PROJOPD: CONFLICT PROJECTION FOR ON-POLICY DISTILLATION IN AGENTIC REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Agentic reinforcement learning typically provides only a single scalar reward at the end of an episode, so the many intermediate decisions within a trajectory all share one signal. Recent work supplements this sparse supervision with on-policy self-distillation, but existing approaches mainly study how to construct the teacher signal or how to weight individual tokens, and pay little attention to *whether the auxiliary update is compatible with the task objective*. We identify an optimization-level conflict between the two: a context teacher expresses nothing more than the local preference of the same policy under an additional prompt, and its gradient can have a negative inner product with the policy gradient; adding the two directly then increases the RL objective to first order, i.e. the auxiliary signal cancels the descent direction dictated by the environment reward. We therefore propose PROJOPD, which builds both logit gradients explicitly on a shared sparse support, removes from the distillation gradient the component opposing the policy gradient within each environment turn, retains only its orthogonal part, and back-propagates through a detached surrogate. The projection is applied per environment turn—a turn corresponds to one complete environment action, whereas deciding token by token ignores how the tokens of a single action jointly point. The construction guarantees to first order that the auxiliary signal cannot cancel the descent direction of the policy gradient, and requires no additional rollouts. In addition, we propose a cheaper teacher construction: the teacher is simply the current policy conditioned on a plan context that the policy itself generates; it only re-scores the tokens the student has already sampled and never re-executes the environment. At test time the planner and the teacher are removed, so inference cost matches an ordinary actor. We evaluate PROJOPD on ALFWorld, WebShop and Search-QA with Qwen2.5-3B and 7B, where it attains the best result on all three benchmarks at 7B; ablations attribute the gains to the conflict projection and to the turn-level projection granularity.

1 INTRODUCTION

Large language models are shifting from passive text generators to agents that perceive an environment, call tools and carry out long-horizon tasks, where early mistakes accumulate over later steps. Reinforcement learning lets the model optimize the final task return directly and has become a central route to improving agent capability, but agentic environments usually provide only a trajectory-level reward that fires when the task succeeds or fails: the many intermediate reasoning steps and actions of one episode share a single sparse signal, leaving the policy unable to tell which local decisions led to success. Recent work addresses this through on-policy self-distillation — collecting trajectories with the current policy, forming a teacher from hindsight skills or planning hints, and using it to supervise the same trajectory at the token or turn level. Because teacher and student operate on the same sampled tokens, such methods avoid the distribution mismatch of offline trajectories; SDAR, RLSD and StepOPSD demonstrate this route through token-level gating, gap-based modulation and step-aware distillation (Lu et al., 2026; Yang et al., 2026; Zhang et al., 2026).

A dense teacher signal, however, does not necessarily produce a correct policy update. Agentic RL ultimately optimizes the environment return, whereas a context-augmented teacher expresses

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

the model's local preference under additional planning information. The two may point in different directions on a given turn. If training simply adopts

$$\mathcal{L} = \mathcal{L}_{\mathrm{RL}} + \lambda \mathcal{L}_{\mathrm{OPD}}, \tag{1}$$

the gradients of both objectives are summed unconditionally. A large OPD gradient can weaken or even cancel the update supplied by the environment return, while simply lowering λ suppresses all of the useful distillation signal at the same time. Existing methods mainly study how to construct the teacher or how to weight individual tokens, and pay little attention to *whether the OPD update is compatible with the outcome-driven policy gradient*.

Our core insight is that whether an auxiliary distillation signal should be adopted depends not only on its magnitude but on its geometric relation to the update direction dictated by the environment return. The two signals fail in incommensurable ways. A trajectory-level reward is *sparse and high-variance* — one scalar spread over a whole trajectory, which is exactly what dense supervision is meant to fix — but it is defined by the task objective and its policy-gradient estimate stays unbiased on-policy. A context teacher is instead *biased*: it is the same policy under an additional prompt, introducing no information independent of that policy, so nothing ties its local preferences to the task return. Prior work likewise keeps RL as the backbone and confines distillation to a controlled auxiliary term (Lu et al., 2026; Zhang et al., 2026). Hence our principle: **when the two conflict on a turn, defer to the environment return, and confine the auxiliary signal to the subspace that does not compete with that direction.** This is a *conservative* trade-off rather than a claim that a single batch's policy gradient is always right; the projection removes only the conflicting component and keeps the orthogonal one, so the teacher's information is not discarded wholesale.

Our main contributions are as follows:

- **Conflict projection.** We formalize the optimization conflict between the outcome-driven policy gradient and planner-conditioned OPD, and remove from the distillation gradient the component opposing the policy gradient. Proposition 1 guarantees the auxiliary gradient cannot cancel the descent direction of RL to first order; ablations put the design at $+3.3$ points ($86.7 \rightarrow 90.0$).
- **Turn-level granularity.** The projection is applied per environment turn rather than per token: deciding token by token ignores how the tokens of one action jointly point, while aggregating over an episode lets unrelated conflicts cancel.
- **Online privileged context.** The context is produced online by the current policy itself, with no hindsight skill, separate teacher or extra rollouts. It beats statically pre-generated context from the same model by 6.43 points and a static 30B planner by 2.86, so what matters is whether the context lies in the policy's high-probability region, not the capability of its producer.

We evaluate on three interactive environments—ALFWorld, Search-QA and WebShop—at two model scales, where PROJOPD improves consistently over GRPO and several strong self-distillation baselines.

2 METHOD

2.1 PROBLEM SETTING

We consider a partially observable multi-turn environment. Given an initial task state o_0, at the t-th environment turn the policy π_θ receives the history h_t and generates a response $y_t = (y_{t,1}, \ldots, y_{t,L_t})$ containing both reasoning and an action. The environment executes the parsed action and returns a new observation. A full trajectory is written as

$$\tau = (o_0, y_1, o_1, \ldots, y_T, o_T), \tag{2}$$

and the environment mainly emits a task-success reward $R(\tau)$ at the end of the episode. For a group of on-policy trajectories sampled on the same task, GRPO derives the advantage A_t from group-relative returns (Shao et al., 2024). We do not change the advantage estimator; we add a constrained OPD auxiliary signal on top of the update it produces.

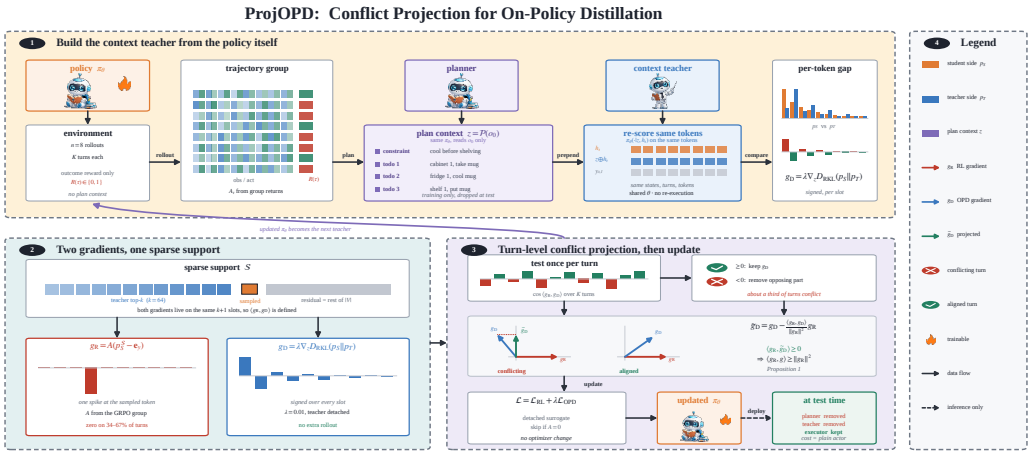


Figure 1: **Overview of PROJOPD.** The student completes its on-policy interaction without access to the plan context; the same current policy, once the planner-generated constraints and todo list are prepended, acts as a context teacher and re-scores the student's tokens. The RL gradient serves as the primary direction, and the sparse OPD gradient enters the update only after turn-level conflict projection. At test time the planner and the teacher are removed and only the executor is deployed.

As shown in Figure 1, we propose PROJOPD. At the start of each episode, the planner produces a privileged context from the initial observation, containing task constraints and an execution plan, and prepends it to the executor's input. The student performs its ordinary on-policy rollout without access to this context; the same policy then acts as a context teacher, with the plan context prepended, and recomputes the distribution over the tokens the student already sampled. We construct the RL and OPD logit gradients explicitly on a sparse grouped distribution formed by the teacher's top-k entries, the student's sampled token and a residual probability bucket, and remove at the turn level the component of the OPD gradient that opposes the policy gradient. The final gradient is back-propagated through a detached surrogate loss, requiring no modification of the optimizer and no explicit manipulation of parameter gradients.

2.2 PLANNER-CONDITIONED ON-POLICY DISTILLATION

Construction of the planner. The planner and the executor are the *same policy* π_θ under different prompts. At the start of each episode the planner reads only the initial observation o_0 and emits a structured plan context $z = \mathcal{P}(o_0) = (\text{hard constraints},\ \text{todo list})$; it is invoked once per episode. Its soundness does not come from the planner being "stronger" than the executor — the parameters are identical — but from three points: z fixes the *global* task constraints and step order at the start of the episode, offsetting the temporal asymmetry of an executor that sees only the history h_t; z is produced by the current policy itself and therefore lies in its high-probability region, keeping the distillation target reachable; and teacher and student share θ and the same Jacobian, which is what makes the angle between the two logit gradients correspond to the direction of the parameter update. The prompt restrictions, field validation and retry policy are given in Appendix A.1, and the shared-Jacobian prerequisite is derived in Appendix A.2.

Precisely because the teacher introduces no external information, what it expresses is only the *local preference* of the same policy under an extra prompt — the source both of its low cost and of its potential conflict with the task objective, which §2.3 addresses.

Teacher distribution. The planner produces a plan context $z = \mathcal{P}(o_0)$ from the initial observation, containing hard task constraints, a todo list and a high-level execution plan. The student $\pi_\theta(\cdot \mid h_t)$ does not access z during rollout. Once trajectories are collected, z is added to the context of the same turn and the teacher distribution is formed with the current policy:

$$p_T(\cdot) = \pi_\theta(\cdot \mid z, h_t), \qquad p_S(\cdot) = \pi_\theta(\cdot \mid h_t). \tag{3}$$

The teacher is not an additional static large model but the conditional distribution of the current policy under the plan context. It only re-scores the response the student has already generated and never re-executes the environment, which guarantees that the student and the teacher are strictly aligned in state, turn and token. If the planner output fails format validation, the episode contributes no OPD signal, while its student trajectory still participates in GRPO as usual.

Both gradients live in the same categorical space. Comparing only the probability of the sampled token cannot describe how the teacher redistributes probability mass across candidate actions. For each response token we construct the support set

$$\mathcal{S}_{t,l} = \mathrm{TopK}(p_T) \cup \{y_{t,l}\} \cup \{\mathrm{res}\}, \tag{4}$$

where res merges the vocabulary mass outside the support into a residual bucket. The implementation first takes the top-64 entries of the teacher logits; if the token actually sampled by the student is not among them, the last support slot is replaced by that token, so the number of support slots remains 64, after which one residual bucket is appended and the sampled-token support rate is always 1. Student and teacher then form normalized grouped distributions $p_S^{\mathcal{S}}$ and $p_T^{\mathcal{S}}$ over the same support. Distributional OPD uses the reverse KL:

$$\mathcal{L}_{\mathrm{OPD}} = D_{\mathrm{KL}}\left(p_S^{\mathcal{S}} \| p_T^{\mathcal{S}}\right), \qquad g_{\mathrm{D}} = \lambda \nabla_z \mathcal{L}_{\mathrm{OPD}} = \lambda\, p_S^{\mathcal{S}} \odot \left(\log \tfrac{p_S^{\mathcal{S}}}{p_T^{\mathcal{S}}} - \mathcal{L}_{\mathrm{OPD}}\right), \tag{5}$$

Throughout, $\mathcal{L}_{\mathrm{OPD}}$ is the *unscaled* divergence and λ enters exactly once, when g_{D} is formed; every geometric quantity below is computed on the λ-scaled g_{D}, with no further coefficient applied. where $\odot$ is the element-wise product. The reverse KL is mode-seeking: it asks the student to place mass on the actions the teacher endorses rather than to cover the whole teacher distribution. Here the teacher is not a stronger model but the same policy conditioned on a lightweight planner, so its tail is not reliable, and the mass-covering behaviour of the forward KL would fit that noise as well (Appendix A.8). The sparse support set is an established device and is not a contribution of this work; here it plays a structural role: **it is precisely because g_{R} and g_{D} live in the same $k{+}1$-dimensional categorical space that their inner product and angle are defined at all**, which is what makes the geometric argument below possible. Compared with token-weighted OPD, which only raises the probability of the sampled token, g_{D} expresses both the directions the teacher promotes and those it suppresses.

We ran an independent diagnostic on support size using Qwen2.5-3B on ALFWorld Valid ALFWorld. Over 125 episodes with valid planner output, covering 358,349 response-token positions, the average teacher probability mass covered by top-16, top-32 and top-64 is 99.13%, 99.59% and 99.79% respectively. Although the average is already high at top-32, top-64 reduces the fraction of tokens with coverage below 95% from 1.53% to 0.38% and raises the first percentile from 93.98% to 96.64%. We therefore use 64 support slots, trading a modest $O(LK)$ overhead for more robust tail coverage; mass outside the support is still retained by the residual bucket. Full quantiles and low-coverage fractions appear in Appendix 3.

2.3 Gradient conflict between RL and OPD

For a token y actually sampled by the student, the gradient of the GRPO loss with respect to the same grouped logits is

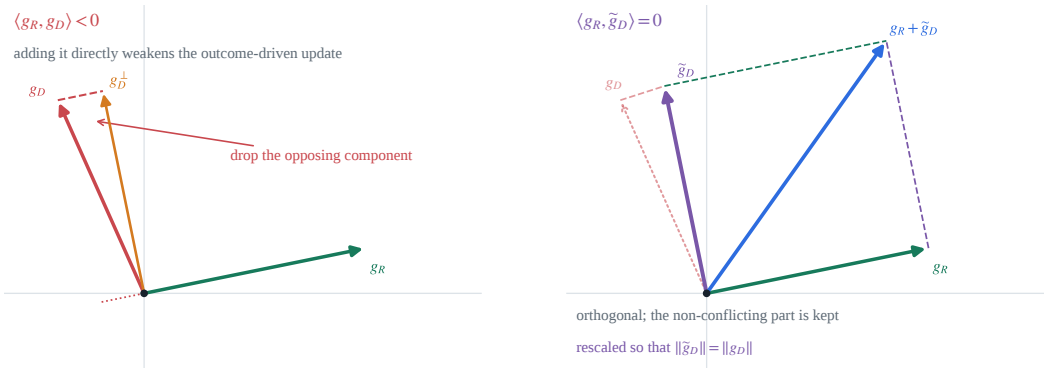
$$g_{\mathrm{R}} = A\left(p_S^{\mathcal{S}} - \mathbf{e}_y\right), \tag{6}$$

where $\mathbf{e}_y$ is the one-hot vector of the sampled token. We flatten and concatenate g_{R} and g_{D} separately over all valid response tokens within the same environment turn. Turn granularity corresponds to one complete environment action; it also avoids the token-level alternative, which would project each token independently and ignore the joint direction inside a single action, as well as the episode-level alternative, in which conflicts between different decisions cancel each other out.

Define the pre-projection cosine as

$$c_{\mathrm{pre}} = \frac{\langle g_{\mathrm{R}}, g_{\mathrm{D}} \rangle}{\|g_{\mathrm{R}}\| \|g_{\mathrm{D}}\| + \epsilon}. \tag{7}$$

When $c_{\mathrm{pre}} < 0$, updating along $-g_{\mathrm{D}}$ increases the RL loss to first order, and we say the turn exhibits gradient conflict. Raw OPD does not distinguish conflicting from aligned turns, and therefore cannot guarantee that the update remains outcome-driven once the auxiliary objective is added.



(a) raw OPD conflicts with the policy gradient

(b) project out the opposing component

Figure 2: **Turn-level gradient constraint.** The raw OPD gradient has a negative inner product with the policy gradient, and adding them directly weakens the RL update. Once the conflicting component is removed, the combined gradient retains the outcome-driven first-order descent direction.

Such conflicts are not rare in practice. In Qwen2.5-3B ALFWorld training, on average 31.20% of valid turns satisfy $c_{\text{pre}} < 0$ before projection, and 91 of 111 training steps exhibit at least one conflict. Even when the planner output is valid and teacher coverage is sufficient, OPD still competes locally with the task objective at the optimization level.

2.4 POLICY-GRADIENT-CONSTRAINED OPD

As illustrated in Figure 2, PROJOPD imposes a directional constraint on every environment turn. For a conflicting turn we remove the component of g_{D} that lies along the negative direction of g_{R}:

$$\tilde{g}_{\text{D}} = \begin{cases} g_{\text{D}} - \dfrac{\langle g_{\text{D}}, g_{\text{R}} \rangle}{\|g_{\text{R}}\|^2} g_{\text{R}}, & \langle g_{\text{D}}, g_{\text{R}} \rangle < 0 \ \text{ and } \ \|g_{\text{R}}\|^2 \geq \epsilon, \\ 0, & \langle g_{\text{D}}, g_{\text{R}} \rangle < 0 \ \text{ and } \ \|g_{\text{R}}\|^2 < \epsilon, \\ g_{\text{D}}, & \text{otherwise.} \end{cases} \tag{8}$$

The second branch exists because the coefficient $\langle g_{\text{D}}, g_{\text{R}} \rangle / \|g_{\text{R}}\|^2$ is not numerically reliable once $\|g_{\text{R}}\|^2$ falls below ϵ; rather than floor the denominator and accept a partially projected gradient, those turns drop the auxiliary gradient entirely. The denominator in the first branch is therefore always exactly $\|g_{\text{R}}\|^2$, and $\langle g_{\text{R}}, \tilde{g}_{\text{D}} \rangle = 0$ holds on all three branches without exception. Aligned turns are left unchanged. Measured over training, the mean cosine over all valid turns therefore moves from -0.0041 before projection to $+0.0161$ after it: the negative inner products on conflicting turns are zeroed while the positive ones on aligned turns are left untouched. When $\|g_{\text{R}}\|$ falls below a threshold we skip the OPD update for that turn, so that the auxiliary objective does not drive policy drift on its own in the absence of RL evidence.

The final direction is $g = g_{\text{R}} + \tilde{g}_{\text{D}}$. The full method is neither a product of two losses nor the untreated form of Eq. (1): the coefficient λ enters g_{D} first, and projection is applied afterwards. We realize the target OPD gradient through a detached surrogate loss, which reproduces $\tilde{g}_{\text{D}}$ exactly and is added to the actor objective without a further coefficient:

$$\mathcal{L}_{\text{C-OPD}} = \sum_i \text{sg}(\tilde{g}_{D,i}) \log p^{\mathcal{S}}_{S,i}, \qquad \mathcal{L}_{\text{actor}} = \mathcal{L}_{\text{GRPO}} + \mathcal{L}_{\text{C-OPD}} + \mathcal{L}_{\text{KL/entropy}}. \tag{9}$$

Proposition 1 (first-order RL non-interference). For every valid turn, the full method satisfies

$$\langle g_{\text{R}}, \tilde{g}_{\text{D}} \rangle \geq 0, \qquad \langle g_{\text{R}}, g \rangle \geq \|g_{\text{R}}\|^2. \tag{10}$$

If the raw gradients do not conflict, their inner product is non-negative and remains so after multiplication by a non-negative scaling factor; if a conflict occurs, Eq. (8) drives the inner product to zero. Hence, at a sufficiently small learning rate, updating along $-g$ cannot yield a worse first-order

descent term than the pure RL gradient. The guarantee holds in the grouped-logit space and presupposes that teacher and student share parameters and therefore the same Jacobian; the full proof, degenerate cases and a formal treatment of turn granularity appear in Appendix A.

Effect of the projection on the gradient norm. Correcting the direction is not free: by the Pythagorean relation $\|\widetilde{g}_{\mathrm{D}}\| = \|g_{\mathrm{D}}\||\sin\theta|$, the projection necessarily shortens the auxiliary gradient. Proposition 1 is scale invariant, however — for any $c > 0$, $\langle g_{\mathrm{R}}, c\,\widetilde{g}_{\mathrm{D}}\rangle = 0$ — so the orthogonal component can be rescaled back to $\|g_{\mathrm{D}}\|$ without affecting the guarantee. PROJOPD **applies this rescaling by default** (`preserve_norm` = True): the projection is responsible only for correcting direction, while the overall strength of the auxiliary signal remains under the sole control of λ. The full derivation, and how the compensation depends on the projection granularity (a measured $|\sin\theta| = 0.997$ at turn level against 0.585 at token level), are given in Appendix A.5.

3 EXPERIMENTS

3.1 ENVIRONMENTS AND METRICS

We evaluate on three interactive environments that differ in interaction horizon, action space and reward sparsity.

ALFWorld. A text-based household environment built on TextWorld/ALFRED with six task types: Pick and Place, Pick Two and Place, Clean, Cool, Heat, and Look At In Light (Shridhar et al., 2021). The model emits textual actions from a dynamic set of admissible commands; an episode succeeds when the goal is reached and otherwise runs for at most 50 steps. We report the episode success rate.

WebShop. The agent must locate a product satisfying natural-language constraints through web actions such as searching, clicking, selecting options and purchasing (Yao et al., 2022). We report $\mathrm{Score} = 100 \times \mathrm{mean}(\text{raw task score})$, which measures partial matching, and $\mathrm{Succ} = 100 \times \mathrm{mean}(\text{task score} = 1)$, which measures full constraint satisfaction.

Search-QA. We adopt the Search-based QA setting of Search-R1 (Jin et al., 2025), matching its retriever, corpus and interaction-turn limit; the configuration is given in Appendix Table 7. Evaluation covers NQ, TriviaQA, PopQA, HotpotQA, 2WikiMultiHopQA, MuSiQue and Bamboogle, reporting answer accuracy on each subset and their unweighted macro average.

3.2 MODELS, BASELINES AND IMPLEMENTATION

We use Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct.

We compare PROJOPD with GRPO and self-distillation baselines, with full results in Table 1. Methods marked * access privileged skills at test time and serve as upper references. Baseline descriptions and ablation configurations are provided in Appendix D.

3.3 MAIN RESULTS

Table 1 reports the main results. Except for the rows marked Ours, every baseline number is quoted from its original paper and was not re-run by us.

On Qwen2.5-3B, PROJOPD achieves the best results on ALFWorld Avg (90.0 vs. 84.4), Heat (91.4 vs. 75.0), Cool (80.6 vs. 75.0) and on both WebShop Score (93.1 vs. 85.0) and Succ (83.6 vs. 68.0), where the margin on WebShop Succ reaches 15.6 points over the second-best baseline; on Search-QA Avg PROJOPD ranks second (44.3 vs. 44.6 for GRPO+OPSD), with the gap arising mainly from the 2WikiMultiHopQA and Bamboogle subsets. On Qwen2.5-7B, PROJOPD achieves the best results on ALFWorld Avg (91.4 vs. 88.4), Search-QA Avg (49.7 vs. 49.0) and both WebShop Score (91.6 vs. 89.4) and Succ (85.2 vs. 82.8), with the largest margins on Look (100.0 vs. 93.8), Pick2 (87.5 vs. 78.9) and TriviaQA (72.4 vs. 64.6); RLSD and StepOPSD remain stronger on Cool (80.0 vs. 70.8).

Table 1: Performance comparison on three representative long-horizon benchmarks (ALFWorld, Search-based QA, and WebShop). We report the success rate (%) on ALFWorld, accuracy on Search-based QA, and task-completion score/success rate on WebShop. An asterisk (*) denotes validation with skills. The best results are highlighted with a green background and boldface; second-best results use a yellow background. Ties share the same highlighting.

Model	Method	ALFWorld							Search-based QA							WebShop	
		Pick	Look	Clean	Heat	Cool	Pick2	Avg	NQ	Triv	Pop	Hotp	2Wk	MuS	Bam	Avg	Score	Succ
Qwen2.5-3B	Vanilla	44.4	11.1	6.2	15.4	28.6	12.5	21.9	24.6	48.1	31.0	26.3	25.3	7.2	59.7	31.7	6.7	0.8
	Skill-Prompt*	51.7	66.7	48.4	0.0	4.3	10.0	28.9	23.7	46.2	30.6	24.4	22.1	7.5	12.5	23.9	0.2	0.8
	OPSD	48.8	41.7	16.7	0.0	15.8	16.7	28.1	0.1	0.1	0.1	0.0	0.0	0.0	0.0	0.0	11.3	3.1
	GRPO	91.2	62.5	96.2	61.9	65.0	47.4	75.0	39.3	60.6	41.1	37.4	34.6	15.4	26.4	36.4	79.8	63.3
	Skill-GRPO	88.9	71.4	58.8	70.6	40.7	29.2	60.2	43.5	58.8	43.0	36.8	32.2	11.7	12.5	34.1	77.3	60.9
	Skill-GRPO*	94.3	57.1	**100.0**	66.7	73.1	57.1	80.5	44.3	59.6	44.3	39.0	36.1	14.5	14.9	36.1	76.3	66.4
	GRPO+OPSD	**100.0**	**82.4**	85.7	75.0	70.0	60.0	81.2	**44.9**	**61.2**	45.2	**40.4**	38.5	16.0	66.1	**44.6**	77.8	66.4
	Skill-SD	88.2	50.0	96.2	52.4	65.0	57.9	73.4	44.4	60.4	44.0	39.5	**40.4**	15.4	64.9	44.1	75.9	64.0
	RLSD	87.9	75.0	90.9	75.0	73.1	68.4	79.7	41.5	58.6	42.3	**40.4**	40.2	**16.8**	**66.9**	43.8	84.4	66.4
	SDAR	97.1	62.5	**100.0**	61.9	75.0	**84.2**	84.4	44.8	58.1	44.3	38.6	36.2	15.7	66.1	43.4	85.0	68.0
	StepOPSD	82.4	66.7	82.6	52.2	73.7	75.0	73.4	43.6	**61.2**	43.8	39.2	38.1	15.8	64.5	43.7	82.4	66.4
	ProjOPD (Ours)	99.0	80.2	98.0	**91.4**	**80.6**	77.8	**90.0**	44.4	60.4	**47.8**	39.3	37.6	15.3	65.3	44.3	**93.1**	**83.6**
Qwen2.5-7B	Vanilla	36.1	22.2	3.1	0.0	0.0	0.0	12.5	25.2	50.8	29.5	29.0	29.0	10.4	63.7	33.9	5.9	1.6
	Skill-Prompt*	51.7	50.0	32.3	5.3	4.3	0.0	23.4	30.9	52.1	32.7	32.7	27.9	12.7	66.1	36.4	1.7	0.8
	OPSD	50.0	60.0	22.7	21.4	17.6	9.5	32.8	8.8	8.6	17.5	2.5	4.2	0.5	1.2	6.2	4.5	2.3
	GRPO	91.2	87.5	96.2	81.0	65.0	57.9	81.2	45.1	63.7	44.0	43.6	43.2	16.8	37.6	42.0	80.9	72.6
	Skill-GRPO	88.5	66.7	65.2	61.1	57.7	73.1	69.5	45.2	63.7	45.7	43.1	43.3	19.6	21.4	40.3	80.4	71.9
	Skill-GRPO*	**100.0**	83.3	96.4	83.3	75.0	78.9	88.3	44.8	63.0	45.1	43.7	43.7	20.5	71.4	47.5	87.0	81.2
	GRPO+OPSD	91.4	61.5	**100.0**	87.5	76.5	52.2	80.4	47.3	64.5	46.9	43.8	39.3	18.0	69.4	47.0	86.8	76.5
	Skill-SD	93.9	93.8	90.9	**100.0**	69.2	68.4	85.1	47.1	64.5	47.8	44.2	42.1	20.2	69.0	47.8	86.1	76.5
	RLSD	**100.0**	87.5	92.3	58.8	**80.0**	65.2	82.0	46.8	63.0	44.4	**45.5**	**48.9**	**21.5**	**73.0**	49.0	87.4	77.3
	SDAR	94.7	75.0	**100.0**	86.7	68.2	78.9	85.9	46.3	63.5	**48.2**	43.8	48.4	19.6	**73.0**	49.0	89.4	82.8
	StepOPSD	98.1	75.0	**100.0**	90.5	**80.0**	63.2	88.4	45.3	64.6	45.1	44.5	44.4	19.3	69.8	48.2	87.2	78.1
	ProjOPD (Ours)	**100.0**	**100.0**	97.0	95.0	70.8	**87.5**	**91.4**	**47.6**	**72.4**	45.8	44.0	43.0	20.3	71.4	**49.7**	**91.6**	**85.2**

Table 2: **Ablation summary for PROJOPD.** Per-subtask and overall episode success (%) of Qwen2.5-3B on ALFWorld. The upper block fixes the teacher source and varies how conflict is handled; the lower block keeps all constraints and varies only how the privileged context is generated. † Quoted from Table 1.

Method	ALFWorld						
	Pick	Look	Clean	Heat	Cool	Pick2	Avg
ProjOPD	**99.0**	80.2	98.0	**91.4**	80.6	**77.8**	**90.0**
How conflict is handled							
w/o orthogonal component	99.0	80.2	**100.0**	86.1	72.2	73.6	87.6
w/o conflict projection	97.9	**89.7**	94.0	89.7	56.9	**87.5**	86.7
w/o OPD (GRPO)†	91.2	62.5	96.2	61.9	65.0	47.4	75.0
Source of the privileged context							
Static 3B planner	97.0	84.9	95.0	70.7	66.7	76.4	83.6
Static 30B planner	96.9	75.0	96.0	89.7	**83.3**	66.7	87.1
Skill distillation	95.9	85.0	96.0	86.2	76.4	70.8	86.4

Training dynamics and computational cost are reported in Appendices D.3 and D.4.

3.4 ABLATIONS

Ablations are fixed to Qwen2.5-3B on ALFWorld and remove one component at a time from the full method. Every number is taken from the checkpoint at the end of training.

The role of dense supervision. The three OPD variants reach 90.0, 87.6 and 86.7 respectively, improving over the policy-optimization baseline without OPD (75.0) by 11.7 to 15.0 points. The gains concentrate on long-horizon subtasks: Heat rises from 61.9 to 86.1–91.4 and Pick2 from 47.4 to 73.6–87.5, precisely the task families that require multi-stage execution and in which trajectory-level reward is least able to localize the decisive step. This indicates that **the gain originates from the dense supervision supplied by the planner context itself, rather than from any particular way of handling conflict**.

How conflict is handled. Retaining the orthogonal component and zeroing the gradient outright differ by 0.9 points (87.6 vs. 86.7), far less than the gain of dense supervision over the baseline without OPD. Proposition 1 holds equally for $\widetilde{g}_{\mathrm{D}} = 0$, so both satisfy the first-order non-interference condition, and this difference is too small to distinguish between them.

Ablating the teacher construction. The lower block of Table 2 keeps every constraint fixed and varies only how the privileged context is produced, comparing four constructions: (i) *online planner*, our method, in which the current policy generates the plan context from the initial observation of each episode; (ii) *static 3B planner*, in which the same model as the student generates a plan for every game once before training and never updates it; (iii) *static 30B planner*, which instead pre-generates plans with a frozen Qwen3-30B-A3B-Instruct and is otherwise identical to (ii); and (iv) *skill distillation*, which produces an episode-level skill from each completed trajectory as context, with the skill generated by the current policy over the full trajectory.

The four arms separate two factors. **Gain from online generation ($+6.43$):** generating the plan online alongside the policy improves success by 6.43 points over generating it once before training with the static 3B planner. **Effect of teacher scale ($+3.57$):** the static 30B planner exceeds the static 3B planner by 3.57 points, showing that a stronger plan generator does help. That gain, however, is smaller than the $+6.43$ from making generation online: relying only on 3B's own planning ability, the online planner still exceeds static plans pre-generated by a model an order of magnitude larger by 2.86 points. The value of a privileged context thus depends not only on the capability of its generator but also on whether it falls in a high-probability region of the current policy—a plan the policy writes itself is naturally phrased and scoped so that the policy can absorb it.

Look-ahead planning beats hindsight skills ($+3.57$): skill distillation derives its context from completed trajectories and is 3.57 points below the online planner overall. The gap concentrates on multi-stage tasks such as Pick2 (77.8 vs. 70.8) and Heat (91.4 vs. 86.2), while hindsight skills are in fact stronger on Look. The two contexts are close in overall success rate but differ markedly in cost: skills require an additional trajectory analysis over all rollouts at every training step, whereas the online planner generates one context per episode. Training-side diagnostics further show that skill quality degrades slowly over training: the fraction of empty skills rises from 6.2% at the first step to 8.2% at the last, and the median length falls from 206 to 104 characters.

Token-level diagnostics support the same conclusion. Comparing how different contexts bias the student's tokens on the same batch of initial observations, we find that the log-probability gain of tokens named by the context relative to unnamed tokens is $+0.49$ for the online planner, clearly higher than for the other constructions, while the overall mean gap is in fact lowest for the online planner. An effective privileged context therefore does not pull the student distribution towards the teacher as a whole, but *selectively* raises the decision tokens relevant to the plan.

4 Related work

Interactive agents and the foundations of policy optimization. ReAct, Toolformer and Voyager demonstrate different routes by which language models complete interactive tasks through explicit reasoning, tool invocation and continual skill accumulation (Yao et al., 2023; Schick et al., 2023; Wang et al., 2024). TRPO, GAE and RLHF respectively established trust-region policy optimization, advantage estimation and post-training from human feedback (Schulman et al., 2015; 2016; Ouyang et al., 2022). Evaluation of such agents now spans API calls, multi-site browsing, desktop operation and code repositories (Liu et al., 2024; Zhou et al., 2023; Xie et al., 2024; Jimenez et al., 2024); tool learning, retrieval-augmented RL, autonomous exploration and agent adaptation without backbone fine-tuning extend their capability boundary along different axes (Shen et al., 2023; Shi et al., 2025; Ye et al., 2026; Zhou et al., 2025; Xu et al., 2024; 2025). Explicit reasoning chains are the output form all of these rely on (Wei et al., 2022).

Agentic RL and credit assignment. PPO, GRPO and their extensions to agent trajectories optimize the policy directly from environment outcomes (Schulman et al., 2017; Shao et al., 2024; Guo et al., 2025), while GiGPO, ARPO and DAPO further address multi-turn agent rollouts, group-relative advantages and large-scale training implementations (Feng et al., 2025; Dong et al., 2025; Yu et al., 2025). Trajectory-level reward, however, broadcasts a single outcome to a large number of intermediate tokens or turns. State grouping, step-level advantages and process rewards can improve credit assignment, but typically rely on additional annotation, environment structure or a reward model. We do not modify the underlying advantage estimator; instead we study how external planning supervision can enter the update without disturbing the original policy gradient.

Agentic on-policy distillation. Knowledge distillation originally trained a student on the soft targets of an offline teacher (Hinton et al., 2015), and sequence-level distillation generalized the idea to a broader choice of divergence (Wen et al., 2023). OPSD and subsequent work study on-policy teacher re-scoring, sequence-level distillation and the stability of OPD (Zhao et al., 2026; Li et al., 2026; Wen et al., 2023). SDAR, Skill-SD and StepOPSD extend this paradigm through token gating, skill-conditioned teachers and turn-level signals respectively (Lu et al., 2026; Wang et al., 2026; Zhang et al., 2026). These works primarily answer how to construct the teacher signal or which tokens and turns deserve larger weight. PROJOPD is orthogonal to them: we study the geometric relation between the auxiliary OPD gradient and the outcome-driven RL gradient, and provide explicit first-order protection when the two conflict.

Gradient conflict. Work on delayed-reward credit assignment likewise shows that trajectory-level return struggles to localize the decisions that actually drive the final outcome (Arjona-Medina et al., 2019). Multi-task learning has established that gradients of different objectives can have negative inner products, so that summing them directly causes mutual interference; PCGrad and related methods improve shared-parameter optimization through projection (Yu et al., 2020). We do not treat multiple peer tasks symmetrically: RL is designated the primary objective and OPD the auxiliary one; the projection happens in an interpretable turn-level sparse-logit space rather than in the full parameter-gradient space; and it is integrated into standard distributed training through a detached surrogate.

5 CONCLUSION AND LIMITATIONS

This paper studies the compatibility between planner-conditioned OPD and the outcome-driven policy gradient. We show that dense teacher supervision can interfere with the reward-driven update in both direction and scale, and propose PROJOPD: the teacher is formed by conditioning the current policy on a plan context and only re-scores the tokens the student has already sampled, so no additional model, extra rollout or trajectory pairing is required; the distillation signal is expressed on a sparse support consisting of the teacher's top-k entries, the sampled token and a residual probability bucket, after which the component conflicting with the policy gradient is removed within each environment turn and back-propagated through a detached surrogate. The construction guarantees to first order that the auxiliary signal cannot cancel the descent direction of RL, and at test time the planner and teacher are removed so that inference cost matches an ordinary actor. On ALFWorld, WebShop and Search-QA the method improves consistently over GRPO and several self-distillation baselines, reaching 91.4% success on ALFWorld with Qwen2.5-7B. Training diagnostics show that gradient conflict is pervasive (on average 31.20% of valid turns have a negative inner product), while ablations indicate that the gain stems mainly from the dense distillation signal itself, with the largest improvements on long-horizon subtasks requiring multi-stage execution. Two limitations are worth stating. First, Proposition 1 is a *first-order* property of the grouped-logit space: it says the auxiliary gradient does not cancel the RL descent direction at the current point, not that the parameter update is globally monotone. Higher-order effects, the Jacobian map and optimizer state may still interfere. This is, however, the price of the construction that makes the method free: the guarantee is obtained from quantities already available in the forward pass, with no extra rollout and no second model, and non-conflicting turns are left untouched so teacher information is never discarded wholesale. Second, because the teacher is the same policy under an additional prompt, it introduces no information independent of the current policy, so what it can supply is bounded by what the policy already knows. That same property is what keeps the distillation target reachable, and empirically an online 3B planner still outperforms a static planner an order of magnitude larger. Extending the guarantee beyond first order, and relaxing the teacher without reintroducing an external model, are the natural next steps.

REPRODUCIBILITY STATEMENT

The method is specified in §2, with the full derivation of the projection, the norm relation and the proof of Proposition 1 in Appendix A. The training loop is given step by step in Appendix B, the optimization and PROJOPD hyper-parameters in Table 6, and the per-environment training and evaluation configuration in Table 7. The planner prompt, the teacher context and the executor prompts

for all three environments are reproduced verbatim in Appendix E, together with sample planner outputs. Benchmarks, splits and metric definitions are described in Appendix C, and the baselines we compare against in Appendix D. An anonymous code archive accompanies this submission as supplementary material: it contains the projection implementation, its call site in the actor, the planner construction, the training and evaluation scripts for every arm reported here, pinned dependency versions, and a single script that takes a bare machine through environment setup, data and model preparation, and training.

AI USE DISCLOSURE

Among the tasks for which ICLR recommends disclosure, we used generative AI tools for retrieval and discovery, that is, for searching for and identifying literature relevant to this submission; for editing and typesetting the paper; and for producing the schematic figures. Details are described in the paper. We did not use generative AI tools for any of the tasks for which ICLR requires disclosure: generating synthetic datasets, developing theoretical models or conceptual frameworks, formulating mathematical claims, providing ingredients for or writing proofs, proposing or refining hypotheses, designing or giving feedback on the research methodology or experiments, implementing the method, translation, cleaning or reformatting datasets, supporting qualitative analysis, or interpreting results.

All code and text arising from this assistance was reviewed and verified by the authors. In particular, every reference surfaced through retrieval was read and checked against its original source before being cited. We take responsibility for the final content of this work, including its text, claims and artifacts.

REFERENCES

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

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, et al. Agentic reinforced policy optimization. *arXiv preprint arXiv:2507.19849*, 2025.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. *arXiv preprint arXiv:2505.10978*, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *NeurIPS Deep Learning and Representation Learning Workshop*, 2015.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? *International Conference on Learning Representations*, 2024.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huanang Gao, Wenkai Yang, Zhiyuan Liu, and Ning Ding. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *arXiv preprint arXiv:2604.13016*, 2026.

Xiao Liu, Hao Yu, Hanfei Zhang, Yifan Xu, Xin Lei, Hanyu Lai, et al. Agentbench: Evaluating llms as agents. In *International Conference on Learning Representations*, 2024.

Zhengxi Lu, Zhiyuan Yao, Zhuowen Han, Zi-Han Wang, Jinyang Wu, Qi Gu, Xunliang Cai, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. Self-distilled agentic reinforcement learning. *arXiv preprint arXiv:2605.15155*, 2026.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 2022.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, 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*, 2023.

John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization. *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. In *arXiv preprint arXiv:1707.06347*, 2017.

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

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. *Advances in Neural Information Processing Systems*, 2023.

Zhengliang Shi, Shen Gao, Lingyong Yan, Yue Feng, Xiuyi Chen, Zhumin Chen, Dawei Yin, Suzan Verberne, and Zhaochun Ren. Tool learning in the wild: Empowering language models as automatic tool agents. In *Proceedings of the ACM Web Conference*, 2025.

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

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018.

Guanzhi Wang, Yunfan Xie, Zi-Yuan Jiang, Aditya Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024.

Hao Wang, Guozhi Wang, Han Xiao, Yufeng Zhou, Yue Pan, Jichao Wang, Ke Xu, Yafei Wen, Xiaohu Ruan, Xiaoxin Chen, and Honggang Qi. Skill-sd: Skill-conditioned self-distillation for multi-turn llm agents. *arXiv preprint arXiv:2604.10674*, 2026.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 2022.

Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Mou. f-divergence minimization for sequence-level knowledge distillation. *arXiv preprint arXiv:2307.15190*, 2023.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.

Tianbao Xie, Danyang Li, Zhiwei Zeng, Tao Zhang, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Advances in Neural Information Processing Systems*, 2024.

Hongshen Xu, Zichen Zhu, Lei Pan, Zihan Wang, Su Zhu, Da Ma, Ruisheng Cao, Lu Chen, and Kai Yu. Reducing tool hallucination via reliability alignment. *arXiv preprint arXiv:2412.04141*, 2024.

Hongshen Xu, Zihan Wang, Zichen Zhu, Lei Pan, Xingyu Chen, Shuai Fan, Lu Chen, and Kai Yu. Alignment for efficient tool calling of large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025.

Chenxu Yang, Chuanyu Qin, Qingyi Si, Minghui Chen, Naibin Gu, Dingyu Yao, Zheng Lin, Weiping Wang, Jiaqi Wang, and Nan Duan. Self-distilled rlvr. *arXiv preprint arXiv:2604.03128*, 2026.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, 2022.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Ziang Ye, Wentao Shi, Yuxin Liu, Yu Wang, Zhengzhou Cai, Yaorui Shi, Qi Gu, Xunliang Cai, and Fuli Feng. Look before you leap: Autonomous exploration for llm agents. *arXiv preprint arXiv:2605.16143*, 2026.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, 2020.

Yanfei Zhang, Xu Lin, and Chenglin Wu. Stepopsd: Step-aware online preference distillation for agent reinforcement learning. *arXiv preprint arXiv:2605.27140*, 2026.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

Huichi Zhou, Yihang Chen, Siyuan Guo, Xue Yan, Kin Hei Lee, Zihan Wang, Ka Yiu Lee, Guchun Zhang, Kun Shao, Linyi Yang, et al. Memento: Fine-tuning llm agents without fine-tuning llms. *arXiv preprint arXiv:2508.16153*, 2025.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, et al. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*, 2023.

A Mathematical supplement

This appendix supplements the derivations for sparse distributional OPD, the gradient projection, and the surrogate loss. All derivations are carried out in the grouped-logit space of a single valid environment turn; vectors belonging to different turns are handled separately in the implementation and are never concatenated across turns.

A.1 Construction of the planner

Construction of the planner. The planner and the executor are the *same policy* π_θ, placed under different prompts. At the start of each episode the planner reads the initial observation o_0 (and *only* o_0) and emits a structured plan context

$$z = \mathcal{P}(o_0) = \big(\text{hard constraints},\ \text{todo list}\big), \tag{11}$$

where the hard constraints are outcome conditions that must hold for the task to succeed and the todo list orders the high-level steps that satisfy them. To keep the planner from taking over the executor's role, its input drops the environment's trailing action-output instructions and it is forbidden to emit `<action>` or to call tools; its output must pass a field-completeness check, a parse failure is retried at most once, and an episode that still fails has its OPD mask zeroed while its student trajectory participates in GRPO as usual. The planner is invoked once per episode, independently of the number of interaction turns.

Why this construction is sound. The construction is not justified by the planner being "stronger" than the executor: the parameters are identical, so it cannot introduce any information independent of the current policy. Its soundness rests on three points. *First, an asymmetry in time*: at turn t the executor sees only the history h_t, whereas z fixes the *global* task constraints and step order at the start of the episode, so z supplies no new knowledge but *unrolls in advance* knowledge the model already has, making it visible at every later local decision. *Second, distributional absorbability*: z is produced by the current policy itself, so its wording and granularity fall naturally inside that policy's high-probability region, the teacher $\pi_\theta(\cdot \mid z, h_t)$ therefore does not drift far from the student, and the distillation target stays reachable. *Third, a prerequisite for the geometric argument*: teacher and student share θ and the same forward network, hence the same Jacobian, so both logit gradients are images of one linear map and the angle between them is well defined (Appendix A.2). With an independent external teacher even that much no longer holds.

Precisely because the teacher introduces no external information, what it expresses is only the *local preference* of the same policy under an extra prompt — the source both of its low cost and of its potential conflict with the task objective, which §2.3 addresses.

A.2 When the two gradients are comparable

We compare the angle between g_{R} and g_{D} in grouped-logit space. That comparison is meaningful only under a structural premise: the two gradients share one Jacobian.

Let $z \in \mathbb{R}^{|\mathcal{S}|}$ be the grouped logits and $J = \partial z / \partial \theta$. For any gradient g given in logit space, the chain rule yields the corresponding parameter gradient

$$\nabla_\theta \mathcal{L} = \sum_i \frac{\partial \mathcal{L}}{\partial z_i} \nabla_\theta z_i = J^\top g. \tag{12}$$

Our teacher is the same policy π_θ conditioned on a plan context z; it re-scores *the very tokens the student has already sampled*, so the student and teacher branches share the parameters θ, share one forward network, and therefore share one J. The two parameter gradients are $J^\top g_{\mathrm{R}}$ and $J^\top g_{\mathrm{D}}$, so both live in the same space and are images of the same linear map, which is what makes an angle between them well defined at all.

We stress what this does *not* establish. The parameter-space inner product is

$$\langle J^\top g_{\mathrm{R}},\, J^\top g_{\mathrm{D}} \rangle = g_{\mathrm{R}}^\top J J^\top g_{\mathrm{D}}, \tag{13}$$

and $JJ^\top$ is positive semi-definite but not the identity, so this differs from $g_{\mathrm{R}}^\top g_{\mathrm{D}}$ in general and need not even share its sign. A shared Jacobian is necessary for the comparison to be meaningful, but it is not sufficient to preserve angles: the network Jacobian is not an isometry. Proposition 1 is accordingly a first-order statement about the grouped-logit space and nothing more — it says the projected auxiliary gradient does not oppose g_{R} *there*. We do not claim that $\langle \nabla_\theta \mathcal{L}_{\mathrm{RL}}, \nabla_\theta \mathcal{L}_{\mathrm{OPD}} \rangle$ is non-negative after projection, and we have not measured it; enforcing the constraint directly in parameter space would require materializing two full parameter gradients per turn, which is precisely the cost the logit-space formulation avoids.

This premise does not hold in general. If the teacher were a separate external model (parameters $\phi \neq \theta$), the Jacobian of its output distribution with respect to the student parameters would differ, the two logit-space gradients would not even be images of a common map, and whether an auxiliary gradient cancels the policy gradient could not be decided by comparing two distributions at all. *Conditioning the teacher on the current policy* is therefore not merely an engineering choice that saves cost, but a necessary condition for the geometric argument of this paper.

A.3 GRADIENT OF THE GROUPED KL

Let the grouped logits be $z \in \mathbb{R}^{|\mathcal{S}|}$, the student distribution $p_S = \mathrm{softmax}(z)$, and the teacher distribution p_T detached before the computation. We use the reverse KL,

$$D_{\mathrm{KL}}(p_S \| p_T) = \sum_{j \in \mathcal{S}} p_{S,j}(\log p_{S,j} - \log p_{T,j}) \triangleq \sum_j p_{S,j}\, \ell_j, \qquad \ell_j = \log \frac{p_{S,j}}{p_{T,j}}. \tag{14}$$

Using $\partial p_{S,j}/\partial z_i = p_{S,j}(\mathbf{1}[i=j] - p_{S,i})$ and $\partial \ell_j/\partial z_i = \mathbf{1}[i=j] - p_{S,i}$, the two terms combine into

$$\frac{\partial D_{\mathrm{KL}}(p_S \| p_T)}{\partial z_i} = p_{S,i}\big(\ell_i + 1\big) - p_{S,i} \sum_j p_{S,j}(\ell_j + 1) = p_{S,i}\big(\ell_i - D_{\mathrm{KL}}(p_S \| p_T)\big), \tag{15}$$

where the constant 1 cancels against its own expectation. With an OPD coefficient λ we obtain $g_D = \lambda\, p_S \odot (\ell - D_{\mathrm{KL}})$. This gradient is also zero-sum,

$$\sum_i g_{D,i} = \lambda\Big(\sum_i p_{S,i}\ell_i - D_{\mathrm{KL}} \sum_i p_{S,i}\Big) = \lambda\big(D_{\mathrm{KL}} - D_{\mathrm{KL}}\big) = 0, \tag{16}$$

so building the detached surrogate as $\sum_i \mathrm{sg}(g_{D,i}) \log p_{S,i}$ makes the logit gradient exactly equal to g_D. This implementation changes only the auxiliary gradient; it never back-propagates through the teacher branch or through the projection computation graph. Appendix A.8 motivates the choice of the reverse KL; the geometric argument below uses only the zero-sum property of g_D and therefore holds for either divergence.

The policy-gradient side. For the same group of logits, the GRPO loss on the sampled token y is $-A \log p_{S,y}$. Reusing the same softmax derivative $\partial \log p_{S,j}/\partial z_i = \mathbf{1}[i=j] - p_{S,i}$ gives

$$g_{R,i} = \frac{\partial(-A \log p_{S,y})}{\partial z_i} = A\big(p_{S,i} - \mathbf{1}[i=y]\big), \qquad \text{that is} \quad g_{\mathrm{R}} = A\,(p_S - \mathbf{e}_y)\,. \tag{17}$$

g_{R} is likewise zero-sum: $\sum_i g_{R,i} = A(1-1) = 0$. Both gradients therefore lie in the same zero-sum hyperplane $\{v : \sum_i v_i = 0\}$ — precisely the subspace of admissible gradients induced by the invariance of softmax to a uniform shift of the logits. Any vector outside that subspace carries a constant component that leaves the distribution unchanged, and its geometric quantities (angle, norm) are meaningless. The projection is a linear operation, so $\widetilde{g}_{\mathrm{D}} = g_{\mathrm{D}} - \alpha g_{\mathrm{R}}$, a linear combination of two zero-sum vectors, remains zero-sum; the surrogate therefore still recovers the target gradient exactly after projection.

Which gradient the projection actually uses. The optimizer does not descend $-A \log p_{S,y}$ directly. The actor objective is the clipped surrogate of GRPO together with a reference-KL term and an entropy term,

$$\mathcal{L}_{\mathrm{actor}} = \underbrace{-\,\mathbb{E}\big[\min\big(rA,\ \mathrm{clip}(r, 1-\varepsilon_{\mathrm{lo}}, 1+\varepsilon_{\mathrm{hi}})A\big)\big]}_{\mathcal{L}_{\mathrm{PG}}} + \beta_{\mathrm{KL}}\, D_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{ref}}) \;-\; \beta_{\mathrm{H}}\, \mathcal{H}(\pi_\theta), \tag{18}$$

where $r = \pi_\theta(y \mid h)/\pi_{\mathrm{old}}(y \mid h)$, and in our runs $\varepsilon_{\mathrm{lo}} = \varepsilon_{\mathrm{hi}} = 0.2$, $\beta_{\mathrm{KL}} = 0.01$ (low-variance KL estimator) and $\beta_{\mathrm{H}} = 0.001$.

The g_{R} used for conflict detection and projection is $\nabla_z(-A \log p_{S,y})$, i.e. the gradient of the policy-gradient term in the form it takes when $r = 1$ and the clip is inactive. Two consequences should be stated plainly. First, g_{R} does not carry the ratio or the clip branch: on a token where the clip is active the true contribution of $\mathcal{L}_{\mathrm{PG}}$ to the logit gradient is zero, whereas g_{R} is not. Second, and more importantly, g_{R} omits the KL and entropy terms of Eq. (18) entirely. The KL term is not negligible at this scale: over the main 8-GPU run its value averages 0.146, so with $\beta_{\mathrm{KL}} = 0.01$ it contributes about 1.5×10^{-3} to the objective against an average $\mathcal{L}_{\mathrm{PG}}$ of 2.3×10^{-3}.

Proposition 1 should therefore be read for what it is: $\widetilde{g}_{\mathrm{D}}$ does not oppose the *outcome-driven policy-gradient term* in grouped-logit space. It is not a statement about $\nabla_z \mathcal{L}_{\mathrm{actor}}$, and we do not claim that the projected auxiliary gradient is non-opposing with respect to the regularization terms. Extending the construction to the full actor objective would mean detecting conflict against $\nabla_z \mathcal{L}_{\mathrm{actor}}$ rather than $\nabla_z \mathcal{L}_{\mathrm{PG}}$, which is a straightforward change to the quantity being compared but is not what the reported experiments do.

A.4 CONFLICT PROJECTION

Two separate thresholds act on the policy gradient, and they are applied to different quantities. The first is a *zero-RL* test on the norm itself: a turn with $\|g_R\| \leq 10^{-8}$ carries no usable outcome direction — this happens when every rollout in the GRPO group shares the same return, so the advantage vanishes — and such turns are removed from conflict handling altogether, their fate decided by `zero_rl_gradient.mode`. The second is a *reliability* test on the squared norm, which is what appears in the projection coefficient: the ratio $\langle g_D, g_R\rangle / \|g_R\|^2$ loses numerical meaning once $\|g_R\|^2 < \epsilon$, $\epsilon = 10^{-6}$ (equivalently $\|g_R\| < 10^{-3}$).

On a conflicting turn that passes both tests, Eq. (8) reads

$$\widetilde{g}_D = g_D - \mathrm{proj}_{g_R}^{\mathrm{opposite}}(g_D), \qquad \mathrm{proj}_{g_R}^{\mathrm{opposite}}(g_D) = \frac{\langle g_D, g_R\rangle}{\|g_R\|^2} g_R, \tag{19}$$

with the denominator exactly $\|g_R\|^2$, so $\langle \widetilde{g}_D, g_R\rangle = 0$ holds without approximation. A conflicting turn in the band $10^{-8} < \|g_R\| < \sqrt{\epsilon}$ fails the reliability test: its auxiliary gradient is set to zero rather than projected, which gives $\langle \widetilde{g}_D, g_R\rangle = 0$ there as well. Had we instead floored the denominator at ϵ, the projection would have been incomplete and would have left a residual $\langle g_D, g_R\rangle(1 - \|g_R\|^2/\epsilon) < 0$; the implementation retains a `clamp` only as a division-by-zero guard, and its output on those rows is discarded. The rate at which the band is entered is logged as `unreliable_rl_turn_rate`; we did not record it in the runs reported here, so we do not state a value for it. Note that the norm relation of Eq. (22) describes the first branch only: on a dropped turn $\widetilde{g}_D = 0$ rather than $\|g_D\||\sin\theta|$.

Non-conflicting turns are not projected, which preserves all the information on which the teacher agrees with the outcome direction. This guarantee is a first-order property of the local logit space; it is not equivalent to a global monotone performance guarantee in parameter space, since the Jacobian map, network nonlinearity and subsequent optimizer state can all produce higher-order effects.

Proof of Proposition 1. Proposition 1 in the main text asserts $\langle g_{\mathrm{R}}, \widetilde{g}_{\mathrm{D}}\rangle \geq 0$ and $\langle g_{\mathrm{R}}, g\rangle \geq \|g_{\mathrm{R}}\|^2$, where $g = g_{\mathrm{R}} + \widetilde{g}_{\mathrm{D}}$. There are two cases.

Non-conflicting turn ($\langle g_{\mathrm{R}}, g_{\mathrm{D}}\rangle \geq 0$): no projection is applied, $\widetilde{g}_{\mathrm{D}} = g_{\mathrm{D}}$, hence $\langle g_{\mathrm{R}}, \widetilde{g}_{\mathrm{D}}\rangle = \langle g_{\mathrm{R}}, g_{\mathrm{D}}\rangle \geq 0$.

Conflicting turn ($\langle g_{\mathrm{R}}, g_{\mathrm{D}}\rangle < 0$): substituting the projection,

$$\langle g_{\mathrm{R}}, \widetilde{g}_{\mathrm{D}}\rangle = \Big\langle g_{\mathrm{R}},\ g_{\mathrm{D}} - \frac{\langle g_{\mathrm{D}}, g_{\mathrm{R}}\rangle}{\|g_{\mathrm{R}}\|^2} g_{\mathrm{R}} \Big\rangle = \langle g_{\mathrm{R}}, g_{\mathrm{D}}\rangle - \frac{\langle g_{\mathrm{D}}, g_{\mathrm{R}}\rangle}{\|g_{\mathrm{R}}\|^2}\|g_{\mathrm{R}}\|^2 = 0. \tag{20}$$

If `orthogonal_keep_ratio` $= \rho \in [0, 1]$ is applied as well, then $\langle g_{\mathrm{R}}, \rho\, \widetilde{g}_{\mathrm{D}}\rangle = \rho \cdot 0 = 0$ still holds. Combining the two cases gives the first inequality, and the second follows:

$$\langle g_{\mathrm{R}}, g\rangle = \langle g_{\mathrm{R}}, g_{\mathrm{R}} + \widetilde{g}_{\mathrm{D}}\rangle = \|g_{\mathrm{R}}\|^2 + \langle g_{\mathrm{R}}, \widetilde{g}_{\mathrm{D}}\rangle \geq \|g_{\mathrm{R}}\|^2. \tag{21}$$

Hence for a sufficiently small learning rate η, the first-order change of $\mathcal{L}_{\mathrm{RL}}$ along $-g$ satisfies $-\eta\langle g_{\mathrm{R}}, g\rangle \leq -\eta\|g_{\mathrm{R}}\|^2$, i.e. it is no worse than a pure RL update.

The conclusion depends only on the projection step and is independent of the *norm* of the auxiliary gradient: for any $c > 0$, $\langle g_{\mathrm{R}}, c\,\widetilde{g}_{\mathrm{D}}\rangle = c \cdot 0 = 0$. The rescaling of the orthogonal component back to $\|g_{\mathrm{D}}\|$ (`preserve_norm`), enabled by default, therefore does not affect the guarantee, and the relative norm cap used in an earlier implementation is not required by it either and has been removed from the method.

The degenerate case $g_{\mathrm{R}} \to 0$. The proposition presumes $g_{\mathrm{R}} \neq 0$. When all rollouts in a group share the same outcome, the GRPO advantage vanishes and $\|g_{\mathrm{R}}\| \to 0$; the projection coefficient $\alpha = \langle g_{\mathrm{D}}, g_{\mathrm{R}}\rangle / \|g_{\mathrm{R}}\|^2$ then diverges and the denominator of the cosine also tends to zero, so "conflicts with the policy gradient" is mathematically undefined. The implementation accordingly excludes such turns from the conflict test (`conflict` $= (\cos < 0) \wedge \neg$`zero_rl` $\wedge (\|g_{\mathrm{D}}\| > \epsilon)$), and their fate is decided separately by `zero_rl_gradient.mode` rather than by Proposition 1.

A.5 Effect of the projection on the gradient norm

Correcting the direction is not free. Substituting the conflicting branch of Eq. (8) into the norm, and writing θ for the angle between g_{D} and g_{R},

$$\|\widetilde{g}_{\mathrm{D}}\|^2 = \|g_{\mathrm{D}}\|^2 - \frac{\langle g_{\mathrm{D}}, g_{\mathrm{R}}\rangle^2}{\|g_{\mathrm{R}}\|^2} = \|g_{\mathrm{D}}\|^2\big(1-\cos^2\theta\big), \qquad \text{that is} \quad \|\widetilde{g}_{\mathrm{D}}\| = \|g_{\mathrm{D}}\|\,|\sin\theta|. \tag{22}$$

In other words, **the projection necessarily shortens the auxiliary gradient, by exactly a factor** $|\sin\theta|$. A conflicting turn thus undergoes two effects at once: its direction is corrected, which is the intent of the method, and its magnitude is attenuated, which is a by-product. In the raw form the two are coupled.

The coupling can be undone, because **Proposition 1 is scale invariant**: for any $c > 0$,

$$\langle g_{\mathrm{R}},\, c\,\widetilde{g}_{\mathrm{D}}\rangle = c\,\langle g_{\mathrm{R}}, \widetilde{g}_{\mathrm{D}}\rangle = 0. \tag{23}$$

Rescaling the orthogonal component back to the original norm,

$$\widetilde{g}_{\mathrm{D}} \;\leftarrow\; \frac{\|g_{\mathrm{D}}\|}{\|\widetilde{g}_{\mathrm{D}}\|}\,\widetilde{g}_{\mathrm{D}} = \frac{\widetilde{g}_{\mathrm{D}}}{|\sin\theta|}, \tag{24}$$

leaves the direction unchanged, the orthogonality unchanged and Proposition 1 intact, while removing the attenuation on conflicting turns. The theory therefore constrains only the *direction* of the auxiliary gradient and says nothing about its *scale*: restoring the norm is a design choice the theory *permits* but does not *dictate*.

Accordingly, PROJOPD **applies the rescaling of Eq.** (24) **by default** (`preserve_norm` = True): the projection is responsible only for correcting direction, while the overall strength of the auxiliary signal remains under the sole control of λ. This choice does not affect Proposition 1, and its cost is one scalar division per conflicting turn.

The size of the compensation is set by $|\sin\theta|$, which depends closely on the projection granularity. At turn granularity $|\cos\theta|$ is small on conflicting turns, with a measured $|\sin\theta| = 0.997$, so the projection removes only about 0.3% of the norm; at token granularity the same diagnostic gives 0.585, i.e. the projection removes more than forty percent of it. **The finer the granularity, the more collateral damage the direction correction does to the magnitude**, and the less dispensable the rescaling becomes.

A.6 Formalizing the turn granularity

Let a turn contain L valid response tokens, each carrying one gradient component on every one of the $|\mathcal{S}| = K{+}1$ support slots, i.e. $g_{\mathrm{R}}, g_{\mathrm{D}} \in \mathbb{R}^{L\times(K+1)}$. The implementation flattens both into $L(K{+}1)$-dimensional vectors before taking the inner product:

$$\langle g_{\mathrm{R}}, g_{\mathrm{D}}\rangle_{\mathrm{turn}} = \sum_{t=1}^{L}\sum_{i=1}^{K+1} g_{\mathrm{R}}[t,i]\,g_{\mathrm{D}}[t,i] = \sum_{t=1}^{L}\big\langle g_{\mathrm{R}}[t,:],\, g_{\mathrm{D}}[t,:]\big\rangle, \tag{25}$$

that is, *the sum of the per-token inner products*, containing only the diagonal terms $t = s$. The norm likewise satisfies $\|g_{\mathrm{R}}\|^2 = \sum_t \|g_{\mathrm{R}}[t,:]\|^2$.

This is equivalent to neither of two other natural formulations. First, summing along the token dimension before taking the inner product introduces meaningless cross terms

$$\Big\langle \sum_t g_{\mathrm{R}}[t,:],\, \sum_s g_{\mathrm{D}}[s,:]\Big\rangle = \sum_t\sum_s \big\langle g_{\mathrm{R}}[t,:], g_{\mathrm{D}}[s,:]\big\rangle, \tag{26}$$

whose $t \neq s$ terms multiply logit components at different positions, even though the category coordinates at different token positions are not in correspondence. Second, normalizing per token and then averaging, $\frac{1}{L}\sum_t \cos_t$, erases magnitude information and gives a token whose gradient is negligible the same weight as one that is decisive.

The substance of the turn granularity is therefore: **the test and the projection coefficient are scalars taken at turn level, while the correction is applied per token**. α is determined jointly by all components of the turn and is then broadcast back to every token: $\widetilde{g}_{\mathrm{D}}[t,i] = g_{\mathrm{D}}[t,i] - \alpha\, g_{\mathrm{R}}[t,i]$. Moving to token level only restricts the summation above to a single t, leaving the rest of the derivation unchanged; moving to episode level would let conflicts on different turns cancel — a turn with $\langle g_{\mathrm{R}}, g_{\mathrm{D}}\rangle < 0$ can be masked by a positive inner product on another turn and thus go undetected.

A.7 EDGE CASES

OPD is skipped when $\|g_R\|$ falls below an implementation threshold, so that the teacher does not drive the policy on its own when no reliable outcome direction is available. When the planner output cannot be parsed or violates the field constraints, the student trajectory still participates in GRPO, but the OPD mask of that episode is set to zero. The student's sampled token is forced into the teacher's top-k; the residual bucket holds the probability of the entire remaining vocabulary, so the sampled-token support rate is 1, while tokens outside the top-k are represented only by one aggregate mass.

A.8 CHOOSING BETWEEN RKL AND FKL

We use the reverse KL:

$$D_{\mathrm{RKL}}(p_S \| p_T) = \sum_i p_{S,i} \log \frac{p_{S,i}}{p_{T,i}}, \qquad \frac{\partial D_{\mathrm{RKL}}}{\partial z_j} = p_{S,j} \left(\log \frac{p_{S,j}}{p_{T,j}} - D_{\mathrm{RKL}} \right), \tag{27}$$

where z are the student logits over the support. The forward KL is instead

$$D_{\mathrm{FKL}}(p_T \| p_S) = \sum_i p_{T,i} \log \frac{p_{T,i}}{p_{S,i}}, \qquad \nabla_z D_{\mathrm{FKL}} = p_S - p_T. \tag{28}$$

Two considerations motivate the reverse KL. The first is the difference between mode-seeking and mass-covering behaviour. The FKL imposes an unbounded penalty on tokens where $p_{T,i}$ is large and $p_{S,i}$ is small, and therefore forces the student to cover the entire support of the teacher; the RKL penalty is weighted by $p_{S,i}$, so the student may abandon regions in which the teacher has low confidence without being penalized. In our setting the teacher is not a stronger model but the same student re-scored under the planner context (§2): its reliable information is concentrated on the few actions the planner explicitly endorses, while its tail mostly reflects the student's own noise. Forcing the student to fit that tail, which is what the FKL does, adds no supervisory signal.

The second concerns the interaction with the sparse support. Vocabulary mass outside the support is merged into a single residual bucket (previous subsection). Under the FKL that bucket enters the loss with weight $p_{T,\mathrm{res}}$, and the teacher's mass on the bucket is an unresolved aggregate: weighting by it turns a quantity that cannot be attributed to any particular token into gradient. Under the RKL the bucket is weighted by $p_{S,\mathrm{res}}$, so once the student has concentrated its mass inside the support the bucket's contribution vanishes on its own.

Because the two divergences couple to the residual bucket differently, they are not related by substituting a single formula, and a strictly matched comparison would have to be designed separately. Related OPD and self-distillation work includes OPSD (Zhao et al., 2026), Rethinking OPD (Li et al., 2026) and SDAR (Lu et al., 2026); f-divergence sequence distillation offers a more general view of the divergence choice (Wen et al., 2023).

We use the RKL uniformly on ALFWorld, WebShop and Search-based QA; the two divergences share exactly the same sparse-support construction and conflict projection implementation.

B ALGORITHM AND IMPLEMENTATION DETAILS

B.1 FULL TRAINING LOOP

The algorithm executes in the following order. The planner, teacher and student share the same initial model parameters by default, but the plan context of the planner/teacher appears only in the re-scoring branch during training.

1. For every initial task observation, generate one tagged planner output and validate the `hard_constraints` and `plan` fields.
2. The student performs an on-policy rollout on observations without the plan context, storing the prompt, response, action, reward, done flag and trajectory id of every environment turn.
3. For every valid turn, insert the plan context into the teacher prompt and compute student/teacher grouped log-probabilities on the same sampled response.

4. Compute g_R from the GRPO group return and g_D from the RKL; the two are aligned on the sparse-logit support of the current turn.

5. Classify each turn: skip it if $\|g_R\| \leq 10^{-8}$ (no usable outcome direction); otherwise mark it conflicting when $\cos\langle g_D, g_R\rangle < 0$. Project a conflicting turn when $\|g_R\|^2 \geq \epsilon = 10^{-6}$, and set its auxiliary gradient to zero when $\|g_R\|^2 < \epsilon$. Back-propagate $g_R + \widetilde{g}_D$ through a detached surrogate.

6. Update the actor; neither the planner/teacher computation graph nor the text context enters the deployed model at inference time.

The procedure is critic-free: no value function participates in the advantage computation. It is consistent with the score-function view of REINFORCE/policy gradients (Williams, 1992; Sutton & Barto, 2018), but uses GRPO's group-normalized return as g_R.

B.2 Sparse support and complexity

Write V for the full vocabulary size, L for the response length and $K = 64$ for the number of sparse support slots. Teacher re-scoring still requires one model forward pass, but the extra tensors for the projection and the KL shrink from $O(LV)$ to $O(L(K+1))$; the residual bucket retains the total probability mass outside the support. Top-k selection is performed on the logits, and the last slot is replaced when the sampled token is not in the top-k. We record the pure top-k mass, the 64-slot mass after replacement and the sampled-token-outside rate, so that a single average cannot hide insufficient tail coverage.

Table 3: Teacher support coverage on Qwen2.5-3B ALFWorld. Computed over 125 episodes with a valid planner output, comprising 358,349 response-token positions; coverage is the probability mass that the support tokens carry under the teacher's full-vocabulary softmax, not the fraction of the vocabulary the support tokens occupy.

Support	Mean	p01	p05	Median	Min	$< 95\%$
Top-16	99.1292	88.4361	94.9034	99.9819	50.4437	5.1255
Top-32	99.5874	93.9765	97.5910	99.9935	65.0573	1.5287
Top-64	99.7885	96.6415	98.7887	99.9974	75.0303	0.3759

As an additional check, the fraction of student sampled tokens falling outside the teacher's original top-64 is 0.00251%. Forcing the sampled token into the 64-slot support therefore has a negligible effect on the total mass, while still guaranteeing that OPD's sampled-token support rate is always 1. The top-16/32/64 comparison is a support-coverage diagnostic and is not equivalent to a full training ablation; production training always fixes top-64.

B.3 Parse failures and data filtering

Planner outputs follow a fail-closed rule: if any required field is missing, the plan is empty, illegal tags are repeated, or an environment action is included, the OPD mask of that episode is set to zero, but the student response is not removed from the GRPO batch. Invalid actions returned by the environment are also kept in the trajectory and counted separately under action validity; retaining only successful trajectories would otherwise introduce a selection bias into both training and evaluation.

C Benchmarks, metrics and evaluation protocols

The Avg column of each environment follows the convention of its own baselines: for ALFWorld it is the overall success rate across all evaluated episodes (successful episodes divided by total episodes) rather than the unweighted mean of the six per-subtask rates, and for Search-based QA it is the unweighted macro average over the seven subsets.

C.1 ALFWORLD

ALFWorld (Shridhar et al., 2021) turns the ALFRED household tasks into a text interaction environment. We use the 140 evaluation tasks, covering Pick and Place, Pick Two and Place, Clean Then Place, Cool Then Place, Heat Then Place and Look At In Light. The executor produces one admissible action per turn, the environment returns a text observation, and an episode runs for at most 50 environment turns. The final metric is the episode success rate of reaching the task goal; results are aggregated by episode count. The training reward of the current wrapper is 10 on success and 0 on failure, and `val/test_score` is not a separate graded metric. We also keep the observation, action, reward, done flag and task type of every trajectory, so that repeated actions and failure causes can be re-examined.

C.2 WEBSHOP

WebShop (Yao et al., 2022) asks the model to search for products, browse results, select attributes and complete a purchase. The main experiments use the 15-turn horizon and GiGPO's fixed 128 validation goals adopted by the quoted baselines (Feng et al., 2025); the 50-turn setting common in the original WebShop paper is a different protocol, and its full500/50-turn result is reported in this appendix as a low-variance supplement. We report two distinct but complementary metrics:

$$\text{Score} = 100 \cdot \frac{1}{N} \sum_{i=1}^{N} s_i, \qquad \text{Succ} = 100 \cdot \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}[s_i = 1], \tag{29}$$

where s_i is the raw WebShop partial task score. Score reflects the degree of attribute matching while Succ counts only fully satisfied tasks, so the two are not related by a fixed factor of ten. Table 4 reports the supplementary full500 evaluation.

Table 4: Supplementary WebShop full500 evaluation of Qwen2.5-3B PROJOPD. It is not mixed with the fixed 128-task main table.

Method	Tasks	Score	Succ
PROJOPD	500	89.78	77.40

This full500 evaluation covers 500 validation goals, of which 387 succeed.

C.3 SEARCH-BASED QA

We adopt the Search-based QA setting of Search-R1 (Jin et al., 2025): the retriever is E5-base-v2, the corpus is Wiki-18, each retrieval returns the top-3 passages, and one episode runs at most 4 interaction turns. After reasoning inside `<think>`, the executor either issues one retrieval or gives a final answer. The training set is formed by sampling 9,600 examples without replacement from each of the NQ and HotpotQA train splits under a fixed seed 0, giving 19,200 in total; during training a stratified 512-example monitor validation keeps evaluation cost down. Final results are computed on the seven subsets NQ, TriviaQA, PopQA, HotpotQA, 2WikiMultiHopQA, MuSiQue and Bamboogle, reporting answer accuracy on each together with their unweighted macro average. All comparisons share the same retrieval and answer-normalization settings, and none of them substitutes the context shipped with the data files for external retrieval.

D BASELINE AND ABLATION DETAILS

D.1 OPTIMIZATION BASELINES

GRPO (Shao et al., 2024). No planner or teacher; only the group-relative advantage of the outcome reward.

Raw OPD. Keeps the planner-conditioned teacher and the RKL, but performs no conflict projection.

No-Projection. Identical to the previous entry, used to isolate the contribution of the orthogonal projection; the removal of the conflicting component is simply never triggered.

Conflict-Drop. Discards the OPD gradient entirely on conflicting turns, testing whether "keeping the orthogonal information" beats a hard drop.

RKL/FKL. Fixes the planner output and the support construction, and replaces only the divergence direction.

The literature baselines in the main table and the papers that introduced them are: OPSD (Zhao et al., 2026), SDAR (Lu et al., 2026), RLSD (Yang et al., 2026), Skill-SD (Wang et al., 2026) and StepOPSD (Zhang et al., 2026); for GRPO see Shao et al. (2024). Vanilla, Skill-Prompt*, Skill-GRPO and GRPO+OPSD are standard configurations combining these methods and do not correspond to a separate original paper.

D.2 Description of the literature baselines

The comparisons fall into three groups: models without post-training, outcome-only reinforcement learning, and self-distillation reinforcement learning. Unless a method is marked with *, evaluation uses only the standard task prompt and the interaction history returned by the environment; * means a retrieved skill is additionally supplied at validation and test time.

Vanilla. The frozen instruction-tuned backbone, evaluated actor-only with no post-training and no planner or skill context. It separates the model's existing ability from the gain brought by RL and distillation.

Skill-Prompt*. The same parameters as Vanilla, with a retrieved task-relevant skill prepended to the context at validation/test time, measuring the direct value of inference-time guidance without any parameter update.

GRPO. GRPO (Shao et al., 2024) is the most important outcome-only baseline. For one prompt it samples G responses, computes the group-relative return

$$\hat{A}_i = \frac{R_i - \mathrm{mean}(R_{1:G})}{\mathrm{std}(R_{1:G}) + \epsilon}, \tag{30}$$

and broadcasts the episode's advantage to its valid response tokens. It produces no planner plan, adds no teacher prefix and computes no OPD loss, so it answers whether outcome-driven RL alone is already sufficient.

Skill-GRPO / Skill-GRPO*. GRPO with a retrieved skill injected into the training prompt. Skill-GRPO removes the skill at inference, testing whether the guidance has been internalized by the executor; Skill-GRPO* keeps it, separating "having learned the skill" from "still depending on it". This work treats the todo list and constraints as an explicit experimental variable and does not conflate them with an external skill library.

OPSD. On-policy self-distillation (Zhao et al., 2026): a teacher branch conditioned on training-only privileged context re-scores the tokens the student has sampled and produces dense token-level targets through distribution matching. Teacher outputs are detached and the privileged context never enters inference. The idea is to derive the distillation supervision from on-policy experience and a same-origin model, avoiding the teacher/student distribution mismatch of the offline setting.

GRPO+OPSD. The most direct joint baseline: optimize the GRPO outcome loss and a plain token-level OPSD loss together, without gradient projection. It answers whether the gain comes merely from adding one more teacher loss, and together with our Raw OPD arm forms the mechanistic control.

Skill-SD. Supplies the retrieved skill only to the teacher branch and trains the student to absorb the skill-conditioned guidance through an importance-weighted distillation loss, requiring no skill at evaluation (Wang et al., 2026).

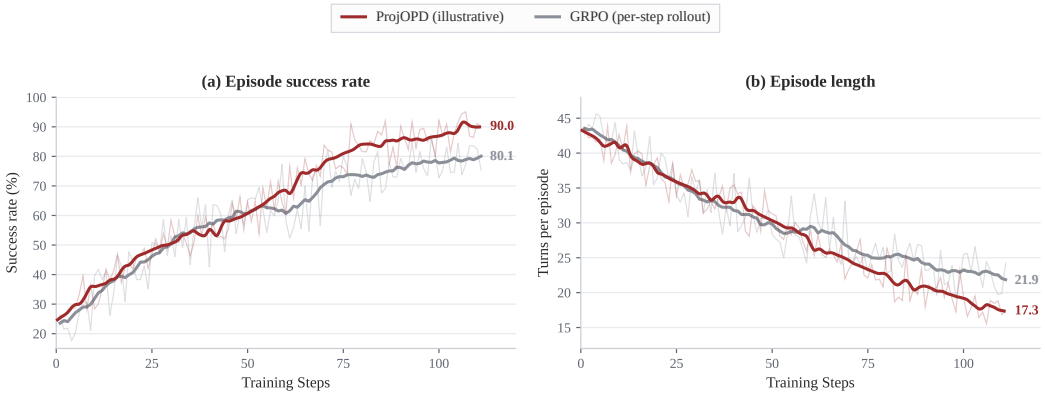


Figure 3: **Training dynamics.** Qwen2.5-3B on ALFWorld.

RLSD. RLSD (Yang et al., 2026) converts the teacher–student log-probability gap into a bounded coefficient that scales the magnitude of each token's GRPO update, while the update direction is still set by the outcome advantage. Like ours it avoids replacing the RL signal outright, but it applies a scalar gate rather than removing the conflicting OPD component in gradient space.

SDAR. Adds a separately gated auxiliary self-distillation loss on top of GRPO, leaving the original GRPO advantage unchanged and modulating each teacher signal with a bounded gate (Lu et al., 2026). It likewise attends to the unevenness of the OPD signal across tokens, but does not address the geometry between the RL and OPD gradients.

StepOPSD. Applies the teacher–student distillation signal at the turn (step) level rather than per token, but uses each step's local signal in isolation (Zhang et al., 2026). Table 5 summarizes how each of these relates to the implementation compared here.

Table 5: Correspondence between the published methods and the implementations compared here. A baseline name denotes an explicit, reproducible experimental configuration.

Method	Original focus	Correspondence / difference here
Vanilla	no post-training	frozen backbone; measures the model's own ability
Skill-Prompt*	skill context at inference	measures inference-only guidance
GRPO	group-relative outcome advantage	RL-only main baseline; no planner or teacher
Skill-GRPO / Skill-GRPO*	skill-conditioned GRPO	control for keeping the skill at train/test
GRPO+OPSD	outcome RL + token OPD	joint baseline without projection
RLSD	gap-based scalar gate	scalar modulation, no vector projection
OPSD	on-policy self-distillation	background of the Raw OPD family; keeps the teacher signal
SDAR	token-level self-distilled advantage	fine-grained weighting; not projection
Skill-SD	skill-aware teacher signal	background for a skill-conditioned teacher
StepOPSD	turn-level distillation signal	each step's local signal used in isolation; no cross-turn direction constraint

D.3 Training dynamics

Figure 3 shows training dynamics. The checkpoint-evaluation success rate of ProjOPD reaches 90.0% at step 111, with a mean episode length of 17.3 turns. The GRPO rollout curves end at

approximately 80.1% success and 21.9 turns after smoothing. These curves use different measurements (checkpoint evaluation versus training rollout), so their vertical difference should not be read as a point-by-point comparison.

D.4 Computational cost

The training overhead of ProjOPD consists of one planner generation per episode and sparse teacher re-scoring of the student's response, without additional environment rollouts. On Qwen2.5-3B with ALFWorld, adding the planner and teacher increases per-step time from 629 s to 647 s ($+2.8\%$) and reduces token throughput from 1030 to 1008 tok/s (-2.2%). Only the executor is deployed at test time, with the inference cost of an ordinary actor.

E Prompts and training implementation

This section reports the prompt contract and the training configuration needed to reproduce the experiments. The planner runs only during training; formal evaluation loads the executor/actor checkpoint alone and removes both the planner output and the teacher context.

E.1 Planner prompt

The planner receives the initial observation of the episode. To stop the planner from simply substituting for the executor, the trailing action-output instructions of the environment are stripped from its input, and the planner is forbidden to emit `<action>` or to call tools. The actual template is:

Planner prompt (shared by ALFWorld / WebShop / Search-based QA)

```
You are the Planner. Produce an episode-level guide for the
    Executor
before execution. Do not choose or emit an environment action. Use
only facts in the initial task state. When an object location or
other fact is unknown, write a conditional search strategy instead
of inventing it.

Field meanings:
- hard_constraints: result conditions that must be true for
    success.
- plan: the ordered strategy for satisfying the hard constraints,
  including conditional branches when needed.

First reason briefly inside <think>...</think>. Then return
    exactly:
<output>
<hard_constraints>
1. one required result condition
</hard_constraints>
<plan>
step1: first strategy step
step2: next strategy step
</plan>
</output>
Keep every section non-empty and concise. Do not use <action>.

Initial task state:
{initial_observation}
```

The output is parsed into {`hard_constraints:[...], plan:[...]`}. Every section must be non-empty, and plan steps are normalized to `step1, step2, ...`. A parse failure is retried at most once; samples that still fail keep their GRPO rollout but receive no OPD.

This template is identical across all three benchmarks; only {`initial_observation`} changes with the environment: ALFWorld inserts the room description and the task goal, WebShop

the initial product search page and the purchase requirement, and Search-based QA the question to be answered. The planner is unaware of the environment type and receives no environment-specific instruction.

The planner's generation budget is 2048 tokens and its plan-step limit is 50.

E.2 Sample planner outputs: Search-based QA

Below are Qwen2.5-3B planner outputs for the same question before and after training on Search-based QA. Both use the template of the previous subsection with `temperature`$= 0.4$, no retry and no environment access, so they show what the planner actually produces rather than what survives a retry.

Sample planner output: Search-based QA, single-hop (before vs. after RL)

```
Q (NQ): who got the first nobel prize in physics

--- Base (Qwen2.5-3B-Instruct)
<think>
To answer the question "who got the first Nobel Prize in Physics",
    I need to
identify the person who won the first Nobel Prize in Physics. This
    requires
searching for historical records related to the Nobel Prizes,
    specifically
focusing on the Physics prize's first award year.
</think>
<output>
<hard_constraints>
1. the answer must be a person who received the first Nobel Prize
    in Physics
</hard_constraints>
<plan>
step1: Search for the first year the Nobel Prize in Physics was
    awarded.
step2: Once the first year is known, identify the laureate of that
    year.
</plan>
</output>

--- After RL (step 150)
<think>
To answer the question "who got the first Nobel Prize in Physics",
    I need to
find the person associated with the first Nobel Prize in Physics.
    The Nobel
Prize in Physics was first awarded in 1901, and I will search for
    the laureate
of that year.
</think>
<output>
<hard_constraints>
1. the person who received the first Nobel Prize in Physics is
    identified
</hard_constraints>
<plan>
step1: search for the first Nobel Prize in Physics laureate
</plan>
</output>
```

Both preserve the decomposition structure on multi-hop questions:

Sample planner output: Search-based QA, multi-hop (before vs. after RL)

```
Q (HotpotQA): Were Scott Derrickson and Ed Wood of the same
    nationality?

--- Base                          --- After RL (step 150)
step1: Confirm Scott              step1: Search for Scott
       Derrickson's nationality.         Derrickson's nationality.
step2: Confirm Ed Wood's          step2: Search for Ed Wood's
       nationality.                      nationality.
step3: Compare the                step3: Compare the nationalities
    of
       nationalities.                    Scott Derrickson and Ed
           Wood.
```

Over 16 questions (four each from NQ / TriviaQA / HotpotQA / MuSiQue), the rate of format-valid outputs rises from $14/16$ at base to $15/16$, the mean generation length falls from 145 to 136 tokens, and the mean number of plan steps falls from 2.4 to 2.0. The trained plans are shorter and more direct: the base model often writes "first look up the year, then look up the person given the year", whereas after RL the model puts the background knowledge into `<think>` and merges the two into a single retrieval step. This differs from the degeneration observed on ALFWorld, and the decomposition structure and field format stay unchanged.

E.3 What the privileged context looks like across ablation arms

Table 2 shows that an online 3B planner outperforms statically pre-generated context from the same model by 6.43 points and a static 30B planner by 2.86. All three arms share the same executor, the same teacher mechanism and the same projection, so the difference must lie in the context itself. This subsection shows what each arm actually produces.

Static planners. In these arms the plan is fixed before training by a frozen model and does not change as the policy updates. Both outputs below are for the same ALFWorld task, `put two alarmclock in desk`.

Static 3B planner (Qwen2.5-3B-Instruct, pre-generated)

```
<hard_constraints>
1. there are two alarmclocks in the room
</hard_constraints>
<plan>
step1: go to drawer 1
step2: if drawer 1 contains an alarmclock, go to desk 1 and put
   the alarmclock there; otherwise, go to drawer 2
step3: if drawer 2 contains an alarmclock, go to desk 1 and put
   the alarmclock there; otherwise, go to shelf 1
step4: if shelf 1 contains an alarmclock, go to desk 1 and put
   the alarmclock there; otherwise, go to drawer 3
step5: inventory to check for any alarmclocks
step6: if no alarmclocks are found, repeat steps 1-5 in a
   different order
</plan>
```

Static 30B planner (Qwen3-30B-A3B-Instruct, pre-generated)

```
<hard_constraints>
1. Two alarm clocks must be placed on desk 1.
</hard_constraints>
<plan>
step1: Search for alarm clocks by checking the locations of bed
   1, bed 2, drawer 1, drawer 2, drawer 3, garbagecan 1,
```

```
   laundryhamper 1, shelf 1, and desk 1, in that order, using
   'go to' actions and 'look' to inspect each.
step2: Once two alarm clocks are found, pick them up one at a
   time using 'take' (if available), then go to desk 1 and place
   each on the desk using 'put on desk 1'.
</plan>
```

The 30B plan is visibly the better-written of the two: it states the goal precisely, enumerates the candidate containers in one sweep and names the concrete action verbs.

The online planner. The online planner is the current policy itself, queried once at the start of each episode, so its plans change as training proceeds. The pair below is the same ALFWorld task before and after training.

Online planner, before training (Qwen2.5-3B-Instruct)

```
<hard_constraints>
1. A clean butterknife must be on diningtable 1.
</hard_constraints>
<plan>
step1: go to cabinet 1
step2: go to cabinet 2
step3: go to cabinet 3
...
step12: go to cabinet 12
step13: go to countertop 1
step14: go to countertop 2
step15: inventory
step16: put a clean butterknife on diningtable 1
</plan>
```

Online planner, after training (step 111, the main method)

```
<hard_constraints>
1. the butterknife must be clean
</hard_constraints>
<plan>
step1: go to countertop 1
step2: if countertop 1 does not contain a butterknife, go to
   countertop 2; otherwise, check countertops 1 and 2 for a
   clean butterknife
step3: if no butterknife is found on the countertops, go to
   cabinets 1-27 to search for a clean butterknife
step4: if a clean butterknife is found in the cabinets, go to
   the diningtable 1 to put the butterknife
step5: if no butterknife is found in the cabinets, go to
   drawers 1-12 to search for a clean butterknife
step6: if a clean butterknife is found in the drawers, go to
   the diningtable 1 to put the butterknife
step7: if no butterknife is found in the drawers, the task
   cannot be completed
</plan>
```

Training changes the plan in three ways. The untrained planner emits a flat enumeration, one container per step, and exhausts the step budget before reaching the goal. The trained planner groups containers into ranges (`cabinets 1-27`, `drawers 1-12`), orders them by where the object is likely to be, and makes each step conditional on the outcome of the previous one. Over the 140 evaluation episodes the mean number of plan steps falls from 8.67 to 7.25.

Why the online context wins. The comparison is instructive precisely because the online planner is *not* the one that writes the cleanest plans: the 30B plan above is better written than anything the 3B

policy produces, yet the online arm wins by 2.86 points. What separates them is where the context sits relative to the student's own distribution. The teacher is the same policy conditioned on the context, so the distillation target is reachable only if that context lies in a region the policy already assigns high probability. A plan written by a different model, or by a frozen snapshot of the same model taken before training, may be better prose while being further from what the policy is prepared to follow. This is the distributional-absorbability argument of §2, and it is visible in the teacher–student log-probability gap measured on the same 229 probe pairs: the distilled branch reaches a mean gap of -0.124 under the static 30B context and -0.156 under the skill context, against -0.226 for the online planner, while the fraction of tokens the teacher raises above the student is comparable across all three (0.59, 0.61, 0.52). The static contexts pull the whole distribution slightly closer without selecting better tokens — the behaviour a reachable but uninformative target would produce.

E.4 Teacher context and input alignment

The teacher generates no new action and does not modify the response the student has already sampled. It only inserts the following into the same observation:

Teacher context: the only difference between the student and teacher inputs

```
**Episode-Level Plan**
Refer to this plan when deciding what action to take:
[<output> ... <hard_constraints> ... <plan> ... </plan> ...
    </output>]
```

Student and teacher therefore compute the log probability of the same token response, the only difference being that the teacher can see the episode-level plan. The context is inserted after the task state and before the current action instruction, and is removed entirely at inference time.

E.5 ALFWorld executor prompt

Executor prompt on ALFWorld

```
You are an expert agent operating in the ALFRED Embodied
    Environment.
Your task is to: {task_description}
Prior to this step, you have already taken {step_count} step(s).
Below are the most recent {history_length} observations and the
corresponding actions you took: {action_history}
You are now at step {current_step} and your current observation
    is:
{current_observation}
Your admissible actions of the current situation are:
[{admissible_actions}].

Now it is your turn to take an action. First reason step-by-step
inside <think>...</think>, then choose one admissible action
    inside
<action>...</action>.
```

ALFWorld accepts only one action from the admissible-action list. The environment issues a terminal reward on success; the maximum number of interaction steps is 50 in both training and formal evaluation.

E.6 WEBSHOP EXECUTOR PROMPT

Executor prompt on WebShop

```
You are an expert autonomous agent operating in the WebShop
E-commerce environment. Your task is to: {task_description}.
Prior to this step, you have already taken {step_count} step(s).
Below are the most recent {history_length} observations and
    actions:
{action_history}
You are now at step {current_step} and your current observation
    is:
{current_observation}.
Your admissible actions are:
[
{available_actions}
].

Reason step-by-step inside <think>...</think>, then choose exactly
    one
admissible action inside <action>...</action>.
```

The WebShop environment returns both a continuous task score and a binary success. The training reward uses the environment reward returned by the wrapper, and the paper reports both Score and Succ. The standard horizon is 50.

E.7 SEARCH-BASED QA EXECUTOR PROMPT

Executor prompt on Search-based QA

```
You are an expert agent tasked with answering the given question
step-by-step. Your question: {task_description}

Prior to this step, you have already taken {step_count} step(s).
The history contains past queries in <search>...</search> and
    results
in <information>...</information>:
{memory_context}

First reason inside <think>...</think>. Then choose exactly one:
(1) retrieve evidence with <search>your query</search>; or
(2) return the final short answer with <answer>answer</answer>.
Do not perform both actions in the same turn.
```

Each `<search>` call queries a Wiki-18 index built with E5-base-v2 and FAISS-GPU and returns the top-3 passages; each question allows at most 4 interaction turns, and the history keeps the most recent 4 turns. Reward is produced only by matching the answer at termination; intermediate retrieval receives no additional shaping reward.

E.8 SHARED OPTIMIZATION AND PROJOPD PARAMETERS

By default we sample 8 rollouts per prompt; the teacher uses 64 sparse support slots and is guaranteed to contain the student's sampled token, with the remaining vocabulary mass merged into a residual bucket.

The OPD coefficient is set to $\lambda = 0.01$ and the gate temperature to $\beta = 5.0$, following SDAR (Lu et al., 2026), which adopts the same $\mathcal{L}_{\mathrm{RL}} + \lambda\mathcal{L}_{\mathrm{OPD}}$ form and selects 0.01 after sweeping $\lambda \in \{0.001, 0.01, 0.1\}$. The projection granularity is one environment turn.

Table 6: Optimization and PROJOPD hyper-parameters shared by all main experiments. Settings not listed use the VERL defaults.

Item	Setting	Item	Setting
Optimizer	AdamW	Learning rate	1×10^{-6}
Weight decay	0.01	Gradient clip	1.0
PPO epochs/update	1	PPO clip range	0.2
GRPO group size	8	Advantage estimator	GRPO
Training rollout	$T = 1.0$, top-$p = 1$	KL loss	low-var KL, coef. 0.01
Entropy coefficient	0.001	Invalid-action penalty	0.01–0.1
Planner max tokens	2048	Planner retries	1
OPD coefficient λ	0.01	Gate β	5.0
Divergence	Reverse KL	Projection granularity	environment turn
Sparse support	Teacher top-64 + sampled token	Residual bucket	enabled
Conflict threshold	cosine < 0	Orthogonal keep ratio	1.0
Zero-RL rule	skip if $\|g_R\| \leq 10^{-8}$	Projection granularity	environment turn
Reliability floor ϵ	10^{-6} on $\|g_R\|^2$	Below ϵ	drop auxiliary gradient
Model parallelism	FSDP data parallel	Actor optimizer offload	disabled
Reference parameter offload	enabled	Gradient checkpointing	enabled

Table 7: Per-environment training and evaluation configuration. Settings shared across all environments are given in Table 6; only the parts that vary with the environment are listed here. Updates are counted over one epoch, and the 3553 ALFWorld training tasks drop the last incomplete batch. Max interaction turns is the environment horizon, not the number of optimizer updates.

	ALFWorld	WebShop	Search-based QA
Training set size	3553	6410	19200
Optimizer updates	222	400	150
Train batch size	16	16	128
Rollout group size G	8	8	8
Max prompt length	4096	5120	4096
Max response length	512	512	512
Max interaction turns	50	50	4
Rollout temperature (train / val)	1.0 / 0.4	1.0 / 0.4	1.0 / 0.4
GPUs	4	4	8
Validation batch	140	20	128
Retriever top-k	–	–	3
Retriever encoder	–	–	E5-base-v2
Retriever corpus	–	–	Wiki-18
Evaluation set	140 tasks	fixed 128	full 51,713

E.9 PER-TASK TRAINING AND EVALUATION CONFIGURATION

The Search training set consists of 9,600 instances sampled without replacement from each of the NQ and HotpotQA training splits with a fixed seed of 0, 19,200 in total. Only a 512-instance stratified monitor validation is used during training to control evaluation cost; final results must be computed on the full 51,713-instance set spanning NQ, TriviaQA, PopQA, HotpotQA, 2WikiMultiHopQA, MuSiQue and Bamboogle. The ALFWorld main table uses the overall success rate; the Avg is the total number of successful episodes divided by the total number of episodes, not a plain average of the six per-category success rates. The WebShop main table uses Score and Succ on the fixed 128 tasks, with full500 serving only as an appendix supplement.

F DIAGNOSTIC METRIC DEFINITIONS

For valid turns, the conflict rate is

$$r_{\text{conflict}} = \frac{1}{M} \sum_{m=1}^{M} \mathbf{1}[\langle g_D^{(m)}, g_R^{(m)} \rangle < 0], \tag{31}$$

and the projection rate counts only those conflicting turns on which a projection was actually performed. The cosines before and after projection are $\langle g_D, g_R \rangle / (\|g_D\| \|g_R\|)$ and

$\langle \widetilde{g}_D, g_R \rangle / (\|\widetilde{g}_D\| \|g_R\|)$ respectively. Teacher support mass is the summed probability mass of the explicit support (including the sampled-token replacement); it is not a success rate and does not indicate the teacher's calibration.

MFU, allocated/reserved GPU memory and CPU host memory are resource diagnostics of the training system, not task-performance metrics. Allocated is the memory the allocator currently holds for tensors, reserved is what the caching allocator has reserved from the CUDA runtime, and neither can be equated with task success rate.

G REPRESENTATIVE EXECUTION TRAJECTORIES

The fragments below come from saved actor-only evaluation JSONL files. They show only the readable action sequence and never treat the training-time planner output as an inference-time input. The ALFWorld example is episode 0 of Qwen2.5-3B, whose task is to put a clean butterknife on the diningtable; the model receives the success reward on turn 16:

Representative trajectory: ALFWorld, Qwen2.5-3B, episode 0

```
0  go to cabinet 1
1  open cabinet 1
2  go to cabinet 2
3  go to cabinet 3
4  go to diningtable 1
5  go to sinkbasin 1
6  go to cabinet 1
7  go to sinkbasin 1
8  go to diningtable 1
9  take butterknife 1 from diningtable 1
10 go to sinkbasin 1
11 clean butterknife 1 with sinkbasin 1
12 move butterknife 1 to diningtable 1
16 move butterknife 1 to diningtable 1  [reward=10, won=true]
```

This example also shows that a high success rate does not mean the shortest action sequence: repeated actions occur in the middle, and the training objective of the method is to improve the final outcome rather than to label every action correct after the fact.

The WebShop example is a successful episode from the full500 JSONL, whose goal is to search for and buy brown-hole228 men's loafers in size 8.5 for less than 70 dollars:

Representative trajectory: WebShop, Qwen2.5-3B

```
0 search[r.brown-hole228 men's loafers slip-ons size 8.5 price <
    70.00]
1 click[b07s7hdc88]
2 click[r.brown-hole228]
3 click[8.5]
4 click[buy now]                    [task_score=1, reward=10]
```

Success in WebShop means `task_score == 1`, not merely having executed `buy now`; the appendix trajectory files retain the full page observations, all actions and the terminal state.