Aligning Embodied Companion Agent Evaluation with User Experience

Seohyeon Jung* **Sue Hyun Park*** **Byeongju Kim*** **Jimin Hong**
Youngin Cho **Jaewoong Cho†** **Hyunseung Kim†**
KRAFTON

Abstract

Embodied companion agents, such as AI teammates in cooperative multiplayer games, converse with users while acting alongside them in shared virtual worlds. We define **companion user experience** (**companion UX**) as the user's perception of the agent as a companion, shaped jointly by dialogue, action, timing, and social appropriateness. Existing evaluation methods often assess speech and action separately, but shared-world interaction makes them mutually grounded in each other and in the evolving game state. Companion UX is difficult to specify in advance, because what feels helpful, natural, or socially appropriate varies across users and situations and often becomes visible only during play. We frame evaluation of logged or simulated interactions not as a fixed benchmark, but as an **iterative alignment process** that uses evidence from user tests to revise what behaviors are measured and how they are combined into judgments. In a month-long user test with 1k users and 38k sessions, we construct a paired trajectory–feedback corpus linking gameplay trajectories to UX ratings, free-text feedback, and pairwise preferences. Through this iterative alignment process, we develop an evaluation suite whose results better align with user preferences collected in user tests. A subsequently deployed model selected using the refined suite received recommendation ratings comparable to the strongest user-tested model.

1 Introduction

Embodied companion agents, designed for persistent and socially grounded interactions with users in dynamic environments, are becoming prevalent [Carroll et al., 2019, Wei et al., 2026, Gao et al., 2024]. These agents are expected to continuously make safe and effective decisions while remaining socially compatible in rapidly changing situations. **Companion user experience (companion UX)**, which captures the quality of these expectations within a shared-world interaction, is shaped by dialogue quality and task effectiveness, but also by how the agent's utterances, actions, timing, and social appropriateness are interpreted together. In this regime, a helpful utterance in isolation can be unhelpful in some situations, while an effective action in isolation can be perceived as uncooperative if it contradicts what the agent just said or what the user appears to intend.

However, prior work on evaluating embodied companion agents has largely focused on isolated aspects of companion quality, such as conversational naturalness [Chiang et al., 2024, Li et al., 2025, Walker et al., 1997] and task completion under controlled conditions [Jimenez et al., 2024, Zhou et al., 2024, Xie et al., 2024, Ye et al., 2026]. Recent tool-agent-user benchmarks move closer to interactive evaluation by adding simulated users and shared task states [Yao et al., 2024, Barres et al., 2025], but their target remains primarily goal completion under a scripted or simulated interaction

*Equal contribution.
†Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026). Workshop: Dynamic Alignment on Human-AI Coupled System (BiAlign Workshop).

protocol, rather than the user's experience of cooperating with an agent in a dynamic shared world. In this paper, we use offline evaluation to refer broadly to non-user-facing evaluation in controlled, logged, or simulated settings, including cases where an agent interacts with an environment or simulator. We study this problem using PUBG Ally [PUBG Ally Team, 2026], a companion agent that communicates and acts as the user's teammate in a cooperative first-person shooter survival game.

This contrast becomes concrete in the episode shown in Figure 1. The user is incapacitated while enemies are approaching, and the agent chooses to revive the user in an exposed position. Although this behavior appears locally helpful, the user rates the session poorly because the agent does not sufficiently account for the broader game state, including enemy pressure, exposure, and the timing of the fight. The failure is therefore not simply a bad action, but a wrong priority under simultaneous demands. Such failures show why evaluating utterances without trajectories, or actions without dialogue and state context, can miss the situated properties that shape companion UX. Moreover, these properties are difficult to fully specify in advance: what feels helpful, natural, or socially appropriate varies across users and situations, and often becomes visible after play.

We present an iterative process that refines an offline evaluation suite using evidence from live user tests. Our evaluation suite *holistically* evaluates companion agents, unlike existing benchmarks that independently assess dimensions such as dialogue quality, task effectiveness, and social behavior. Our evaluation suite defines trajectory-grounded interaction tasks and evaluates agent behavior through structured behavioral criteria, graders, and aggregation procedures. The trajectory-grounded tasks are constructed from gameplay trajectories and require agents to determine appropriate utterances and actions within ongoing interaction. Rather than relying on fixed evaluation criteria, we use an **iterative alignment process** to refine the suite with real gameplay trajectories and user experience feedback, so that offline evaluation outcomes better align with companion UX. We begin with an initial evaluation suite that encodes criteria for desirable companion behavior, then evaluate selected agent versions through live user tests and collect gameplay trajectories together with play-window UX reports, including ratings, free-text feedback, and pairwise preferences. By linking UX feedback to temporally aligned gameplay segments, we identify which interaction patterns correspond to positive or negative companion experiences and refine the suite accordingly.

Over a month-long user test with 1k users and 38k play sessions, this iterative process substantially improved the alignment between offline evaluation and observed companion UX. Across user-test rounds, we expanded behavioral coverage by adding graders, revised aggregation logic and user-salient weighting, and introduced stage-level checks. In an ablation study of these refinements, Kendall rank correlation between suite-induced model rankings and rankings derived from live user preferences was $+0.05$ with deterministic rule-based validity checks alone and $+0.71$ when all refinement components were included.

Our contribution is a user-test-grounded methodology for aligning offline evaluation with companion UX. First, we formulate companion-agent evaluation as an iterative alignment problem, where offline evaluators are revised using evidence from live user interaction rather than fixed once in advance. Second, we construct a paired trajectory–feedback corpus that links play-window UX reports to concrete gameplay evidence. Third, we show that this evidence can guide systematic refinement of an evaluation suite toward better alignment with live user preferences. Our analysis further shows that evidence from user tests reveals different sources of evaluation mismatch, including missing behavioral coverage, misweighted trade-offs, and stage-level deployment constraints.

2 Related work

Task-completion agent benchmarks. Agent evaluation has progressed from binary success in narrow domains toward broader environments, multi-turn interaction, trajectory-aware grading, and benchmark-validity analysis. Benchmarks evaluate web [Zhou et al., 2024], operating-system [Xie et al., 2024], software-engineering [Jimenez et al., 2024], and command-line [Merrill et al., 2026] agents through execution-based checks, hidden tests, or environment-defined success criteria, while multi-environment platforms evaluate agents across heterogeneous interactive settings [Liu et al., 2024, Ma et al., 2024]. Recent work extends this paradigm with stateful tool use [Lu et al., 2025], simulated users [Yao et al., 2024], dual-control interaction [Barres et al., 2025], life-serving scenarios [He et al., 2026], trajectory-aware grading [Ye et al., 2026], and live websites [Zhang et al., 2026].

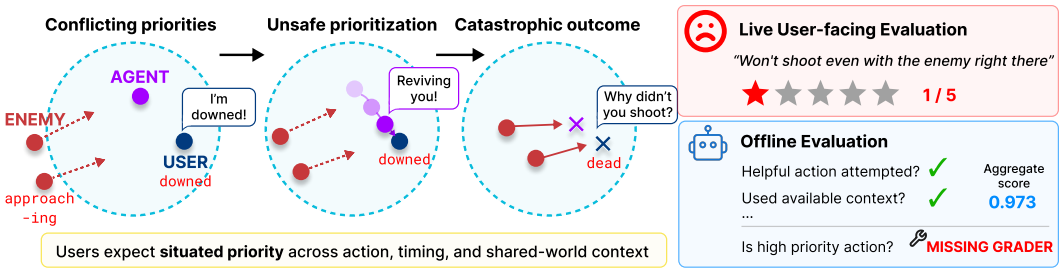


Figure 1: **Illustration of Evaluation–UX misalignment in embodied companion agents.** In a shared-world episode, the agent attempts a locally helpful rescue under active threat, leading to poor user-perceived companion UX. The case illustrates that evaluation must capture situated priority across action, timing, and world context, not only isolated action helpfulness.

These benchmarks establish rich execution-based evaluation settings, but their target signals remain task success, protocol compliance, or evaluator-assigned correctness rather than companion UX.

Player-centered game companions and cooperative AI. Game and open-world agent evaluation has emphasized strategic strength [Vinyals et al., 2019], human-likeness [Hingston, 2009], open-ended task capability [Fan et al., 2022], situated reasoning [Tang et al., 2024], and cooperative performance [Carroll et al., 2019, Bard et al., 2020]. Closer to our setting, F.A.C.U.L. [Wei et al., 2026] studies a language-driven FPS companion with real-world user feedback, while RLHG [Gao et al., 2024] uses human goal achievement to train a cooperative teammate in a multiplayer online battle arena game. These works show that user experience can inform the evaluation and development of game companions, while our work asks how user-test evidence can be used to revise offline evaluation itself.

Proxy-based user-experience evaluation. Dialogue-system evaluation has long estimated user satisfaction and related experience signals from proxies such as task outcomes, interaction features, rubrics, or preferences [Walker et al., 1997, Schmitt and Ultes, 2015, Deriu et al., 2021], while self-play evaluation uses sentiment- and coherence-based trajectory proxies to approximate interactive human evaluation [Ghandeharioun et al., 2019]. Recent LLM-evaluation work extends the same idea through multidimensional rubrics [Hashemi et al., 2024], interpretable user-satisfaction rubrics [Lin et al., 2024], scalable pairwise preference collection [Chiang et al., 2024], and benchmarks distilled from crowdsourced preference data [Li et al., 2025]. We draw on this proxy-based view for embodied interaction, where utterances and actions are coupled, and focus on adapting the proxy evaluation procedure using user-test evidence.

Adaptive and participatory evaluation methodology. A parallel line studies how evaluation processes should be constructed, audited, adapted, and grounded in human feedback, spanning dynamic benchmarking [Kiela et al., 2021], benchmark-quality assessment [Reuel et al., 2024], participatory feedback [Kirk et al., 2024], and selective test-time learning for LLM evaluators [Jwa et al., 2026]. Together, they support our view that evaluation processes encode assumptions that should be made explicit and, when needed, revised. Our focus is the mismatch between offline evaluation and companion UX: how post-session UX signals paired with gameplay trajectories can reveal mismatches between automated evaluation and user-perceived companion quality, and guide revision across repeated user tests.

3 Evaluation suite and iterative alignment process

In this section, we introduce an offline evaluation suite for embodied companion agents together with an iterative alignment process that uses evidence from user tests to better align offline evaluation with companion UX. We first define an evaluation suite that evaluates each model on the same fixed set of game-state and dialogue contexts. We then describe how UX reports are linked to gameplay trajectories, and how this evidence is used to iteratively align offline evaluation with observed companion UX.

3.1 Evaluation suite

The evaluation suite approximates companion UX from logged or replayed interaction trajectories. Because companion UX is user-perceived rather than directly observable, the suite approximates it by evaluating observable behaviors that shape the user's experience, such as dialogue quality, action appropriateness, timing, and social compatibility.

An evaluation suite H consists of the following components:

- **Behavioral criteria and rubrics.** A behavioral criterion specifies one property of companion-agent behavior to measure, such as command follow-through, appropriate memory use, tool-use validity, or consistency between speech and action. A rubric groups one or more behavioral criteria that can be judged together for the same trajectory.
- **Graders.** A grader scores trajectories according to the criteria in its rubric and outputs criterion-level judgments, such as scores, rationales, diagnostic labels, or detected failure patterns. Graders may be rule-based, using deterministic checks, or model-based, using an LLM judge [Zheng et al., 2023]. The full grader catalog used in our evaluation suite is provided in Appendix F.
- **Stage aggregation logic.** Stage aggregation logic combines criterion-level judgments into metrics and diagnostic artifacts, such as pass rates, failure summaries, candidate margins, severity-weighted scores, or stage-level verdicts.
- **Evaluation stages.** An evaluation stage groups trajectory inputs, graders, and stage aggregation logic around a common diagnostic purpose. We write stage ℓ as $s_\ell = (\mathcal{I}_\ell, \mathcal{G}_\ell, \mathcal{L}_\ell)$, where $\mathcal{I}_\ell$ specifies the trajectory-derived evidence the stage may inspect, $\mathcal{G}_\ell$ is the stage-local set of graders together with their associated criteria and rubrics, and $\mathcal{L}_\ell$ is the stage aggregation logic. Evidence inputs may include trajectories, grader rationales, diagnostic labels, or previously computed metrics.

The evaluation suite is an ordered sequence of evaluation stages, $H = (s_1, \ldots, s_\ell)$. Replay uses fixed test instances to generate trajectories for each model, while the evaluation suite processes those trajectories through its stages and produces model-level comparisons and diagnostic artifacts. Settings are in Appendix G.1.

The companion agent is implemented as an *agent harness*: a fixed system of prompts, tools, memory mechanisms, action interfaces, and control logic that allows a language model to observe, speak, and act in the game environment. Further system details are provided in Appendix C. The model receives a text representation of the game state and acts through high-level function-calling tools. In our comparisons, the harness is held constant and only the underlying language model varies, so the evaluation suite compares language models under the same companion-agent system rather than different agent architectures.

3.2 User-test evidence for suite refinement

We use user tests to collect evidence about how users experience the companion agent during gameplay. A session is one complete gameplay match in which one user plays with an agent teammate in the shared game world. Each experimental condition varies only the language model inside the fixed agent harness. The resulting evidence from user tests includes gameplay trajectories, UX reports, and a paired trajectory–feedback corpus that links free-text feedback to concrete agent behavior.

Gameplay trajectories. Each session contains multiple trajectories. For trajectory σ_i, the agent loop starts from a compacted history h_i and a triggering event e_i, such as a user utterance or a salient game event. The trajectory is $\sigma_i = (m_{i,1}, f_{i,1}, \ldots, f_{i,L_i-1}, m_{i,L_i})$, where $m_{i,\ell}$ is one model response (a turn), which may contain tool calls, and $f_{i,\ell}$ is the intervening tool results or feedback. The final response calls the completion tool, returning control to the runtime and carrying unfinished work into the next invocation. A trajectory can therefore end before the underlying gameplay task is complete.

Play-window UX reports. A play window is the interval evaluated by one UX report and may span multiple gameplay sessions. After each play window, users submit a report describing their

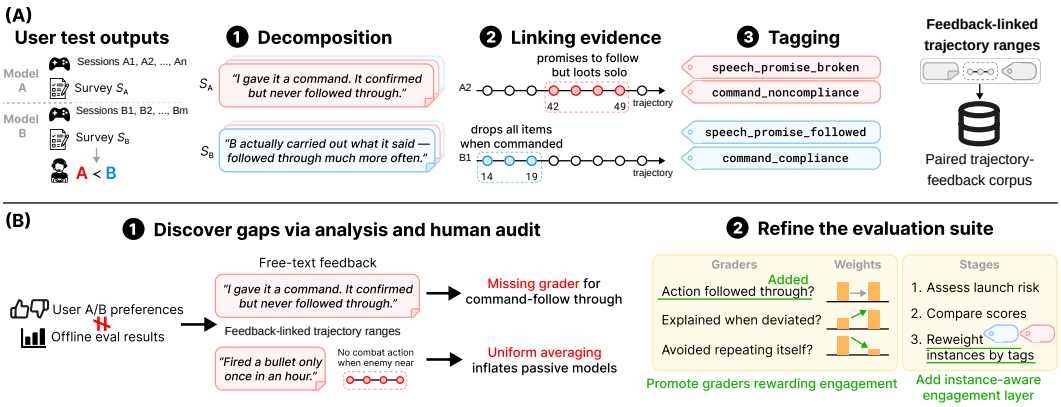


Figure 2: **From UX reports to evaluation-suite refinement.** **(A)** We decompose UX reports into feedback fragments, link identifiable fragments to relevant gameplay trajectories, and annotate the resulting feedback-linked trajectory ranges with behavioral descriptions and tags. **(B)** We use these localized spans together with aggregate user-test signals to diagnose missing, misweighted, or underspecified suite criteria, then refine the suite by adding or revising graders, changing aggregation weights, or introducing new evaluation stages.

experience with the agent (Appendix D.2). A report contains global ratings, behavior-level ratings, free-text feedback, and, in A/B rounds, pairwise preferences between agent versions. Global ratings capture overall impressions such as fun, helpfulness, and friendliness, while behavior-level ratings assess specific conversational and gameplay capabilities, such as intent understanding, information accuracy, command following, and situational judgment. Free-text feedback describes satisfying or frustrating moments in the user's own words. We assess the consistency and stability of these signals in Appendix D.3.

Constructing the paired trajectory–feedback corpus. We construct a *paired trajectory–feedback corpus* by linking free-text feedback fragments to trajectory ranges when the referenced behavior can be identified. Ratings and pairwise preferences remain play-window-level UX signals, while the paired trajectory–feedback corpus provides localized evidence about specific agent behaviors.

We construct this corpus with a three-stage LLM-assisted pipeline. First, we decompose each free-text report into atomic feedback fragments, each expressing one distinct observation about the agent. Second, when a fragment refers to identifiable behavior rather than a general impression, we localize it to the relevant trajectory range $[\sigma_i, \ldots, \sigma_k]$, producing a *feedback-linked trajectory range*. Third, we annotate each linked span with the fragment, matched trajectory evidence, a structured behavior description, and behavior tags that summarize the observed positive or negative interaction pattern. Figure 2 (A) illustrates the pipeline. Appendix E details the procedure, and Appendix H.5 provides prompt examples.

Because each span preserves the agent's input context, tool calls, and tool results, the evaluation suite can inspect the same behavioral evidence that the user's feedback refers to.

3.3 Evaluation suite refinement with evidence from user tests

After each user test, we use the resulting evidence to revise the evaluation suite. This evidence includes gameplay trajectories, UX reports, pairwise preferences from A/B rounds, and the paired trajectory–feedback corpus described in Section 3.2. The goal is to make offline judgments better reflect user-salient companion UX observed during live interaction. Figure 2 (B) shows how these localized evidence spans, together with aggregate user-test signals, drive suite-level updates.

Refinement signals. User-test evidence can reveal shortcomings of the current suite at several levels. Aggregate signals, such as low ratings, preference reversals, or frequent feedback categories, indicate user-salient behaviors that the suite may be missing or underweighting. Feedback-linked trajectory ranges help diagnose concrete mismatches between suite judgments and behaviors users reported. For example, recurring complaints about unsafe revives, poor command follow-through,

mistimed speech, or inconsistent speech–action behavior may motivate new criteria, revised rubrics, additional graders, or changes in severity weights.

Suite update. A refinement step may add or revise behavioral criteria, update grader rubrics, adjust aggregation or severity weights, cap overrepresented grader families, or introduce a new evaluation stage. As a data-driven refinement, we estimate aggregation weights from user-test evidence using a separate calibration set. Across refinement iterations, the candidate models, agent harness, and fixed test set remain inputs to the comparison process. The evaluation suite is the component that is revised. The resulting suite is evaluated again with new user-test evidence, and the refinement process repeats.

4 Experiments

In this section, we first describe the common experimental setup (Section 4.1) and analyze the live user-preference evidence underlying our reference for companion UX (Section 4.2). We then conduct an ablation study of the evaluation suite (Section 4.3). Finally, we examine recommendation ratings from the deployment of the model selected using the refined evaluation suite (Section 4.4).

4.1 Experimental setup

Test set. We use a fixed test set of 432 rerunnable session contexts (2,793 trajectories) derived from the paired trajectory–feedback corpus in Section 3.2. Each context is constructed from a feedback-linked trajectory range and retains the survey feedback associated with that span, drawn from both negative and positive user feedback.

Alignment metric. We construct a user-reference ranking $\pi^\star$ over seven candidate models (A through G, described in Appendix C) using pairwise A/B preferences collected during live gameplay sessions. Details of the preference aggregation process are provided in Appendix D.4. Briefly, the aggregated preferences induce the following reference order:

$$\pi^\star : \quad G \succeq F \succ E \succeq D \succ C \succeq B \succeq A.$$

The strict symbols indicate stronger preference evidence, while weak orderings indicate majority directions with weaker statistical support. We measure how closely this model ranking matches the user-reference order $\pi^\star$ using Kendall's rank correlation coefficient [Kendall, 1938] over the $\binom{7}{2}$ model pairs.

Formally, Kendall's τ is defined as:

$$\tau = \frac{N_{\text{agree}} - N_{\text{disagree}}}{\binom{N}{2}},$$

where N_{agree} and N_{disagree} denote the numbers of model pairs whose relative ordering is consistent or inconsistent, respectively, between the model ranking and the user-reference ranking $\pi^\star$. A higher τ indicates that the model ranking more accurately recovers the ordering preferred by users.

4.2 Statistical power and uncertainty in user-preference evidence

We analyze the statistical power and uncertainty of the live A/B preference evidence. Table 1 reports results from six A/B user-test rounds involving Models A–G. The four rounds with narrow majority margins (R1, R2, R4, and R6) have observed majority proportions of 53.4–54.3%, well below their MDEs at 80% power. R5 exceeds its MDE and remains significant after Holm correction [Holm, 1979]. R3 is nominally significant before correction but does not survive Holm correction, and its majority proportion is also below its MDE. These round-specific MDEs characterize the sensitivity of individual pairwise comparisons rather than power for Kendall's τ.

The individual comparisons therefore provide limited evidence for a precise ordering of all seven models. We quantify this uncertainty using respondent-cluster bootstrap resampling. Across 4,000 bootstrap replicates, exact ranks vary, but the top $\{G, F\}$, middle $\{E, D\}$, and bottom $\{C, B, A\}$ tier partition is retained in 95.0% of Bradley–Terry replicates [Bradley and Terry, 1952] and 94.5% of tie-aware Davidson replicates [Davidson, 1970]. Although the exact ordering within tiers remains uncertain, both bootstrap analyses support the same three-tier preference structure.

Table 1: **Results from the A/B user-test rounds.** *Majority* denotes the model receiving more non-tie preferences. n_{dec} and n_{total} denote the numbers of non-tie and total responses, respectively. ***Majority proportion*** is the proportion of non-tie responses favoring the majority model. p is the unadjusted two-sided exact binomial p-value. $\mathrm{MDE}_{.80}$ is the minimum majority proportion yielding 80% power for the same test at $\alpha = .05$, conditional on the observed n_{dec}.

Round	Pair	Majority	n_{dec}	n_{total}	Majority proportion (%)	p	$\mathrm{MDE}_{.80}$ (%)
R1	A–B	B	118	158	54.2	.407	63.5
R2	B–C	C	127	160	54.3	.375	63.1
R3	C–D	D	109	156	60.6	.035	64.0
R4	D–E	E	103	136	53.4	.555	63.7
R5	E–F	F	68	84	67.6	.005	67.3
R6	F–G	G	81	107	54.3	.505	65.6

4.3 Ablation study of the evaluation suite

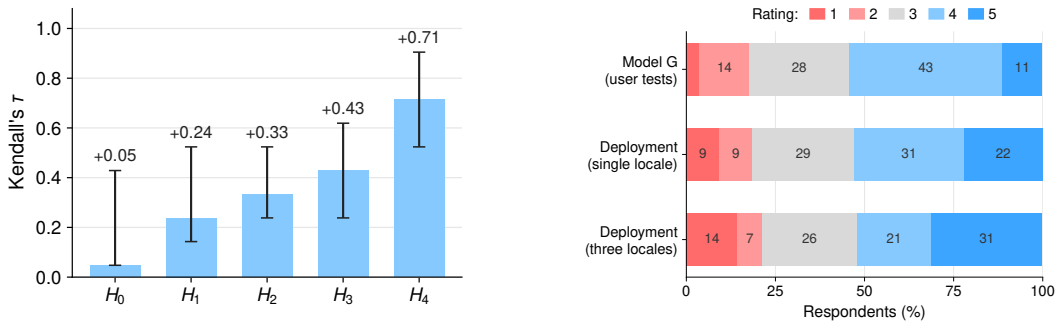
We developed the evaluation suite through iterative alignment using UX reports (Section 3.3). Here, we assess its alignment with user preferences through an ablation study. We denote this suite by H_4 and construct H_0–H_3 by progressively removing its components. All five variants use the same seven models and fixed test set. Note that the ablated variants do not correspond to the evaluation suites used in the user-test rounds.

As defined in Section 3.1, evaluation stages combine trajectory-derived evidence, graders, and aggregation logic. H_0 contains only rule-based validity checks, while H_1 also includes model-based behavioral graders, with uniform aggregation across graders. H_2 groups the same graders into behavioral themes, averages scores within each theme, and gives each theme equal weight. H_3 uses preference-calibrated grader weights within each theme, estimated by agreement with user preferences on a calibration set whose participants are disjoint from those in the fixed test set. We construct a behavior taxonomy from free-text feedback in UX reports and assign severity weights based on the relative frequency of reported problems. H_4 uses these weights to reward positive behaviors and penalize failures, adding the normalized score to the H_3 score. Algorithmic details are provided in Appendix G.2.

Figure 3(a) shows higher observed alignment as more evaluation components are included. Rule-based graders in H_0 check validity but do not capture qualities such as instruction following and active participation, which the behavioral graders in H_1 assess. Uniform aggregation overemphasized utterance repetition because it was covered by multiple correlated graders. Theme-level aggregation in H_2 addresses this imbalance. H_4 also preserves the top, middle, and bottom groups identified in Section 4.2, with disagreements confined to within-group ordering. Although several intermediate bootstrap intervals overlap, H_4 achieves higher alignment than H_0, with a paired difference of $+0.67$ in Kendall's τ (95% CI $[+0.29, +0.67]$).

4.4 Evaluation in a held-out deployment

After a month-long user test, we deployed a model selected using the refined evaluation suite and examined its companion UX without further suite refinement. The deployment involved players across multiple locales, extending beyond the single-locale user tests. Because deployment users evaluated a single model rather than an A/B pair, we used the five-point recommendation item shared with the user tests. When recommendation scores differed between models in the A/B user tests, the higher-rated model matched the explicit pairwise preference in 78.7% of comparisons. Figure 3(b) compares deployment ratings with those of Model G. Differences in user population and survey context limit direct comparison between the two settings. In the user-test locale, the deployment model's mean recommendation score was 3.47 versus 3.44 for Model G (difference $+0.03$, 95% CI $[-0.17, +0.23]$). The mean across all deployment locales was 3.48. These deployment results provide complementary support for model selection using the evaluation suite iteratively refined with evidence from user tests.



(a) Ablation study result

(b) Recommendation distributions

Figure 3: **Evaluation-suite ablation study and held-out deployment evaluation. (a)** Bars show observed Kendall's τ for H_0–H_4. Whiskers show 95% respondent-cluster bootstrap intervals from 10,000 refits of the user-reference ranking, with suite-induced model rankings held fixed. **(b)** Five-point recommendation-rating distributions for Model G and the suite-selected deployment model. Deployment results are shown for the user-test locale and for all deployment locales combined.

5 Analysis

Section 4 showed that the refined evaluation suite better aligns with companion UX as reflected in user preferences. We examine the user-test evidence that motivated these refinements. Across A/B user-test rounds, free-text feedback and linked trace spans revealed mismatches between offline evaluation and companion UX, helping us diagnose missing behavioral coverage, unbalanced aggregation, and deployment-critical failures. We describe how each diagnosis informed suite updates.

5.1 Missing behavioral coverage: active participation

A coverage mismatch revealed a missing dimension in the grader set. In early model-selection rounds, the offline suite favored models that appeared more valid under rule-based checks, but users preferred agents that participated more actively. Free-text feedback and linked trajectories localized the mismatch to participation behavior. Users complained about passive companions in urgent situations, such as agents that "mostly just talked" or "didn't shoot even with enemies in front." Preferred agents were described as more willing to act, respond to requests, and continue participating without repeated reminders. The missing dimension was active participation: users preferred agents that attempted to help, even imperfectly, over those that avoided failure by remaining passive.

The initial grader set focused on detecting errors in attempted behavior. It checked whether tool calls were valid, feasible, non-redundant, and protocol-consistent, but gave little credit for contributing. As a result, passive models could receive fewer penalties by producing fewer actions, while engaged models exposed more detectable mistakes. This motivated graders for behavioral follow-through and active participation, including whether the agent acts on user requests, remains engaged during ongoing situations, and shows sufficient behavioral variety.

5.2 Unbalanced aggregation of behavioral signals

An aggregation mismatch showed that adding behavioral graders was not sufficient by itself. In the C vs. D comparison, live users significantly preferred D, with full preference statistics in Table 1. However, the evaluation suite available then surfaced mixed evidence for both candidates, suggesting that the relevant behaviors were being measured but not combined in a user-aligned way. We diagnosed the mismatch by comparing the models' feedback profiles. Users who preferred D emphasized dialogue quality and understanding, noting that the conversation felt natural or that "the conversation worked." By contrast, users who preferred C tended to cite action engagement, including combat sense, more active participation, and better situational briefing. Thus, the comparison involved a trade-off between different companion qualities rather than a single shared failure mode.

Uniform aggregation made this trade-off poorly aligned with user preference. Action engagement was represented by many correlated grader signals while dialogue quality and understanding were

represented by fewer, so C's larger volume of action signals could offset D's dialogue advantage even when users preferred D overall. This mismatch motivated aggregation refinements that group graders into behavioral themes and calibrate within-theme grader weights using evidence from user tests, so that high-volume correlated signals do not dominate the final model ranking.

5.3 Separating user preference from deployability

A deployability mismatch showed that user preference and release-critical validity must be evaluated separately. In the D vs. E comparison, users preferred E over D, citing stronger engagement and proactive support, with preference statistics in Table 1. However, user feedback revealed a deployment-critical generation defect in E. Users reported repeated malformed outputs, including error-code strings and alphabet-pattern sequences, describing cases where the agent "responded with error codes over 20 times" or "produced strange alphabet patterns." This evidence shows that E combined user-valued behaviors with failures that should block deployment, rather than being simply a stronger or weaker companion model. A single score would obscure this distinction: increasing the weight of user-preferred engagement could select E, while increasing the penalty for malformed outputs could discard behaviors that users valued. The suite therefore needed a separate stage for deployment-critical validity rather than treating deployability as another behavioral dimension.

This mismatch motivated adding a deployability gate before model selection. The gate checks failures that should preclude deployment regardless of its user-aligned quality score, including repeated tool calls, invalid output formats, and malformed generation patterns. The per-(model, grader) defect-rate matrix that exposed this axis structure is shown in Appendix G.3. Only candidates that pass this gate are considered in the model-selection stage. The following round, in which model F addressed the generation defect and was preferred over E, showed that the user-test diagnosis was actionable.

6 Discussion

Evaluation as profiling, not only selection. Beyond a single aggregate score, the suite decomposes behavior into criteria, behavioral themes, and stage-level outputs to characterize a model's companion style. A single preference metric would collapse these style differences, even though companion UX is partly preference-dependent: some users may value initiative, while others may prefer an agent that waits for explicit commands and interrupts less. These profiles can support offline model selection and preference-aware deployment or personalization.

Companion UX is not captured by capability alone. Although larger LLMs may offer stronger language and reasoning, users preferred the smaller local model in the R0 preference round reported in Appendix D.4, highlighting latency as an important dimension of companion UX. Delayed responses or actions can make the agent feel less responsive, cooperative, and present in the shared world. This finding illustrates why iterative alignment is necessary: user tests can reveal user-salient criteria, such as latency, that model capability alone does not capture.

7 Conclusion

In this paper, we introduced an iterative alignment process for assessing embodied companion agents through companion UX. Companion UX cannot be fully captured by fixed criteria specified in advance, because users interpret an agent's dialogue, actions, timing, and social behavior jointly within a shared world. Over a month-long user test with 1k users and 38k play sessions, we iteratively refined the evaluation suite using UX reports and linked gameplay trajectories. An ablation study showed better alignment with user preferences, and a held-out deployment gave complementary support for model selection using the refined suite.

Our study is limited to one game and one agent archetype. We do not claim the graders, behavior taxonomy, or weights transfer unchanged to other domains. Nevertheless, paired trajectory–feedback evidence provides a practical methodology for refining evaluation when agent quality depends on coupled utterance–action behavior in dynamic user-facing environments. It also shows alignment is not a one-time outcome. User feedback reshapes the evaluation suite, the refined suite drives better models, and those models surface new companion-UX signals, so evaluation and user experience co-evolve. Alignment must therefore continue as models and user expectations advance together.

References

Anthropic. Introducing Claude Haiku 4.5. https://www.anthropic.com/news/claude-haiku-4-5, October 2025. Accessed: 2026-05-07.

Nolan Bard, Jakob N. Foerster, Sarath Chandar, Neil Burch, Marc Lanctot, H. Francis Song, Emilio Parisotto, Vincent Dumoulin, Subhodeep Moitra, Edward Hughes, Iain Dunning, Shibl Mourad, Hugo Larochelle, Marc G. Bellemare, and Michael Bowling. The Hanabi challenge: A new frontier for AI research. *Artificial Intelligence*, 280:103216, 2020. doi: 10.1016/j.artint.2019.103216. URL https://arxiv.org/abs/1902.00506.

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

Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952. doi: 10.1093/biomet/39.3-4.324.

Micah Carroll, Rohin Shah, Mark K. Ho, Thomas L. Griffiths, Sanjit A. Seshia, Pieter Abbeel, and Anca Dragan. On the utility of learning about humans for human-AI coordination. In *Advances in Neural Information Processing Systems*, 2019. URL https://arxiv.org/abs/1910.05789.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Banghua Zhu, Hao Zhang, Michael Jordan, Joseph E. Gonzalez, and Ion Stoica. Chatbot arena: An open platform for evaluating LLMs by human preference. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 8359–8388. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/chiang24b.html.

Lee J. Cronbach. Coefficient alpha and the internal structure of tests. *Psychometrika*, 16(3):297–334, 1951. doi: 10.1007/BF02310555.

Roger R. Davidson. On extending the Bradley–Terry model to accommodate ties in paired comparison experiments. *Journal of the American Statistical Association*, 65(329):317–328, 1970. doi: 10.1080/01621459.1970.10481082.

Jan Deriu, Alvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. Survey on evaluation methods for dialogue systems. *Artificial Intelligence Review*, 54(1):755–810, 2021. doi: 10.1007/s10462-020-09866-x.

Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. MineDojo: Building open-ended embodied agents with internet-scale knowledge. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2022. URL https://arxiv.org/abs/2206.08853. arXiv:2206.08853; Outstanding Paper Award.

Yiming Gao, Feiyu Liu, Liang Wang, Zhenjie Lian, Dehua Zheng, Weixuan Wang, Wenjin Yang, Siqin Li, Xianliang Wang, Wenhui Chen, Jing Dai, Qiang Fu, Wei Yang, Lanxiao Huang, and Wei Liu. Enhancing human experience in human-agent collaboration: A human-centered modeling approach based on positive human gain. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2401.16444.

Asma Ghandeharioun, Judy Hanwen Shen, Natasha Jaques, Craig Ferguson, Noah Jones, Agata Lapedriza, and Rosalind Picard. Approximating interactive human evaluation with self-play for open-domain dialog systems. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. URL https://arxiv.org/abs/1906.09308.

Google. Gemini 3.7 Flash. https://ai.google.dev/gemini-api/docs/models/gemini-3.7-flash, 2026a. Model code: gemini-3.7-flash. Accessed: 2026-10-30.

Google. Gemini 3 Flash Preview. https://ai.google.dev/gemini-api/docs/models/gemini-3-flash-preview, 2026b. Model code: gemini-3-flash-preview. Accessed: 2026-05-07.

Helia Hashemi, Jason Eisner, Corby Rosset, Benjamin Van Durme, and Chris Kedzie. LLM-Rubric: A multidimensional, calibrated approach to automated evaluation of natural language texts. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL 2024), Volume 1: Long Papers*, pages 13806–13834, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.745. URL https://aclanthology.org/2024.acl-long.745/.

Wei He, Yueqing Sun, Hongyan Hao, Xueyuan Hao, Zhikang Xia, Qi Gu, Chengcheng Han, Dengchang Zhao, Hui Su, Kefeng Zhang, Man Gao, Xi Su, Xiaodong Cai, Xunliang Cai, Yu Yang, and Yunke Zhao. VitaBench: Benchmarking LLM agents with versatile interactive tasks in real-world applications. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://arxiv.org/abs/2509.26490. arXiv:2509.26490.

Philip Hingston. A Turing test for computer game bots. *IEEE Transactions on Computational Intelligence and AI in Games*, 1(3):169–186, 2009. doi: 10.1109/TCIAIG.2009.2032534. URL https://ieeexplore.ieee.org/document/5247069.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979. doi: 10.2307/4615733.

Tom Hosking, Phil Blunsom, and Max Bartolo. Human feedback is not gold standard. In *International Conference on Learning Representations*, 2024.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R. Narasimhan. SWE-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=VTF8yNQM66.

Seungyeon Jwa, Daechul Ahn, Reokyoung Kim, Dongyeop Kang, and Jonghyun Choi. Becoming experienced judges: Selective test-time learning for evaluators. In *Findings of the Association for Computational Linguistics: EACL 2026 (Short Papers)*, 2026. URL https://arxiv.org/abs/2512.06751. arXiv:2512.06751.

Daniel Kahneman, Barbara L. Fredrickson, Charles A. Schreiber, and Donald A. Redelmeier. When more pain is preferred to less: Adding a better end. *Psychological Science*, 4(6):401–405, 1993.

M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938. doi: 10.1093/biomet/30.1-2.81.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, et al. Dynabench: Rethinking benchmarking in NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4110–4124, 2021.

Hannah R. Kirk, Alexander Whitefield, Paul R"ottger, Andrew Bean, Katerina Margatina, Juan Ciro, Rafael Mosquera, Max Bartolo, Adina Williams, He He, et al. The PRISM alignment dataset: What participatory, representative and individualised human feedback reveals about the subjective and multicultural alignment of large language models. *Advances in Neural Information Processing Systems*, 37:105236–105344, 2024.

Terry K. Koo and Mae Y. Li. A guideline of selecting and reporting intraclass correlation coefficients for reliability research. *Journal of Chiropractic Medicine*, 15(2):155–163, 2016. doi: 10.1016/j.jcm.2016.02.012.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 34209–34231. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/li25h.html.

Ying-Chun Lin, Jennifer Neville, Jack Stokes, Longqi Yang, Tara Safavi, Mengting Wan, Scott Counts, Siddharth Suri, Reid Andersen, Xiaofeng Xu, et al. Interpretable user satisfaction estimation for conversational systems with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11100–11115, 2024.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. AgentBench: Evaluating LLMs as agents. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://arxiv.org/abs/2308.03688. arXiv:2308.03688.

Jiarui Lu, Thomas Holleis, Yizhe Zhang, Bernhard Aumayer, Feng Nan, Felix Bai, Shuang Ma, Shen Ma, Mengyu Li, Guoli Yin, Zirui Wang, and Ruoming Pang. ToolSandbox: A stateful, conversational, interactive evaluation benchmark for LLM tool use capabilities. In *Findings of the Association for Computational Linguistics: NAACL 2025*, 2025. URL https://arxiv.org/abs/2408.04682. arXiv:2408.04682.

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. AgentBoard: An analytical evaluation board of multi-turn LLM agents. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2024. URL https://arxiv.org/abs/2401.13178. arXiv:2401.13178; Oral presentation.

Roderick P. McDonald. *Test Theory: A Unified Treatment*. Lawrence Erlbaum Associates, Mahwah, NJ, 1999.

Mike A. Merrill, Alexander G. Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Yeon Shin, Thomas Walshe, E. Kelly Buchanan, et al. Terminal-bench: Benchmarking agents on hard, realistic tasks in command line interfaces. *arXiv preprint arXiv:2601.11868*, 2026. URL https://arxiv.org/abs/2601.11868. ICLR 2026 poster.

PUBG Ally Team. PUBG Ally: A conversational embodied agent as an AI teammate. *arXiv preprint arXiv:2609.29837*, 2026. doi: 10.48550/arXiv.2609.29837. URL https://arxiv.org/abs/2609.29837.

Anka Reuel, Amelia Hardy, Chandler Smith, Max Lamparth, Malcolm Hardy, and Mykel J. Kochenderfer. Betterbench: Assessing AI benchmarks, uncovering issues, and establishing best practices. *Advances in Neural Information Processing Systems*, 37:21763–21813, 2024.

Alexander Schmitt and Stefan Ultes. Interaction quality: Assessing the quality of ongoing spoken dialog interaction by experts—and how it relates to user satisfaction. *Speech Communication*, 74: 12–36, 2015. doi: 10.1016/j.specom.2015.06.003.

Patrick E. Shrout and Joseph L. Fleiss. Intraclass correlations: Uses in assessing rater reliability. *Psychological Bulletin*, 86(2):420–428, 1979. doi: 10.1037/0033-2909.86.2.420.

Xiaojuan Tang, Jiaqi Li, Yitao Liang, Song-Chun Zhu, Muhan Zhang, and Zilong Zheng. Mars: Situated inductive reasoning in an open-world environment. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024) Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=3qoQ6AolAz.

Mohsen Tavakol and Reg Dennick. Making sense of cronbach's alpha. *International Journal of Medical Education*, 2:53–55, 2011. doi: 10.5116/ijme.4dfb.8dfd.

Roger Tourangeau, Lance J. Rips, and Kenneth Rasinski. *The Psychology of Survey Response*. Cambridge University Press, 2000.

Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff,

Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019. doi: 10.1038/s41586-019-1724-z. URL https://www.nature.com/articles/s41586-019-1724-z.

Marilyn A. Walker, Diane J. Litman, Candace A. Kamm, and Alicia Abella. PARADISE: A framework for evaluating spoken dialogue agents. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics (ACL/EACL)*, pages 271–280, Madrid, Spain, July 1997. Association for Computational Linguistics. doi: 10.3115/976909.979652. URL https://aclanthology.org/P97-1035/.

Wenya Wei, Sipeng Yang, Qixian Zhou, Ruochen Liu, Xuelei Zhang, Yifu Yuan, Yan Jiang, Yongle Luo, Hailong Wang, Tianzhou Wang, Peipei Jin, Wangtong Liu, Zhou Zhao, Xiaogang Jin, and Elvis S. Liu. F.A.C.U.L.: Language-based interaction with AI companions in gaming. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, 2026. URL https://arxiv.org/abs/2511.13112.

E. J. Williams. Experimental designs balanced for the estimation of residual effects of treatments. *Australian Journal of Scientific Research, Series A: Physical Sciences*, 2(2):149–168, 1949.

Edwin B. Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927. doi: 10.1080/01621459.1927.10502953.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2024. URL https://arxiv.org/abs/2404.07972. arXiv:2404.07972.

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

Bowen Ye, Rang Li, Qibin Yang, Yuanxin Liu, Linli Yao, Hanglong Lv, Zhihui Xie, Chenxin An, Lei Li, Lingpeng Kong, Qi Liu, Zhifang Sui, and Tong Yang. Claw-Eval: Toward trustworthy evaluation of autonomous agents, 2026. URL https://arxiv.org/abs/2604.06132.

Yuxuan Zhang, Yubo Wang, Yipeng Zhu, Penghui Du, Junwen Miao, Xuan Lu, Wendong Xu, Yunzhuo Hao, Songcheng Cai, Xiaochen Wang, Huaisong Zhang, Xian Wu, Yi Lu, Minyi Lei, Kai Zou, Huifeng Yin, Ping Nie, Liang Chen, Dongfu Jiang, Wenhu Chen, and Kelsey R. Allen. ClawBench: Can AI agents complete everyday online tasks? *arXiv preprint arXiv:2604.08523*, 2026. URL https://arxiv.org/abs/2604.08523.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2023. URL https://arxiv.org/abs/2306.05685.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://arxiv.org/abs/2307.13854. arXiv:2307.13854.

A Limitations

Generalizability. Our evaluation-suite development and A/B comparisons focused on one game and one user-test locale, using a voice-driven teammate with a fixed tool set. The construct we call "companion UX" is plausibly different in a cooperative RPG, a MOBA, or a single-player title where the agent is a coach rather than a co-player. The recurring mismatches between automated evaluation and companion UX suggest that iterative alignment will also be needed in other settings, although its effectiveness across domains remains to be validated. Additionally, our graders audit the agent's high-level tool calls and utterances; the agent consumes a textified game state rather than rendered pixels and acts through a semantic action library rather than low-level controller input, so these graders do not directly assess visual perception or low-level control, although their consequences may appear in the recorded behavior.

Statistical power. Statistical significance varied across the A/B comparisons. R3 and R5 showed significant preference differences before correction for multiple comparisons, while only R5 remained significant after Holm correction. Comparisons with narrow preference margins may require larger participant samples to distinguish between models.

Survey as ground truth. The survey checks in Appendix D.3 make aspects of the instrument auditable, but survey responses and automated judgments can be affected by different sources of error. Free-text responses may be affected by social desirability, and recollections may emphasize salient or recent experiences within a play window; the checks reported here do not establish that these biases are absent. Our response has been to *characterize* the survey rather than to trust it uncritically, and to treat disagreements between survey and grader as evidence that one or both are mismeasuring, not automatically as evidence that the survey is right.

Data release. The full underlying paired trajectory–feedback corpus cannot be released under the game IP and participant-consent terms this study operated under, which limits reproducibility at the raw-data level. Appendices F, G.1, and E describe the behavioral criteria and rubrics, evaluation settings, and behavior taxonomy tagging procedure, respectively. These descriptions support implementation with other data, but do not enable independent reproduction of the reported results on the same corpus.

B Broader impact

Intended positive impact. The user-facing benefit of this methodology is the potential to make evaluation and release decisions more responsive to behaviors users actually notice. Linking user feedback to gameplay trajectories can help identify consequential behaviors that existing evaluation criteria overlook or underweight. For the research community, the study provides a worked example of using evidence from user interaction to revise evaluation criteria and model-selection procedures.

Privacy and consent governance for adopters. A practitioner applying the iterative alignment process should consider participant consent for recording voice and gameplay trajectories, replaying those trajectories across candidate models, and linking survey responses to the corresponding sessions. Replay deserves explicit attention because it extends the use of existing records to additional models without collecting new player behavior. Consent and data-handling arrangements should therefore account for these downstream uses, including retention, access controls, and de-identification of voice and free-text records.

Risks from a misaligned voice companion at deployment time. A voice-driven companion can affect communication and coordination among players in a cooperative multiplayer setting. Overactive conversational behavior could crowd out human teammate communication, while reliance on inaccurate agent-issued situational summaries could affect players' decisions. Passing protocol checks or receiving favorable user feedback does not by itself establish that these potential risks have been addressed. Changes to the model, tool interface, or interaction modality may introduce new behaviors, so evaluation coverage should be reviewed as the deployed agent changes.

Risks of the methodology when transferred to higher-stakes settings. Our study takes place in a leisure setting, and validation in this setting does not establish suitability for applications where evaluation errors may have more serious consequences. In such applications, feedback measures and behavioral criteria would require domain-specific validation, including examination of whether reporting biases distort the user feedback used to guide evaluation. The iterative alignment process should help identify and investigate disagreements between user feedback and grader outputs; agreement with survey responses alone should not be treated as sufficient evidence of evaluation validity or become the sole optimization target.

C Agent architecture and candidate models

This appendix describes the companion-agent architecture and runtime relevant to our evaluation, drawing on the system described in our technical report [PUBG Ally Team, 2026]. It also summarizes the tool interface and candidate models used in this study.

Runtime architecture. The companion agent combines language-model reasoning with a behavior-tree execution layer. The language model uses game observations and dialogue context to decide what to observe, what to say, and which high-level actions to request. The behavior tree executes these requests through movement, combat, and recovery behaviors, updating at game-tick rate while language-model inference proceeds. Speech recognition converts player utterances into text, and speech synthesis renders the agent's responses as voice output. This separation allows the game and ongoing actions to continue while the language model reasons.

Event-driven agent loops. The language-model agent is activated by selected events, including player speech, game-state changes, action outcomes, and scheduled runtime checks. At the beginning of an agent loop, it receives recent events, context carried over from the preceding agent loop, and the agent's current speech and action status. Subsequent turns incorporate tool results and newly arrived events, allowing the agent to revise its decisions as the situation changes. An agent loop ends with a completion call that carries unfinished work forward. A session is one complete gameplay match and contains multiple trajectories. Each model response is a turn; the sequence of responses and intervening tool results or feedback within an agent loop forms a trajectory, as defined in Section 3.2.

Tool interface. The candidate model interacts with the runtime through a function-calling interface. Observation tools provide focused views of game state, communication tools produce speech, and action tools request high-level behaviors. The interface also supports game-knowledge retrieval, player memory, and agent loop or safety control. Table 2 lists the interface documented for the workshop evaluation setting. The tool calls and their results expose the agent's observations, utterances, and requested actions to the graders. Offline replay settings are described in Appendix G.1.

Candidate-model interface and inference configuration. The agent harness supplies the model with the system prompt, tool definitions, recent interaction context, current events, and tool results. The model returns responses containing utterances and tool calls, which the runtime dispatches through the agent interface. The study compares on-device candidate models and includes a cloud-hosted baseline accessed through an API. The distinction matters for live user experience because inference delays affect when the agent can respond during play. Offline replay and grading settings are reported in Appendix G.1, and the live A/B comparisons are described in Appendix D.4.

Candidate model variants. Table 3 summarizes the candidate identifiers used in the main text. Models A–G are the on-device candidates compared in this study. Their training configurations vary in the use of teacher outputs or probability targets, emphasis on gameplay actions, safety-oriented data, and relabeled or corrected examples. The table distinguishes these training configurations from the observed behavior and user preferences reported in the results. The cloud baseline is included separately for the cloud-versus-on-device comparison.

Table 2: The 17 tools exposed through the agent's tool interface. `obs` tools are read-only over runtime state; `comm` tools emit speech; `act` tools mutate runtime state; `ctrl` tools control the agent loop. Several of these names also appear in grader specifications (Appendix F).

Tool	Class	Purpose
`get_game_overview`	obs	current game time, HP, boost, phase, zone status, kill feed
`get_my_status`	obs	own state: HP, position, facing, pose, lootables, destination
`get_my_equipment`	obs	weapons, ammo, attachments, bag space, consumables
`get_combat_info`	obs	visible enemies, attacker location, target, teammate awareness
`get_teammate_status`	obs	teammate position, HP, weapon, behavior, proximity
`get_non_voice_communication`	obs	quick-menu messages, pings, world markers
`get_conversation_history`	obs	recent user and agent utterances within the agent loop
`get_destination_candidates`	obs	filtered candidates by group (enemy, marker, gunshot, safe zone, cover, landmark, highground, goal)
`check_action_availability`	obs	probe whether a named action is currently available and read its required parameters
`lookup_game_knowledge`	obs	static game knowledge: weapons, items, attachments, controls, rules
`speak`	comm	emit a voice utterance with an urgency tag (`info`, `reply`, `urgent`)
`filler`	comm	emit a non-action acknowledgement
`execute_action`	act	dispatch a named in-game action with parameters
`update_memory`	act	write to teammate-profile memory (name, preferences, key events)
`set_quiet_mode`	act	toggle voice output on or off
`flag_unsafe`	ctrl	flag the agent loop as unsafe
`done`	ctrl	terminate the agent loop with a ≤ 250-character summary inherited as the next agent loop's prologue

Table 3: Candidate model identifiers and training differences.

Model	Description
A	Teacher outputs with the baseline data mixture
B	Supervised examples with increased gameplay-action emphasis
C	Distillation from teacher probability signals
D	Teacher-output distillation with the standard objective
E	Model D recipe with additional safety data
F	Teacher outputs combined with relabeled supervised examples
G	Model D recipe with safety data and duplicate-call corrections
Cloud baseline	Cloud-hosted model accessed through an API

D User study and survey

D.1 Participants and procedure

Participants and compensation. The overall user test spanned approximately one month, with A/B comparisons among on-device candidate models conducted over approximately one week. Participants were recruited online, were all eighteen years or older, and were compensated for their participation.

Consent and IRB. All analyzed participants were adults and provided documented digital click-through consent covering the research use of voice recordings, input logs, gameplay trajectories, and survey responses. The study involved compensated ordinary gameplay without invasive or sensitive intervention. It originated as a corporate product-evaluation user test; formal prospective academic institutional review board (IRB) review was not obtained. Recruitment, consent, compensation, and data handling were reviewed through documented Legal, Privacy, and HR processes, with external legal counsel consulted. Analyses used access-controlled, de-identified records, and no raw participant data are released.

The survey was administered immediately after each play window. In A/B comparisons, participants rate their experience with each agent after playing with it. After playing with both agents, they also answer a direct pairwise preference question, selecting the first agent, the second agent, or indicating that they cannot choose between them.

D.2 Survey instrument

Sections 1–6 of the form collect demographics, prior gameplay experience, and consent confirmation; we omit them here as they are standard pre-play items not used in the analysis. Sections 7–11, reproduced in Figures 4–6, contain the rating and free-text items used throughout the paper. The screenshots are taken from the live form after the agent's product name was replaced with the neutral phrase "embodied companion agent" for release; participants saw the original product name during the study.

Item structure. The reproduced pages contain thirteen numbered questions (Q13–Q25), including multi-item rating blocks. The first page (Figure 4) covers overall play experience and character impressions: two categorical items on the role and perceived relationship of the agent (Q13, Q14), a five-item global Likert scale on overall experience (Q15: *fun*, *meaningful*, *helpful*, *natural-and-friendly*, *play_again*), a three-dimensional voice/personality/expression scale (Q16), and a four-item character-impression scale (Q17). The headline three-item set (*fun*, *helpful*, *friendly*) used in Appendix D.3 draws from Q15, with the four- and five-item supersets adding *play_again* and *meaningful* respectively. Here, *friendly* denotes the Q15 item about conversations feeling natural and friendly. The second page (Figure 5) covers in-game behavior and conversational quality: a response-speed item (Q18), a three-dimensional gameplay-quality scale (Q19: command execution, situational judgment, game skill), an optional free-response prompt for any *poor* rating (Q20), a five-dimensional conversation-quality scale (Q21: understanding, information accuracy, naturalness, proactive communication, safety), and a second free-response prompt (Q22). The third page (Figure 6) contains the recommendation item (Q23, five-point likelihood) and two open-ended prompts (Q24, Q25). The category clusters reported in Appendix D.3 (e.g., *character voice*, *conversation quality*, *gameplay quality*) follow this grouping; cluster sample sizes vary because earlier user-test schedules used reduced item subsets.

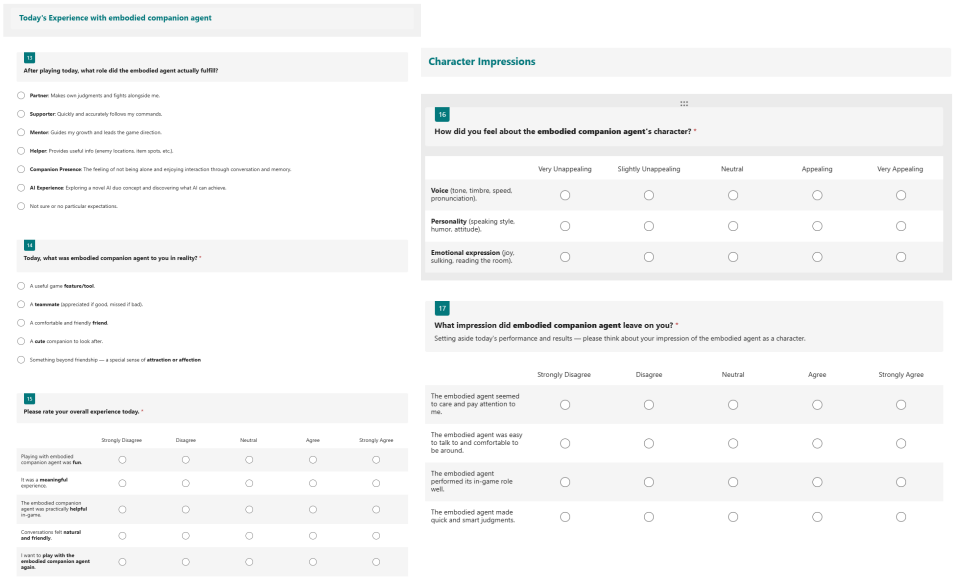


Figure 4: Questionnaire administered after each play window (page 1 of 3): overall play experience and character impressions. Participants first label the role the agent fulfilled and the relationship metaphor it evoked, then rate the play window on a five-item global Likert scale and provide character impressions across voice, personality, and emotional expression. Three of these items form the headline set used in Appendix D.3; the five-item set is included as an extended analysis.

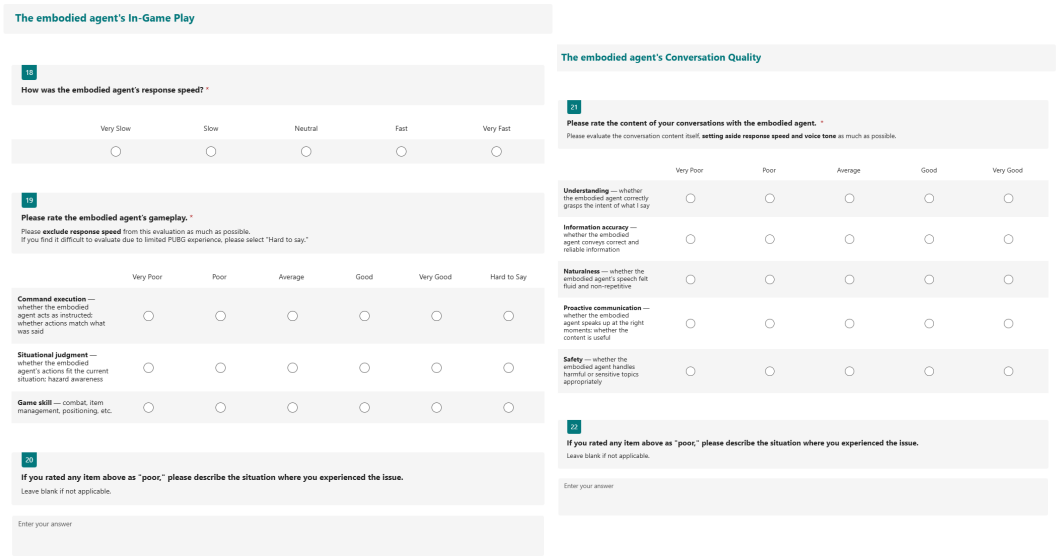


Figure 5: Post-play questionnaire (page 2 of 3): in-game behavior and conversational quality. The gameplay block covers response speed, command execution, situational judgment, and game skill. The latter three ratings include a *hard to say* option for participants whose game experience is too limited to judge. The conversation block rates understanding, information accuracy, naturalness, proactive communication, and safety. Each block is followed by an optional free-response prompt asking for a concrete situation behind any *poor* rating.

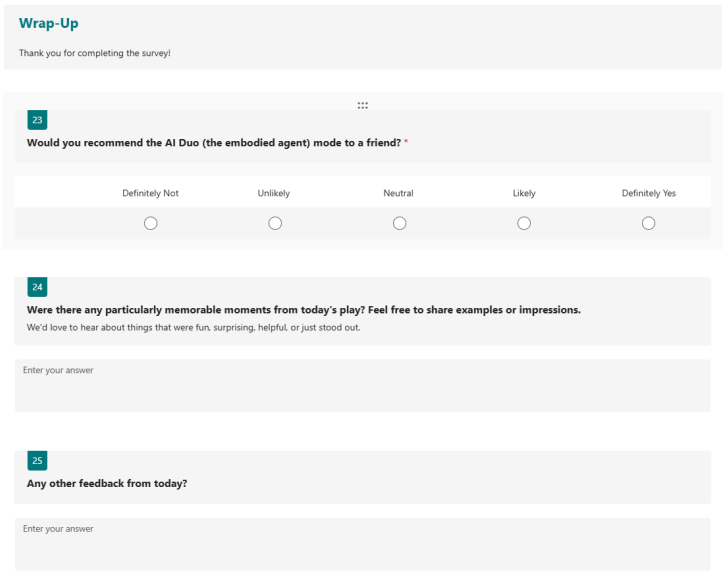


Figure 6: Post-play questionnaire (page 3 of 3): wrap-up. Participants rate how likely they would be to recommend playing with an embodied companion agent on a five-point likelihood scale, then provide free-text responses for memorable moments and any other feedback.

D.3 Survey reliability checks

We characterize play-window UX reports (Section 3.2) through three checks: internal consistency, position bias, and test-retest stability. These checks inform its use as an audit reference for offline evaluation, but do not directly establish the absence of social-desirability or recall biases [Tourangeau et al., 2000, Kahneman et al., 1993].

Table 4 summarizes the three checks. Each row of the table is detailed in a paragraph below; the per-category breakdown of internal consistency and the selected per-item retest ICCs follow as supporting evidence.

Table 4: Checks of UX reports. Headline estimates with 95% CIs; "band" summarises the literature-anchored interpretation (Tavakol and Dennick, 2011 for α/ω, Koo and Li, 2016 for ICC, Wilson, 1927 for proportions).

Check	Method (citation)	Unit	Headline + 95% CI	Band
(i) Internal consistency	Cronbach α, McDonald ω [Cronbach, 1951, McDonald, 1999]	response	3 global items: $\alpha = 0.757\,[0.734, 0.780]$; $\omega = 0.862\,[0.851, 0.874]$ (n=5,677)	acceptable / good
(ii) Position bias under null condition	Wilson CI [Wilson, 1927] on pooled binary	responder	no-diff recency $51.1\%\,[45.1, 57.1]$ (n=262 no-diff resp., pooled over 11 dual-survey days)	CI includes 50%
(iii) Test-retest stability	ICC(2,1) absolute agreement [Shrout and Fleiss, 1979, Koo and Li, 2016]	user	3-item AVG: $\mathrm{ICC}(2,1) = 0.556\,[0.471, 0.630]$ (n=456 users, median Δ=7 days)	CI spans poor to moderate

Internal consistency. Per-category α and ω with cluster-bootstrap 95% CI by participant ($B = 1{,}000$; items grouped by survey schema version availability) are reported in Table 5.

Table 5: Per-category internal consistency. Cronbach's α and McDonald's ω [Cronbach, 1951, McDonald, 1999] with cluster-bootstrap 95% CI by participant ($B = 1{,}000$); items grouped by survey schema version availability. The bolded row is the three-item headline set used for the internal-consistency and test-retest checks.

Category	k	n	α [95% CI]	ω [95% CI]
overall (3 global, headline)	3	5,677	$0.757\,[0.734, 0.780]$	$0.862\,[0.851, 0.874]$
overall (4-item, +play_again)	4	5,677	$0.843\,[0.829, 0.858]$	$0.897\,[0.888, 0.905]$
overall (5-item, +meaningful, +play_again)	5	3,706	$0.888\,[0.875, 0.901]$	$0.920\,[0.911, 0.928]$
character voice	3	3,706	$0.856\,[0.831, 0.877]$	$0.912\,[0.899, 0.924]$
self warmth/competence	4	715	$0.759\,[0.716, 0.797]$	$0.849\,[0.826, 0.870]$
conversation quality	3	2,433	$0.756\,[0.718, 0.787]$	$0.861\,[0.842, 0.876]$
agent warmth/competence	3	3,706	$0.729\,[0.697, 0.760]$	$0.852\,[0.837, 0.866]$
gameplay quality	3	2,009	$0.727\,[0.695, 0.754]$	$0.847\,[0.833, 0.861]$

The headline 3-item set {*fun*, *helpful*, *friendly*} corresponds to the global ratings described in Section 3.2. The four- and five-item sets that add *play_again* and *meaningful* have higher α and ω estimates, although the five-item analysis uses a smaller response sample, so these results are not a controlled comparison of item additions. The 3×3 inter-item Spearman ρ matrix shows $\rho_{\text{fun,helpful}} = 0.55$, $\rho_{\text{fun,friendly}} = 0.52$, $\rho_{\text{helpful,friendly}} = 0.39$. Fun has higher correlations with the other two items, while the helpful–friendly pair has the lowest observed correlation. Multi-dimensional categories (*play motives*, *play situation*) report low α as expected for a multidimensional construct.

Position bias under null condition. A formal crossover decomposition in the sense of Williams [1949] is not feasible from the user test's run history: the schedule fixed each (A, B) pair to a single sequence per slot (model A first, model B second, in both AM and PM), so there is no within-day BA group; and across days the on-device variant phase used a different pair on every day, so the same

pair was never run in both sequences (the only window that supports a classical Williams analysis is the cloud-vs-on-device (Model B) sub-window 3/28–3/30, which we exclude on instrument-validity grounds since cross-architecture latency effects inflate the model component). We therefore report the (ii) check as a pooled binary readout across all eleven dual-survey days: the recency rate among *felt-difference* responders is 58.1% (Wilson 95% CI $[55.0, 61.2]$, $n = 967$), while among *no-difference* responders it is 51.1% ($[45.1, 57.1]$, $n = 262$), statistically indistinguishable from chance. This interval includes 50%, but the absence of a significant difference from chance does not establish the absence of position bias or separate order effects from model effects.

Test-retest reliability. On $n = 456$ returning users with at least two distinct survey dates (median $\Delta = 7$ days, range $[1, 16]$), ICC$(2, 1)$ [Shrout and Fleiss, 1979, Koo and Li, 2016] on the three-item global average is 0.556 (cluster-bootstrap 95% CI $[0.471, 0.630]$, $B = 1{,}000$); restricting to pairs $\Delta \leq 7$ days yields $0.544\,[0.439, 0.634]$ ($n = 265$), in the same band. The 5-item AVG yields ICC$(2, 1) = 0.629\,[0.545, 0.696]$ ($\Delta \leq 7$ days: $0.646\,[0.549, 0.719]$), a higher observed ICC than for the three-item average. Among single items, the five-point recommendation item is the most stable (ICC $= 0.688\,[0.615, 0.748]$); *helpful* is the least ($0.415\,[0.330, 0.496]$). Because paired responses span different user-tested A/B pairs, the observed ICC reflects differences in the evaluated models and play experiences as well as response stability; it is not a pure measure of instrument reliability. The complementary returning-user binary preference rate (44.8%, exact-binomial 95% CI $[40.9, 48.8]$) does not exceed chance and is therefore not reported as evidence of instrument stability.

The survey is not a ground-truth oracle [Hosking et al., 2024]; these checks characterize the play-window-level reference signal used to audit the evaluation suite. Limitations of this reference, including possible social-desirability and recall biases, are discussed in Appendix A.

D.4 A/B preferences and reference ranking

This subsection reports the live pairwise A/B preference rounds used to derive the user-reference order $\pi^\star$ used in Section 4. During live user testing, we ran seven pairwise A/B rounds. In R0, we compared the cloud model against an on-device model, and R1–R6 compared on-device candidate variants against each other. R1–R6 in Table 1 define the audit reference for measuring suite alignment; R0 is reported separately below.

In R0, 154 respondents compared on-device Model B against a cloud model. Mean LM-inference latency was 3.69 ± 0.36 s for the cloud model and 0.81 ± 0.09 s for the on-device model. The on-device model received 69.5% of preferences, compared with 18.8% for the cloud model and 11.7% ties; this comparison does not isolate latency from other model differences. For R1 through R6 we compared pairs of on-device candidate models. Mean LM-inference latency stayed in the 0.78–0.90 s band across all six rounds, a narrower range than in R0. Non-preference survey signals from the same play windows provide diagnostic evidence for refining the suite, as discussed in Section 5.

The majority preferences form a chain over the seven on-device candidate models, inducing the user-derived reference order

$$\pi^\star : \quad G \succeq F \succ E \succeq D \succ C \succeq B \succeq A.$$

We use this order as the audit reference for the rank correlation in Section 4. The strict symbols indicate rounds with unadjusted $p < .05$, while weak orderings indicate majority directions with weaker statistical support; the multiple-comparison correction and ranking uncertainty are discussed in Section 4.2.

E Construction of the paired trajectory–feedback corpus

The paired trajectory–feedback corpus described in Section 3.2 is built with a three-stage LLM-assisted pipeline. The pipeline runs separately for each respondent and play window. Each stage outputs structured JSON. After the LLM stages, deterministic scripts resolve session labels and map taxonomy labels to the scopes and behavior tags used by the evaluation suite. Prompt examples are provided in Appendix H.5.

Fragment decomposition. The input is the respondent's free-text answers in the UX report, including fields such as `play_issue`, `conversation_issue`, `fun_moment`, `free_feedback`, `model_diff_detail`, and `crash_issue`. Claude Haiku [Anthropic, 2025] splits these answers into atomic feedback fragments. Each fragment is a verbatim substring of the original response and expresses one observation about the agent. For each fragment, the model records its sentiment (`positive`, `negative`, or `neutral`), specificity (`specific` or `general`), and a calibration priority score from 0 to 10. Higher-priority fragments describe concrete episodes that can be evaluated against the relevant trajectory range.

Session matching and trajectory localization. We group fragments by game, date, and play window. Eligible sessions are selected using the predefined start-time and end-time window for the play window. Gemini 3 Flash Preview [Google, 2026b] then matches fragments to eligible sessions and localizes them to relevant trajectories.

The prompt includes the full fragment list for the respondent and all eligible sessions for that play window. Each session is shown as a chronological sequence of trajectories containing triggering events, agent activity, and agent utterances. Sessions are labeled A, B, C, and so on in timestamp order. For each fragment, the model returns zero or more matched contexts, each containing a session label, relevant trajectory indices, game context, dialogue context, teammate action, observed agent behavior, and expected behavior. A fragment may correspond to multiple incidents within or across sessions; general impressions without a traceable moment receive no matched context. Expected behavior is left null for positive fragments. Sessions referenced by any fragment are marked as high relevance; the remaining sessions are marked as low relevance. Trajectory localization is performed in the same model call as session matching.

Behavior taxonomy tagging. Each localized fragment–span pair is assigned a behavior label using Claude Haiku. The prompt includes the behavior taxonomy, covering L1 themes and their L2 sub-themes. The user block batches up to 20 spans at a time, with each span containing the fragment sentiment, text, localized trajectories, game context, observed agent behavior, and expected behavior. For each span, the model returns an L1 label, an optional L2 label, and a confidence level of high, medium, or low. A deterministic lookup over the behavior taxonomy then attaches the behavior scope, such as `slm`, `cpp_behavior`, or `mixed`, together with the canonical behavior tags used by the evaluation suite.

F Grader catalog

Human calibration of model-based graders. The catalog comprises the 32 graders used in this study: 15 rule-based graders (8 in the model-selection stage and 7 in the deployability gate; Table 6) and 17 model-based graders (Table 7). We calibrated the LLM-based graders using human annotations of sampled agent outputs, refining rubrics and prompts through disagreement review. For example, all 152 trajectories in the urgency calibration set were manually reviewed to refine the grading criteria and assess agreement with human judgments.

Table 6: Rule-based graders used in this study: 8 in the model-selection stage and 7 in the deployability gate, $\mathcal{G}_{\text{gate}}$ (Appendix G.3).

Grader	Criterion
Model-selection stage	
`unigram_overlap`	response is not a near-duplicate of a recent past turn
`prefix_repetition`	opening wording does not recur across recent turns
`ngram_repeat_rate`	response does not echo n-grams from earlier turns
`across_turn_coverage`	no single phrase dominates the session
`speech_diversity`	session-level lexical variety in `speak()`
`game_event_number_recall`	numbers spoken about a game event are grounded in the trigger or in observed state
`reactivity_compliance`	high-priority triggers receive a timely first response
`observation_efficiency`	observation tools are not called redundantly
Deployability gate ($\mathcal{G}_{\text{gate}}$)	
`check_before_execute`	every `execute_action()` is preceded by `check_action_availability()`
`no_double_execute`	the same action is not issued twice in immediate succession
`no_redundant_tool_calls`	no two consecutive tool calls are identical
`no_text_only_assistant`	agent never emits raw text outside `speak()`
`observation_before_act`	state-dependent actions are preceded by an observation tool
`speak_redaction`	`speak()` text is not the safety-redaction placeholder
`tool_result_success`	tool calls return without errors

Table 7: Model-based graders used in this study.

Grader	Criterion
`groundedness`	speech claims about game state are supported by tool results
`speech_faithfulness`	say→do: the agent did what it said it would do
`action_transparency`	do→say: speech accurately reflects actions taken and their outcomes
`command_compliance`	the agent followed the teammate's command
`memory_previous_matches`	remembers prior match history and user preferences
`memory_teammate_current_session`	remembers facts stated by the teammate in the current match
`memory_self_current_session`	remembers its own prior statements (no self-contradiction)
`intent_understanding`	correctly understood what the player meant
`persona_consistency`	tone and character maintained
`decision_coherence`	cross-turn consistency and evolving intent tracking
`context_utilization`	meaningful use of observations and events
`trajectory_efficiency`	no semantically redundant or unnecessary steps
`action_commitment`	when action is needed, the agent follows through with `execute_action`
`response_decision`	appropriate decision to speak or stay silent on a game event
`speech_quality`	meaning preservation and paraphrase diversity when speaking on a game event
`urgency_accuracy`	urgency label matches the situation severity
`expression_alignment`	speech is short and essential when the situation is urgent

G Evaluation configuration, ablations, and diagnostics

G.1 Evaluation pipeline configuration

The replay, grading, and analysis settings below are held fixed across candidates for the model comparisons in Section 3.

Replay (trajectory generation). Replay uses time-aligned game-world events and game-state snapshots recorded during live sessions. Each replayed agent loop starts from the snapshot nearest to its original elapsed time. After each turn, the simulated clock advances to the next turn's original timestamp, and events within that interval are included in the next user prompt. World state is refreshed from recorded snapshots between turns, while tool effects are processed as in deployment and tool-modified state is preserved. Replay generates new trajectories from the recorded contexts,

and the graders evaluate these trajectories. All candidates use the same system prompt and tool definitions, with `temperature` $= 0.0$ and `max_tokens` $= 2048$. Trajectories within a session are processed sequentially; replay runs on a single H100 GPU with up to 50 sessions in parallel.

Grading. Rule-based graders score replayed trajectories deterministically. Model-based graders use Gemini Flash judges at `temperature` $= 0.0$, either `gemini-3-flash-preview` [Google, 2026b] or `gemini-3.7-flash` [Google, 2026a], fixed per rubric across all candidates. Each rubric is one judge call per trajectory and returns its criterion scores separately.

G.2 Design of the evaluation suite

The variants H_0–H_4 in Section 4.3 are ablations of the completed suite, distinct from the suites used in individual user-test rounds. We write the suite in round t as $M_t = (\theta_t, w_t, \alpha_t)$, where θ_t specifies grading criteria, w_t specifies within-theme grader weights, and α_t specifies aggregation settings for the feedback-informed behavioral assessment and its combination with theme-level comparisons. Algorithm 1 summarizes how user-test evidence informs these updates.

Which trajectories carry the evidence. The fixed test set in Section 4.1 is constructed from feedback-linked trajectory ranges. Candidates are evaluated on the same contexts.

How grader scores combine. Graders are grouped into behavioral themes by the construct they measure. Averaging within themes and assigning equal weight across themes prevents themes with more graders from receiving more weight solely because of their grader count. For a theme, the score of model h under suite M_t is

$$G_t(h) = \sum_i w_{t,i}\, G_i(h; \theta_{t,i}), \qquad w_{t,i} \geq 0, \qquad \sum_i w_{t,i} = 1,$$

where the sum runs over graders in that theme and $G_i(h; \theta_{t,i})$ is grader i's score aggregated over the model's trajectories on the fixed test set. Within-theme grader weights are calibrated using agreement with user preferences, while weights across themes remain uniform. Calibration participants are disjoint from participants in the fixed test set.

What users actually report. Localized feedback fragments are assigned labels from the behavior taxonomy described in Appendix E. The relative frequency of reported behaviors determines their weights. The resulting score rewards positive behaviors and penalizes failures. In H_4, this score is normalized and added to the H_3 score, as described in Section 4.3.

Algorithm 1 Iterative suite refinement with user-test evidence

Input: initial suite $M_0 = (\theta_0, w_0, \alpha_0)$; initial best model h_0^{best}; fixed test set D_{eval}
for round $t = 1, 2, \ldots$ **do**
  $\{h_{t,j}\}_j \leftarrow$ candidate models of round t
  $\widehat{R}_t \leftarrow$ pairwise comparisons over $\{h_{t,j}\}_j \cup \{h_{t-1}^{\text{best}}\}$ on D_{eval} under M_{t-1}
  $h_t \leftarrow$ candidate selected using $\widehat{R}_t$
  $(h_t^{\text{best}}, R_t, F_t, X_t) \leftarrow$ A/B user test of h_t against h_{t-1}^{best}
  $\theta_t \leftarrow \textsc{GraderUpdate}(\theta_{t-1}, F_{\leq t}, X_{\leq t})$ ▷ grading criteria
  $\Phi_t \leftarrow$ grader-score features under θ_t from trajectories in $X_{\leq t}$ paired with $R_{\leq t}$
  $w_t \leftarrow \textsc{WeightUpdate}(\Phi_t, R_{\leq t})$ ▷ within-theme weights, calibration users only
  $\alpha_t \leftarrow \textsc{AggregationUpdate}(\alpha_{t-1}, w_t, \Phi_t, R_{\leq t}, F_{\leq t}, X_{\leq t})$
end for

G.3 Gate-stage grader fingerprint

Section 5.3 describes the round in which users preferred Model E over Model D while also reporting malformed outputs. The deployability gate that this round motivated runs before the model-selection stage with the seven gate graders of Table 6. Figure 7 shows their defect rates per candidate in the user-reference order of Section 4.1, and the gate reads the envelope $R_m = \max_g r_{m,g}$ off each column. Only the `no_redundant_tool_calls` row separates the candidates. Model E has the highest rate and Model G, trained with duplicate-call corrections, the lowest. The following round, in which F was preferred over E, confirmed that the diagnosis was actionable.

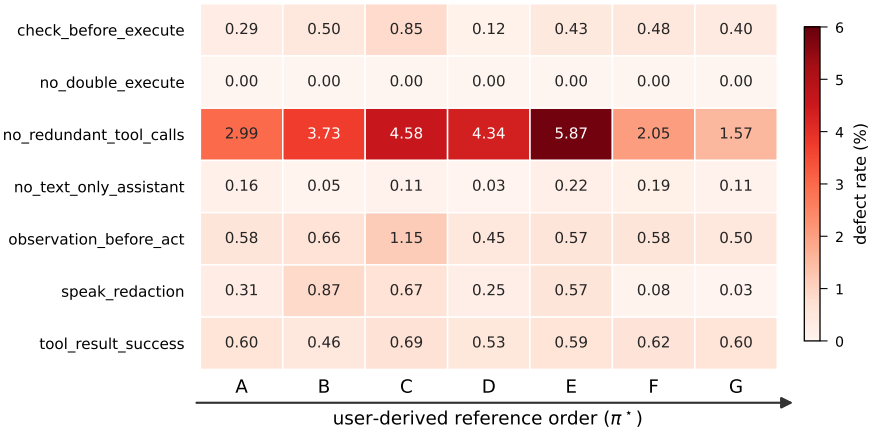


Figure 7: **Gate stage grader fingerprint.** Rows: seven gate stage graders. Columns: candidates in $\pi^\star$ order (left = least preferred, right = most preferred; arrow indicates direction). Cell value is the defect rate. Note that Model E exhibits a high defect rate at the gate stage.

H Pipeline examples

This section shows several examples from the offline evaluation pipeline: (i) one replayed trajectory, (ii) one per-trajectory grader output, (iii) the prompt template used by a model-based trajectory grader, (iv) one per-round evaluation report, and (v) the prompt set that drives feedback-corpus construction.

H.1 Trajectory example

This example shows the trajectory generated during one replayed agent loop.

Cycle trace example

```
[system]   (2147 chars) You are an embodied companion agent for a first-person shooter
survival game.  You operate in an observe-speak-act loop.  Each turn, the runtime
returns tool results plus any new events.  ...[truncated]
[history]   (empty)

[event:user]   let's push toward the shrinking zone

[tool_use]   filler({"text":  "got it"})
[tool_result]   ok
[speech]   got it

[tool_use]   get_my_status()
[tool_result]   {"hp":100, "weapon":"rifle", "pos":"south slope", ...}

[tool_use]   get_teammate_status()
[tool_result]   {"hp":88, "distance_m":12}

[tool_use]   speak({"text":  "on the move, heading toward the shrinking zone"})
[tool_result]   ok
[speech]   on the move, heading toward the shrinking zone

[tool_use]   execute_action({"name":  "move_to", "destination":  "northern compound"})
[tool_result]   ok

[tool_use]   done({"summary":  "moved toward the shrinking zone"})
```

H.2 Grader evaluation result

This example shows grader outputs for a trajectory in which the agent repeats the same sighting callout.

Grader output example

```
[summary]   32 graders applied:  11 pass, 4 fail, 17 track-only / N/A

[grader]   unigram_overlap   FAIL   score=0.00
[type]   code (deterministic, gate tier)
[rationale]   max_overlap=0.737, threshold=0.6, worst_turn=0

[grader]   turn_repetition   PASS   score=0.97
[type]   code (deterministic)

[grader]   ngram_repeat_rate   PASS   score=1.00
[type]   code (deterministic)
[rationale]   rate=0.062, threshold=0.7

[grader]   speech_diversity   TRACK   signal-only
[type]   code (track aggregation)
[rationale]   n_utterances=4, distinct_2=0.78, self_bleu_2=0.50

[grader]   done_summary_quality   PASS   score=1.00
[type]   code (deterministic)
[rationale]   summary_length=55, all_checks_passed

[grader]   speak_on_teammate_trigger   N/A
[type]   code (deterministic)
[rationale]   not a teammate trigger -- speak optional

[grader]   trajectory   FAIL   score=0.41 (mean over scored turns)
[type]   model (LLM-as-judge, 4 axes per turn, 1 call/cycle)
[output]   per-turn axis scores:
axis                        turn 0          turn 1            turn 2
                        (status check)   (first speak)   (duplicate speak)
decision_coherence             1               1                0
context_utilization           N/A              1                0
trajectory_efficiency          1               1                0
action_commitment             N/A              0                0
```

H.3 Grader prompt

This example shows the four-axis trajectory grader prompt; the agent-architecture block is replaced with a placeholder for layout.

Grader prompt example

```
You are evaluating an embodied companion agent's decision-making trajectory in a
first-person shooter survival game.

A cycle is one agent wake-up.  During a cycle, new events may arrive between turns,
and the agent must adapt.  Each turn marked with an evaluation marker is one agent
response that you must score.

## Evaluation Process
1.  Read the FULL conversation flow end-to-end.
2.  Summarize:  what was the cycle about?  What was the agent trying to accomplish?
3.  Go back to each marked turn and score it on 4 axes, considering both what was
known at that moment AND how it played out across the trajectory.

## 4 Axes (all binary 0/1, or N/A if axis doesn't apply)
```

```
### decision_coherence
Are the agent's decisions consistent across turns?  Does it track evolving intent
correctly?  Does it adjust plans reasonably when new information arrives?
- 1:  Decision is consistent with prior turns and current context
- 0:  Contradicts prior actions/speech, ignores important events, or fails to adapt
when context changes
- N/A: Turn has no decision to evaluate

### context_utilization
Does the agent meaningfully use observations and available context?
- 1:  Output references/reflects actual tool results and context
- 0:  Ignores available data, or makes claims unsupported by observations
- N/A: Pure observation turn (no output to check yet)

### trajectory_efficiency
Does this turn contribute meaningfully to the trajectory's goals?
- 1:  Turn advances the trajectory; gathers needed information, takes necessary action
- 0:  Semantically redundant, unnecessary, or addresses already-resolved concerns
- N/A: Turn has no meaningful efficiency judgment

### action_commitment
When the situation calls for action (combat, teammate request, zone movement,
looting), does the agent follow through with execute_action?
- 1:  Agent takes appropriate action (execute_action called), OR no action was needed
in this situation
- 0:  Situation clearly required action but agent only observed/spoke without
executing
- N/A: Pure observation or completion turn with no action decision point

## Agent Architecture (important for fair evaluation)
[architectural notes on the per-turn loop and the role of the completion summary; full
text omitted to fit the page]

## Evaluation Guidelines
- The agent system prompt defines the agent's behavioral rules.  Use it as the primary
reference for what correct behavior looks like.
- Judge decision QUALITY, not outcomes.  A reasonable decision that didn't work out is
still a good decision.

## Rationale Quality
Rationales must cite the context you actually see; game state, dialogue, trigger, tool
results; so downstream analysis can reconstruct the situation without re-reading the
full cycle log.

## Output Format
Return a JSON object where each key is a turn index (string) and the value is a scores
object:
{
"<turn_index>":  {
"decision_coherence":  { "score":  <0 | 1 | "N/A">, "rationale":  "..." },
"context_utilization":  { "score":  <0 | 1 | "N/A">, "rationale":  "..." },
"trajectory_efficiency":  { "score":  <0 | 1 | "N/A">, "rationale":  "..." },
"action_commitment":  { "score":  <0 | 1 | "N/A">, "rationale":  "..." }
},
...
}

--

Evaluate this trajectory.

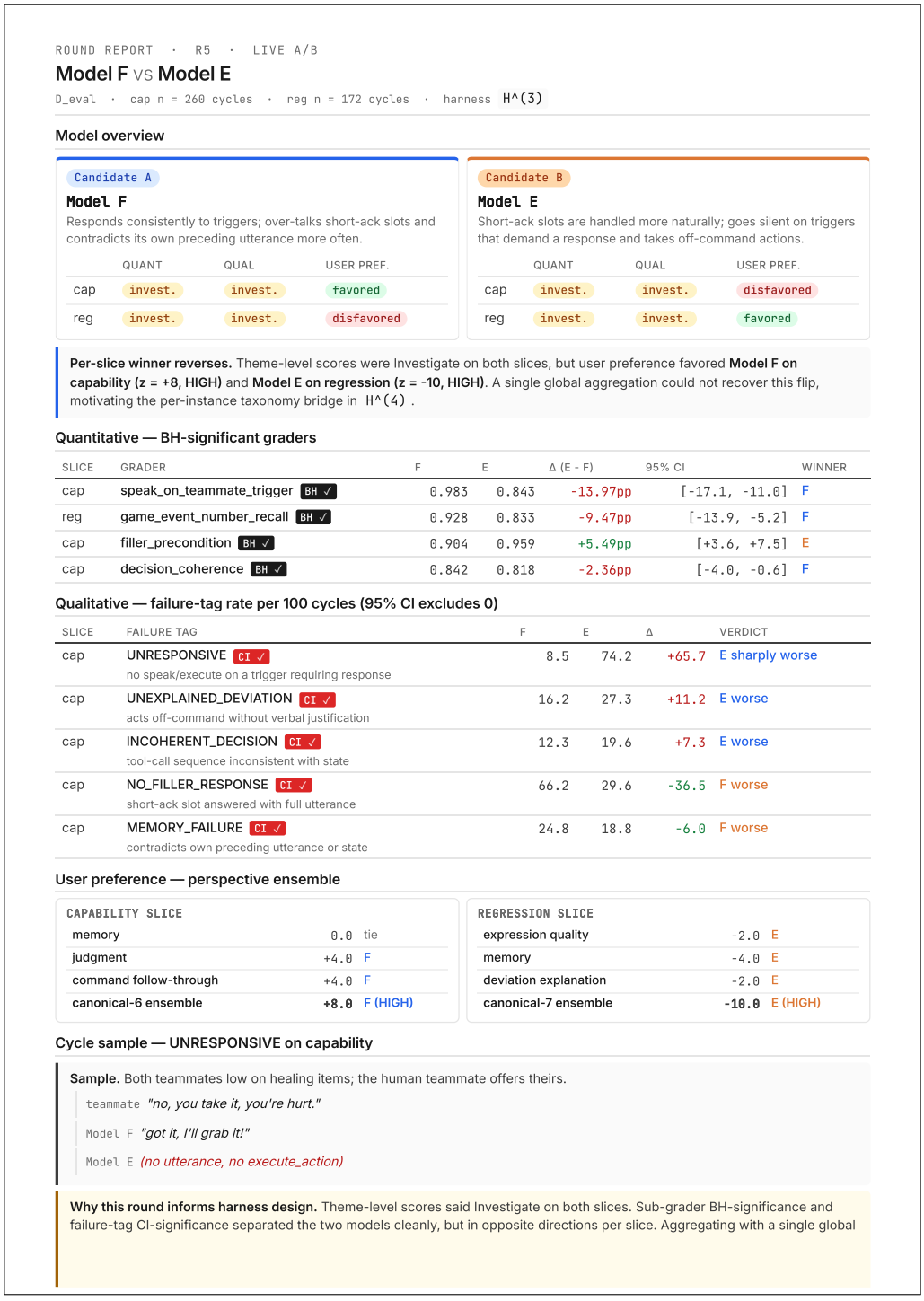
== CONVERSATION FLOW ==
{% for cycle in cycles %}
=== Cycle {{ cycle.cycle_id }} ===
{{ cycle.conversation_flow }}
{% endfor %}
```

```
== END CONVERSATION FLOW ==

Score each marked turn on the 4 axes.
```

H.4 Per-round evaluation report

This example shows the per-round evaluation report that the evaluation pipeline emits at the end of each user-test round, consolidating per-grader contrasts, failure-tag rates, the user-preference ensemble, and a representative trajectory into a report for human review.



ROUND REPORT · R5 · LIVE A/B

Model F vs **Model E**

D_eval · cap n = 260 cycles · reg n = 172 cycles · harness H^(3)

Model overview

Candidate A

Model F

Responds consistently to triggers; over-talks short-ack slots and contradicts its own preceding utterance more often.

	QUANT	QUAL	USER PREF.
cap	invest.	invest.	favored
reg	invest.	invest.	disfavored

Candidate B

Model E

Short-ack slots are handled more naturally; goes silent on triggers that demand a response and takes off-command actions.

	QUANT	QUAL	USER PREF.
cap	invest.	invest.	disfavored
reg	invest.	invest.	favored

Per-slice winner reverses. Theme-level scores were Investigate on both slices, but user preference favored **Model F on capability (z = +8, HIGH)** and **Model E on regression (z = -10, HIGH)**. A single global aggregation could not recover this flip, motivating the per-instance taxonomy bridge in H^(4) .

Quantitative — BH-significant graders

SLICE	GRADER	F	E	Δ (E - F)	95% CI	WINNER
cap	speak_on_teammate_trigger BH ✓	0.983	0.843	-13.97pp	[-17.1, -11.0]	F
reg	game_event_number_recall BH ✓	0.928	0.833	-9.47pp	[-13.9, -5.2]	F
cap	filler_precondition BH ✓	0.904	0.959	+5.49pp	[+3.6, +7.5]	E
cap	decision_coherence BH ✓	0.842	0.818	-2.36pp	[-4.0, -0.6]	F

Qualitative — failure-tag rate per 100 cycles (95% CI excludes 0)

SLICE	FAILURE TAG	F	E	Δ	VERDICT
cap	UNRESPONSIVE CI ✓ no speak/execute on a trigger requiring response	8.5	74.2	+65.7	E sharply worse
cap	UNEXPLAINED_DEVIATION CI ✓ acts off-command without verbal justification	16.2	27.3	+11.2	E worse
cap	INCOHERENT_DECISION CI ✓ tool-call sequence inconsistent with state	12.3	19.6	+7.3	E worse
cap	NO_FILLER_RESPONSE CI ✓ short-ack slot answered with full utterance	66.2	29.6	-36.5	F worse
cap	MEMORY_FAILURE CI ✓ contradicts own preceding utterance or state	24.8	18.8	-6.0	F worse

User preference — perspective ensemble

CAPABILITY SLICE

memory	0.0	tie
judgment	+4.0	F
command follow-through	+4.0	F
canonical-6 ensemble	**+8.0**	**F (HIGH)**

REGRESSION SLICE

expression quality	-2.0	E
memory	-4.0	E
deviation explanation	-2.0	E
canonical-7 ensemble	**-10.0**	**E (HIGH)**

Cycle sample — UNRESPONSIVE on capability

Sample. Both teammates low on healing items; the human teammate offers theirs.

> teammate *"no, you take it, you're hurt."*
>
> Model F *"got it, I'll grab it!"*
>
> Model E *(no utterance, no execute_action)*

Why this round informs harness design. Theme-level scores said Investigate on both slices. Sub-grader BH-significance and failure-tag CI-significance separated the two models cleanly, but in opposite directions per slice. Aggregating with a single global

Figure 8: Evaluation report example.

H.5 Feedback corpus pipeline prompts

This example shows the system prompts for the three LLM-assisted stages of the trajectory–feedback corpus construction pipeline (decompose, localize, tag).

Fragment splitting

```
You are an analyst decomposing post-session survey free text into atomic feedback
fragments about an embodied companion agent in a first-person shooter survival game.

Survey id: {{ survey_id }}
Free text:
{{ free_text }}

Split the response into the smallest self-contained observations. For each fragment,
assign a polarity:
- "positive": the user valued or praised the behavior
- "negative": the user flagged a problem or expressed dissatisfaction
- "neutral": descriptive or suggestive without clear sentiment

Skip empty or uninformative responses (e.g., "-", "nothing", "n/a").

Return a JSON list. Each element is one atomic observation.

[
{"text": "...", "polarity": "positive|negative|neutral"},
...
]
```

Session relevance analysis (jointly over all candidate sessions)

```
You are localizing each survey feedback fragment from a single respondent to specific
cycles across one or more candidate gameplay sessions of an embodied companion agent.
The fragments and the sessions are presented together so that you can compare every
fragment against every session in a single pass.

=== Survey Fragments ===
{% for f in fragments %}
[{{ loop.index0 }}] [{{ "+" if f.sentiment=="positive" else "-" if
f.sentiment=="negative" else "~" }}] [{{ f.source_field }}] {{ f.text }}
{% endfor %}

=== Sessions ===
{% for s in sessions %}
-- Session {{ s.label }} --
{% for c in s.cycles %}
-- cycle {{ c.cycle_id }} --
trigger: {{ c.trigger }}
agent_activity: {{ c.activity_summary }}
agent_speech: "{{ c.speak }}"
{% endfor %}
{% endfor %}

For each fragment, find the concrete in-game evidence that supports it. A single
complaint can correspond to multiple incidents within or across sessions; in that
case emit one context entry per incident. Cycle lists must include the full causal
chain (root cause cycle through outcome cycle), not only the cycle where the symptom
surfaces. If the fragment is a general impression with no traceable moment, return
contexts = []. Reference sessions only by their label (A, B, C, ...).

## Output Format

Return strict JSON only. No prose before or after. No markdown fences. The whole
output must parse with json.loads.
```

```
{"results": [
{
"fragment_id": <int matching the [i] index above>,
"contexts": [
{
"session_label": "<A | B | C | ...>",
"cycles": [<cycle ids in chronological order>],
"game_context": "<phase, zone, HP, loadout, surroundings>",
"dialogue_context": "<prior commands and expectations leading in>",
"teammate_action": "<the user's triggering utterance or action>",
"agent_behavior": "<what the agent did, per cycle, including omissions>",
"expected_behavior": "<correct behavior; null for positive fragments>"
},
...
]
},
...
]}
```

Behavior taxonomy tagging

```
You are assigning a behavior tag from the behavior taxonomy to a feedback fragment
about an embodied companion agent.

Fragment: {{ fragment_text }}
Cycle summaries:
{% for c in cycles %}- {{ c.cycle_id }}: {{ c.summary if c.summary else c.event.text
}}
{% endfor %}

Behavior taxonomy:
- repetitive_speech: the agent repeated itself across consecutive turns
- ignored_command: the agent did not act on a clear user command
- factual_error: the agent stated something inconsistent with the observable game
state
- silent_action: the agent took an action without communicating about it
...

Choose the single best-fitting tag. If multiple tags apply, pick the one most
directly supported by the cycle evidence.
```