HADP: A HIERARCHICAL ADAPTIVE DIALOGUE POLICY FOR ALIGNING WHAT LLM AGENTS SAY AND DO IN MOBA TEAM COORDINATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Competitive multiplayer games require AI teammates not only to choose useful actions but also to honor what they tell players. We study this say–do gap in MOBA team coordination and propose Hierarchical Adaptive Dialogue Policy (HADP). HADP grounds player requests in a shared Mission representation, generates coordinated action and dialogue responses, and reviews candidate outputs against evolving context and memory. We evaluate alignment at three levels: semantic consistency between utterances and issued plans, execution consistency between activated plans and agent behavior, and interaction consistency of commitments across turns. In the evaluated closed-loop scenarios in Honor of Kings, HADP achieves 91.5%, 86.3%, and 76.5% consistency at these levels, respectively, compared with 81.3%, 67.5%, and 51.5% for a flat policy with budget-matched self-refinement. Module ablations show that removing Understanding lowers semantic consistency by 18.0 percentage points, while removing Adaptive Review lowers interaction consistency by 15.3 points. These results support complementary roles for mission grounding and contextual review within the tested setting. Supporting trace-based experiments assess macro-action prediction and command quality after supervised fine-tuning, GRPO, and command-channel self-distillation.

1 INTRODUCTION

Effective AI teammates in competitive multiplayer games must jointly decide what to do and what to say in real time (Ye et al., 2020; 2022). In the Commander Mode of Honor of Kings (Wei et al., 2022; Qu et al., 2023) (Figure 1), this poses a concrete Dialogue Policy Learning (DPL) challenge: dynamically aligning AI utterances with player commands and executed behaviors. We explicitly evaluate this plan-behavior agreement rather than human-likeness or perceived agency.

Flat DPL architectures (Wen et al., 2017) often suffer from a *say–do gap*, where utterances diverge from executed actions. We address three underlying sources of inconsistency: (i) *input grounding* of player references, (ii) *action–dialogue coordination*, and (iii) *commitment tracking* as conditions change. To resolve them, we introduce explicit mission grounding and contextual review, evaluating their contributions through closed-loop ablations.

The Hierarchical Adaptive Dialogue Policy (HADP) coordinates actions and dialogue via a shared Mission representation across three layers: (i) an Understanding module that converts game state and player inputs into structured missions; (ii) an LLM-based Response Generator (DeepSeek-AI et al., 2024; Sun et al., 2024; Bai et al., 2023; Yang et al., 2025) that generates mission-conditioned actions and utterances; and (iii) an Adaptive Review module that validates candidates against STM/LTM for multi-aspect consistency before execution.

In summary, this work makes three key contributions:

- We operationalize the *say–do gap* in multi-agent DPL through three formal metrics covering semantic agreement, plan execution, and interaction-level commitment consistency.
- We propose HADP, a hierarchical DPL framework that unifies action and dialogue through a shared *Mission* and explicit commitment review over STM/LTM context.

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



Figure 1: In Commander Mode of **Honor of Kings**, the AI interprets the game state and player signals to engage in dialogue.

- We demonstrate that HADP outperforms budget-matched flat baselines by up to 25% across closed-loop consistency metrics in *Honor of Kings*, with thorough ablations isolating the impact of each architectural layer.

2 Related Work

Task-oriented Dialogue System. Traditional task-oriented dialogue (TOD) systems follow a modular pipeline (Li et al., 2020; Wu et al., 2019), effective in constrained domains such as restaurant booking or travel planning (Zhang et al., 2020), and are evaluated mainly on task completion and dialogue efficiency (Walker et al., 2000). In games, however, dialogue is not about efficiency but about coherence across linguistic and behavioral actions under evolving context—a regime where modular pipelines struggle to scale.

Memory and Persona Consistency in NPCs. Generative agents (Park et al., 2023) log observations in natural language and periodically synthesize higher-level reflections to plan actions;Hu et al. (2024) survey LLM-based game agents across diverse game genres, reviewing architectural components such as perception, memory, and action; Li et al. (2026) model character profiles through high-dimensional programmable persona spaces to support multi-turn dialogue consistency. This line of work studies memory-based behavior and persona-conditioned dialogue, primarily in sandbox or narrative settings. We instead evaluate agreement between utterances, issued plans, and executed behavior in competitive multiplayer coordination; persona consistency and perceived human-likeness are distinct from these behavioral criteria.

Multi-Agent Cooperation. Cooperative multi-agent reinforcement learning has achieved superhuman team play in video games—e.g., AlphaStar (StarCraft II) and OpenAI Five (Dota 2) (Li et al., 2025)—demonstrating that agents can learn complex coordination in dynamic environments. These results establish strong tactical coordination, but do not by themselves evaluate whether player-facing dialogue agrees with the agents' subsequent actions.

LLM-Powered NPCs. LLMs have recently been explored as the core engine of NPC behavior: Zhou et al. (2023) integrate dialogue exchanged with NPCs into an RL agent's policy graph as a training signal; Gallotta et al. (2024) observe that LLMs generate dynamic, contextually appropriate responses, making interactions less repetitive; Song (2025) build a cross-platform LLM-NPC system whose cloud-logged chat history maintains coherence across platforms. These prototypes target narrative-rich or sandbox environments, whereas our MOBA setting evaluates whether dialogue remains consistent with tactical plans and subsequent actions under rapidly changing game states.

RL-Based Post-Training for LLM Reasoning.

Search-R1 Jin et al. (2025) extends outcome-reward RL to retrieval-augmented reasoning by interleaving reasoning with multi-turn search. While post-training heavily uses verifiable outcome rewards via GRPO (Shao et al., 2024; DeepSeek-AI, 2025), existing work like Liao et al. (2025) applies SFT-plus-GRPO merely to offline, single-channel MOBA trace prediction without human interaction. In contrast, HADP operates in live, conversational team play, where actions and utterances are generated jointly and any plan-utterance contradiction leads to policy failure. To address this, we adapt outcome GRPO strictly for decision-making (Section 4.2), complemented by format injection and self-distillation to preserve faithful, player-facing communication.

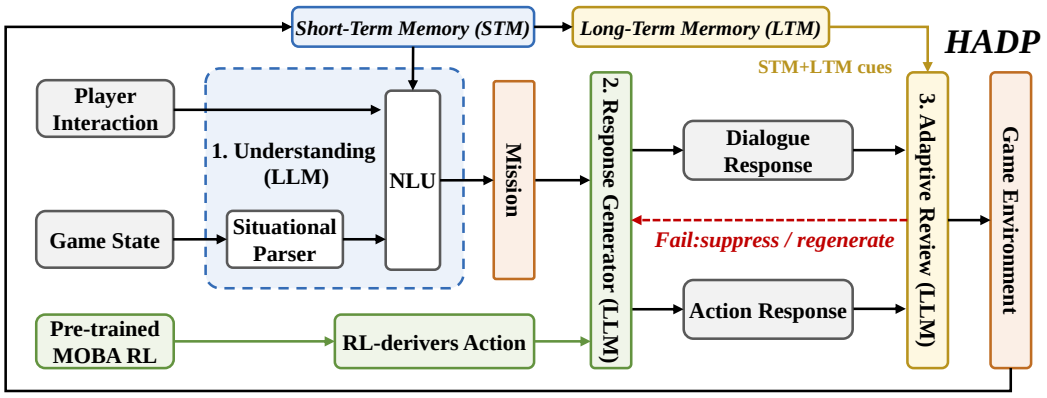


Figure 2: The hierarchical architecture of the HADP framework.

Human-Agent Collaboration. Memory and hierarchical control provide relevant mechanisms for human-agent collaboration: Park et al. (2023) use observation records and reflection to condition later behavior, while Liu et al. (2024) pair a reasoning "Slow Mind" with a reactive "Fast Mind" in Overcooked, combining hierarchical LLM reasoning with low-level policies in a real-time game. Building on these approaches, HADP focuses on an explicit shared mission and pre-execution review for competitive team coordination. Our evaluation distinguishes language quality from execution and multi-turn commitment consistency.

3 The HADP Framework

To overcome the limitations of flat DPL, the HADP framework introduces a hierarchical architecture (Figure 2) that structures the workflow into three layers—**Understanding**, **Response Generator**, and **Adaptive Review**—each instantiated by a fine-tuned compact LLM. All modules operate in the Commander Mode of Honor of Kings; Figure 3 (Appendix B) shows a representative in-game scene. Appendix H distinguishes the reusable workflow and semantic interfaces from environment-specific components and outlines adaptation guidelines; cross-environment transfer remains untested.

Understanding The Understanding module transforms multimodal inputs (player utterances, signals, game states, and STM history) into structured Missions to guide downstream policy execution. This explicit Mission-based representation serves as an essential and rational bridge, decoupling high-level semantic reasoning from fine-grained policy control, which drastically reduces the search space for execution.

Specifically, a fine-tuned LLM acts as a Situational Parser, distilling raw, timestamped map statistics into a concise Situational Description (Figure 4). Conditioned on this description, STM context, and predefined schemas (Figure 5), a second LLM parses player utterances into a 5-element Mission: **Is Command**, **Topic**, **Executor**, **Target**, and **Temporal Cue**.

Constructing such a structured Mission tuple is crucial and necessary for several reasons:

- **Unambiguous Grounding:** It strictly grounds ambiguous references (e.g., resolving implicit pronouns like "you" or minimap spatial tokens like "here" into concrete game entities) while filtering out irrelevant non-command chatter.
- **Contextual Continuity:** By pairing chronological Utterance Logs with State Snapshots in STM, the agent reliably resolves complex multi-turn dependencies (e.g., binding "supp" with "come with me").

Ultimately, formulating unstructured human intent into this standardized 5-element schema guarantees deterministic goal specification, ensuring robust, verifiable, and precise downstream execution.

Response Generator The Response Generator module produces dual-channel outputs conditioned on two sources: (i) the *Mission*, encoding the semantic interpretation of the player's utterance, and (ii) the *RL-derived Action* from a pre-trained MOBA reinforcement learning model (Ye et al., 2020), reflecting purely gameplay-driven planning without semantic conditioning. A fine-tuned

LLM generates an *Action Response*—an executable command bridging high-level intent with low-level movement—and a *Dialogue Response*—a player-facing utterance conditioned on the same inputs. Depending on the interaction context, either or both channels are activated. Before responding, the model selects among three collaborative stances—*immediate cooperation*, *refusal*, and *delayed cooperation*—to represent acceptance, rejection, or conditional acceptance of a request explicitly.

Adaptive Review The Adaptive Review module checks each candidate utterance or action before final execution (Madaan et al., 2023; Wang et al., 2024; Bo et al., 2025), using a fine-tuned LLM conditioned on the most recent STM record, the LTM, and the candidate Action and Dialogue Responses. The latest STM provides context for identifying outdated instructions—e.g., an attack command the player has just retracted; the LTM stores player preferences (communication style, tone), historical interactions, and team-dynamic tendencies as additional review context (Li et al., 2026). Candidates are judged on three criteria: **Contextual Relevance** (consistency with the latest STM), **Tactical Consistency** (alignment with the player's strategic objectives), and **Social Coherence** (conformity to the player's communication style). The last criterion concerns communication style, not a measure of social intelligence or commitment fulfillment. Candidates flagged as failing are suppressed or trigger regeneration; approved outputs are executed and logged back into the STM for subsequent review. This procedure is intended to reduce inconsistencies rather than guarantee their absence. Appendix G provides a conditional theoretical analysis of shared-commitment consistency, the net benefit of review, and memory-based resolution of commitment ambiguity; it does not assume that hierarchy is inherently superior to a flat policy.

4 Experiments and Analysis

We organize the evaluation around three questions: **(Q1)** Does HADP reduce the say–do gap relative to flat policies? **(Q2)** Which architectural modules and training stages contribute to the observed gains? **(Q3)** Do the specialized models retain useful decision and language capabilities, and do production observations support their practical utility? Closed-loop trajectories provide the primary evidence for Q1 and the architectural part of Q2 (Section 4.1). Offline replay isolates supporting capabilities and command-channel training effects; it does not measure executed commitments. Production cohorts provide a separate view of product-level outcomes (Section 4.6).

The reported closed-loop evaluation comprises 400 independently executed task trials for each of five policies or variants ($N_v = 400$), totaling 2,000 closed-loop trajectory evaluations; offline evaluation uses 512 held-out scenarios. Dataset units, subset sizes, and rating protocols are summarized in Table 5 in Appendix A; metric definitions and evidence boundaries are detailed below.

4.1 Main Results: Closed-Loop Say–Do Consistency

Role-based trial design. For each policy, the 400 closed-loop trials are organized by five hero positions (roles), with 80 trials per position. For each position, we test eight of its most commonly used commands, with ten trials per command, yielding $5 \times 8 \times 10 = 400$ trials per policy and $5 \times 400 = 2{,}000$ trajectory evaluations across the five policies. The command sets are position-specific: for example, jungle commands include "Start attacking the dragon," whereas marksman commands include "Check the brush" and "Recover health." Each trial tests one command for the corresponding position, and its utterances, plans, and subsequent agent behavior are assessed using the consistency criteria below. The five positions are sampling strata, distinct from the five policy variants compared in Table 1; the 80-trial allocation applies to each position within each policy. The position-specific command sets need not be disjoint, so this design comprises 40 position–command combinations rather than necessarily 40 distinct commands.

Evaluation criteria. Using timestamped utterances, versioned plans, and agent-generated closed-loop trajectories, we evaluate three complementary criteria. Each concerns consistency rather than tactical success: fulfilling a poor plan can be consistent without being strategically useful.

- **Semantic consistency (SC):** Does the utterance agree with the issued plan? A pass requires agreement on the collaborative stance and material conditions; non-actionable utterances are excluded.

- **Execution consistency (EC):** Does the agent carry out an activated plan? A pass requires satisfying the specified behavioral predicates and deadlines without conflicting actions. Dispatch alone is insufficient, and interrupted but unfulfilled plans fail.
- **Interaction consistency (IC):** Does the agent honor its commitment across turns? A pass requires timely cooperation, non-execution after refusal, or action when a delayed-cooperation condition is triggered. Retractions or state changes require timely, explicit cancellation or justified revision before conflicting behavior; silent abandonment fails.

An untriggered conditional commitment may pass IC if respected throughout observation, but is excluded from EC. Revising a plan does not retroactively repair its EC score, and human actions cannot substitute for agent execution.

Aggregation and controls. Let $N_v = 400$ be the number of independently executed trials for each policy v, with $N_{\text{total}} = \sum_{v=1}^{5} N_v = 2{,}000$. For each policy v and criterion $L \in \{\text{SC}, \text{EC}, \text{IC}\}$, let $\mathcal{E}_{v,L}$ contain applicable, fully observed episodes, let $N_{\text{eligible},v,L} = |\mathcal{E}_{v,L}| \leq N_v$, and let $z_i^{v,L}$ denote a binary pass:

$$C_{v,L} = \frac{100}{N_{\text{eligible},v,L}} \sum_{i \in \mathcal{E}_{v,L}} z_i^{v,L}. \qquad (1)$$

Episodes with missing follow-up are censored; an empty denominator yields N/A. In the reported evaluation, all 400 trials for each policy are eligible for each criterion ($N_{\text{eligible},v,L} = 400$), with no inapplicable or censored trials in the supplied counts (Table 7). SC, EC, and IC assess distinct properties, not successive stages of a success funnel, even though their denominators coincide here. Comparisons match initial scenarios, backbone, training data, and inference budget, with separate rollouts for each policy. Two model-blind annotators resolve ambiguous cases. Flat DPL directly maps context to action and dialogue; budget-matched self-refinement tests whether additional unstructured revision explains HADP's gains. The two HADP ablations remove Understanding or Adaptive Review to assess module-level contributions. Table 6 in Appendix A summarizes the component configurations and clarifies the scope of the reported budget matching and ablation wiring.

Table 1: Closed-loop SC, EC, and IC rates (%) with 95% confidence intervals; 400 trials per policy, 2,000 in total.

Variant	SC	EC	IC
Flat DPL	78.5 [74.1, 82.4]	62.0 [57.0, 66.7]	45.3 [40.3, 50.3]
Flat DPL + bmf	81.3 [77.0, 84.9]	67.5 [62.6, 72.0]	51.5 [46.5, 56.5]
HADP w/o U	73.5 [68.8, 77.7]	74.3 [69.6, 78.4]	63.8 [58.8, 68.4]
HADP w/o AR	89.0 [85.4, 91.8]	81.0 [76.7, 84.7]	61.3 [56.3, 66.0]
Full HADP	**91.5** [88.2, 94.0]	**86.3** [82.4, 89.4]	**76.5** [72.0, 80.5]

Notes. bmf: budget-matched self-refinement; U: Understanding; AR: Adaptive Review. CIs use continuity-corrected Wilson scores with $N_{\text{eligible}} = 400$ for every entry; pass counts and computation details appear in Table 7. Rates, interval endpoints, and percentage-point differences are calculated from unrounded counts and rounded half up to one decimal place.

Comparison with flat policies (Q1). Table 1 reports the primary alignment results. Full HADP attains 91.5% SC, 86.3% EC, and 76.5% IC, exceeding budget-matched flat self-refinement by 10.3, 18.8, and 25.0 percentage points, respectively. The larger numerical gains on EC and IC suggest that the benefit is not confined to matching an utterance to an issued plan. This comparison supports the hierarchical design under the reported controls, but does not isolate Mission grounding from all other architectural differences. The remaining interaction-level failures also show that high semantic agreement does not guarantee reliable commitment handling.

Mechanism ablations: grounding versus review. Removing Understanding reduces SC from 91.5% to 73.5% (18.0 points), EC from 86.3% to 74.3% (12.0 points), and IC from 76.5% to 63.8% (12.8 points). This is consistent with grounded mission construction supporting both language–plan agreement and downstream commitment handling. Removing Adaptive Review reduces SC by 2.5 points and EC by 5.3 points, but IC by 15.3 points, supporting a complementary role for contextual review in maintaining commitments across turns. These module-level contrasts do not isolate individual Mission fields or STM/LTM contributions. All reported policy–criterion denominators equal 400; nevertheless, the aggregate rates should be interpreted alongside refusal/activation statistics rather than as proof that avoidance is absent.

4.2 Offline Evaluation: Tasks, Data, and Training

Task formulation A match is represented as a sequence of game states $\{s_t\}$, each serialized as a structured JSON object capturing all information visible from the primary player's perspective (teammate attributes, visible turrets, and map vision data; hidden enemy information is excluded for

realism). Given s_t, the model predicts the next macro-level action a_t from $K = 40$ predefined team objectives(Table 8), produces a reasoning chain c_t justifying its choice, and distills it into a concise command u_t directed at the player (e.g., "Jointly push down the enemy mid-lane tier-one tower; be mindful of a potential enemy ambush"). This auxiliary task probes state-grounded decision and command generation. It does not directly test interpretation of a live player's request, execution by the agent, or commitment handling across turns.

Datasets and metrics. The held-out scenarios summarized in Table 5 are reconstructed from ranked matches within a single recent patch window, stratified by game phase and team advantage, with at most one scenario per match. Evaluation matches are temporally later than every training trace. The evaluation output format is slightly modified; this is a format-robustness check rather than a guarantee against leakage. The probes have complementary interpretations:

- **Macro-action prediction:** Top-k accuracy ($k \in \{1, 3, 5\}$) measures whether the reference label occurs among the first k predictions. Multiple objectives may be tactically valid, so this metric measures reference agreement, not an exhaustive assessment of decision quality.
- **Command quality:** Functional ratings assess whether the command agrees with the selected action, is actionable, and uses concise, appropriate wording; the LLM cross-check uses the rubric prompt in Figure 6. Perceptual ratings assess the player-facing wording. Human-likeness refers only to the perceived wording, not to agent behavior.
- **Open-ended game QA:** Questions probe state grounding, decision reasoning, game knowledge, and actionability. Per-dimension accuracy is the percentage of answers rated fully correct; hallucination rate measures fabricated game-state claims, and the overall score averages the four rubric dimensions (Table 10).
- **General-capability retention:** Ape210K, MMLU, CEval, School-Chinese, BBH, IfEval, and CharacterEval Tu et al. (2024) assess changes in non-game capabilities (Table 9).

Baselines and compute. We compare Qwen3-8B/14B/32B-Instruct, GPT-4o, and DeepSeek-R1 (DeepSeek-AI, 2025), and evaluate training variants of the 8B and 14B backbones. Training uses four servers with eight NVIDIA H200 GPUs each, Megatron-LM for SFT, and OpenRLHF for GRPO, with prompt and generation length limits of 8,360 and 2,048 tokens, respectively.

Scope of replay-based evidence. Replay evaluation exposes all models to the same observed state history, enabling controlled comparisons of their predictions and command choices. However, the subsequent human actions in a historical trace are not counterfactual rollouts of the evaluated policy: the trace does not branch according to the model's own commands. Therefore, metrics such as reference-action accuracy and command–action agreement should be interpreted as offline proxy evidence, not as substitutes for the agent-generated execution and interaction measurements reported in Section 4.1.

Training recipe. Training combines SFT with GRPO (Shao et al., 2024; DeepSeek-AI, 2025) on real gameplay traces. Here, *offline* describes the source states and evaluation setting; GRPO uses on-policy text rollouts from the current model, not closed-loop gameplay. Its rule-based binary reward scores macro-action correctness, with neither a neural reward model nor a format reward. To train the command channel, (i) *format injection* introduces the output contract $y = [c;\ a;\ u]$ during SFT, and (ii) *self-distillation* tunes command generation on the model's own correct rollouts. Both Qwen3 backbones use the same GRPO step budgets: 500 steps for GRPO alone and 500 or 3,000 steps after SFT. Equal step counts do not imply equal compute across model sizes. Table 4(a) isolates the command-channel stages; objectives and training details appear in Appendix C.

4.3 Offline Decision and Language Capabilities

Macro-action prediction (Q3). Table 2 shows that task-specific training improves reference-action prediction at both scales. With SFT + GRPO (3,000 steps), Qwen3-8B improves from 61.13% to 87.70% Top-1 accuracy (+26.57 percentage points), and Qwen3-14B from 79.69% to 91.99% (+12.30 points). The latter has the highest reported accuracy at all three cutoffs (91.99%/96.88%/99.02%), with Top-1 gains of 4.88 and 5.27 points over GPT-4o and DeepSeek-R1. The trained 8B model is also numerically higher at Top-1, although its Top-3 and Top-5 scores re-

main below those baselines. These are descriptive comparisons; paired uncertainty estimates require per-scenario predictions.

SFT and GRPO contribute differently. GRPO alone remains below SFT alone on both backbones (74.80% vs. 75.20% for 8B; 84.18% vs. 85.16% for 14B). Adding 500 GRPO steps after SFT improves Top-1 by 8.20 and 3.51 points, respectively; extending to 3,000 steps adds 4.30 and 3.32 points. Thus, correctness feedback complements imitation in this offline task, although the longer runs also use more training compute. Higher Top-3/5 scores indicate that the reference action is often retained below the first rank, not that all ranked alternatives are tactically valid.

General-capability retention. Table 9 reports relatively small changes after specialization: accuracy-scale benchmarks change by at most 1.1 points, while BBH changes by at most 0.01 on its separate raw-score scale. This provides no indication of a broad collapse on the measured benchmarks, but does not establish equivalence or guarantee retention in every HADP module.

Table 2: Macro-action prediction Top-k accuracy (%) on the 512 held-out evaluation scenarios reconstructed from real match logs. The full Qwen3-14B configuration correctly predicts 471 of 512 scenarios (91.99% Top-1).

Model	Training	Top-1	Top-3	Top-5
Qwen3-32B	zero-shot	84.18	93.36	96.88
DeepSeek-R1	zero-shot	86.72	95.31	98.44
GPT-4o	zero-shot	87.11	95.31	98.63
Qwen3-8B	zero-shot	61.13	71.29	84.77
Qwen3-8B	SFT	75.20	84.38	91.41
Qwen3-8B	GRPO (500 steps)	74.80	84.18	90.43
Qwen3-8B	SFT + GRPO (500 steps)	83.40	90.62	95.51
Qwen3-8B	SFT + GRPO (3000 steps)	**87.70**	**92.97**	**96.48**
Qwen3-14B	zero-shot	79.69	88.48	93.36
Qwen3-14B	SFT	85.16	90.82	95.12
Qwen3-14B	GRPO (500 steps)	84.18	90.04	94.73
Qwen3-14B	SFT + GRPO (500 steps)	88.67	95.90	97.46
Qwen3-14B	SFT + GRPO (3000 steps)	**91.99**	**96.88**	**99.02**

Open-ended game QA. Table 10 evaluates whether task-specific gains extend beyond selecting an action label. The full recipe improves state grounding, decision reasoning, and actionability, while hallucination falls from 21.5% to 10.1% for Qwen3-8B and from 16.2% to 7.7% for Qwen3-14B. The trained 14B model has a mean score of 1.56, numerically above DeepSeek-R1 (1.52) and GPT-4o (1.51). Game-knowledge scores, however, do not improve (42.1% to 41.2% for 8B; 56.2% to 55.0% for 14B). The gains therefore concern using the observed state and expressing decisions rather than acquiring broader game knowledge.

Quality of player-facing output Table 3 separates functional command quality from perceived wording quality. The full Qwen3-14B recipe has the highest reported mean in both views (1.73 functional and 4.21 perceptual), compared with 1.67 and 4.06 for DeepSeek-R1. Expert–judge agreement is Spearman's $\rho = 0.81$ and Cohen's $\kappa = 0.72$ (binarized at 1.5); agreement among blind expert raters is Kendall's $W = 0.68$. These statistics characterize rating agreement, not the uncertainty of between-model differences. Human-likeness is the lowest-rated perceptual dimension for every model, suggesting that useful commands need not receive equally high ratings for human-like phrasing. Functional action consistency here is a 0–2 text rubric, distinct from binary closed-loop SC. Neither view establishes execution or isolates the effect of Adaptive Review on communication style.

4.4 Ablations: Command Training and Hierarchical Modules

Command-channel training (Q2). With Qwen3-14B and the 3,000-step SFT + GRPO configuration fixed, Table 4(a) compares three cumulative variants: a post-hoc command baseline without either command-channel stage, SFT format injection, and format injection followed by self-distillation. Format injection raises command consistency/actionability from 1.54/1.47 to 1.69/1.64, while Top-1 accuracy decreases from 90.82% to 90.23% (0.59 points). Adding self-distillation raises these quality scores to 1.78/1.72 and accuracy to 91.99%, exceeding the post-hoc baseline by 1.17 points. The full recipe therefore improves command quality without an observed loss in reference-action accuracy. This staged comparison evaluates the training recipe as a whole; without an equal-data, equal-compute control, it does not isolate reasoning-token masking from the effects of additional tuning.

Hierarchical modules (Q2). With the backbone and training recipe fixed, Table 4(b) compares Flat DPL, the two module ablations, and Full HADP on offline action and command quality. Full HADP exceeds Flat DPL by 3.71 Top-1 points (91.99% vs. 88.28%) and by 0.25 in functional quality

Table 3: Quality of player-facing output on stratified subsets of the 512 held-out scenarios of Table 2: 150 scenarios for functional evaluation and 120 for perceptual evaluation. *Functional* columns (0–2 expert rubric) score the distilled command u_t on action consistency (Cons.), actionability (Act.), and brevity/tone (Brev.); F-Ov. is their mean and Judge-Ov. the rubric-prompted LLM-judge overall. *Perceptual* columns are blind expert 1–5 Likert ratings of generated utterances on naturalness (Nat.), human-likeness (Hum.), and command experience (Cmd.); P-Ov. is their mean. All ratings are averaged over three expert raters. Bold denotes the best score in each quality column.

		Functional (0–2)					Perceptual (1–5)			
Model	**Training**	Cons.	Act.	Brev.	F-Ov.	Judge-Ov.	Nat.	Hum.	Cmd.	P-Ov.
Qwen3-32B	zero-shot	1.55	1.48	1.52	1.52	1.56	3.86	3.55	3.78	3.73
DeepSeek-R1	zero-shot	1.74	1.67	1.61	1.67	1.71	4.15	3.92	4.10	4.06
GPT-4o	zero-shot	1.70	1.62	1.66	1.66	1.70	4.22	3.95	4.15	4.11
Qwen3-8B	zero-shot	0.92	0.85	1.05	0.94	0.98	2.85	2.41	2.62	2.63
Qwen3-8B	SFT	1.30	1.22	1.36	1.29	1.33	3.42	3.02	3.31	3.25
Qwen3-8B	GRPO (500 steps)	1.34	1.28	1.30	1.31	1.35	3.38	3.05	3.36	3.26
Qwen3-8B	SFT + GRPO (500 steps)	1.52	1.46	1.50	1.49	1.53	3.72	3.38	3.66	3.59
Qwen3-8B	SFT + GRPO (3000 steps)	1.64	1.58	1.60	1.61	1.65	3.95	3.66	3.90	3.84
Qwen3-14B	zero-shot	1.18	1.10	1.24	1.17	1.22	3.25	2.88	3.12	3.08
Qwen3-14B	SFT	1.52	1.45	1.50	1.49	1.53	3.78	3.44	3.70	3.64
Qwen3-14B	GRPO (500 steps)	1.55	1.50	1.48	1.51	1.55	3.80	3.50	3.78	3.69
Qwen3-14B	SFT + GRPO (500 steps)	1.66	1.60	1.58	1.61	1.65	4.02	3.78	3.98	3.93
Qwen3-14B	SFT + GRPO (3000 steps)	**1.78**	**1.72**	**1.68**	**1.73**	**1.76**	**4.31**	**4.08**	**4.24**	**4.21**

Table 4: Ablation studies: macro-action accuracy is evaluated on all 512 held-out scenarios; functional and perceptual quality use stratified subsets of 150 and 120 scenarios, respectively, following the protocol and column layout of Table 3, with the backbone and recipe fixed to Qwen3-14B + SFT + GRPO (3,000 steps). (a) Command-channel training stages (Section 4.2); (b) HADP's hierarchical modules. Top-1 Acc. = macro-action accuracy (%); QA-Ov. = open-ended-QA overall score (protocol of Table 10, Appendix B). Panel (b) uses a shared QA result, shown only in the Full HADP row; it is not an architecture-specific comparison.

			Functional (0–2)					Perceptual (1–5)			
Variant	**Top-1 Acc.**	**QA-Ov.**	Cons.	Act.	Brev.	F-Ov.	Judge-Ov.	Nat.	Hum.	Cmd.	P-Ov.
(a) Command-channel training stages											
SFT + GRPO (w/o stages (i), (ii))	90.82	1.48	1.54	1.47	1.50	1.50	1.55	3.85	3.60	3.80	3.75
+ stage (i) format injection	90.23	1.52	1.69	1.64	1.62	1.65	1.69	4.12	3.86	4.05	4.01
+ stage (ii) self-distillation	**91.99**	**1.56**	**1.78**	**1.72**	**1.68**	**1.73**	**1.76**	**4.31**	**4.08**	**4.24**	**4.21**
(b) Hierarchical modules											
Flat DPL (no Mission, no Review)	88.28	–	1.52	1.44	1.49	1.48	1.52	3.80	3.55	3.71	3.69
HADP w/o Understanding	90.43	–	1.63	1.57	1.60	1.60	1.64	4.02	3.78	3.96	3.92
HADP w/o Adaptive Review	91.21	–	1.71	1.65	1.63	1.66	1.70	4.18	3.95	4.10	4.08
Full HADP	**91.99**	**1.56**	**1.78**	**1.72**	**1.68**	**1.73**	**1.76**	**4.31**	**4.08**	**4.24**	**4.21**

(1.73 vs. 1.48). Removing Understanding lowers command consistency/actionability by 0.15/0.15, compared with 0.07/0.07 when removing Adaptive Review. These offline contrasts suggest that grounding particularly benefits immediate command construction. They do not diminish the role of review in longitudinal interaction: removing Adaptive Review lowers closed-loop IC by 15.3 points (Table 1). Training-time command alignment and inference-time contextual review thus address different evaluation targets.

4.5 Qualitative Errors and Case Analysis

Residual failure modes. Manual inspection of open-ended answers identifies five recurring categories: basic game-knowledge errors, game-state misinterpretation, critical-event oversight, situational misjudgment, and spatio-temporal miscoordination. This taxonomy suggests where interventions may help: grounding and event extraction address input interpretation, while generation and review can check tactical choices and timing. It does not uniquely localize errors to a module, since an upstream mistake can propagate and several modules can contribute to the same failure. Without category counts or a closed-loop failure breakdown, these examples should be treated as qualitative diagnostics rather than evidence of a particular error distribution.

Reasoning-to-command example. Appendix D illustrates how state analysis and objective prioritization are compressed into a concise tower-push instruction with an ambush warning. The inter-

mediate reasoning also contains a factual error, showing that an apparently useful final command can coexist with faulty analysis. The example illustrates the output contract, but provides neither an execution trace nor a multi-turn commitment test; it therefore cannot establish EC or IC.

4.6 Production Deployment in Commander Mode

Serving setting. The command-channel model used by HADP's Response Generator was deployed in production Commander Mode through a six-week staged release. Mission-relevant context is assembled in the background alongside player input collection; prefix-cache reuse and batched quantized decoding support a reported end-to-end P99 latency of a few hundred milliseconds at peak load. This approximate figure describes the deployed stack, not a controlled latency comparison among architectures.

Product-level outcomes (Q3). Players were deterministically assigned to cohorts differing in whether the upgraded dialogue policy served their sessions. Table 12 reports relative changes against the matched control across rollout stages: $+6.2\%$–$+8.4\%$ for command adoption, $+2.1\%$–$+3.5\%$ for session satisfaction feedback, and $+0.7\%$–$+1.3\%$ for seven-day retention. These are relative lifts, not percentage-point differences or confidence intervals. Cohort sizes and absolute rates are proprietary, preventing an independent assessment of uncertainty from the disclosed data. The positive trends support practical utility in this deployment, but neither measure SC/EC/IC directly nor attribute gains to individual HADP layers.

Operational failure modes. Deployment exposed noisy or ambiguous utterances, including accidental voice triggers, dialects, and mid-thought revisions; latency variance during team fights, when replies can become stale; and personalization needs only partly captured by static long-term memory. These observations motivate input robustness, timely cancellation, and longitudinal memory evaluation in Section 5. Taken together, the experiments distinguish three kinds of evidence: closed-loop alignment, offline component capability, and product-level utility, each with a different scope of inference.

5 Limitations and Future Directions

We acknowledge four key limitations: (i) **Evaluation Scope:** Scenarios and deployment metrics establish closed-loop consistency but do not isolate layer-wise impacts or prove long-term win-rate durability in unrestricted play. (ii) **Domain Focus:** Experiments are confined to a single MOBA title. (iii) **Data & Rating Bias:** Logged labels reflect source-player styles, and blind studies rely on a small expert pool ($N = 3$, Kendall's $W = 0.68$). (iv) **Agency Claim:** Metrics evaluate consistency and satisfaction, not perceived agency.

Accordingly, future work will focus on: (i) cross-game generalization, (ii) handling noisy live inputs, (iii) robust rewarding for the verbal channel, and (iv) personalized long-term memory at scale.

6 Conclusion

This paper presented HADP, a hierarchical framework for reducing the say–do gap in MOBA teammates through shared *Mission* grounding and Adaptive Review. The central evaluation distinguishes semantic agreement, execution consistency, and interaction consistency rather than treating action prediction as evidence of commitment fulfillment. In the reported closed-loop comparison, HADP attains SC/EC/IC of 91.5%/86.3%/76.5%, compared with 81.3%/67.5%/51.5% for the compute-matched flat self-reflection baseline. Module ablations suggest complementary roles: removing Understanding most strongly reduces semantic consistency, whereas removing Adaptive Review particularly reduces interaction consistency. These comparisons do not separately isolate individual Mission fields or memory components. Supporting offline experiments characterize macro-action prediction and command quality, while production observations provide complementary product-level evidence rather than direct measurements of promise fulfillment. The results motivate explicit grounding and review for action-aligned dialogue within the tested setting; fuller closed-loop reporting, dedicated user studies, and evaluation beyond a single game remain necessary to establish reproducibility and broader applicability.

REFERENCES

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report, 2023. URL https://arxiv.org/abs/2309.16609. 1

Xiaohe Bo, Zeyu Zhang, Quanyu Dai, Xueyang Feng, Lei Wang, Rui Li, Xu Chen, and Ji-Rong Wen. Reflective multi-agent collaboration based on large language models. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2025. Curran Associates Inc. ISBN 9798331314385. 4

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948. 2, 6, 15

DeepSeek-AI, :, Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, Huazuo Gao, Kaige Gao, Wenjun Gao, Ruiqi Ge, Kang Guan, Daya Guo, Jianzhong Guo, Guangbo Hao, Zhewen Hao, Ying He, Wenjie Hu, Panpan Huang, Erhang Li, Guowei Li, Jiashi Li, Yao Li, Y. K. Li, Wenfeng Liang, Fangyun Lin, A. X. Liu, Bo Liu, Wen Liu, Xiaodong Liu, Xin Liu, Yiyuan Liu, Haoyu Lu, Shanghao Lu, Fuli Luo, Shirong Ma, Xiaotao Nie, Tian Pei, Yishi Piao, Junjie Qiu, Hui Qu, Tongzheng Ren, Zehui Ren, Chong Ruan, Zhangli Sha, Zhihong Shao, Junxiao Song, Xuecheng Su, Jingxiang Sun, Yaofeng Sun, Minghui Tang, Bingxuan Wang, Peiyi Wang, Shiyu Wang, Yaohui Wang, Yongji Wang, Tong Wu, Y. Wu, Xin Xie, Zhenda Xie, Ziwei Xie, Yiliang Xiong, Hanwei Xu, R. X. Xu, Yanhong Xu, Dejian Yang, Yuxiang You, Shuiping Yu, Xingkai Yu, B. Zhang, Haowei Zhang, Lecong Zhang, Liyue Zhang, Mingchuan Zhang, Minghua Zhang, Wentao Zhang, Yichao Zhang, Chenggang Zhao, Yao Zhao, Shangyan Zhou, Shunfeng Zhou, Qihao Zhu, and Yuheng Zou. Deepseek llm: Scaling open-source language models with longtermism, 2024. URL https://arxiv.org/abs/2401.02954. 1

Roberto Gallotta, Graham Todd, Marvin Zammit, Sam Earle, Antonios Liapis, Julian Togelius, and Georgios N. Yannakakis. Large language models and games: A survey and roadmap. *IEEE Transactions on Games*, pp. 1–18, 2024. doi: 10.1109/TG.2024.3461510. 2

Sihao Hu, Tiansheng Huang, Gaowen Liu, Ramana Rao Kompella, Fatih Ilhan, Selim Furkan Tekin, Yichang Xu, Zachary Yahn, and Ling Liu. A survey on large language model-based game agents. *arXiv preprint arXiv:2404.02039*, 2024. 2

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. 2

Chao Li, HongSong Tang, XiaoFeng Qi, BoHao Yang, and Dong Liu. High-dimensional programmable personas: A comprehensive framework for synthesizing customisable role-playing agents. *IEEE Access*, pp. 1–1, 2026. doi: 10.1109/ACCESS.2026.3732026. 2, 4

Yangming Li, Kaisheng Yao, Libo Qin, Wanxiang Che, Xiaolong Li, and Ting Liu. Slot-consistent NLG for task-oriented dialogue systems with iterative rectification network. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 97–106, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.10. URL https://aclanthology.org/2020.acl-main.10/. 2

Zhengyang Li, Qijin Ji, Xinghong Ling, and Quan Liu. A comprehensive review of multi-agent reinforcement learning in video games. *IEEE Transactions on Games*, pp. 1–21, 2025. doi: 10.1109/TG.2025.3588809. 2

Yi Liao, Yu Gu, Yuan Sui, Zining Zhu, Yifan Lu, Guohua Tang, Zhongqian Sun, and Wei Yang. Think in games: Learning to reason in games via reinforcement learning with large language models, 2025. URL https://arxiv.org/abs/2508.21365. 2

Jijia Liu, Chao Yu, Jiaxuan Gao, Yuqing Xie, Qingmin Liao, Yi Wu, and Yu Wang. Llm-powered hierarchical language agent for real-time human-ai coordination. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '24, pp. 1219–1228, Richland, SC, 2024. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9798400704864. 3

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: iterative refinement with self-feedback. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc. 4

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, UIST '23, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701320. doi: 10.1145/3586183.3606763. URL https://doi.org/10.1145/3586183.3606763. 2, 3

Yun Qu, Boyuan Wang, Jianzhun Shao, Yuhang Jiang, Chen Chen, Zhenbin Ye, Lin Liu, Junfeng Yang, Lin Lai, Hongyang Qin, Minwen Deng, Juchao Zhuo, Deheng Ye, Qiang Fu, Wei Yang, Guang Yang, Lanxiao Huang, and Xiangyang Ji. Hokoff: real game dataset from honor of kings and its offline reinforcement learning benchmarks. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc. 1

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, 2024. URL https://arxiv.org/abs/2402.03300. 2, 6, 16

Li Song. Llm-driven npcs: Cross-platform dialogue system for games and social platforms, 2025. URL https://arxiv.org/abs/2504.13928. 2

Xingwu Sun, Yanfeng Chen, Yiqing Huang, Ruobing Xie, Jiaqi Zhu, Kai Zhang, Shuaipeng Li, Zhen Yang, Jonny Han, Xiaobo Shu, Jiahao Bu, Zhongzhi Chen, Xuemeng Huang, Fengzong Lian, Saiyong Yang, Jianfeng Yan, Yuyuan Zeng, Xiaoqin Ren, Chao Yu, Lulu Wu, Yue Mao, Jun Xia, Tao Yang, Suncong Zheng, Kan Wu, Dian Jiao, Jinbao Xue, Xipeng Zhang, Decheng Wu, Kai Liu, Dengpeng Wu, Guanghui Xu, Shaohua Chen, Shuang Chen, Xiao Feng, Yigeng Hong, Junqiang Zheng, Chengcheng Xu, Zongwei Li, Xiong Kuang, Jianglu Hu, Yiqi Chen, Yuchi Deng, Guiyang Li, Ao Liu, Chenchen Zhang, Shihui Hu, Zilong Zhao, Zifan Wu, Yao Ding, Weichao Wang, Han Liu, Roberts Wang, Hao Fei, Peijie Yu, Ze Zhao, Xun Cao, Hai Wang, Fusheng Xiang, Mengyuan Huang, Zhiyuan Xiong, Bin Hu, Xuebin Hou, Lei Jiang, Jianqiang Ma, Jiajia Wu, Yaping Deng, Yi Shen, Qian Wang, Weijie Liu, Jie Liu, Meng Chen, Liang Dong, Weiwen Jia, Hu Chen, Feifei Liu, Rui Yuan, Huilin Xu, Zhenxiang Yan, Tengfei Cao, Zhichao Hu, Xinhua Feng, Dong Du, Tinghao Yu, Yangyu Tao, Feng Zhang, Jianchen Zhu, Chengzhong Xu, Xirui Li, Chong Zha, Wen Ouyang, Yinben Xia, Xiang Li, Zekun He, Rongpeng Chen, Jiawei Song, Ruibin Chen, Fan Jiang, Chongqing Zhao, Bo Wang, Hao Gong, Rong Gan, Winston Hu, Zhanhui Kang, Yong Yang, Yuhong Liu, Di Wang, and Jie Jiang. Hunyuan-large: An open-source moe model with 52 billion activated parameters by tencent, 2024. URL https://arxiv.org/abs/2411.02265. 1

Quan Tu, Shilong Fan, Zihang Tian, Tianhao Shen, Shuo Shang, Xin Gao, and Rui Yan. Charactereval: A chinese benchmark for role-playing conversational agent evaluation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11836–11850, 2024. 6

Marilyn Walker, Candace Kamm, and Diane Litman. Towards developing general models of usability with paradise. *Nat. Lang. Eng.*, 6(3–4):363–377, September 2000. ISSN 1351-3249. doi: 10.1017/S1351324900002503. URL https://doi.org/10.1017/S1351324900002503. 2

Haoyu Wang, Tao Li, Zhiwei Deng, Dan Roth, and Yang Li. Devil's advocate: Anticipatory reflection for LLM agents. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 966–978, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.53. URL https://aclanthology.org/2024.findings-emnlp.53/. 4

Hua Wei, Jingxiao Chen, Xiyang Ji, Hongyang Qin, Minwen Deng, Siqin Li, Liang Wang, Weinan Zhang, Yong Yu, Lin Liu, Lanxiao Huang, Deheng Ye, Qiang Fu, and Wei Yang. Honor of kings arena: an environment for generalization in competitive reinforcement learning. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088. 1

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. A network-based end-to-end trainable task-oriented dialogue system. In Mirella Lapata, Phil Blunsom, and Alexander Koller (eds.), *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pp. 438–449, Valencia, Spain, April 2017. Association for Computational Linguistics. URL https://aclanthology.org/E17-1042/. 1

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. Transferable multi-domain state generator for task-oriented dialogue systems. In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 808–819, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1078. URL https://aclanthology.org/P19-1078/. 2

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388. 1

Deheng Ye, Guibin Chen, Wen Zhang, Sheng Chen, Bo Yuan, Bo Liu, Jia Chen, Zhao Liu, Fuhao Qiu, Hongsheng Yu, Yinyuting Yin, Bei Shi, Liang Wang, Tengfei Shi, Qiang Fu, Wei Yang, Lanxiao Huang, and Wei Liu. Towards playing full moba games with deep reinforcement learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546. 1, 3

Deheng Ye, Guibin Chen, Peilin Zhao, Fuhao Qiu, Bo Yuan, Wen Zhang, Sheng Chen, Mingfei Sun, Xiaoqian Li, Siqin Li, Jing Liang, Zhenjie Lian, Bei Shi, Liang Wang, Tengfei Shi, Qiang Fu, Wei Yang, and Lanxiao Huang. Supervised learning achieves human-level performance in moba games: A case study of honor of kings. *IEEE Transactions on Neural Networks and Learning Systems*, 33(3):908–918, 2022. doi: 10.1109/TNNLS.2020.3029475. 1

Zheng Zhang, Ryuichi Takanobu, Qi Zhu, MinLie Huang, and XiaoYan Zhu. Recent advances and challenges in task-oriented dialog systems. *Science China Technological Sciences*, 63(10): 2011–2027, Oct 2020. ISSN 1869-1900. doi: 10.1007/s11431-020-1692-3. URL https://doi.org/10.1007/s11431-020-1692-3. 2

Wei Zhou, Xiangyu Peng, and Mark Riedl. Dialogue shaping: Empowering agents through npc interaction, 2023. URL https://arxiv.org/abs/2307.15833. 2

A Data and Evaluation Protocols

Table 5 summarizes the game-specific datasets, evaluation units, rating protocols, and metrics used in Section 4.

Table 5: Data and evaluation settings for the game-specific experiments. Evaluation units and subset conditions are explicitly aligned with reported results.

Evaluation task	Data source	Size / unit	Evaluation protocol	Metrics
Closed-loop consistency	Agent-generated trajectories, utterances, and versioned plans	400 trials per policy: 5 positions × 8 commands × 10 trials; five policies, 2,000 trajectory evaluations total.	Criterion-specific pass/fail checks; two model-blind annotators adjudicate ambiguous cases.	SC, EC, IC pass rates (%); 95% Wilson CIs.
Macro-action prediction	Held-out states from ranked matches	512 scenarios; 40 action classes	Compare ranked predictions with the reference action label.	Top-1/3/5 accuracy (%).
Functional command quality	Stratified subset of the held-out scenarios	150 scenarios	Three experts; 0–2 rubric; LLM-judge cross-check.	Action consistency, actionability, brevity/tone; mean score.
Perceptual quality	Stratified subset of the held-out scenarios	120 scenarios	Three model-blind experienced players; 1–5 Likert scale.	Naturalness, human-likeness, command experience; mean score.
Open-ended game QA	Questions paired with held-out game states	512 source states; 512 paired QA prompts	Three blind experts; 0–2 rubric across four dimensions.	Fully correct answers (%), hallucination rate (%), mean score.

Notes. The 150-scenario functional and 120-scenario perceptual sets are stratified subsets of the main 512-scenario testbed (their exact overlap is unspecified). Source-state and QA prompt counts are aligned 1:1 ($N = 512$). Within each policy, each of the five hero positions contributes 80 trials, covering eight position-specific common commands with ten trials per command. The 40 position–command combinations do not imply 40 distinct commands and are separate from the 40 action classes in the offline macro-action task. Each of the five policies is independently executed for $N_v = 400$ trials, yielding $N_{\text{total}} = \sum_{v=1}^{5} N_v = 2{,}000$ closed-loop trajectory evaluations. Eligible denominators are computed separately for each policy v and criterion $L \in \{\text{SC}, \text{EC}, \text{IC}\}$, excluding inapplicable or censored episodes when present. In this evaluation, all 15 denominators equal $N_{\text{eligible},v,L} = N_v = 400$, with no exclusions or censoring in the reported counts. Table 7 provides the pass counts; the pooled total of 2,000 is not used as each policy's denominator. The reported 95% Wilson intervals use continuity correction.

Policy variants and ablation configurations. Table 6 summarizes the five policies used in the closed-loop comparison. It separates explicit Mission grounding, the action–dialogue generation architecture, memory components, and the dedicated Adaptive Review module. These are module-level contrasts, not isolated tests of individual Mission fields or memory mechanisms.

Table 7: Closed-loop pass counts. Each cell reports k/N_{eligible}; each policy executes 400 trials, and the same trials are scored on three criteria rather than counted as additional trajectories.

Variant	SC	EC	IC
Flat DPL	314/400	248/400	181/400
Flat DPL + bmf	325/400	270/400	206/400
HADP w/o U	294/400	297/400	255/400
HADP w/o AR	356/400	324/400	245/400
Full HADP	366/400	345/400	306/400

Confidence-interval computation. For a pass count k and eligible denominator $n = 400$, the rate is $100k/n$. We compute two-sided 95% continuity-corrected Wilson score intervals by inverting the binomial score test with a half-count correction. With $z = \Phi^{-1}(0.975)$, $q_- = (k - 0.5)/n$, and

Table 6: Policy variants and ablation configurations for the closed-loop evaluation. All variants produce action and dialogue outputs; generation architecture, rather than the presence of two output channels, distinguishes the flat and hierarchical policies.

Variant	Mission grounding	Generation architecture	Explicit STM/LTM	Adaptive Review	Comparison purpose
Flat DPL	No	Flat dual-channel	No	No	Basic flat-policy baseline
Flat DPL + bmf	No	Flat dual-channel with self-refinement	No	No	Control for additional unstructured revision
HADP w/o U	No	Hierarchical dual-channel; U removed	Retained	Yes	Assess the contribution of Understanding
HADP w/o AR	Yes	Hierarchical dual-channel; AR removed	Retained*	No	Assess the contribution of contextual review
Full HADP	Yes	Hierarchical dual-channel	Yes	Yes	Complete framework

Notes. U: Understanding; AR: Adaptive Review; bmf: budget-matched self-refinement. "No" under explicit STM/LTM means no dedicated HADP memory components, not necessarily an absence of dialogue history in the policy input. Ordinary self-refinement is not the dedicated Adaptive Review module. "Retained" denotes component retention, not identical memory access across variants. *Without AR, STM remains available to Understanding; the full architecture's AR-mediated LTM access is removed, and any remaining LTM reader is unspecified.

Budget and wiring scope. The closed-loop protocol reports matching initial scenarios, backbone, training data, and inference budget across policies, with separate rollouts. Exact per-variant token and model-call allocations, and whether matching refers to budget ceilings or realized consumption, are not specified; this table does not establish equal realized compute or latency. For the variant without U, the replacement inputs to the Response Generator and any changes to the STM update path remain unspecified. Component retention alone does not establish that downstream inputs are otherwise identical. These configuration details are needed for full reproduction; the table does not assume identical wiring for the offline ablations.

$q_+ = (k + 0.5)/n$, the bounds for the reported counts ($0 < k < n$) are

$$\ell = \frac{q_- + z^2/(2n) - z\sqrt{q_-(1-q_-)/n + z^2/(4n^2)}}{1 + z^2/n},$$
$$u = \frac{q_+ + z^2/(2n) + z\sqrt{q_+(1-q_+)/n + z^2/(4n^2)}}{1 + z^2/n}. \tag{2}$$

Table 1 displays $[100\ell, 100u]$. Rates, endpoints, and between-policy differences are computed before rounding and then rounded half up to one decimal place. For example, the IC decrease from Full HADP to HADP w/o AR is $100(306 - 245)/400 = 15.25$ percentage points, reported as 15.3; subtracting already-rounded table entries can therefore differ by 0.1 point. These are marginal binomial intervals for each policy–criterion pair, not confidence intervals for paired between-policy differences.

Macro-action label space. The macro-action label space in Table 8 was constructed through an expert-driven summarization process. Specifically, experienced game designers reviewed common cooperative behaviors and strategic intentions observed in real matches, and grouped semantically similar actions according to practical gameplay requirements. As a result, the original fine-grained action labels were merged into 40 macro-action categories. This label space covers the major cooperation-oriented action types required in actual gameplay, including objective contesting, tower attacking and defending, hero combat, lane clearing, jungle resource acquisition, team grouping, and recall-related states.

Table 8: Final 40-label macro-action space.

ID	Category	Macro-action Label	ID	Category	Macro-action Label
0	Other	All Other Cases	20	Lane Minions	Mid-lane Minions
1	Dragon Objective	Dragon Objective	21	Lane Minions	Bottom-lane Minions
2	Tower / Crystal	Enemy Crystal	22	Lane Minions	Allied High-ground Minions
3	Tower / Crystal	Top Tower	23	Lane Minions	Enemy High-ground Minions
4	Tower / Crystal	Mid Tower	24	Buff	Allied Buff
5	Tower / Crystal	Bottom Tower	25	Buff	Enemy Buff
6	Defense	Defend Allied Crystal	26	Jungle	Allied Jungle Camps
7	Defense	Defend Top Tower	27	Jungle	Enemy Jungle Camps
8	Defense	Defend Mid Tower	28	River Objective	River Neutral Objective
9	Defense	Defend Bottom Tower	29	Vision / Safety	Bush Checking
10	Hero Combat	Top-lane Hero Combat	30	Grouping	Top-lane Grouping
11	Hero Combat	Mid-lane Hero Combat	31	Grouping	Mid-lane Grouping
12	Hero Combat	Bottom-lane Hero Combat	32	Grouping	Bottom-lane Grouping
13	Hero Combat	Upper-river Hero Combat	33	Grouping	Upper-river Grouping
14	Hero Combat	Lower-river Hero Combat	34	Grouping	Lower-river Grouping
15	Hero Combat	Allied-jungle Hero Combat	35	Grouping	Allied-jungle Grouping
16	Hero Combat	Enemy-jungle Hero Combat	36	Grouping	Enemy-jungle Grouping
17	Hero Combat	Allied High-ground Hero Combat	37	Grouping	Allied High-ground Grouping
18	Hero Combat	Enemy High-ground Hero Combat	38	Grouping	Enemy High-ground Grouping
19	Lane Minions	Top-lane Minions	39	Recovery	Recovery / Recall State

Macro-action label space. As shown in Table 8, the final macro-action label space was defined based on an expert-driven analysis of real gameplay. Experienced game designers summarized the cooperative behaviors and strategic intentions that frequently occur in actual matches, and merged actions with similar tactical meanings into unified macro-action categories. The resulting 40-label taxonomy is designed to cover the practical label requirements of macro-action prediction, including neutral-objective contesting, tower-related actions, defensive behaviors, hero combat, minion clearing, jungle resource control, river-objective contesting, bush checking and vision-safety behaviors, team grouping, and recovery or recall states.

B Prompt Templates and Supplementary Material

This appendix collects the full prompt templates of the LLM modules referenced in the main text, the rubric prompt of the LLM judge, an in-game screenshot of the production Commander Mode, and the general benchmark retention results.



Figure 3: An in-game screenshot from Commander Mode.

C Training Details

Table 11 summarizes the reported training configuration; objectives and command-channel supervision are detailed below.

As illustrated in Figure 7, following the multi-stage recipe popularized by DeepSeek-R1 (DeepSeek-AI, 2025), training combines supervised fine-tuning and online reinforcement learning. The SFT

[System] You are a tactical analyst for a 5v5 MOBA match (Honor of Kings). You read timestamped game statistics and report the current situation accurately and concisely.

[User]
– *Map & game structure*: `<arena layout, lane configuration, key objectives>`
– *State statistics at time t*:
◇ Heroes: `<kills/deaths/assists, health ratio, position, proximity to allies/enemies>`
◇ Minions, monsters, towers: `<health ratio, position>`
Task: (1) Summarize the current match situation concisely. (2) Provide tactical suggestions for the primary player's team.

Figure 4: Prompt template of the **Situational Parser**. Placeholders in `<...>` are filled with serialized game data.

[System] You are the command interpreter of an AI teammate in a MOBA match. Interpret the player's latest interaction and output a structured mission in JSON.

[User]
– *Map & game structure*: `<arena layout>`
– *Player identity*: `<hero, team affiliation>`
– *Current interaction*: `<utterance or signal>`
– *Dialog history (STM)*: `<recent interactions>`
– *Situational description*: `<output of the Situational Parser>`
– *Candidate command themes*: `<list with explanations>`
– *Candidate executors*: `<list with explanations>`
– *Candidate targets*: `<list with explanations>`
Task: Output `{"is_command": <bool>, "topic": <theme>, "executor": <who>, "target": <what>, "temporal_cue": <when>}`. If the interaction is not a command, set `"is_command"` to `false` and leave the remaining fields empty. Ground pronouns (e.g., "you", "here") in the game state and STM.

Figure 5: Prompt template of the **Natural Language Understanding** module, which produces a structured *Mission*.

stage distills reasoning data from DeepSeek-R1, whose strong game-state analysis provides high-quality supervision for smaller models; the RL stage applies GRPO (Shao et al., 2024) on real gameplay traces with a rule-based binary reward on macro-action correctness—neither a neural reward model nor format-based rewards are used, keeping the training signal reliable and directly aligned with strategic correctness. We train the two open-source backbones Qwen3-8B and Qwen3-14B with this recipe, keeping training budgets consistent across scales so that the results in Table 2 are directly comparable: for both model sizes, GRPO alone is run for 500 steps and the SFT+GRPO pipeline is trained for 500 steps; to observe long-run training dynamics, SFT+GRPO is further extended to 3,000 steps for both Qwen3-8B and Qwen3-14B.

Teaching the command channel. We extend the multi-stage recipe so that the distilled command u_t becomes part of the training target rather than a post-hoc decoding step. (i) *Format injection in SFT*: each teacher-distilled sample is augmented with a final instruction line produced by the teacher, so every training example follows a three-part output contract—situational analysis, macro action, and a concise player-facing command—and the stylistic properties of the command channel (brevity, directness, actionability) are learned by imitation. (ii) *Self-distillation of the command channel*: after SFT+GRPO training, we re-sample the model's own *correct* rollouts and have the model compress its full reasoning into a (reasoning, command) pair, forming a small auxiliary corpus of 61,983 examples per backbone for a two-epoch instruction-tuning round that specifically reinforces reasoning-to-command compression; because the samples are drawn from the model's own distribution, this stage also mitigates exposure bias. The RL reward is left unchanged—binary macro-action correctness—keeping the training signal free of verbal-channel noise.

Both stages use standard response-only token-level cross-entropy, differing in which tokens the loss covers. In stage (i), the objective spans the entire three-part output,

$$\mathcal{L}_{\mathrm{SFT}} = -\sum_{t} \log \pi_\theta(y_t \mid y_{<t}, x), \qquad y = [c;\ a;\ u], \tag{3}$$

[System] You are an expert evaluator of commander instructions in MOBA matches. Score the given player-facing command with an explicit rubric; be strict and do not reward verbosity.

[User]
– *Game state*: <serialized state s_t>
– *Chosen action*: <a_t>
– *Model reasoning*: <c_t>
– *Command to evaluate*: <u_t>
Rubric (score each dimension on 0–2):
◇ *Action consistency* — 2: the command fully follows from a_t and c_t; 1: partially consistent or omits key conditions; 0: contradicts the action or reasoning.
◇ *Actionability* — 2: directly executable without ambiguity; 1: executable but requires minor interpretation; 0: vague or not executable.
◇ *Brevity & tone* — 2: concise and appropriate for real-time in-game communication; 1: slightly verbose or stylistically off; 0: verbose, confusing, or inappropriate tone.
Task: Output {"consistency": <0--2>, "actionability": <0--2>, "brevity": <0--2>}.

Figure 6: Rubric prompt of the **LLM judge** used to score distilled command expressions (Table 3). Expert players followed the same rubric.

Table 9: General benchmark retention after game-specific training. All columns except BBH are 0–100 accuracies; BBH uses the native raw-score scale of the evaluation harness and is comparable only within each model pair.

Model	Ape210K	MMLU	CEval	School-Chinese	BBH	IfEval	CharacterEval
Qwen3-8B	79.0	95.7	76.2	76.7	3.05	60.7	64.8
Qwen3-8B + SFT + GRPO	78.8	95.8	76.3	76.2	3.04	61.3	64.9
Qwen3-14B	93.5	91.6	80.3	83.1	3.01	65.8	65.8
Qwen3-14B + SFT + GRPO	94.1	91.3	80.4	82.8	3.01	66.9	65.3

where x is the prompt (game state, history, and task instruction) and prompt tokens are masked. Because the command line shares this objective, both its style and the reasoning-to-command mapping are learned by imitation from the teacher. In stage (ii), the loss is restricted to the command tokens while the reasoning is treated as conditioning context and masked out,

$$\mathcal{L}_{\text{cmd}} = -\sum_{t \in u} \log \pi_\theta(u_t \mid x, c, u_{<t}), \tag{4}$$

so that gradients flow only through the reasoning-to-command compression rather than re-teaching the reasoning itself (which would risk self-plagiarism of already-learned traces). Stage (ii) uses one-tenth of the stage-(i) learning rate (1.0×10^{-6} vs. 1.0×10^{-5}) and mixes in 10% stage-(i) data as replay to avoid catastrophic forgetting. An optional safeguard, should the command line degrade action accuracy, is to down-weight it inside Eq. (3) as $\mathcal{L} = \mathcal{L}_{\text{analysis+action}} + \lambda \mathcal{L}_{\text{command}}$ with $\lambda < 1$; in our pilot runs, keeping $\lambda = 1$ preserved action accuracy and was therefore used throughout.

D Case Study Walkthrough

The representative case involves the player hero A Gu Duo pushing the mid lane with the ally Jiang Ziya against a weakened enemy tier-one tower. The model's reasoning unfolds in four stages: (i) *situation analysis*—recognizing that the early-game protection mechanisms have expired, spotting the low-health enemy tower as the primary opportunity, and flagging risks from unknown enemy positions and A Gu Duo's own low health; (ii) *objective prioritization*—ranking the mid-lane tower as the most immediately achievable goal and emphasizing coordination with Jiang Ziya's crowd-control abilities for a safer push; (iii) *strategy formulation*—"join Jiang Ziya at the enemy mid-lane tier-one tower and focus fire", paired with an explicit ambush warning; and (iv) *hero-specific playstyle integration*—advising the marksman-role hero to "maintain a safe distance for output" in tandem with the ally's control effects. The full chain is finally distilled into one concise, executable instruction to the player: "Jointly push down the enemy mid-lane tier-one tower with Jiang Ziya; be mindful of a potential enemy ambush." Even though the intermediate reasoning contains a factual slip (miscounting surviving heroes), the distilled guidance remains tactically sound—precisely

Table 10: Open-ended game-QA on questions paired with the 512 held-out scenarios of Table 2, stratified over early/mid/late game stages. Expert players score each answer on four dimensions of the 0/1/2 rubric—state grounding (Groun.: faithful use of in-game information), decision reasoning (Decis.), game knowledge (Knowl.), and actionability (Act.: whether the answer translates into an executable player action)—and each column reports the percentage of answers scored fully correct (score 2). Hall. = percentage of answers containing fabricated game-state claims (lower is better). Ov. = mean 0–2 score across the four dimensions. All scores are averaged over three blind expert raters.

Model	**Training**	**Groun.**	**Decis.**	**Knowl.**	**Act.**	**Hall.↓**	**Ov.**
Qwen3-32B	zero-shot	50.1	52.3	66.9	59.1	5.1	1.26
DeepSeek-R1	zero-shot	58.9	61.3	79.4	56.2	8.6	1.52
GPT-4o	zero-shot	55.7	57.4	74.8	59.1	9.8	1.51
Qwen3-8B	zero-shot	30.4	28.9	42.1	26.8	21.5	0.79
Qwen3-8B	SFT	43.1	40.2	41.3	38.6	15.2	0.99
Qwen3-8B	SFT + GRPO (500 steps)	49.2	51.0	41.7	47.1	11.6	1.18
Qwen3-8B	SFT + GRPO (3000 steps)	52.6	56.1	41.2	50.8	10.1	1.29
Qwen3-14B	zero-shot	41.3	43.0	56.2	39.4	16.2	1.06
Qwen3-14B	SFT	50.6	52.1	55.3	47.9	12.0	1.25
Qwen3-14B	SFT + GRPO (500 steps)	56.9	59.6	55.7	55.2	9.3	1.42
Qwen3-14B	SFT + GRPO (3000 steps)	61.2	64.1	55.0	59.0	7.7	1.56

Table 11: Training configuration summary for Qwen3-8B and Qwen3-14B. Dataset size and optimization budget are reported separately; settings are shared across backbones unless specified otherwise.

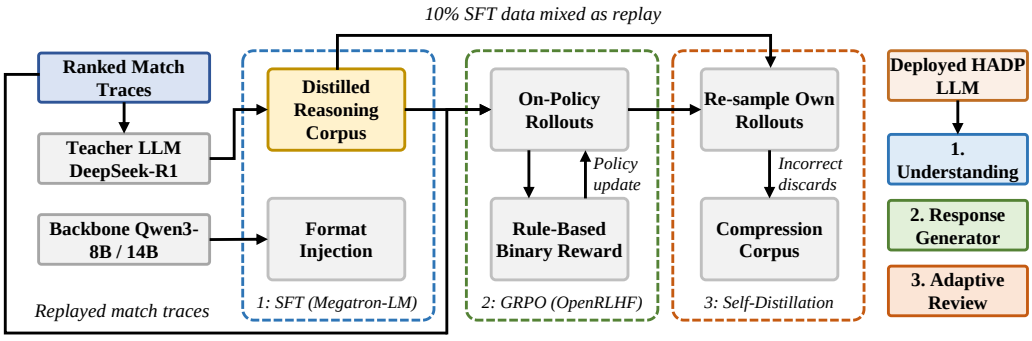
Stage	Data	Dataset size	Training budget	Objective / reward	Settings / implementation
SFT with format injection	Teacher-distilled gameplay traces with reasoning, actions, and commands	25,000 examples	2 epochs	Response-only cross-entropy over reasoning, action, and command tokens; prompt tokens masked	LR: 10^{-5}; Megatron-LM
GRPO	Gameplay-derived states; on-policy text rollouts from the current model	5,000 distinct real-match scenarios	GRPO-only: 500 steps; after SFT: 500 or 3,000 steps	Binary macro-action correctness; no neural reward model or format reward	OpenRLHF; LR: 10^{-6} (8B), 5×10^{-7} (14B)
Command self-distillation	Correct model rollouts compressed into (reasoning, command) pairs	61,983 examples	2 epochs	Command-token cross-entropy; prompt and reasoning tokens masked	LR: 10^{-6}; 10% SFT-data replay; Megatron-LM (SFT) / OpenRLHF (fine-tuning)

Notes. Dataset sizes and training budgets are per backbone, not totals summed across Qwen3-8B and Qwen3-14B; self-distillation continues tuning the corresponding backbone. The 5,000 GRPO inputs count distinct real-match training scenarios, not sampled rollouts; the held-out evaluation set contains 512 scenarios. Equal GRPO step counts across backbones do not imply equal compute. GRPO rollouts are text generations from trace-derived states, not closed-loop gameplay. Training uses 4×8 NVIDIA H200 GPUs. Deployment-time inference limits are not specified by this table.

the input–output contract HADP requires, with rich situational reasoning inside the Understanding/Generator pipeline and compact, player-facing language at the surface.

E Game Commands in Commander Mode

Figure 8 shows the existing command interface in Commander Mode. It provides nine command types and twelve command–interaction combinations: *Attack*, *Retreat*, and *Rally* each support both clicking and dragging to a designated area (six combinations), whereas the other six commands support clicking only. Here, the count of twelve distinguishes interaction variants rather than twelve distinct tactical intents.



Infra: 4 servers × 8 NVIDIA H200 · prompt length 8,360 · generation length 2,048

Figure 7: The multi-stage training pipeline: distillation-based SFT with format injection over the three-part output $y = [c; a; u]$, online GRPO with a rule-based binary reward, and command-channel self-distillation.

Click-and-drag commands. As shown in Figure 8(a), *Attack* requests an offensive action, *Retreat* requests disengagement, and *Rally* requests regrouping. Each can be issued by clicking its button or by dragging it to a designated area; the latter explicitly specifies the spatial target of the instruction.

Table 12: Relative changes of key in-product metrics for the treated versus control cohort during the six-week staged (gray) roll-out of the deployed dialogue policy in production Commander Mode. Absolute rates are proprietary and omitted; ranges span the successive roll-out stages.

Level	Metric	Relative change
Interaction	Command-adoption rate	$+6.2\% \sim +8.4\%$
Experience	Session satisfaction feedback	$+2.1\% \sim +3.5\%$
Retention	Seven-day retention	$+0.7\% \sim +1.3\%$

Click-only commands. Figure 8(b) shows the six click-only commands. *Push Tower* requests pressure on an enemy tower; *Initiate Teamfight* requests starting a team engagement; *Clear Minion Wave* requests clearing lane minions; *Farm* requests accumulating resources; *Ambush in Brush* requests waiting in brush for an ambush opportunity; and *Scout Brush* requests checking brush for hidden opponents. These six commands are issued by clicking, without a drag-to-area variant.

Text input. In addition to the predefined commands, players can type their own instructions through the text-input interface shown in Figure 8(c). This allows players to express requests in their own words rather than selecting a predefined command. Text input is an additional interaction channel and is not included in the twelve predefined command–interaction combinations above.

F Production Deployment Details

This appendix supplements Section 4.6 with the cohort-based A/B metrics of the six-week staged (gray) roll-out in production Commander Mode. Players were deterministically bucketed into cohorts differing only in whether the upgraded dialogue policy served their sessions; per-stage cohort sizes and absolute rates are proprietary and omitted, so Table 12 reports only relative changes of the treated cohort against its matched control, with ranges spanning the successive roll-out stages.

G Conditional Theoretical Analysis of HADP

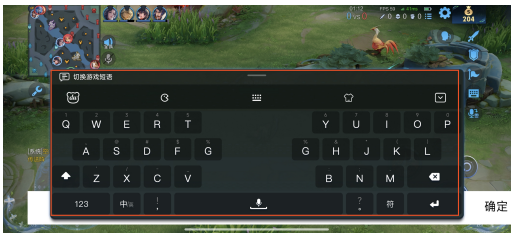
This appendix examines three mechanisms underlying HADP: a shared semantic interface, pre-execution review, and memory of prior commitments. The results are conditional statements, not guarantees that hierarchy outperforms a flat policy. They explain how particular information and error-control mechanisms can help; they do not establish improved sample complexity, optimal game return, or error-free execution.



(a) Click-and-drag commands.



(b) Click-only commands.



(c) Text-input interface.

Figure 8: The existing Commander Mode command and text-input interfaces (original Chinese UI). (a) Attack, Retreat, and Rally support both clicking and dragging to a designated area. (b) Push Tower, Initiate Teamfight, Clear Minion Wave, Farm, Ambush in Brush, and Scout Brush support clicking only. Together, the nine predefined command types provide twelve command–interaction combinations. (c) Players can also type instructions in their own words through an additional text-input channel.

G.1 Setup and Shared-Commitment Consistency

Let H_t denote the interaction history available at decision time t, including observed game states and player messages. Understanding constructs a Mission M_t, and the Response Generator selects a collaborative stance S_t and produces a candidate plan A_t and utterance U_t, also conditioned on the RL-derived action. For analysis, write C_t for the commitment specified by the Mission together with the selected stance and any material activation conditions or deadlines. This is an analytical description of existing outputs, not an additional implemented module or a claim that the five Mission fields alone specify a complete commitment. Here A_t denotes the issued plan rather than the subsequent environment trajectory.

Fix an evaluation population of actionable plan–utterance pairs. For each commitment c, define sets $\mathcal{A}(c)$ and $\mathcal{U}(c)$ of admissible plans and utterances. Assume the interface is sufficiently specific that every pair in $\mathcal{A}(c) \times \mathcal{U}(c)$ agrees on the stance and all material conditions. This *compatibility assumption* is substantive: merely conditioning both channels on the same Mission does not imply it.

Proposition 1 (interface-based error decomposition). Let $E_A = \{A_t \notin \mathcal{A}(C_t)\}$ and $E_U = \{U_t \notin \mathcal{U}(C_t)\}$, with probabilities ϵ_A and ϵ_U. Under the compatibility assumption, the semantic inconsistency event F_{SC} satisfies

$$\Pr(F_{\mathrm{SC}}) \le \min\{1, \epsilon_A + \epsilon_U\}. \tag{5}$$

Proof. If neither E_A nor E_U occurs, both outputs belong to the admissible sets for the same commitment and are compatible by assumption. Thus $F_{\mathrm{SC}} \subseteq E_A \cup E_U$, and the union bound gives the result. No independence between channels is required. □

Grounding versus mutual agreement. Both channels can agree while jointly misinterpreting the request. Let G be the event that C_t is an invalid contextual response to the request, with $\Pr(G) = \epsilon_G$. Validity permits justified refusal or conditional acceptance; it does not require unconditional obedience. Define $\bar{\epsilon}_A = \Pr(E_A \mid G^c)$ and $\bar{\epsilon}_U = \Pr(E_U \mid G^c)$ when $\Pr(G^c) > 0$. For the composite failure $F_{\mathrm{ground}} = G \cup F_{\mathrm{SC}}$,

$$\Pr(F_{\mathrm{ground}}) \le \epsilon_G + (1 - \epsilon_G) \min\{1, \bar{\epsilon}_A + \bar{\epsilon}_U\}. \tag{6}$$

This follows by conditioning on G and applying Proposition 1 on G^c; if $\epsilon_G = 1$, the bound is simply one. The composite event is an analytical diagnostic, not a redefinition of the reported SC metric. The structural benefit is an explicit interface for separating grounding errors from channel-realization errors. Neither bound proves that HADP has smaller component errors than a flat baseline, or that a compatible plan will actually be executed.

G.2 When Adaptive Review Reduces Failure Risk

Consider a fixed distribution of review inputs and a fixed binary failure criterion. Let Y be a candidate output and Y^+ the final outcome of the review procedure, including any regeneration, suppression, and delay. Write $L(Y)$ and $L(Y^+)$ for their failure indicators under the same criterion. For a trajectory-level criterion, these denote the respective potential rollout outcomes from the same pre-review context, with environmental randomness included; observing both requires a paired or otherwise controlled evaluation. The following identity is not a claim that such counterfactual outcomes are available in the reported experiments.

Proposition 2 (net benefit of review). Suppose $0 < p < 1$, where $p = \Pr(L(Y) = 1)$. Define the effective repair rate $r = \Pr(L(Y^+) = 0 \mid L(Y) = 1)$ and the damage rate $f = \Pr(L(Y^+) = 1 \mid L(Y) = 0)$. Then

$$p^+ = \Pr(L(Y^+) = 1) = p(1 - r) + (1 - p)f, \qquad p - p^+ = pr - (1 - p)f. \tag{7}$$

Consequently, review strictly reduces failure risk if and only if $pr > (1 - p)f$.

Proof. Partition the evaluation population according to $L(Y)$. Failed candidates remain failures with conditional probability $1 - r$; successful candidates become failures with conditional probability f. The law of total probability yields the first equality, and rearranging yields the second and the strict-improvement condition. □

At $p = 0$, review cannot strictly improve an already zero failure rate; at $p = 1$, it improves the rate exactly when it repairs a positive fraction of failures. Undefined conditional rates at these endpoints are unnecessary. Crucially, r measures successful final repair, not merely error detection. A detected error followed by failed regeneration is not a repair. Suppression of a required action or a missed deadline must be scored as failure when the criterion requires timely execution; justified refusal or cancellation is evaluated according to its own commitment conditions.

Scope and testable implication. This identity applies to any reviewer, including flat self-refinement. HADP's proposed advantage is that explicit mission information and recent memory may improve effective repair while limiting damage, not that hierarchy changes the identity. Estimating this mechanism requires candidate/final-outcome pairs on a common population, with unchanged eligibility rules and recorded suppression, latency, and activation. Comparing failure rates only among accepted outputs can manufacture an apparent gain by dropping difficult cases. The aggregate ablation rates in Section 4.1 do not identify r or f, and SC, EC, and IC must be analyzed separately rather than multiplied as stages of a success funnel.

G.3 Memory and Commitment-State Ambiguity

Let X_t be the current observation exposed to a policy that does not retain commitment history. Consider a two-context population with histories h_1 and h_0, occurring with probabilities q and $1 - q$, where $0 < q < 1$. Both histories produce the same $X_t = x$, but their prior commitments differ. At a fixed decision deadline, consistency in h_1 requires executing a specified feasible action, whereas consistency in h_0 prohibits that action. For example, a delayed-cooperation condition has become active in the first history, while the corresponding request was explicitly retracted in the second. Assume these obligations are valid under the current game state and no alternative revision discharges the first obligation at this deadline.

Proposition 3 (irreducible ambiguity without history). Any policy whose action distribution depends only on X_t has commitment-violation probability at least $\min\{q, 1 - q\}$ on this population. If a memory variable B_t accurately distinguishes the two histories, a policy using (X_t, B_t) can attain

zero violation in this idealized decision problem, provided it can reliably execute the required action or withhold it.

Proof. Let α be the probability that the observation-only policy executes the specified action. Because its input is identical in both histories, α is the same in each. Its violation probability is

$$R(\alpha) = q(1-\alpha) + (1-q)\alpha, \qquad \min_{\alpha \in [0,1]} R(\alpha) = \min\{q, 1-q\}. \tag{8}$$

The minimum of this affine function is attained at an endpoint. With a distinguishing memory, choose the action in h_1 and withhold it in h_0; both obligations are satisfied under the stated execution assumption. □

If memory instead predicts the relevant binary commitment state with error probability η, following that prediction has violation probability η in the same idealized setting. If the probability of execution deviating from this selected decision is at most δ, the union bound gives a violation probability at most $\min\{1, \eta + \delta\}$. Thus a sufficient condition for a strict advantage over the observation-only lower bound is $\eta + \delta < \min\{q, 1-q\}$. This condition highlights that stale or inaccurate memory and execution failures can erase the information advantage.

What this establishes for HADP. Memory can resolve a genuine information deficit when current observations omit prior commitments. HADP's STM provides a mechanism for retaining such context, but Proposition 3 does not establish that its actual retrieval is sufficient or accurate. Nor does it imply that longer memory is always better or that LTM is necessary: the relevant information may already reside in STM. A flat policy with the same usable history or an equivalent recurrent state is not subject to this observation-only comparison.

G.4 Interpretation and Limits

Together, these results motivate *structured inductive bias and conditional risk control*: a shared commitment makes channel errors inspectable, review helps when effective repairs outweigh introduced failures, and memory can distinguish otherwise ambiguous obligations. These mechanisms are compatible with the grounding and review ablations, but those ablations do not verify the propositions' assumptions or estimate their component error rates. In particular, mutual semantic agreement is distinct from correct grounding, successful execution, and strategically useful behavior.

An unrestricted flat policy with the same information and sufficient computational capacity can in principle simulate the composed hierarchical computation. Therefore, the analysis establishes no unconditional expressivity advantage, finite-sample learning guarantee, or optimal-return dominance for HADP. Mission compression can omit useful information; module errors can propagate; and review can introduce errors or latency. Establishing a practical advantage under matched data and inference budgets remains an empirical question. The propositions identify conditions and diagnostics for that question rather than substitute for closed-loop evidence.

H Architectural Reusability and Adaptation Guidelines

Scope of reuse. HADP separates request interpretation, coordinated action–dialogue generation, and contextual review through a shared Mission and memory. The reusable element is this organization of responsibilities and semantic interfaces, not a claim that the Honor of Kings model weights, prompts, or execution stack transfer unchanged. Our experiments evaluate a single game. The guidelines below describe a proposed adaptation pathway, not an implemented cross-environment framework or evidence of zero-shot transfer.

H.1 Reusable Core and Environment-Specific Components

The core workflow is to ground a request, select an explicit collaborative stance, generate compatible action and dialogue candidates, and review them against current context before execution. Immediate cooperation, refusal, and delayed cooperation describe responses to requests rather than game-specific actions. Likewise, distinguishing semantic, execution, and interaction consistency is useful wherever an agent communicates commitments and subsequently acts. Table 13 separates these reusable concepts from the components that require adaptation.

Table 13: Conceptual reuse boundaries for adapting HADP. These are design guidelines, not measured transfer results or claims of an existing plug-in API.

Component	Reusable core	Environment-specific adaptation
Understanding	Ground requests into an explicit Mission using state and recent interactions.	Observation encoding, entity resolution, command vocabulary, executor/target sets, and temporal-condition semantics.
Response Generator	Condition action and dialogue on a shared task interpretation and collaborative stance.	Action proposal source, executable command vocabulary, feasibility constraints, and control interface.
Adaptive Review	Check candidates against current context before release; suppress or regenerate when appropriate.	Task-specific validity rules, deadlines, cancellation behavior, review examples, and latency budget.
STM/LTM	Retain relevant interaction context and, where needed, longer-term preferences.	State/event representation, retrieval, retention, privacy rules, and updates after execution or revision.
Evaluation	Separate SC, EC, and IC from task success and language quality.	Behavioral predicates, trigger conditions, observation windows, eligibility, and task-success criteria.

Mission as an adaptation boundary. The five fields—Is Command, Topic, Executor, Target, and Temporal Cue—provide a starting schema. Reuse preserves their semantic roles while replacing environment-specific values and grounding rules. It does not require retaining MOBA entities or treating natural-language temporal cues as directly executable conditions. A target environment must resolve references and map relevant cues to observable events or explicitly represent uncertainty. Tasks with multiple objects, resource dependencies, or compound goals may require schema extensions; both generation and review must then interpret those extensions consistently. The selected stance and any material conditions must remain aligned across outputs, as discussed in Appendix G.1.

Planner and model replaceability. The current implementation uses an RL-derived MOBA action as an input to the Response Generator. Adapting the architecture could replace this source with a target-environment policy or planner, provided its proposals can be mapped to the task interpretation and executable commands. This is a proposed replacement boundary, not an evaluated interchangeable-planner interface. Models and prompts may require target-domain supervision or retraining: architectural reuse alone does not establish model-weight transfer, lower annotation cost, or equivalent performance. Action dispatch, feedback, and cancellation must also be implemented in the target environment; producing a well-formed command is not evidence that the environment executed it.

H.2 Illustrative Adaptation to Cooperative Cooking

The following example is hypothetical and has not been implemented or evaluated in this work. Consider a cooperative cooking environment in which a player tells an AI teammate, "When this pot is free, start cooking soup." Assume the referenced pot and teammate can be identified, the ingredients are available, and the environment exposes pot occupancy and cooking events.

1. **Ground the request.** Understanding maps the instruction to Is Command = true, Topic = cook soup, Executor = the addressed teammate, Target = the referenced pot, and Temporal Cue = when the pot becomes free. Pot identity and occupancy replace the corresponding game-specific target and state information; an unresolved reference should trigger clarification rather than an invented target.
2. **Coordinate the response.** If the request is feasible but the pot is occupied, the generator selects delayed cooperation and produces a conditional plan with a matching utterance, such as "I'll start the soup when this pot is free." This statement does not imply that cooking has already begun. An infeasible request instead requires an appropriate refusal or explicit qualification.
3. **Review against updated context.** An adapted runtime would retain the pending commitment and re-evaluate it when relevant events occur. If the pot becomes free while the request remains

valid, it would generate and review the activation command before release. If the player retracts the request, it would cancel the pending plan and explicitly acknowledge the revision. If ingredients become unavailable, it would communicate a justified revision rather than silently abandon the promise. Event monitoring and pending-plan management are integration requirements, not additional components claimed to have been tested here.

4. **Evaluate the outcome.** SC checks agreement between the utterance and the conditional plan. EC applies once the plan is activated and checks the required cooking behavior within a predefined deadline. IC checks whether the conditional commitment, refusal, or explicit revision is honored throughout the observation window. Overall cooking success is measured separately.

This example preserves the grounding–generation–review organization while changing entities, observations, action semantics, and success predicates. It illustrates an interface mapping, not a prediction of performance in another game.

H.3 Adaptation and Validation Procedure

A prospective reuse study should document the following steps and distinguish unchanged components from rewritten or retrained ones.

1. **Specify the environment contract.** Identify available observations and timestamps, valid actions, controllable agents, execution feedback, and cancellation support. Define timing constraints and distinguish unavailable information from negative observations. If condition monitoring or cancellation is unsupported, narrow the commitments the system can make rather than promise unsupported behavior.
2. **Adapt the semantic interface.** Define Mission vocabularies, reference-resolution rules, temporal triggers, and any schema extensions. Specify how the selected stance, conditions, and deadlines are shared by action generation, dialogue generation, and review.
3. **Connect and adapt the modules.** Replace state parsing and action execution adapters, integrate a suitable action proposal source, and adapt prompts or model parameters as necessary. Define memory updates and retrieval for accepted, activated, completed, and revised commitments. Verify that changes propagate to downstream modules rather than leave stale interpretations in use.
4. **Exercise commitment-sensitive cases.** Test immediate cooperation, refusal, delayed activation, retraction, conflicting instructions, missing observations, and execution failures. Check both action behavior and player-facing communication; count harmful suppression, failed regeneration, and missed deadlines rather than hiding them through selective acceptance.
5. **Measure reuse and transfer separately.** Under matched target-domain data, backbone, information access, and inference budgets where feasible, compare the adapted hierarchy with a flat policy and budget-matched self-refinement. Report SC/EC/IC with eligibility and censoring counts, refusal and activation rates, task success, and latency. Record implementation effort, adapted components, training examples, and training/inference cost to substantiate claims about reuse effort. None of these cross-environment measurements is supplied by the present study.

Applicability and limits. The proposed reuse is most natural for interactive agents that receive requests, expose meaningful executable actions, and can observe the events needed to track commitments. Environments with inaccessible execution feedback, unobservable triggers, or strict control deadlines may require substantial redesign or may not support the same consistency checks. The framework is not a safety certification for physical or high-stakes systems. Accordingly, the current contribution is a reusable architectural pattern with explicit adaptation boundaries; cross-environment effectiveness and reduced adaptation cost remain empirical questions.

Ethics Statement

AI-controlled teammates raise considerations around transparency, fair play, and data use. In the production deployment, players are informed that teammate agents are AI-driven, and agent behavior is constrained to in-game tactical communication without access to personal player data beyond consented session statistics. Fluent dialogue can nevertheless encourage players to overestimate an agent's reliability; language-quality ratings and the reported consistency scores do not establish

trustworthiness in unrestricted play. Transparency about AI control and data-use boundaries remains relevant regardless of perceived human-likeness.

Reproducibility Statement

We describe the supporting offline evaluation protocol, including the construction of the held-out testbed from real match traces, temporal separation from training data, macro-action prediction metrics, command-quality scoring rubrics, and blind human evaluation procedure, in Section 4. Section 4.1 defines the primary closed-loop consistency metrics and comparison conditions; reproducing these results additionally requires the trajectory provenance, per-configuration denominators, censoring counts, observation windows, behavioral criteria, and interval-estimation procedure identified there. Training pipelines, stages (SFT, command-channel format injection and self-distillation, and GRPO), and key hyperparameters are detailed in Section 4.2; all prompt templates used by the LLM modules and by the LLM-as-judge evaluator are given in the figures referenced therein (see Appendix B), and the inference-time command length and generation budget are reported alongside the baselines. Judge–expert agreement (Spearman's $\rho = 0.81$; Cohen's $\kappa = 0.72$, binarized at 1.5) is reported to calibrate the evaluator. Compute is a modest post-training budget of 4×8 H200 GPUs. The raw match logs and the production serving stack are proprietary and cannot be released; to facilitate verification, we will release the anonymized testbed scenario statistics, the scoring rubrics, and the set of distilled command-quality annotations upon acceptance.

AI Use Statement

Large language model assistants were used to polish English wording, support LaTeX editing, and assist in formulating and drafting the conditional theoretical analysis and proofs in Appendix G. This theoretical assistance did not generate or modify experimental observations. The authors retain responsibility for reviewing and verifying all claims, assumptions, proofs, and results, and for the full content of the paper.