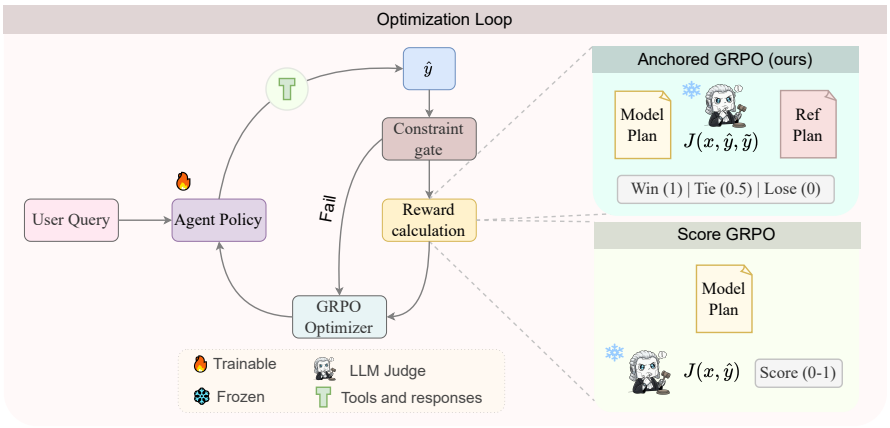


Figure 1: **Anchored GRPO training overview.** (Left) The optimization loop (single rollout): the agent policy generates a plan $\hat{y}$ from a user query, which passes through a constraint gate before reward calculation. (Right) Comparison of reward formulations: *Anchored GRPO* (ours) compares the model's plan against a reference plan $\tilde{y}$ via an LLM judge $J(x, \hat{y}, \tilde{y})$, yielding win/tie/lose outcomes. *Score GRPO* uses absolute scoring $J(x, \hat{y})$ without a reference, which can be exploited through superficial artifacts.

a comparative outcome from their relative scores. This anchored setup is tailored to constrained planning where high-quality reference outcomes exist. Pairwise-RL [Xu et al., 2025] and DPO with reference-guided judges [Shi et al., 2026] also train against a reference, and self-play fine-tuning [Chen et al., 2024b] contrasts a model's generations with human responses. These methods address single-turn generation. Our reference has no tool calls, and only plans that pass the constraint gate are compared with it.

LLM-as-a-judge. Using strong LLMs as evaluators is now common in open-ended generation, and MT-Bench helped operationalize judge-based pairwise comparisons while documenting practical biases such as position and verbosity effects [Zheng et al., 2023]. More recent meta-evaluations and surveys systematize how to design, prompt, and audit LLM judges [Gu et al., 2024]. When a judge is used for training, its vulnerabilities can induce reward hacking, including attacks that exploit superficial prompt artifacts [Zhao et al., 2025]. These findings motivate our emphasis on robustness checks (e.g., order randomization, multiple judge families, rubric variation) when using LLM judge output as optimization signals.

Group-based policy optimization for LLMs. GRPO was introduced as a PPO-family optimizer using multiple samples per prompt and within-group baselines to avoid training an explicit value function [Shao et al., 2024]. While GRPO is agnostic to the source of reward, most of the existing research focuses on verifiable or automatically checkable signals (e.g., correctness in reasoning tasks). Our setting differs in that we combine (i) verifiable constraint checks with (ii) comparative, reference-anchored preference rewards, and study how this composition affects constraint satisfaction and plan quality in tool-augmented planning.

Outcome-level anchors and imputing intermediate steps. Several lines of work exploit knowledge of desired outcomes to shape intermediate decisions, including approaches that select or refine intermediate actions based on information gain signals in multi-turn agents [Wang et al., 2025a] and methods that learn latent reasoning structure for long-form generation [Gurung and Lapata, 2025]. These are complementary to our approach. We do not impute or supervise intermediate tool traces. We use the anchor outcome directly as a *comparison target* for preference optimization and enforce explicit verifiable constraints during training.

3 Methodology

3.1 Problem setup

We study itinerary generation as a tool-using planning problem. Each instance provides a user request x (destination, dates, budget, preferences) and access to external tools that query a sandbox

database of hotels, restaurants, attractions, and transportation. Let $\tau = (a_{1:T}, o_{1:T})$ denote a full interaction trajectory, where each a_t is either a tool call or final plan token emission, and each o_t is the corresponding tool response. A policy π_θ generates trajectories by interleaving tool calls with tool-conditioned continuation; each action conditions on the history of previous actions and observations. The trajectory terminates by emitting a structured JSON plan $y = y(\tau)$. The schema is provided in Appendix F. There is no user interaction; the agent generates the plan in a single pass. We disable thinking tokens, so the agent emits tool calls and the final plan directly with no interleaved reasoning.

For each request, we have access to a reference plan $\tilde{y}$ (human-authored or synthetic), but we do not observe the intermediate tool traces that produced it. The policy must learn its own tool-use sequence, since the reference comes without one. The reference is also not necessarily optimal, and the goal is to surpass it.

3.2 Anchored comparative feedback

The central idea is to train the policy to improve upon a reference plan under a comparative LLM judge. For each request x, we provide the reference $\tilde{y}$ and ask a judge J to compare the candidate plan y against $\tilde{y}$ under a fixed rubric:

$$s = J(x, y, \tilde{y}) \in \{\text{loss}, \text{tie}, \text{win}\}. \tag{1}$$

The anchored design defines what the model should surpass. This matches how preference-based evaluation is operationalized in benchmarks where multiple valid solutions exist. In practice, the judge scores both plans on a 1–5 scale; we derive win if the model's score exceeds the reference's, tie if equal, and loss otherwise (see Appendix P for prompts).

3.3 Hard-gated reward

Plans must satisfy verifiable constraints (valid entities, budget feasibility, no repeated venues, appropriate meal prices, reasonable visit durations; see Appendix C for complete definitions) in addition to being preferred by the judge. Let $\mathcal{C}(\tau) \in \{0, 1\}$ indicate whether all constraints pass for trajectory τ. We combine constraint satisfaction with comparative preference via a gated reward:

$$R(\tau) = \mathcal{C}(\tau) \cdot \big(r_{\text{pref}}(s) + r_{\text{rule}}(\tau)\big), \quad \text{where } r_{\text{pref}}(s) = \begin{cases} 1 & s = \text{win} \\ 0.5 & s = \text{tie} \\ 0 & s = \text{loss} \end{cases} \tag{2}$$

Here $r_{\text{rule}}(\tau) \geq 0$ is a weighted sum of rule-based checks on the plan, such as budget and route efficiency (Appendix C.1). The preference term is the only part of the reward that uses the judge. If any constraint fails, the trajectory receives $R(\tau) = 0$ regardless of the preference of the judge LLM. This treats constraints as non-negotiable feasibility gates: the policy receives optimization pressure only within the feasible set, preventing it from trading constraint satisfaction for judge-preferred artifacts. The tie reward reflects that matching a high-quality anchor provides a useful signal, though surpassing it is preferred.

3.4 Policy optimization

We optimize π_θ with Group Relative Policy Optimization (GRPO), a critic-free method that estimates baselines from multiple samples per prompt [Shao et al., 2024]. For each request x, we sample K trajectories from the current policy and compute their gated rewards. GRPO forms within-group advantages by centering and normalizing, then applies a clipped policy-gradient update. Because rewards are gated, infeasible samples contribute zero reward and only influence training through their effect on the group baseline; the update thus emphasizes improving over alternative feasible rollouts for the same prompt. We provide the complete algorithm specification in Appendix A.

During training, each trajectory includes the full tool interaction sequence and final JSON plan. We compute the policy-gradient loss only over model-emitted tokens (tool-call strings and the final JSON plan); tool responses are included in the context as observations but are never scored under the policy loss. At inference, the trained policy produces a complete plan in one pass.

3.5 Mitigating LLM-as-a-judge biases

Using LLM judges as training signals raises concerns about reward hacking: the policy might exploit judge-specific biases without improving plan quality. Recent work has cataloged systematic vulnerabilities in LLM evaluation [Ye et al., 2025, Thakur et al., 2025], including demonstrations that adversarial phrases can inflate scores without improving content [Raina et al., 2024] and that even "null models" outputting constant responses can achieve high win rates [Zheng et al., 2025]. We design our training and evaluation protocol to address the major known failure modes, making it difficult for the policy to learn superficial shortcuts. These choices address known biases; Section 4.6 verifies robustness empirically across judge families and rubrics. We cannot rule out yet-uncharacterized vulnerabilities, but improvements transfer across all evaluation conditions tested.

Position bias. The tendency to favor responses appearing in certain positions is well-documented [Wang et al., 2024, Shi et al., 2025, Thakur et al., 2025]. During evaluation, we control for this by running each pairwise comparison twice with swapped presentation order and averaging outcomes. During training, we randomly swap the model output and the anchor positions for each sample, preventing the policy from learning position-dependent artifacts.

Self-enhancement bias. LLM evaluators tend to prefer their own outputs [Ye et al., 2025, Zheng et al., 2023]. We address this by using Gemini-2.5-Flash-Lite as the training judge while evaluating with GPT-4o and Claude-4.5-Haiku, models from different providers with distinct architectures. This cross-family design provides evidence that learned improvements generalize beyond the training judge's specific preferences.

Length and style biases. Judges tend to prefer longer or more visually formatted responses regardless of content quality [Park et al., 2024, Huang et al., 2025, Chen et al., 2024a]. We substantially reduce this attack surface by requiring all policies to output pure JSON with a fixed schema, then rendering plans to natural language via a deterministic template before judge presentation. Compared plans thus have nearly identical length and formatting, removing surface variation that judges might exploit. We also disable chain-of-thought during tool interaction.

4 Experiments

We evaluate anchored comparative feedback on TripTailor [Wang et al., 2025b], a tool-augmented travel planning benchmark where models call tools against a sandbox database to retrieve valid entities and prices, condition on tool outputs across multiple calls, and emit complete multi-day itineraries as structured plans. The sandbox design enables fully reproducible experiments without real API calls, while the multi-day planning task requires coordination across accommodations, meals, attractions, and transportation. TripTailor includes human-curated reference plans for each query, which we use as anchors for comparative feedback.

4.1 Benchmark and evaluation metrics

We use the TripTailor test split containing 703 user requests spanning 2 to 7-day itineraries. Each request specifies destination, dates, budget, and preference constraints. The environment exposes a sandbox database of points of interest across multiple cities, including hotels, restaurants, attractions, and transportation options. Following TripTailor [Wang et al., 2025b], we report metrics across three dimensions. **Feasibility** checks that all referenced entities exist in the sandbox (Within Sandbox) and all mandatory fields are present (Complete Information). **Rationality** checks five verifiable constraints: total cost within budget, no repeated restaurants or attractions, meal prices within the requested tier, and activity durations within recommended ranges. For both, given m sub-constraints over n samples, we report micro $= \frac{1}{mn}\sum_{i,j}\mathbf{1}[c_{ij}]$ (average sub-constraint pass rate) and macro $= \frac{1}{n}\sum_{i}\mathbf{1}[\bigwedge_{j} c_{ij}]$ (fraction where all sub-constraints pass). A plan is *Valid* if all feasibility and rationality sub-constraints pass. **Personalization** combines two complementary signals against the human reference. (1) The *LLM-Win* metric aggregates four comparisons (two judges, GPT-4o and Claude-4.5-Haiku, times two presentation orderings); each judge scores both plans on a 1–5 scale, and a sample counts as a win only if the model's averaged score strictly exceeds the reference's, controlling for judge-specific and position effects [Zheng et al., 2023, Wang et al., 2024]. When a judge response cannot be parsed, that judgment is left out of the comparison. (2) The *Reward Model (RM)*, following Wang et al. [2025b], is a Qwen2.5-1.5B-Instruct model fine-tuned with a

Table 1: Main results on TripTailor (703 requests). Micro/Macro = average/all sub-constraints satisfied. LLM/RM = preference win rate against human references via LLM judge / reward model. Valid = all sub-constraints satisfied. Final = Valid $\wedge$ (LLM $\vee$ RM), following TripTailor; Strict = Valid $\wedge$ LLM $\wedge$ RM. Best results in **bold**. † = from Wang et al. [2025b].

	Feasibility		Rationality		Personalization			Final	
Model	Micro	Macro	Micro	Macro	LLM	RM	Valid	Surpass	Strict
$\text{Workflow}_{\text{4o-mini}}$ †	98.3	97.3	91.6	63.7	15.8	11.8	–	16.2	–
Baseline (Qwen3-4B)	97.7	95.4	84.6	47.5	45.4	46.5	46.7	34.1	11.1
SFT (imputed traces)	98.6	97.9	89.1	60.2	39.5	48.5	60.2	43.7	12.2
Score GRPO	96.0	92.2	**99.5**	**97.9**	38.5	66.3	90.3	71.3	24.5
Syn-Anchored GRPO	**99.0**	**98.3**	98.2	92.3	43.2	**68.1**	91.2	74.4	26.5
Gemini-3.0-Pro	95.2	91.7	93.3	74.7	59.6	44.1	72.0	52.6	20.9
Anchored GRPO	97.9	96.6	98.5	94.5	**61.6**	67.7	**91.6**	**80.1**	**38.4**

contrastive ranking objective on pairwise preference data (Appendix D); it is never used during policy optimization and serves purely as a held-out evaluator, differing from the LLM judges in architecture and training signal (contrastive ranking on human-versus-retrieved negatives, where the judges use prompted pairwise comparison), so a win under both is stronger evidence of quality than either alone. **Final Surpass**, our primary metric, is Valid $\wedge$ (LLM $\vee$ RM), following TripTailor. We also report *Strict* Final Surpass, Valid $\wedge$ LLM $\wedge$ RM, which requires both evaluators to prefer the plan.

4.2 Training data and baselines

TripTailor provides 3,145 training queries with human-curated itineraries in free-form narrative text. We convert these to structured JSON using Gemini-3.0-Pro (see Appendix B). These outcome-level plans serve as anchors during GRPO training; no intermediate tool traces are available or used.

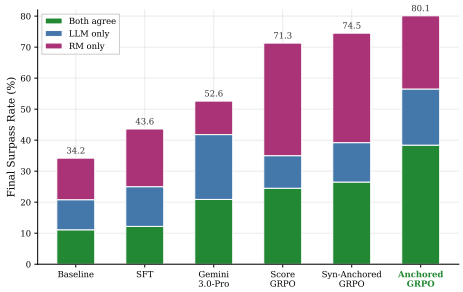
We use Qwen3-4B-Instruct-2507 as our base model and fine-tune using GRPO [Shao et al., 2024] with hard-gated rewards: plans violating any constraint receive reward 0, while valid trajectories receive the reward of Equation 2, whose preference term gives 1 for win, 0.5 for tie, and 0 for loss. Full training configurations appear in Appendix E.

We compare against five alternatives. The **Baseline** is Qwen3-4B-Instruct-2507 without post-training. **SFT** applies supervised fine-tuning on imputed tool traces following the trajectory-construction paradigm common in tool-learning [Schick et al., 2023, Qin et al., 2023]: we reverse-engineer tool call trajectories from the human-authored final plans by imputing the search queries and tool responses that would retrieve exactly those entities, an oracle reconstruction that is correct by design (see Appendix G). **Score GRPO** applies GRPO with pointwise judge scores instead of anchored comparison, using the same rubric with comparison-specific language removed; this isolates the effect of anchored versus absolute scoring. **Syn-Anchored GRPO** applies GRPO with synthetic teacher anchors instead of human references, where anchors are generated using Gemini-3.0-Pro with the **workflow** method from Wang et al. [2025b], selected as the best-performing approach in their evaluation; applying this method directly to test queries yields our Gemini-3.0-Pro baseline. The three GRPO variants share all rule-based reward terms and differ only in the preference term.

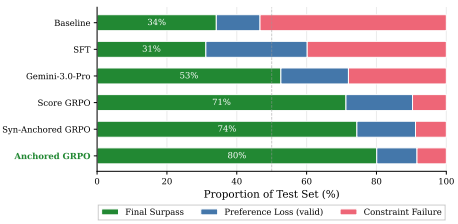
4.3 Main results

Table 1 presents our main results. The baseline Qwen3-4B achieves reasonable feasibility (95.4% macro) but struggles with rationality constraints (47.5% macro), yielding only 34.1% Final Surpass. Trajectory imitation via SFT improves rationality to 60.2% and achieves 43.7% Final Surpass, a 9.6-point gain. However, SFT's LLM win rate actually decreases relative to the baseline (39.5% versus 45.4%). This is expected, since imitation learning optimizes to match the reference distribution and its outputs tend to be similar to the anchor in quality. Under our strict-win criterion, such outputs often register as ties.

Anchored GRPO achieves 80.1% Final Surpass, more than doubling the baseline and substantially outperforming both SFT and the Gemini-3.0-Pro teacher (52.6%). The improvements reflect gains on both constraint satisfaction and preference alignment: macro rationality rises from 47.5% to 94.5%,



(a) Final Surpass decomposed by evaluator agreement. Both = Valid ∧ LLM ∧ RM; LLM-only = Valid ∧ LLM ∧¬RM; RM-only = Valid ∧ RM ∧¬LLM.



(b) Outcome decomposition: Final Surpass, valid-but-loses (preference loss), invalid (constraint failure). The baseline's failures are dominated by constraint violations (53.3%).

Table 2: Inference-time selection and other baselines on TripTailor (703 requests). Best-of-8 selects among eight base-model samples per request. Valid∧RM counts valid plans that the reward model prefers over the reference.

	Baseline (Qwen3-4B)	Final-plan only SFT	Best-of-8 + filter	Best-of-8 + LLM rerank	Anchored DPO	Constraint-only GRPO	Anchored GRPO
Valid	46.7	0.0	64.6	46.1	64.2	88.6	**91.6**
RM win	46.5	19.1	62.3	53.9	57.5	61.6	**67.7**
Valid∧RM	24.5	0.0	42.8	28.7	39.3	54.5	**62.0**

while LLM wins increase from 45.4% to 61.6%. Anchored GRPO's Final Surpass is significantly higher than that of every other method, including the closest competitor, Syn-Anchored GRPO (Appendix J). The same recipe also carries over to a second model family. On Gemma-4-E4B-it, it raises validity from 46.7% to 84.6% and the reward-model win rate from 60.9% to 81.2% (Appendix S).

Score GRPO, which uses pointwise scores without anchoring, achieves the highest rationality (97.9% macro) but the lowest LLM win rate among GRPO methods (38.5%). This suggests that without an anchor to ground the preference signal, optimization pressure concentrates on verifiable constraints at the expense of holistic quality. Syn-Anchored GRPO, which uses synthetic Gemini-generated anchors, achieves strong Final Surpass (74.4%) but lower LLM wins than Anchored GRPO (43.2% versus 61.6%). The gap indicates that anchor quality affects preference alignment: human-curated anchors provide a stronger signal for what constitutes a well-personalized plan (18.3 points more LLM wins; Appendix J).

Figure 2a decomposes Final Surpass by evaluator agreement. Anchored GRPO achieves the highest rate of plans that win under both LLM and RM evaluation (38.4%), indicating that its improvements are recognized across evaluation paradigms. This component is the Strict column of Table 1, where Anchored GRPO leads the next best system by 11.9 points. In contrast, Score GRPO derives most of its Final Surpass from RM-only wins (36.3% versus only 10.5% LLM-only), while Gemini-3.0-Pro shows the opposite pattern with more LLM-only wins (20.9%) than RM-only (10.8%). The balanced agreement profile of Anchored GRPO suggests that its improvements do not come from gaming a single evaluator.

4.4 Inference-time selection

We next ask whether choosing among more samples at inference time can match the trained model, and how much of the gain the gate provides on its own. Table 2 compares Anchored GRPO with the base model and five further baselines for these questions (Appendix R). With a constraint filter, best-of-N sampling from the base model reaches 64.6% validity at $N = 8$, which is the share of requests where at least one of the eight samples is valid. Reranking the same samples with an LLM judge lowers validity as N grows, since the judge does not check the constraints. Anchored DPO uses the same comparison against the reference and learns offline from pairs of final plans, so the tool calls get no direct learning signal. It reaches 64.2% validity. Final-plan-only SFT never produces a valid plan, since without tool calls it cannot ground entities in the sandbox. Constraint-only GRPO

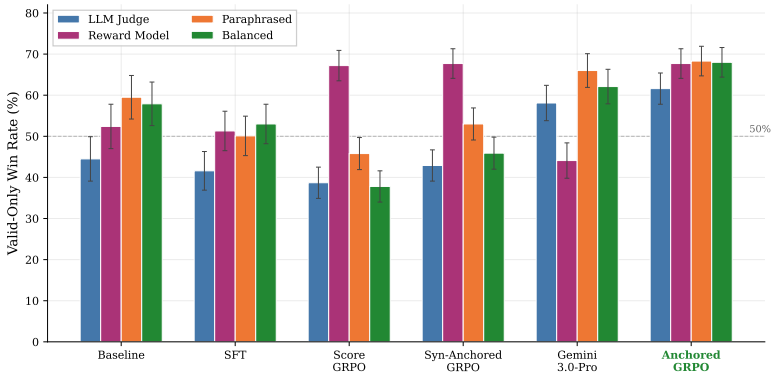


Figure 3: Evaluator robustness: valid-only win rates (%) across four evaluation conditions. Anchored GRPO achieves the highest win rate under all conditions, including held-out rubrics with different evaluation criteria, and ties with Syn-Anchored GRPO under the reward model. Dashed line indicates 50% (better than reference on average). For Anchored GRPO under the Paraphrased rubric, 223 samples are scored by one judge only.

keeps the gate and the rule-based terms and drops the judge. It comes close to Anchored GRPO on validity but has a lower reward-model win rate, and this difference is what the anchored comparison adds. Anchored GRPO is significantly better than every baseline on all three measures (Appendix R).

4.5 Understanding the improvements

To better understand where gains originate and whether they generalize across query types, we analyze the composition of outcomes and stratify by query characteristics (Appendix M). Figure 2b categorizes the 703 test samples into three mutually exclusive outcomes: Final Surpass (valid and preferred by at least one evaluator), constraint failure (fails feasibility or rationality), and preference loss (valid but loses to reference on both evaluators). For the baseline, the dominant failure mode is constraint violation: 53.3% of samples fail constraints, while only 12.5% are valid but lose on preference.

Anchored GRPO reduces constraint failures from 53.3% to 8.4% while maintaining a similar preference loss rate (11.5% versus 12.5%). The 46-point improvement in Final Surpass thus comes primarily from learning to satisfy constraints, not from gaming preference evaluators. SFT shows intermediate constraint failure (39.8%) and a higher preference loss than the baseline (16.5% against 12.5%), since its valid plans often do not beat the reference. Score GRPO has a similar constraint failure rate (9.7%) but a higher preference loss (19.1%) than Anchored GRPO, consistent with its over-optimization for verifiable metrics. Finally, Anchored GRPO retains 87.1% of baseline successes while converting 76.5% of baseline failures (net $+323$ samples), so few gains come at the expense of queries the baseline already solved.

Anchored GRPO also makes more tool calls (8.73 against 5.78 per plan). In the trajectory analysis of Appendix O, 46.6% of baseline plans with 7 to 8 calls are valid, against 89.9% for Anchored GRPO, so more calls alone do not explain the gain. At inference, neither the judge nor the anchor is used.

4.6 Evaluator robustness and human evaluation

Section 3.5 described our protocol for mitigating known LLM-as-a-judge biases. Here we empirically verify that improvements transfer beyond the training evaluation setup by testing with held-out rubrics the policy never encountered during optimization. The *Paraphrased* rubric (Appendix P) covers the same underlying dimensions as the training rubric but adopts user-centric framing, asking whether the plan addresses "what the traveler specifically asked for". The *Balanced* rubric takes a deliberately orthogonal approach: it replaces the domain-specific criteria with preference alignment, logical flow between activities, plan completeness, and overall recommendation quality. Evaluation with all rubrics follows the same debiasing protocol: aggregation across both judge families and both presentation orderings.

Table 3: LLM win rate (%) against the human reference under three judge configurations (Appendix T).

Model	GPT-4o + Claude-4.5-Haiku	Llama-3.3-70B	Mistral-Small-3.2
Baseline (Qwen3-4B)	45.4	33.0	31.7
SFT (imputed traces)	39.5	28.9	30.3
Score GRPO	38.5	27.5	22.8
Syn-Anchored GRPO	43.2	31.5	28.6
Gemini-3.0-Pro	59.6	38.2	36.7
Anchored GRPO	**61.6**	**43.7**	**40.3**

Figure 3 shows that Anchored GRPO achieves the highest win rate under all four evaluation conditions, tied with Syn-Anchored GRPO under the reward model. Under the three LLM-judge conditions, Gemini-3.0-Pro comes second and ahead of Syn-Anchored GRPO, and under the Paraphrased rubric it comes close to Anchored GRPO. Improvements also transfer to the Balanced rubric, which uses different criteria from training: Anchored GRPO achieves 68.0% versus 62.1% for Gemini-3.0-Pro. Among valid plans, Score GRPO again has a high RM win rate (67.2%) and a low LLM win rate (38.7%), and its win rate stays low under both held-out rubrics. Without an anchor, optimization favors constraint satisfaction over holistic preference.

Syn-Anchored GRPO matches Anchored GRPO under the reward model but trails it under all three LLM-judge conditions, and the gap is largest under the orthogonal Balanced rubric. This suggests that synthetic anchors may encode teacher-specific biases that transfer less than human-curated preferences. Anchored GRPO also has the highest win rate when a win requires a margin of 0.5 or 1.0 points (Appendix L). Re-scoring all systems with two open-weights judges, Llama-3.3-70B and Mistral-Small-3.2, lowers every system's win rate and keeps Anchored GRPO first and Gemini-3.0-Pro second under both (Table 3). Under both open judges, the gap between Anchored GRPO and Gemini-3.0-Pro is larger than under the judges of Table 1.

In a human evaluation of three systems on the full test set (Appendix Q), seven professional annotators compared each plan with the human reference on the TripTailor rubric, with both plans unlabeled and in random order. The Anchored GRPO plans for this study come from an inference run made during development. With the human judgment in place of the automated preference term, Final Surpass is 16.9% for Anchored GRPO, more than twice the baseline's 7.8%, and on par with Gemini-3.0-Pro (16.1%). Human raters are stricter than the automated evaluators and often give the two plans the same rating (in 56% of comparisons), which lowers the win rates of all systems.

4.7 Transfer to held-out benchmarks

We subsequently test whether the skills learned on TripTailor transfer to planning tasks with different structure. We evaluate on three held-out benchmarks spanning different geographic scopes, constraint types, and task formats.

TRAVELPLANNER [Xie et al., 2024] requires US-centric, multi-city itineraries over three to seven days with flights, accommodations, driving, and explicit hard constraints. Unlike TripTailor's single-city focus, this benchmark tests coordination across multiple destinations. Evaluation is purely rule-based with no LLM judge. This benchmark is challenging: Zhu et al. [2025] report that Qwen3-8B and Qwen3-32B achieve near-zero final pass rates (0% and 0.6% respectively), indicating that scale alone does not solve this task.

NATURALPLAN [Zheng et al., 2024] evaluates constraint satisfaction across three scheduling tasks: Trip Planning (sequencing city visits given direct flight connectivity and timing constraints), Meeting Planning (maximizing meetings given locations and travel times), and Calendar Scheduling (finding available slots across attendees). This benchmark tests whether TripTailor training affects general constraint reasoning beyond tool-augmented planning.

τ^2-BENCH [Barres et al., 2025] measures goal completion in simulated customer service environments where an agent engages in multi-turn dialogue with a user simulator while making tool calls to satisfy requests. We evaluate on the retail domain, which involves product returns, order modifications, and account management. This benchmark is substantially out-of-distribution: the task domain, tool

types, and interaction pattern all differ from travel planning, testing whether general tool-calling competence transfers.

Table 4 summarizes transfer performance. On TRAVELPLANNER, Anchored GRPO improves Final Pass from 1.1% to 5.6%, a fivefold increase, with hard constraint micro improving by 9.3 percentage points and hard constraint macro rising from 3.9% to 8.3%. Given that much larger models (Qwen3-8B and Qwen3-32B) achieve near-zero performance on this benchmark, this improvement is meaningful despite the low absolute numbers.

On NATURALPLAN, performance is essentially unchanged across all three tasks (differences within 0.2 percentage points). TripTailor training improves tool-augmented itinerary planning and leaves pure constraint reasoning over static context unchanged.

Table 4: Transfer to held-out benchmarks (%). Our model is trained exclusively on TripTailor; no target-domain data is used.

	Baseline	Anchored GRPO
TRAVELPLANNER		
Final Pass Rate	1.1	**5.6**
Hard Constraint Micro	53.6	**62.9**
Hard Constraint Macro	3.9	**8.3**
NATURALPLAN *(Exact Match)*		
Trip Planning	4.9	5.1
Meeting Planning	16.8	16.7
Calendar Scheduling	18.6	18.7
τ^2-BENCH *(Retail)*		
Pass@1	39.0 $\pm$3.1	**43.1** $\pm$3.5

On τ^2-BENCH, we observe a modest improvement from 39.0% to 43.1% (+4.1 points). Due to the inherent stochasticity of this benchmark (LLM-based user simulator, multi-turn interaction), we run 10 independent trials per task and report the mean with standard error. The improvement suggests some transfer of general tool-calling competence to out-of-distribution domains.

Appendix N adds two tool-planning benchmarks with tool inventories unrelated to travel and deterministic metrics, TaskBench [Shen et al., 2024] and UltraTool [Huang et al., 2024]. On the daily-life tools of TaskBench, Anchored GRPO is the only trained variant that improves edge-F1, the metric that checks dependencies between predicted tool calls, from 59.4 to 61.1. Anchored GRPO also improves all three UltraTool metrics, by 1.9 to 2.6 points.

4.8 When does anchored comparative feedback help?

Our method applies in a specific regime defined by two axes: trace availability and objective specifiability. Standard methods handle three of the four cases: SFT + RLVR (traces, verifiable), SFT + RLAIF or DPO (traces, partial), and GRPO with verifier rewards alone (outcome-only, verifiable). Constraint-only GRPO illustrates the last by reaching 88.6% validity without a judge (Table 2), though its reward-model win rate is lower (61.6% vs. 67.7% for Anchored GRPO), and so is the share of valid plans that the reward model prefers (54.5% vs. 62.0%). Our method addresses the remaining case, outcome-only with partially specifiable objectives, beating SFT by 36 points of Final Surpass and Score GRPO by 9. It degrades when no high-quality references exist (synthetic anchors transfer worse under orthogonal rubrics; Section 4.6), when the judge does not reliably correlate with quality, or when held-out tasks exercise skills the training reward never reached (Section 4.7). Our evidence comes from one training domain, and plan quality is measured mostly with LLM judges and a learned reward model. Appendix U discusses these and further limitations.

5 Conclusion

We studied anchored comparative feedback for training tool-using planning agents from outcome-only supervision. On TripTailor, anchoring GRPO against human reference plans yields a 4B-parameter model that matches Gemini-3.0-Pro on preference metrics while nearly doubling the base model's constraint satisfaction. The approach outperforms both trajectory-imputation baselines and non-anchored GRPO variants, and its improvements transfer across held-out judges, rubrics, a contrastively trained reward model, a human-rated version of the primary metric, and held-out benchmarks. These results suggest that when high-quality outcomes exist but tool traces are unavailable, optimizing to surpass reference outcomes under a comparative judge can provide an effective training signal for learning transferable tool-using planners, offering a practical alternative to synthetic trajectory construction.

References

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment, 2025. URL https://arxiv.org/abs/2506.07982.

Guiming Hardy Chen, Shunian Chen, Ziche Liu, Feng Jiang, and Benyou Wang. Humans or LLMs as the judge? a study on judgement bias. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8301–8327, Miami, Florida, USA, November 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.474. URL https://aclanthology.org/2024.emnlp-main.474/.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 6621–6642. PMLR, 21–27 Jul 2024b. URL https://proceedings.mlr.press/v235/chen24j.html.

Jade Copet, Quentin Carbonneaux, Gal Cohen, Jonas Gehring, Jacob Kahn, Jannik Kossen, Felix Kreuk, Emily McMilin, Michel Meyer, Yuxiang Wei, et al. Cwm: An open-weights llm for research on code generation with world models. *arXiv preprint arXiv:2510.02387*, 2025.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, et al. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*, 2024.

Alexander Gurung and Mirella Lapata. Learning to reason for long-form story generation. *arXiv preprint arXiv:2503.22828*, 2025.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Hui Huang, Xingyuan Bu, Hongli Zhou, Yingqi Qu, Jing Liu, Muyun Yang, Bing Xu, and Tiejun Zhao. An empirical study of LLM-as-a-judge for LLM evaluation: Fine-tuned judge model is not a general substitute for GPT-4. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Findings of the Association for Computational Linguistics: ACL 2025*, pages 5880–5895, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.306. URL https://aclanthology.org/2025.findings-acl.306/.

Shijue Huang, Wanjun Zhong, Jianqiao Lu, Qi Zhu, Jiahui Gao, Weiwen Liu, Yutai Hou, Xingshan Zeng, Yasheng Wang, Lifeng Shang, et al. Planning, creation, usage: Benchmarking llms for comprehensive tool utilization in real-world complex scenarios. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4363–4400, 2024.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Ren Lu, Thomas Mesnard, Johan Ferret, Colton Bishop, Ethan Hall, Victor Carbune, and Abhinav Rastogi. RLAIF: Scaling reinforcement learning from human feedback with AI feedback, 2024. URL https://openreview.net/forum?id=AAxIs3D2ZZ.

Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. API-bank: A comprehensive benchmark for tool-augmented LLMs. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3102–3116, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.187. URL https://aclanthology.org/2023.emnlp-main.187/.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=5PAF7PAY2Y.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=TG8KACxEON.

Junsoo Park, Seungyeon Jwa, Ren Meiying, Daeyoung Kim, and Sanghyuk Choi. OffsetBias: Leveraging debiased data for tuning evaluators. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1043–1067, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.57. URL https://aclanthology.org/2024.findings-emnlp.57/.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. Toolllm: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789*, 2023.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Vyas Raina, Adian Liusie, and Mark Gales. Is LLM-as-a-judge robust? investigating universal adversarial attacks on zero-shot LLM assessment. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7499–7517, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.427. URL https://aclanthology.org/2024.emnlp-main.427/.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36:68539–68551, 2023.

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

Yongliang Shen, Kaitao Song, Xu Tan, Wenqi Zhang, Kan Ren, Siyu Yuan, Weiming Lu, Dongsheng Li, and Yueting Zhuang. Taskbench: Benchmarking large language models for task automation. *Advances in Neural Information Processing Systems*, 37:4540–4574, 2024.

Kejian Shi, Yixin Liu, Peifeng Wang, Alexander Fabbri, Shafiq Joty, and Arman Cohan. References improve llm alignment in non-verifiable domains. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust, editors, *International Conference on Learning Representations*, volume 2026, pages 3503–3542, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/0721476d8b04a329a1c76145d328a2c5-Paper-Conference.pdf.

Lin Shi, Chiyu Ma, Wenhua Liang, Xingjian Diao, Weicheng Ma, and Soroush Vosoughi. Judging the judges: A systematic study of position bias in llm-as-a-judge, 2025. URL https://arxiv.org/abs/2406.07791.

Aman Singh Thakur, Kartik Choudhary, Venkat Srinik Ramayapally, Sankaran Vaidyanathan, and Dieuwke Hupkes. Judging the judges: Evaluating alignment and vulnerabilities in llms-as-judges. In *Proceedings of the Fourth Workshop on Generation, Evaluation and Metrics (GEM^2)*, pages 404–430, 2025.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, and Zhenzhe Ying. Information gain-based policy optimization: A simple and effective approach for multi-turn llm agents. *arXiv preprint arXiv:2510.14967*, 2025a.

Kaimin Wang, Yuanzhe Shen, Changze Lv, Xiaoqing Zheng, and Xuanjing Huang. TripTailor: A real-world benchmark for personalized travel planning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Findings of the Association for Computational Linguistics: ACL 2025*, pages 9705–9723, Vienna, Austria, July 2025b. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.503. URL https://aclanthology.org/2025.findings-acl.503/.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9440–9450, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.511. URL https://aclanthology.org/2024.acl-long.511/.

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 *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Wenyuan Xu, Xiaochen Zuo, Chao Xin, Yu Yue, Lin Yan, and Yonghui Wu. A unified pairwise framework for rlhf: Bridging generative reward modeling and policy optimization. *arXiv preprint arXiv:2504.04950*, 2025.

Zhihao Xu, Rumei Li, Jiahuan Li, Rongxiang Weng, Jingang Wang, Xunliang Cai, and Xiting Wang. Unlocking implicit experience: Synthesizing tool-use trajectories from text, 2026. URL https://arxiv.org/abs/2601.10355.

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*, 2022.

Jiayi Ye, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin-Yu Chen, Nitesh V Chawla, and Xiangliang Zhang. Justice or prejudice? quantifying biases in LLM-as-a-judge. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=3GTtZFiajM.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, YuYue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=2a36EMSSTp.

Yulai Zhao, Haolin Liu, Dian Yu, Sunyuan Kung, Meijia Chen, Haitao Mi, and Dong Yu. One token to fool llm-as-a-judge. *arXiv preprint arXiv:2507.08794*, 2025.

Huaixiu Steven Zheng, Swaroop Mishra, Hugh Zhang, Xinyun Chen, Minmin Chen, Azade Nova, Le Hou, Heng-Tze Cheng, Quoc V Le, Ed H Chi, et al. Natural plan: Benchmarking llms on natural language planning. *arXiv preprint arXiv:2406.04520*, 2024.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Xiaosen Zheng, Tianyu Pang, Chao Du, Qian Liu, Jing Jiang, and Min Lin. Cheating automatic LLM benchmarks: Null models achieve high win rates. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=syThiTmWWm.

Siyu Zhu, Yanbin Jiang, Hejian Sang, Shao Tang, Qingquan Song, Biao He, Rohit Jain, Zhipeng Wang, and Alborz Geramifard. Planner-r1: Reward shaping enables efficient agentic rl with smaller llms. *arXiv preprint arXiv:2509.25779*, 2025.

A Anchored GRPO algorithm

We provide additional details on the GRPO optimization procedure. The algorithm operates in an episodic setting where each episode consists of a planning request x sampled from the dataset $\mathcal{D}$.

A.1 Advantage computation

For a group of K trajectories $\{\tau^{(k)}\}_{k=1}^{K}$ sampled for a single request x, we compute rewards $R^{(k)} = R(\tau^{(k)})$ using the gated reward from Equation 2. GRPO forms within-group advantages by centering and normalizing:

$$A^{(k)} = \frac{R^{(k)} - \bar{R}}{\sigma_R + \delta}, \tag{3}$$
$$\bar{R} = \frac{1}{K}\sum_{j=1}^{K} R^{(j)}, \quad \sigma_R = \sqrt{\frac{1}{K}\sum_{j=1}^{K}(R^{(j)} - \bar{R})^2}.$$

Dividing by the within-group standard deviation σ_R stabilizes updates when reward variance differs across prompts, and a small constant δ avoids division by zero.

A.2 Policy update

The policy is updated using a clipped surrogate objective:

$$\mathcal{L}(\theta) = \mathbb{E}\left[\sum_{k=1}^{K} \min\left(\rho^{(k)}A^{(k)},\ \bar{\rho}^{(k)}A^{(k)}\right)\right], \tag{4}$$

where $\bar{\rho}^{(k)} = \text{clip}(\rho^{(k)}, 1-\epsilon_{\text{low}}, 1+\epsilon_{\text{high}})$ and $\rho^{(k)} = \pi_\theta(\tau^{(k)} \mid x)/\pi_{\text{old}}(\tau^{(k)} \mid x)$ is the importance ratio. Following recent practice [Yu et al., 2025], we use asymmetric clipping with $\epsilon_{\text{low}} = 0.2$ and $\epsilon_{\text{high}} = 0.25$. We omit KL regularization against a reference policy, as recent work has found it unnecessary when clipping provides sufficient stability [Copet et al., 2025]. We also avoid per-token length normalization; as noted by Liu et al. [2025], dividing loss by trajectory length incentivizes longer outputs on harder problems.

A.3 Training algorithm

Algorithm 1 Anchored GRPO

Require: Dataset $\mathcal{D}$, policy π_θ, judge J, reference plans $\{\tilde{y}_x\}$, group size K

Initialize policy π_θ from pretrained checkpoint
for iteration $t = 1$ to T **do**
  Sample batch $\mathcal{B} = \{x_j\}_{j=1}^{B} \sim \mathcal{D}$
  for each $x_j \in \mathcal{B}$ **do**
    for $k = 1$ to K **do**
      Generate trajectory: $\tau^{(k,j)} \sim \pi_\theta(\cdot \mid x_j)$ with tool execution
      Extract final plan: $y^{(k,j)} = y(\tau^{(k,j)})$
      Check constraints: $c^{(k,j)} = \mathcal{C}(\tau^{(k,j)})$
      if $c^{(k,j)} = 1$ **then**
        Query judge: $s^{(k,j)} = J(x_j, y^{(k,j)}, \tilde{y}_{x_j})$
        Compute reward: $R^{(k,j)} = r_{\text{pref}}(s^{(k,j)}) + r_{\text{rule}}(\tau^{(k,j)})$
      else
        Set reward: $R^{(k,j)} = 0$
      end if
    end for
    Compute advantages $\{A^{(k,j)}\}_{k=1}^{K}$ via within-group normalization
  end for
  Update policy: $\theta \leftarrow \theta + \eta\nabla_\theta\mathcal{L}(\theta)$
end for

B Data preparation

The TripTailor training set provides human-curated itineraries as free-form narrative text. Since our method requires structured plans for both training and evaluation, we convert these narratives to JSON using Gemini-3.0-Pro. This is a straightforward extraction task: the model structures the existing information (hotels, restaurants, attractions, transportation, timings) into our fixed JSON schema without adding or removing content. We manually verified a sample of conversions to confirm fidelity to the original plans.

C Constraint definitions

The hard-gated reward requires all constraints to pass before a trajectory receives any reward. We define two constraint categories:

Feasibility constraints verify structural validity:

- **Within Sandbox**: All entity IDs (hotels, restaurants, attractions) must exist in the TripTailor database.
- **Complete Information**: All required fields must be present (hotel for each night, meals for each day, at least one attraction per day).

Rationality constraints verify semantic correctness:

- **Budget**: Total plan cost (accommodation + meals + attractions + transportation) must not exceed the user's specified budget.
- **Restaurant Diversity (DivRest)**: No restaurant may appear more than once across the entire itinerary.
- **Attraction Diversity (DivAttr)**: No attraction may appear more than once across the entire itinerary.
- **Meal Prices**: Restaurant prices must fall within the user's requested price tier (e.g., budget, moderate, upscale).
- **Visit Duration (VisDur)**: Time allocated to each attraction must fall within its recommended duration range from the database.

A trajectory passes the constraint gate ($\mathcal{C}(\tau) = 1$) if and only if all seven sub-constraints are satisfied. During training, constraint checking is performed programmatically against the sandbox database, providing a fully reproducible and deterministic signal.

C.1 Reward terms

Table 5 lists the terms of the training reward for feasible plans. Equation 2 writes this reward with all weights divided by 7, so that r_{pref} has weight 1. GRPO normalizes rewards within each group, so this rescaling does not change the advantages. Score GRPO and Syn-Anchored GRPO use the same terms and weights. Score GRPO replaces r_{pref} with the pointwise judge score, and Syn-Anchored GRPO compares against synthetic references. Constraint-only GRPO (Appendix R) keeps the gate and r_{rule} and drops r_{pref}.

D Reward model training

We train a reward model (RM) following the architecture and procedure of Wang et al. [2025b]. The RM is a Qwen2.5-1.5B-Instruct model fine-tuned on pairwise preference data. Following TripTailor, training pairs are constructed by treating the real human itinerary as the positive sample for each query; negative samples are obtained by retrieving a different query with high TF-IDF similarity and using its corresponding itinerary. This contrastive setup teaches the RM to distinguish plans

Table 5: Terms of the training reward for feasible plans and their weights.

Term	Weight
Anchored preference r_{pref}	7
Entity accuracy	5
Route efficiency	3
Within budget	2
Transport validity	1
Activity count	1
Day coverage	1
Diverse restaurants	1
Diverse attractions	1
Reasonable meal prices	1
Visit duration	1

Table 6: GRPO training configuration.

Parameter	**Value**
Model	
Base model	Qwen3-4B-Instruct-2507
LoRA rank / alpha	64 / 128
LoRA dropout	0.05
Target modules	All attention + MLP projections
Precision	BF16
Optimization	
Learning rate	2.0e-6
Weight decay	0.01
Adam betas	(0.9, 0.95)
Global batch size	128
GRPO	
Group size (K)	8
PPO epochs	4
Gamma / Lambda	1.0 / 0.95
KL regularization	None
Advantage normalization	By std
Sequence lengths	
Max prompt length	1,792 tokens
Max response length	8,192 tokens

that match user preferences from plausible but misaligned alternatives. We reuse the code from the official TripTailor benchmark implementation[1].

To match our evaluation setting, we adapt the RM to our JSON output format. TripTailor's original RM was trained on free-form narrative plans; we instead render JSON plans to natural language using the same deterministic template applied during LLM evaluation. With this "narrativized" format, the RM sees the same textual representation as the LLM judges, and the original training procedure stays unchanged. We use identical hyperparameters to Wang et al. [2025b]: learning rate 1×10^{-5}, batch size 4, maximum sequence length 4096, weight decay 0.01, and 2 training epochs.

E Implementation details

E.1 Training configuration

Table 6 summarizes the GRPO training configuration. We use LoRA [Hu et al., 2022] for parameter-efficient fine-tuning, targeting all attention and MLP projection layers.

[1] https://github.com/swxkfm/TripTailor

Table 7: SFT training configuration.

Parameter	Value
Base model	Qwen3-4B-Instruct-2507
LoRA rank / alpha	32 / 64
Target modules	All linear layers
Precision	BF16
Learning rate	5.0e-5
LR scheduler	Cosine with 3% warmup
Weight decay	0.01
Gradient clipping	1.0
Global batch size	128
Max sequence length	5,120 tokens
Epochs	6

Table 7 summarizes the SFT configuration used for the trajectory imitation baseline.

E.2 Hardware and runtime

All experiments with Qwen3-4B-Instruct-2507 were conducted on a single node with 8 NVIDIA A100-SXM4-40GB GPUs using FSDP (Fully Sharded Data Parallel). Training the reported Anchored GRPO model took 31.4 hours on this node (251 GPU-hours). The training judge was called 22,018 times on training rollouts, since only rollouts that pass the gate are judged. SFT training completed in approximately 6 hours. Inference used vLLM with tensor parallelism.

E.3 Inference configuration

During evaluation, we use the following generation parameters:

- Engine: vLLM
- Temperature: 0.7
- Top-p: 1.0
- Repetition penalty: 1.0
- Max output tokens: 16,384

F Data processing

F.1 Output schema

All models output structured plans in the JSON schema shown in Figure 4. Plans are validated against this schema before evaluation; malformed outputs are marked as constraint failures.

F.2 Narrative conversion

TripTailor's training set provides human-curated itineraries as free-form narrative text. We convert these to structured JSON using Gemini-3.0-Pro with temperature 0.0 and a few-shot prompt containing two extraction examples. We manually verified a sample of conversions to confirm fidelity.

F.3 Plan narrativization for evaluation

For LLM judge evaluation, JSON plans are rendered to natural language via a deterministic template. Both model and reference plans are formatted identically, producing sections for accommodation, transportation, and day-by-day activities with times and prices. Judges therefore compare content and not formatting.

```
{
  "hotel": [{
    "day": int,
    "name": string,
    "price_per_night": float
  }],
  "transportation": [{
    "day": int,
    "mode": "Train" | "Flight",
    "route": string,
    "number": string,
    "time": "HH:MM-HH:MM",
    "price": float
  }],
  "itinerary": {
    "day_1": [{
      "time": "HH:MM-HH:MM",
      "action": "sightseeing" | "dining",
      "location": string,
      "price": float
    }], ...
  }
}
```

Figure 4: JSON output schema. Each plan specifies accommodations, transportation (outbound and return), and a day-by-day itinerary of activities.

G SFT baseline: trajectory imputation

The SFT baseline requires tool-call trajectories, but TripTailor provides only final plans without intermediate traces. We reconstruct trajectories by imputing the tool calls that would produce the plan's entities.

Imputation procedure. Given a final plan, we generate tool calls in a fixed order: (1) outbound transportation search, (2) attractions search, (3) accommodations search, (4) restaurants search, (5) return transportation search. Tool arguments are inferred from the plan and query metadata:

- **Transport mode**: Inferred from plan's `mode` field (Train or Flight).
- **Hotel category**: Inferred from price per night: Economy ($<$¥200), Midscale ($<$¥400), Upscale ($<$¥800), Luxury ($\geq$¥800).
- **Meal price range**: From user query's stated preference.
- **Cities**: From query metadata.

Tool responses. Tool responses are generated by querying the TripTailor sandbox database. Tool outputs include the exact entities from the reference plan (oracle items, placed first) alongside distractor items sampled uniformly at random from the same destination city. We include 4 hotel distractors, 12 attraction distractors, and 6 restaurant distractors per response. A deterministic per-sample seed based on query identifier and destination makes the imputation reproducible.

This oracle reconstruction gives SFT the best possible chance: the imputed trajectories are correct by construction and lead directly to valid plans. Despite this advantage, SFT underperforms Anchored GRPO, demonstrating the value of preference-based optimization over imitation.

H Tool definitions

Table 8 describes the five tools for querying the TripTailor sandbox database. Results are shuffled to provide variety.

All tools return results from the TripTailor sandbox CSV databases, which contain real points of interest across multiple Chinese cities. Search results are shuffled to provide variety, preventing the model from memorizing fixed orderings.

Table 8: Available tools. *Required parameters.

Tool	Parameters
`search_flights` *Flights by day of week*	origin*, destination*, departure_date, max_results
`search_trains` *Train schedules*	origin*, destination*, departure_date, max_results
`search_accommodations` *Hotels by price tier*	city*, hotel_cost, sample_num
`search_attractions` *Tourist sites & POIs*	city*, sample_num
`search_restaurants` *Dining by price/location*	city*, meal_price_min, meal_price_max, top_n

I System prompt

The following system prompt is used for both training and inference:

System Prompt

`/no_think`
You are a travel planner. Follow these steps:
Step 1: Search for transportation (outbound and return). Use `search_trains` for short distances, `search_flights` for longer distances.
Step 2: Search for hotels in the destination city
Step 3: Search for attractions
Step 4: Search for restaurants
Step 5: Output your final plan as JSON
Output Rules:
- Transportation: Exactly 2 entries (outbound Day 1, return last day)
- Hotel: Use exact name and price from search results
- Itinerary: 2-4 activities per day, use exact names from searches
- Budget: Total cost should not exceed user's budget

Output ONLY valid JSON in the specified format. No markdown or explanations.

The `/no_think` prefix disables chain-of-thought reasoning tokens, so the model emits tool calls and the final plan directly without interleaved reasoning.

J Statistical significance

We assess statistical significance of the comparisons reported in Section 4.3 using paired bootstrap resampling on the TripTailor test set ($N = 703$, $B = 10000$ resamples). For each resample, we draw 703 query indices with replacement and recompute every method's metrics on the resampled subset; pairing on query index controls for query-level variation in difficulty and reference quality. We report bias-corrected percentile 95% confidence intervals for both per-model metrics and pairwise differences, plus two-sided bootstrap p-values for the pairwise tests.

J.1 Per-model confidence intervals

Table 9 reports bootstrap 95% CIs for each method on the six headline metrics: Feasibility Macro (all feasibility sub-constraints satisfied), Rationality Macro (all rationality sub-constraints satisfied), LLM and RM preference win rates against the human reference, the composite Valid metric (Feasibility $\wedge$ Rationality), and Final Surpass (Valid $\wedge$ (LLM $\vee$ RM)).

Table 9: Per-model bootstrap 95% CIs on the TripTailor test set ($N = 703$, $B = 10000$ resamples). All values in percent. Point estimates appear in Table 1.

Model	Feas Mac.	Rat Mac.	LLM	RM	Valid	Final Surp.
Baseline	[93.9, 96.9]	[43.8, 51.2]	[41.7, 49.1]	[42.8, 50.1]	[43.0, 50.5]	[30.7, 37.7]
SFT	[96.7, 98.9]	[56.5, 63.9]	[35.8, 43.1]	[44.8, 52.2]	[56.5, 63.9]	[40.0, 47.4]
Score GRPO	[90.2, 94.2]	[96.7, 98.9]	[34.9, 42.1]	[62.7, 69.7]	[88.1, 92.5]	[67.9, 74.7]
Syn-Anch. GRPO	[97.3, 99.1]	[90.3, 94.3]	[39.5, 46.9]	[64.6, 71.6]	[89.0, 93.2]	[71.1, 77.5]
Gemini-3.0-Pro	[89.6, 93.7]	[71.4, 77.8]	[55.9, 63.2]	[40.4, 47.8]	[68.6, 75.4]	[48.9, 56.3]
Anchored GRPO	[95.2, 97.9]	[92.7, 96.0]	[57.9, 65.1]	[64.3, 71.1]	[89.5, 93.6]	[77.1, 82.9]

J.2 Pairwise comparisons across all metrics

Tables 10–14 report paired-bootstrap 95% CIs for the difference $\Delta = (\text{row}) - (\text{column})$ across the five core metrics. Significance markers indicate p-values from two-sided bootstrap tests: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$. Negative values indicate that the row method scores lower than the column method on that metric; positive values indicate the row method scores higher.

Table 10: Paired-bootstrap 95% CIs for the difference $\Delta = (\text{row}) - (\text{column})$ on **Feasibility Macro** ($N = 703$, $B = 10000$ paired resamples). Values are percentage points. Stars indicate two-sided bootstrap p: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

	SFT	Score GRPO	Syn-Anchored GRPO	Gemini-3.0-Pro	Anchored GRPO
Baseline	-2.4 [-4.3, -0.6]*	+3.3 [+1.0, +5.5]**	-2.8 [-4.6, -1.1]**	+3.7 [+1.3, +6.1]**	-1.1 [-3.0, +0.7]
SFT		+5.7 [+3.6, +8.0]***	-0.4 [-1.7, +0.9]	+6.1 [+3.8, +8.5]***	+1.3 [-0.4, +3.0]
Score GRPO			-6.1 [-8.3, -4.1]***	+0.4 [-2.4, +3.3]	-4.4 [-6.4, -2.6]***
Syn-Anchored GRPO				+6.5 [+4.3, +8.8]***	+1.7 [+0.4, +3.1]*
Gemini-3.0-Pro					-4.8 [-7.3, -2.4]***

Table 11: Paired-bootstrap 95% CIs for the difference $\Delta = (\text{row}) - (\text{column})$ on **Rationality Macro** ($N = 703$, $B = 10000$ paired resamples). Values are percentage points. Stars indicate two-sided bootstrap p: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

	SFT	Score GRPO	Syn-Anchored GRPO	Gemini-3.0-Pro	Anchored GRPO
Baseline	-12.7 [-17.5, -7.8]***	-50.4 [-54.1, -46.5]***	-44.8 [-48.6, -41.0]***	-27.2 [-31.7, -22.5]***	-46.9 [-50.8, -43.0]***
SFT		-37.7 [-41.5, -33.9]***	-32.1 [-36.0, -28.3]***	-14.5 [-19.1, -10.0]***	-34.3 [-38.1, -30.4]***
Score GRPO			+5.5 [+3.7, +7.5]***	+23.2 [+19.8, +26.6]***	+3.4 [+1.8, +5.1]***
Syn-Anchored GRPO				+17.6 [+13.9, +21.5]***	-2.1 [-4.3, +0.0]
Gemini-3.0-Pro					-19.8 [-23.3, -16.2]***

Table 12: Paired-bootstrap 95% CIs for the difference $\Delta = (\text{row}) - (\text{column})$ on **LLM win rate** ($N = 703$, $B = 10000$ paired resamples). Values are percentage points. Stars indicate two-sided bootstrap p: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

	SFT	Score GRPO	Syn-Anchored GRPO	Gemini-3.0-Pro	Anchored GRPO
Baseline	+5.8 [+1.3, +10.4]*	+6.8 [+2.7, +11.1]**	+2.1 [-2.0, +6.1]	-14.2 [-18.8, -9.5]***	-16.2 [-20.2, -12.1]***
SFT		+1.0 [-3.6, +5.7]	-3.7 [-8.3, +1.0]	-20.1 [-24.3, -15.6]***	-22.0 [-26.7, -17.4]***
Score GRPO			-4.7 [-8.5, -0.7]*	-21.1 [-25.5, -16.6]***	-23.0 [-26.9, -19.2]***
Syn-Anchored GRPO				-16.4 [-20.8, -11.8]***	-18.3 [-22.2, -14.7]***
Gemini-3.0-Pro					-2.0 [-6.5, +2.4]

Table 13: Paired-bootstrap 95% CIs for the difference $\Delta = (\text{row}) - (\text{column})$ on **RM win rate** ($N = 703$, $B = 10000$ paired resamples). Values are percentage points. Stars indicate two-sided bootstrap p: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

	SFT	Score GRPO	Syn-Anchored GRPO	Gemini-3.0-Pro	Anchored GRPO
Baseline	-2.0 [-6.5, +2.4]	-19.8 [-23.8, -15.8]***	-21.6 [-25.5, -17.8]***	+2.4 [-2.3, +7.0]	-21.2 [-25.9, -16.5]***
SFT		-17.8 [-22.5, -13.2]***	-19.6 [-24.0, -15.1]***	+4.4 [+0.1, +8.8]*	-19.2 [-23.9, -14.5]***
Score GRPO			-1.8 [-5.3, +1.6]	+22.2 [+17.6, +26.7]***	-1.4 [-5.7, +2.8]
Syn-Anchored GRPO				+24.0 [+19.6, +28.4]***	+0.4 [-3.8, +4.6]
Gemini-3.0-Pro					-23.6 [-28.2, -19.2]***

Table 14: Paired-bootstrap 95% CIs for the difference Δ = (row) − (column) on **Final Surpass** ($N = 703$, $B = 10000$ paired resamples). Values are percentage points. Stars indicate two-sided bootstrap p: $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

	SFT	Score GRPO	Syn-Anchored GRPO	Gemini-3.0-Pro	Anchored GRPO
Baseline	-9.5 [-14.2, -4.7]***	-37.1 [-41.5, -32.4]***	-40.3 [-44.2, -36.1]***	-18.5 [-23.3, -13.5]***	-45.9 [-50.2, -41.5]***
SFT		-27.6 [-32.3, -22.8]***	-30.7 [-35.3, -26.0]***	-9.0 [-13.7, -4.1]***	-36.4 [-41.0, -31.7]***
Score GRPO			-3.1 [-7.1, +0.9]	+18.6 [+13.7, +23.3]***	-8.8 [-12.9, -4.8]***
Syn-Anchored GRPO				+21.8 [+16.9, +26.5]***	-5.7 [-9.4, -2.0]**
Gemini-3.0-Pro					-27.5 [-32.0, -22.8]***

J.3 Summary of statistical findings

Anchored GRPO's Final Surpass is significantly higher than that of every other method, at $p < 0.01$ against Syn-Anchored GRPO and at $p < 0.001$ against all others. On LLM win rate, its difference to Gemini-3.0-Pro (2.0 points) is not significant. Score GRPO is significantly higher than Anchored GRPO on rationality macro (by 3.4 points) and significantly lower on LLM win rate (by 23.0 points) and on Final Surpass (by 8.8 points), all at $p < 0.001$. Under Strict Final Surpass (Table 1), Anchored GRPO is ahead of every other system at $p < 0.001$, and its margins over Score GRPO and Syn-Anchored GRPO grow from 8.8 and 5.7 points to 13.9 and 11.9 points.

K Detailed constraint analysis

We provide detailed breakdowns of constraint satisfaction to complement the aggregate results in the main paper.

K.1 Sub-constraint failure rates

Table 15 shows the percentage of the 703 test samples failing each individual constraint. The baseline struggles most with visit duration (33.1% failure rate) and restaurant diversity (22.3% failure rate). These two constraints account for the majority of rationality failures. Anchored GRPO reduces failure rates across all rationality sub-constraints, with the largest absolute improvements on visit duration (reduced from 33.1% to 2.1%) and restaurant diversity (reduced from 22.3% to 0.9%).

Table 15: Sub-constraint failure rates (% of 703 samples failing each constraint). Lower is better. Best results per constraint in **bold**.

Constraint	Baseline	SFT	Gemini-3.0-Pro	Score GRPO	Syn-Anchored	Anchored
Within Sandbox	4.6	2.1	8.3	7.8	**1.7**	3.4
Complete Information	**0.1**	0.6	1.4	**0.1**	0.3	0.9
Budget Limit	11.4	10.8	3.6	**1.0**	2.7	3.3
Diverse Restaurants	22.3	6.5	9.7	**0.1**	0.4	0.9
Diverse Attractions	8.3	15.5	1.7	**0.1**	**0.1**	0.6
Reasonable Meal Prices	2.1	0.7	2.1	**0.1**	0.4	0.4
Visit Duration	33.1	20.9	16.4	**1.3**	5.1	2.1
Feasibility (aggregate)	4.6	2.1	8.3	7.8	**1.7**	3.4
Rationality (aggregate)	52.5	39.8	25.3	**2.1**	7.7	5.5

Score GRPO achieves the lowest failure rates on most individual constraints but exhibits elevated sandbox failures (7.8%), suggesting that aggressive constraint optimization can sometimes lead to invalid entity references. Anchored GRPO achieves a better balance, with low failure rates across all constraints while maintaining higher feasibility than Score GRPO.

K.2 Constraint success heatmap

Figure 5 visualizes sub-constraint success rates across all models as a heatmap. The baseline struggles most with visit duration (66.9% success) and restaurant diversity (77.7% success), while every GRPO variant passes each sub-constraint on at least 92% of samples. Score GRPO achieves the highest individual constraint success rates but shows slightly elevated sandbox failures, while Anchored

GRPO maintains strong performance across all constraints with better balance between feasibility and rationality.

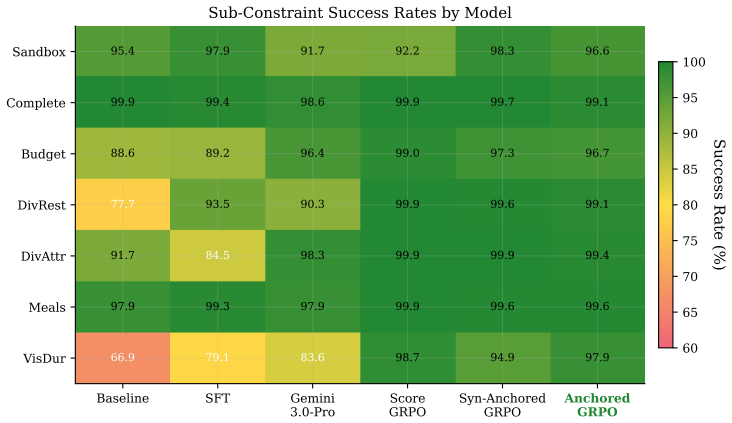


Figure 5: Sub-constraint success rates by model. Darker green indicates higher success rates. The baseline struggles most with visit duration and restaurant diversity, while every GRPO variant passes each sub-constraint on at least 92% of samples.

L Preference margin analysis

We analyze the margins of preference wins to assess whether improvements are decisive or concentrated at decision boundaries.

L.1 Score margins

Table 16 reports statistics on LLM and RM score margins, computed as the model's score minus the reference's score. For LLM margins, scores are on a 1 to 5 scale averaged across both judges and presentation orderings. The RM outputs unbounded logits, so absolute values are not directly interpretable, but relative comparisons are meaningful. Positive margins indicate model preference; negative margins indicate reference preference.

Table 16: Score margin statistics across all 703 samples (model minus reference). Higher mean/median indicates stronger preference for model outputs. The RM outputs unbounded logits; relative comparisons are meaningful.

	LLM judge margin					Reward model margin				
Model	Mean	Median	Std	Min	Max	Mean	Median	Std	Min	Max
Baseline	0.04	0.00	0.80	−1.75	2.00	−0.33	−0.25	4.08	−14.5	16.8
Gemini-3.0-Pro	0.24	0.25	0.65	−2.25	2.00	−0.57	−0.40	4.01	−12.2	12.9
Score GRPO	−0.21	−0.25	0.88	−3.33	2.00	1.92	1.86	4.94	−15.1	19.1
Syn-Anchored GRPO	−0.07	−0.10	0.89	−3.20	2.38	2.00	1.80	4.56	−13.6	16.2
Anchored GRPO	**0.32**	**0.50**	0.79	−3.00	2.00	**2.97**	**2.47**	6.02	−12.0	21.0

Anchored GRPO achieves the highest mean margin for both LLM (0.32) and RM (2.97) evaluations.

L.2 Win rate at margin thresholds

Table 17 shows LLM win rates at increasingly strict margin thresholds. A margin greater than 0 indicates any win; greater than 0.5 indicates a moderate win (half-point advantage); greater than 1.0 indicates a decisive win (full point difference on the 1 to 5 scale).

Anchored GRPO maintains the highest win rate at all thresholds. Its plan leads the reference by more than 0.5 points in 38.5% of comparisons and by more than 1.0 point in 16.8%, the highest rates among all methods.

Table 17: LLM win rates (%) at different margin thresholds. Higher thresholds indicate more decisive wins.

Model	>0	>0.25	>0.5	>1.0
Baseline	45.4	35.6	27.5	10.4
Gemini-3.0-Pro	59.6	46.2	33.3	8.3
Score GRPO	38.5	29.0	23.3	11.0
Syn-Anchored GRPO	43.2	35.8	27.0	14.2
Anchored GRPO	**61.6**	**50.9**	**38.5**	**16.8**

L.3 Margin distribution

Figure 6 shows the distribution of score margins for the main models using kernel density estimation. Panel (a) shows LLM judge margins on the 1–5 scale: Anchored GRPO exhibits the most positive shift with mean margin $+0.32$. Panel (b) shows reward model margins: Anchored GRPO shows a clear rightward shift compared to Baseline and Gemini-3.0-Pro, with a mean margin of $+2.97$. The vertical dashed line indicates the decision boundary where model and reference scores are equal.

M Additional stratification analysis

We provide additional stratification analyses to complement the main paper results.

M.1 Surpass rate among valid samples

Table 18 restricts analysis to valid samples only, showing the conditional surpass rate (Final Surpass divided by Valid count) stratified by query characteristics. This isolates preference performance from constraint satisfaction.

Table 18: Conditional surpass rate among valid Anchored GRPO samples. Surpass Rate = Final Surpass / Valid.

Segment	Valid	Final Surpass	Surpass Rate
By trip length			
2 days	98	84	85.7%
3 days	219	196	89.5%
4 days	183	159	86.9%
5 days	109	94	86.2%
6–7 days	35	30	85.7%
By budget tercile			
$<$¥3,400	204	173	84.8%
¥3,400–5,400	219	193	88.1%
$\geq$¥5,400	221	197	89.1%

Conditional surpass rates among valid samples are consistent (84.8% to 89.5% across all segments), indicating that once plans satisfy constraints, preference performance is stable regardless of trip length or budget level.

N Extended held-out evaluation

We extend Section 4.7 to a five-method comparison on TravelPlanner and NaturalPlan, and to two additional tool-planning benchmarks beyond Table 4: TaskBench daily-life APIs [Shen et al., 2024] and UltraTool English [Huang et al., 2024]. All inference uses the trained checkpoints from Section 4.2. Statistical significance is reported as paired-bootstrap standard error of the per-instance metric difference vs. Baseline ($B = 10000$ resamples).

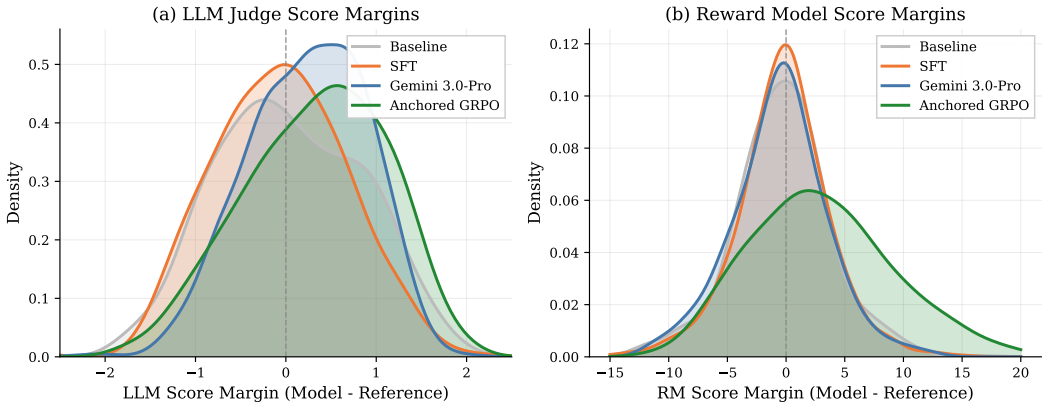


Figure 6: Score margin distributions. (a) LLM judge margins: Anchored GRPO shows the most positive shift (mean $+0.32$). (b) Reward model margins: Anchored GRPO shows the largest positive shift (mean $+2.97$). The dashed line indicates the decision boundary.

N.1 All methods on TravelPlanner and NaturalPlan

Tables 19 and 20 extend the two held-out benchmarks of Table 4 to all five methods. On TRAVELPLANNER, Anchored GRPO retains the highest Final Pass Rate at 5.6%; Score GRPO follows at 3.6%, Syn-Anchored at 2.4%, and SFT collapses to 0.0% because it emits TripTailor entity names instead of TravelPlanner-tool-returned entities, driving sandbox-validity to zero and propagating to zero on every hard constraint. On NATURALPLAN, the three GRPO variants stay within ± 1.5 points of Baseline on all three tasks, while SFT regresses on Calendar Scheduling by about 6 points. Since the GRPO variants stay close to Baseline, TripTailor training improves *tool-augmented* itinerary planning specifically and does not extend to constraint reasoning over static context.

Table 19: TravelPlanner test set (%). HC = hard constraints.

	HC		Final
Method	Micro	Macro	Pass
Baseline	53.6	3.9	1.1
SFT (imputed)	0.0	0.0	0.0
Score GRPO	51.6	6.1	3.6
Syn-Anchored GRPO	54.2	6.2	2.4
Anchored GRPO	**62.9**	**8.3**	**5.6**

Table 20: NaturalPlan exact match (%). Trip $n = 1600$; Meeting and Calendar $n = 1000$.

Method	Trip	Meeting	Calendar
Baseline	4.9	16.8	18.6
SFT (imputed)	2.8	16.0	12.3
Score GRPO	5.6	16.2	18.8
Syn-Anchored GRPO	5.7	16.7	19.2
Anchored GRPO	5.1	16.7	18.7

N.2 TaskBench daily-life APIs

TaskBench [Shen et al., 2024] evaluates LLMs on task automation through three stages: task decomposition, tool selection, and parameter prediction. We use the daily-life APIs domain (40 productivity tools spanning weather, music, calendar, shopping, and similar APIs; $n = 4318$). Models receive a free-form user request and a description of the available tools, and emit a JSON dictionary containing both `task_nodes` (which tools to invoke with which arguments) and `task_links` (the dependency graph indicating which tool's output feeds which tool's input). Following Shen et al. [2024] we report node-F1 over the predicted set of tool names, argument-F1 over predicted (tool, argument-name, argument-value) triples, and edge-F1 over the predicted dependency edges. Edge-F1 is the metric most directly testing whether the model sequences tool calls with correct data flow. No TaskBench data was used for training.

Table 21 reports results. Anchored GRPO is the only trained variant that improves edge-F1 over Baseline, by 1.7 points ($p < 0.001$). Score GRPO and Syn-Anchored GRPO are below Baseline on the same metric, by 0.8 and 2.3 points. SFT's argument-F1 drops from 76.55 to 2.07 while node-F1 stays at 95.89. The model selects roughly the right tools, but most of its responses write the arguments in a format that does not follow the TaskBench output schema, so the scorer cannot read them.

Table 21: TaskBench daily-life APIs ($n = 4318$). Δ edge-F1 is the paired per-instance difference vs. Baseline.

Method	node-F1	arg-F1	edge-F1	parse-ok	Δ edge-F1 $\pm$ SE
Baseline	**96.49**	**76.55**	59.44	99.93	(reference)
SFT (imputed)	95.89	2.07	37.54	99.98	-21.90 ± 0.57
Score GRPO	96.34	76.32	58.68	99.91	-0.76 ± 0.20
Syn-Anchored GRPO	96.28	76.33	57.10	99.91	-2.34 ± 0.27
Anchored GRPO	96.35	76.43	**61.11**	99.91	$\mathbf{+1.66 \pm 0.27}$

N.3 UltraTool English

UltraTool [Huang et al., 2024] evaluates LLMs on real-world tool-utilization scenarios spanning planning, creation, and usage; we use the English split test set ($n = 1000$, with 1–7 tools per instance and per-instance argument-JSON schemas). The model receives a user question and a list of tools and produces a hierarchical plan with top-level steps and sub-steps that are either `No tool required` or concrete `tool_name(arg='value', ...)` invocations. We report tool-set-F1 over the multiset of invoked tool names, sequence match (LCS over the predicted vs. gold tool-call sequence normalized by max length), and argument score (per-key normalized Levenshtein similarity averaged across argument keys, with greedy positional matching by tool name). All three metrics are deterministic and no UltraTool data is used for training.

Table 22 reports results. Anchored GRPO improves over Baseline on all three metrics, by 1.9 points on tool-set-F1, 2.6 on sequence match and 2.6 on argument score ($p < 0.001$ for each). Score GRPO and Syn-Anchored GRPO stay within two standard errors of Baseline on every metric. SFT has the highest tool-set-F1 (89.33) and sequence match (83.39), so it has learned the multi-step plan format, but its argument score drops to 16.12, 33 points below Baseline. This mirrors the TaskBench argument-F1 drop on a different benchmark and tool inventory.

Table 22: UltraTool English ($n = 1000$). Paired $\Delta \pm$ SE rows are per-instance differences vs. Baseline.

Method	tool-set-F1	seq-match	arg-score
Baseline	86.64	79.65	49.34
SFT (imputed)	**89.33**	**83.39**	16.12
Score GRPO	86.96	80.08	49.94
Syn-Anchored GRPO	86.29	78.94	48.20
Anchored GRPO	88.50	82.28	**51.96**
paired $\Delta \pm$ SE *vs. Baseline*			
SFT	$+2.69 \pm 0.65$	$+3.74 \pm 0.85$	-33.22 ± 1.38
Score GRPO	$+0.33 \pm 0.32$	$+0.44 \pm 0.42$	$+0.60 \pm 0.66$
Syn-Anchored GRPO	-0.35 ± 0.37	-0.71 ± 0.49	-1.15 ± 0.76
Anchored GRPO	$\mathbf{+1.87 \pm 0.38}$	$\mathbf{+2.63 \pm 0.51}$	$\mathbf{+2.61 \pm 0.73}$

N.4 SFT on held-out tools

A consistent SFT failure mode appears across the extended evaluation. On TravelPlanner (Table 19) Final Pass is 0.0%, with the model emitting TripTailor entity names that drive sandbox-validity to zero. On TaskBench daily-life APIs (Table 21) argument-F1 drops by 74 points while node-F1 stays near baseline, because most responses do not follow the argument schema. On UltraTool English (Table 22) argument score drops by 33 points while tool-set-F1 reaches the highest value among all methods. SFT retains tool-name selection across all three benchmarks but does not follow the entity and argument formats of the held-out tools.

O Trajectory analysis

We analyze tool-calling behavior to understand how Anchored GRPO achieves improved constraint satisfaction. The trajectories in this section come from a separate inference run of the Baseline

and Anchored GRPO models with a different system prompt, so their validity differs slightly from Table 1.

O.1 Tool usage statistics

Table 23 compares tool usage between models. Anchored GRPO makes approximately 51% more tool calls on average, with a similar spread (standard deviation 1.13 against 1.15).

Table 23: Tool usage statistics across all 703 test samples.

Model	N	Mean	Std	Min	Max
Baseline	703	5.78	1.15	0	10
Anchored GRPO	703	8.73	1.13	0	12

Table 24 gives the token counts per plan, measured with the Qwen3 tokenizer. Generated tokens include the tool-call blocks, and the final context is the full conversation at the end of the trajectory without the tool schemas.

Table 24: Tokens per plan, averaged over 703 test samples.

	Baseline	Anchored GRPO	Ratio
Tool calls	5.78	8.73	1.51
Generated tokens	1,015	1,185	1.17
Final context	2,357	2,917	1.24

O.2 Tool frequency by type

Table 25 shows the total number of calls to each tool type across all test samples. Anchored GRPO substantially increases usage of search_trains (+186%), search_attractions (+74%), and search_restaurants (+67%), which reflects more exploration of the options.

Table 25: Total tool calls by type across 703 test samples. A. GRPO = Anchored GRPO.

Tool	Baseline	A. GRPO	Change
search_flights	1342	1278	−5%
search_trains	488	1397	+186%
search_accommodations	706	855	+21%
search_attractions	817	1422	+74%
search_restaurants	713	1188	+67%
Total	4066	6140	+51%

O.3 Validity rate by tool call count

Table 26 shows validity rates stratified by the number of tool calls in each trajectory. Most baseline trajectories use 5 to 8 calls, and their validity stays near 50% in both ranges (51.5% and 46.6%). Almost all Anchored GRPO trajectories use 7 or more calls, with 89.9% validity at 7 to 8 calls and 93.5% at 9 or more. Anchored GRPO learned to use the additional calls, while more calls alone do not help the baseline.

O.4 Turn distribution

Anchored GRPO converges to a consistent 4-turn interaction pattern (97.0% of samples), while the baseline shows more variance, with the majority at 2 or 3 turns. This suggests that Anchored GRPO learned a systematic planning approach: initial search, refinement via additional tool calls, and finalization.

Table 26: Validity rate by tool call count.

	Baseline		Anchored GRPO	
Tool Calls	Count	Valid %	Count	Valid %
$\leq$4	3	0.0	2	0.0
5–6	472	51.5	5	100.0
7–8	223	46.6	298	89.9
$\geq$9	5	60.0	398	93.5

Table 27: Distribution of conversation turns (assistant responses) per trajectory.

Turns	Baseline	Anchored GRPO
1	2 (0.3%)	2 (0.3%)
2	394 (56.0%)	0 (0.0%)
3	228 (32.4%)	17 (2.4%)
4	79 (11.2%)	682 (97.0%)
5	0 (0.0%)	2 (0.3%)

P Evaluation rubrics

We provide the complete prompts for all evaluation rubrics used in our experiments. All rubrics use a 1 to 5 scoring scale. Plans are rendered via a deterministic JSON-to-narrative template before presentation, so formatting is identical across comparisons.

P.1 TripTailor training rubric

This is the rubric from Wang et al. [2025b], used during GRPO training with Gemini-2.5-Flash-Lite as the judge.

TripTailor Personalization Rubric

Evaluation Criteria and Key Factors:
Experiences: Consider both variety and depth. While a diverse range of activities is beneficial, immersive and well-planned experiences that align closely with traveler interests should also be recognized.
Itinerary Intensity: Evaluate how well the plan matches the traveler's desired itinerary intensity (e.g., relaxed, moderate, packed). Consider the balance between activities and free time, as well as the pacing of the trip.
Cuisine: Assess the suitability of dining choices to the traveler's stated preferences, including cuisine category and alignment with budget and meal price range.
Accommodations: Evaluate the quality, comfort, and overall fit with the traveler's stated preferences, including accommodation category and budget range.
Transportation: Assess the practicality of transportation options with a focus on departure and return times, convenience, cost, and suitability for the traveler's preferences.
Total Budget Consideration: Staying within the budget is essential, but an itinerary that justifies slightly higher costs through premium experiences is viewed positively, whereas strict cost-cutting at the expense of premium experiences is seen as unfavorable.
Scoring: 5 = exceeds expectations, perfectly aligned; 4 = largely meets needs, minor gaps; 3 = partially satisfies, lacks depth; 2 = barely meets expectations, significant gaps; 1 = fails to address query entirely.

P.2 Paraphrased rubric

This rubric assesses the same underlying dimensions as the TripTailor training rubric but uses user-centric framing and no technical constraint language.

Paraphrased Rubric

You are comparing two travel itineraries for the same trip request. Evaluate which plan better meets the traveler's needs.
Preference Alignment: Do the selected attractions, dining venues, and activities match what the traveler specifically asked for? Consider stated interests, cuisine preferences, and activity types.
Practical Fit: Are the hotel category and price appropriate? Are transportation choices sensible for the route and budget? Is the overall cost reasonable given the stated budget?
Schedule Quality: Does the daily pacing match the traveler's stated intensity preference (relaxed/moderate/packed)? Are time allocations realistic for each activity?
Overall Value: Considering the budget, does the plan deliver good value? Does it make smart tradeoffs, or does it waste money on some things while skimping on others?
Scoring: 5 = excellent, thoughtfully addresses all preferences; 4 = good, meets most needs well; 3 = adequate, covers basics; 2 = weak, misses important preferences; 1 = poor, fails to address needs.

P.3 Balanced rubric

This rubric takes a deliberately orthogonal approach. It replaces the domain-specific criteria (cuisine, accommodations, transportation) with general plan quality dimensions.

Balanced Rubric

Compare two travel itineraries for the same trip request. Determine which plan would result in a better travel experience.
Preference Alignment: Does the plan address what the traveler specifically asked for? Consider their stated interests, budget tier, dining preferences, and activity types.
Logical Flow: Do activities connect sensibly? Consider geographic proximity, time-of-day appropriateness, and natural progression through the trip.
Completeness: Does the plan feel finished and ready to execute? Are there clear details for accommodation, transportation, and activities? No unexplained gaps?
Overall Quality: Stepping back, which plan would you recommend to a friend? Which delivers a more satisfying trip experience?
Scoring: 5 = excellent, would enthusiastically recommend; 4 = good, solid plan; 3 = adequate, acceptable; 2 = weak, notable problems; 1 = poor, would not recommend.

Q Human evaluation

We ran a human evaluation of three systems on the full TripTailor test set: the baseline, Gemini-3.0-Pro and Anchored GRPO. This gives 2,109 comparisons. Seven professional annotators, contracted through a commercial annotation vendor, rated them on an annotation platform. The study was approved through an internal review at Booking.com. Each comparison showed a model plan and the human reference plan for the same request. Both plans were unlabeled, appeared in random order and were rendered from their JSON by the same template. Raters scored each plan from 1 to 5 using the six criteria and the scale of the TripTailor rubric (Appendix P). As for the LLM judges, a model plan wins only if its rating is strictly higher than the reference's, and Final Surpass (human) is Valid $\wedge$ human win.

The Anchored GRPO plans for this study come from an inference run made during development, with one additional sentence in the system prompt. They are valid on 79.9% of requests, against 91.6% for the plans in Table 1. Validity in Table 28 is computed on the plans shown to raters, so it differs slightly from Table 1 for all three systems.

Table 28: Human evaluation on the TripTailor test set (703 requests per system).

System	Valid	Human win	Final Surpass (human)
Baseline (Qwen3-4B)	46.5	18.2	7.8
Gemini-3.0-Pro	73.1	23.9	16.1
Anchored GRPO	79.9	21.6	16.9

Under Final Surpass (human), Anchored GRPO more than doubles the baseline ($p < 0.001$, paired bootstrap), and the difference from Gemini-3.0-Pro is not significant ($p = 0.68$). As in Section 4.5,

most of the gain over the baseline comes from validity. The comparison with Gemini-3.0-Pro is indirect, since both systems were rated against the same references and not against each other.

Human raters gave the model plan and the reference the same rating in 56% of the comparisons, and a tie does not count as a win, which lowers human win rates for every system. Each comparison was rated by one annotator, so we cannot report agreement between raters. Raters gave slightly higher ratings to the plan shown first. The model plan was shown first in 369 of the 703 requests, with the same order for all three systems, so this changes the reported win rates by less than half a point.

R Additional baselines

Final-plan-only SFT fine-tunes the base model to write the reference plan directly from the request, without tool calls. **Best-of-N with a constraint filter** draws N trajectories from the base model and keeps one that passes all constraints, if there is one. **LLM reranking** scores the same candidates with an LLM judge and keeps the one with the highest score. **Anchored DPO** samples candidate trajectories from the base model through the same tool harness, scores them with the training reward and builds preference pairs between their final plans. **Constraint-only GRPO** trains like Anchored GRPO, with the same gate and rule-based terms and without the preference term.

For Anchored DPO we evaluated nine configurations that vary β, the learning rate, the number of epochs and the data size. Table 2 reports the configuration with the highest validity, retrained on 1,800 pairs. The other configurations, and three rounds of an iterated on-policy variant, reach between 47.8% and 57.9% validity. With the constraint filter, best-of-N reaches 55.2%, 60.3% and 64.6% validity at $N = 2$, 4 and 8. This is the share of requests where at least one of the N candidates is valid, so no selection rule over these candidates can reach a higher validity.

These runs were judged with a single call to the training judge, so we do not compare their LLM win rates with Table 1. Table 29 gives the constraint metrics, which use the checker of Table 1. Constraint-only GRPO, the closest baseline, is significantly below Anchored GRPO on all three measures of Table 2 ($p \leq 0.02$, paired bootstrap).

Table 29: Full metrics for the additional baselines (703 requests).

	Feasibility		Rationality				
Method	Micro	Macro	Micro	Macro	Valid	RM win	Valid∧RM
Final-plan-only SFT	47.7	0.0	38.0	0.0	0.0	19.1	0.0
Best-of-8, constraint filter	97.1	94.6	90.8	66.3	64.6	62.3	42.8
LLM reranking, $N = 2$	96.1	92.3	86.7	49.9	48.2	52.5	28.2
LLM reranking, $N = 4$	96.2	92.7	86.1	48.9	47.5	53.1	28.6
LLM reranking, $N = 8$	96.6	93.2	86.0	47.9	46.1	53.9	28.7
Anchored DPO	96.0	92.2	92.9	68.7	64.2	57.5	39.3
Constraint-only GRPO	95.7	92.7	98.5	94.6	88.6	61.6	54.5
Anchored GRPO	97.9	96.6	98.5	94.5	91.6	67.7	62.0

S Other base models

We train Qwen3-8B and Gemma-4-E4B-it with the same recipe as the main model, on 4 H100 GPUs. Table 30 gives the results. Gemma-4-E4B-it starts with the same validity as Qwen3-4B-Instruct-2507 and a higher reward-model win rate, and training raises both. Qwen3-8B predates the 2507 release of Qwen3-4B and starts lower on validity. Training raises its validity by 21.5 points, and its reward-model win rate stays close to that of the untrained model. As for the baselines of Appendix R, these runs were judged by the training judge only, so we do not report LLM win rates.

T Open-weights judges

We re-score the plans of Table 1 with Llama-3.3-70B and Mistral-Small-3.2, accessed through OpenRouter at temperature 0, with the TripTailor rubric and both presentation orders. As in Table 1,

Table 30: Anchored GRPO on other base models (703 requests).

Base model		Feas. Macro	Rat. Macro	Valid	RM win	Valid∧RM
Qwen3-4B-Instruct-2507	Untrained	95.4	47.5	46.7	46.5	24.5
	Anchored GRPO	96.6	94.5	91.6	67.7	62.0
Gemma-4-E4B-it	Untrained	78.5	53.1	46.7	60.9	34.0
	Anchored GRPO	94.0	88.6	84.6	81.2	68.3
Qwen3-8B	Untrained	46.5	57.9	35.4	45.0	15.9
	Anchored GRPO	68.3	76.5	56.9	43.4	26.3

a sample counts as a win if the model's average score strictly exceeds the reference's. We leave out the few requests where one ordering could not be scored (at most five per system). Table 3 in Section 4.6 gives the win rates. Both open judges are stricter than GPT-4o and Claude-4.5-Haiku for every system, and both rank Anchored GRPO first and Gemini-3.0-Pro second.

U Limitations

We train on one domain, travel planning on TripTailor, with base models of 4B to 8B parameters. The method needs reference outcomes and programmatic constraint checks, and it calls an LLM judge during training. Plan quality is mostly measured with LLM judges and a learned reward model. We check the ranking with two open-weights judges (Appendix T) and with a human evaluation (Appendix Q). Data conversion, the synthetic anchors and the training judge use closed models. On held-out benchmarks such as TravelPlanner, absolute success rates remain low for all systems we tested.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims in the abstract and introduction are supported by the experimental results; the scoping subsection (Section 4.8) explicitly defines the regime in which they apply.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 3.5 discusses LLM-judge biases our protocol mitigates but cannot fully eliminate and explains the caveat of non-identified biases; Section 4.8 characterizes the regime where the method applies and where it degrades. Appendix U lists further limitations of the method and of our evaluation.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper presents an empirical contribution; no theorems or formal proofs are claimed.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and crossreferenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Algorithm 1 specifies the training procedure; Appendix C.1 gives the reward weights; Appendix E provides full hyperparameters; Appendix P provides exact judge prompts; Appendix G documents the SFT baseline; the TripTailor benchmark and base model (Qwen3-4B-Instruct-2507) are publicly available.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: We release the trained LoRA adapters. Section 3 and Appendices A, C.1, E and P describe the algorithm, reward, training configuration and judge prompts. All evaluation datasets and the base model are publicly available.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4.1 and Section 4.2 describe data splits, baselines, and evaluation protocol; Appendix E provides full training and inference configurations including optimizer, learning rate, batch size, and sequence lengths.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: Appendix J reports paired-bootstrap 95% CIs and two-sided p-values for the comparisons of Table 1. Appendices N, Q and R report significance for the held-out benchmarks, the human evaluation and the additional baselines. τ^2-BENCH numbers in Table 4 are mean $\pm$ standard error over 10 independent trials per task.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix E reports hardware (8$\times$ A100-SXM4-40GB), runtime (31.4 hours, or 251 GPU-hours, for the reported GRPO model; 6 hours for SFT), and inference setup. The runs on other base models in Appendix S use 4 H100 GPUs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses publicly available benchmarks and pretrained models. The human evaluation (Appendix Q) was carried out by professional annotators contracted through a commercial vendor. The work conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This is foundational methodological work on training tool-using agents; we are not aware of direct societal impacts beyond those general to tool-using LLM research.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The released artifact is a LoRA adapter for a 4B-parameter model for travel-itinerary planning; it does not pose elevated misuse risk relative to its base model.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We cite the TripTailor benchmark [Wang et al., 2025b], TravelPlanner [Xie et al., 2024], NaturalPlan [Zheng et al., 2024], τ^2-BENCH [Barres et al., 2025], TaskBench [Shen et al., 2024], UltraTool [Huang et al., 2024], the Qwen3-4B-Instruct-2507 base model, and GRPO [Shao et al., 2024]; all are used under their respective open-source licenses.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The trained LoRA adapters are released with a model card that documents the base model, the training data and the evaluation setup.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [Yes]

Justification: Appendix Q describes the human evaluation, the information shown to raters, and the rubric they used, which is the TripTailor rubric reproduced in Appendix P. Raters were professional annotators recruited and compensated through a commercial annotation vendor.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [Yes]

    Justification: Raters compared travel itineraries and were exposed to no sensitive content and no personal data, so the study posed minimal risk to them. The study was approved through an internal review at Booking.com and carried out by an external annotation vendor (Appendix Q).

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLMs are central to the method and evaluation. Section 3.5 and Section 4.1 document the judges used at training and evaluation time (Gemini-2.5-Flash-Lite for training-time comparative feedback; GPT-4o and Claude-4.5-Haiku as held-out evaluation judges; Llama-3.3-70B and Mistral-Small-3.2 as additional open-weights judges in Appendix T); Section 4.2 and Appendix B document the use of Gemini-3.0-Pro for narrative-to-JSON conversion of the TripTailor training set and as the source of synthetic anchors for the Syn-Anchored GRPO baseline. Full judge prompts appear in Appendix P. LLMs were not used as a non-trivial component of paper writing.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.