DIAGNOSING PERCEPTUAL BOTTLENECKS OF LLM AGENTS IN INTERACTIVE GAMES WITH UNKNOWN RULES

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce a framework for evaluating LLM agents' perception through their action-selection reports on ARC-AGI-3, a benchmark of interactive games with unknown rules, controls, and goals. It automatically labels screen changes and uses an LLM judge to compare them with agents' reports, identifying omissions, inaccurate descriptions, and unsupported claims. Across four models, errors vary by model and change type; even explicitly reported changes are sometimes misdescribed. To test whether detected errors hinder progress, we resume execution from identical states and histories, with and without a message describing a selected unreported change. The intervention improves both reporting accuracy for the selected changes and overall level attainment in Qwen3.8-Max, highlighting the importance of accurate perception for reasoning and planning.

1 INTRODUCTION

Autonomous agents often need to act before they understand an unfamiliar environment. They must use interaction efficiently to learn how their actions affect the environment and what counts as success, updating their beliefs as they gather evidence. ARC-AGI-3 captures this problem in novel interactive environments whose rules, action meanings, and goals are not given in advance (ARC Prize Foundation, 2026b; Courtis et al., 2026). Each action can both advance the task and provide information about it. Recent work has used ARC-AGI-3 to study systematic exploration (Rudakov et al., 2026), online world-model construction and verification (Courtis et al., 2026; Skoutnev et al., 2026; Lehmann et al., 2026), goal inference (Skoutnev et al., 2026), and structured test-time learning (Arjmandi, 2026).

Although these tasks emphasize reasoning and planning, errors can begin in *perception* and propagate to later stages. Wen et al. (2025) document this pattern in rule generation and execution on static ARC-AGI, and Arjmandi (2026) reports a similar effect on an agent's knowledge updates. In unfamiliar interactive environments, perceptual errors can distort both the current state estimate and the evidence used to infer rules and goals. Yet task failure alone cannot distinguish correct perception followed by faulty reasoning or planning from an earlier perceptual error (Figure 1a). Reasoning traces are also difficult to diagnose because observation descriptions, rule hypotheses, and action plans are often mixed in free-form text. Even when perceptual claims can be identified, checking them against visual or structured observations requires semantic comparison rather than exact text matching. Evaluating perception separately from task performance therefore requires both a reference for what was observable and a way to compare it with the agent's reports.

To address this gap, we develop a framework for evaluating LLM agents' perception through the reports they use to select their next actions. Applied to ARC-AGI-3, the framework automatically generates ground-truth annotations of observable screen changes. An LLM judge compares these annotations with the reports, and we check its judgments against those of a blinded human evaluator. The evaluation identifies unreported changes, errors about what changed or how it changed, and unsupported claims about changes that did not occur. Our first research question is: **RQ1: How accurately do LLM agents perceive screen changes after their actions?** We evaluate report completeness and accuracy across models, games, and change types.

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

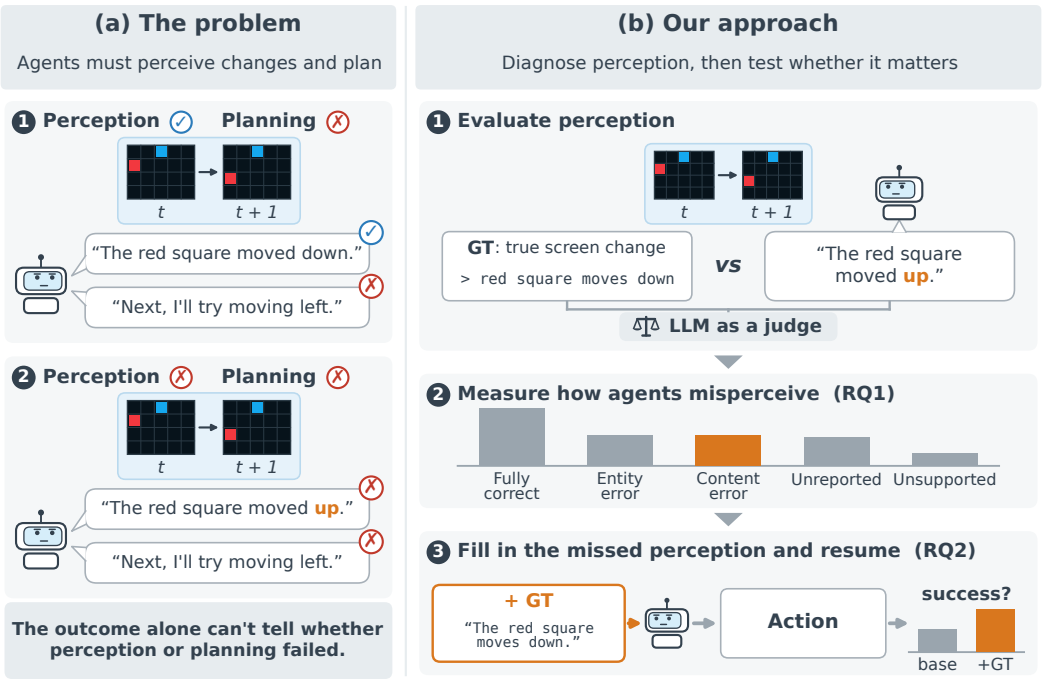


Figure 1: Overview of the diagnostic problem and our framework. **(a) Ambiguous task failure.** The same failed outcome can follow an accurate report with an incorrect plan (①) or an inaccurate report of a screen change (②). The outcome alone cannot distinguish these cases. **(b) Evaluation and intervention.** ① We automatically label screen changes and use an LLM judge to compare these ground-truth (GT) annotations with the report the agent uses to select its next action. ② The judge evaluates how each GT change is reported and classifies each change claim as linked, unsupported, out of scope, or indeterminate (RQ1). ③ We resume execution from the same state and history, with and without a message describing an unreported change, and compare target-report accuracy and subsequent level attainment (RQ2).

A reporting error need not limit task performance: the change may be irrelevant to progress, or the agent may fail for other reasons. Our second research question is therefore: **RQ2: Are the detected perceptual problems bottlenecks for task execution?** We use a common rule across games to select intervention targets from the RQ1 diagnosis, such as an object movement the agent failed to report. We then resume execution from the same state and history, with and without a single message describing the selected change. The message provides neither correct actions nor game-solving rules. We compare the next report's accuracy on the target change and subsequent level attainment across conditions, testing whether the supplied information improves reporting and task execution.

Our contributions are threefold.

1. **A perception evaluation framework for interactive environments with unknown rules.** Instantiated on ARC-AGI-3, the framework combines automatic screen-change labeling with LLM judging to identify unreported or misreported changes separately from task outcomes. We will release the change-extraction rules, judge prompts, intervention messages, and execution logs to support evaluation of other agents and games.
2. **An analysis of agents' perception during execution.** Across Gemma4 31B (Google DeepMind, 2026), Qwen3.8-27B (Qwen Team, 2026), Qwen3.8-Max (Alibaba Cloud, 2026), and Claude Opus 5 (Anthropic, 2026), patterns of unreported changes and reporting errors differ by model and change type. Agents also misdescribe changes they mention, so mention alone does not establish report accuracy.

3. **A targeted intervention framework for testing perceptual bottlenecks.** By supplying only a diagnosed screen change and resuming from the same state and history, we test whether correcting that information improves subsequent reporting and task execution. The intervention improves target-report accuracy in both tested models and increases aggregate level attainment for Qwen3.8-Max, demonstrating how the framework can distinguish correctable perceptual deficits from limitations arising downstream.

2 Related Work

ARC-AGI-3 and agent failure analysis. ARC-AGI-3 (ARC Prize Foundation, 2026b) evaluates agents in unfamiliar interactive games whose action effects and success conditions must be inferred through play. Game suites such as BALROG (Paglieri et al., 2025) and VideoGameBench (Zhang et al., 2025a) evaluate agents in existing game environments, whereas Ying et al. (2025) propose novel games for studying adaptive world models. lmgame-Bench (Hu et al., 2026) additionally provides configurable perception, memory, and reasoning modules for probing component capabilities. In ARC-AGI-3, Sensi (Arjmandi, 2026) separates observation from action selection and documents how erroneous descriptions of screen changes propagate into subsequent hypotheses and knowledge updates. Perceptual bottlenecks and error propagation have also been studied in static abstract-reasoning tasks (Wang et al., 2026; Wen et al., 2025). These studies motivate examining the information used for decisions alongside task outcomes. We focus on whether agents correctly report the visible consequences of their own actions.

Perception evaluation in agents Existing approaches differ in their reference information, environment coverage, and required agent outputs. EmbodiedBench (Yang et al., 2025) and CUADebug (Zhang et al., 2026b) diagnose perceptual failures through episode analysis or human annotation, while AGMem (Choi et al., 2026) automates failure classification without constructing environment-derived perception GT. Theory of Space (Zhang et al., 2026a) provides simulator-derived spatial GT but evaluates explicitly elicited cognitive maps in spatial-exploration tasks. VAGEN (Wang et al., 2025) and HalluWorld (Liu et al., 2026) provide environment-derived references across heterogeneous environments, but evaluate designated state-estimation/prediction fields and separately elicited probe answers, respectively. Our framework automatically extracts visible-change GT across games with different dynamics and goals, then evaluates the free-form reports used for action selection. It imposes no fixed state or map schema on the agent's reports and requires no additional diagnostic answers: reports may mix observations, hypotheses, and plans.

Information interventions and downstream behavior Zenkri & Brock (2026) compare RGB, RGB-D, and ground-truth symbolic observations in a sequential mechanical puzzle, finding that greater observation fidelity does not necessarily improve success. AgentDebug (Zhu et al., 2025) and CUADebug (Zhang et al., 2026b) use diagnosed execution errors to provide corrective feedback for re-execution. Our intervention supplies a description of only one diagnosed visible change, once, without game-solving rules or correct actions, and compares continuations from the same state and history. By measuring both correction of the targeted report and subsequent task attainment, we test whether repairing a reporting deficit also improves execution.

3 Evaluation Framework

Figure 1(b) gives an overview of the diagnosis and the intervention. We compare action-induced changes with the reports used for the next decision. For diagnosed discrepancies, we then supply the corresponding change information and test its effect on subsequent reports and task execution.

3.1 Agent Execution and External Evaluation

We view *perception* as updating the agent's estimate of the environment state from the new observation, the preceding action, and retained information, as in state estimation under partial observability (Kaelbling et al., 1998). At step t, the environment is in state $s_t \in \mathcal{S}$. The agent receives the observation $o_t = (f_{t,1}, \ldots, f_{t,K_t})$, where $f_{t,k}$ is the k-th of the $K_t \geq 1$ frames presented after action a_{t-1} and each frame is an integer grid with values in $\{0, \ldots, 15\}$. It then generates a

free-form report, the *Reasoning Log* ℓ_t, and selects an action $a_t \in \mathcal{A}$. Actual Reasoning Logs can contain observations, rule hypotheses, and action plans. We compare the changes that a_{t-1} causes in o_t with ℓ_t, which the agent then uses to select a_t. We evaluate the change descriptions in the Reasoning Log, not the agent's unobserved internal state.

The same evaluated model receives an observation, produces the Reasoning Log, and selects the next action using that log. A common harness constructs inputs, retains history, executes actions, and saves logs. Built on the official ARC-AGI-3 agent implementation (ARC Prize Foundation, 2026a), the harness presents screens as textual integer grids. After execution, an external LLM judge (Zheng et al., 2023) scores the stored logs against reference records of actual screen changes, which we call Ground Truth (GT). The judge is separate from the evaluated agent and does not influence the agent's actions in RQ1. Its prompt combines a common judging procedure with game-specific evaluation rules.

3.2 Reference Records of Visible Changes

Each GT record identifies an entity and its visible change. For example, the scene in Figure 1 yields the record "the red square moved one cell down." We call one such record a GT event.

We construct extraction rules from game code and screen transitions, checking object identity, visible attributes, and correspondence with the rendered screen. Extraction then runs deterministically on environment records and the frames actually presented to the agent. Human input is limited to these per-game codebooks and the game-specific judging rules; extraction, judging, target selection, and the intervention message then follow fixed procedures. GT extraction excludes hidden state changes and does not use the Reasoning Log. A game-specific codebook groups evaluated phenomena into *game mechanics*, such as the movement of an object, a change in a switch's state, or a decrease in a counter. These are operational categories for evaluation, not a claim of a unique minimal decomposition of the game.

Formally, the GT event set is defined as

$$m_t = \pi(s_t), \qquad \Delta_t = \delta(m_{t-1}, m_t;\, o_t). \tag{1}$$

Here, $\pi : \mathcal{S} \to \mathcal{M}$ maps a state to the entities selected for evaluation together with their visible attributes, such as position, orientation, and color, where $\mathcal{M}$ is the set of such records. The function δ returns $\Delta_t \subseteq \mathcal{E}$, a finite set of GT events drawn from the set $\mathcal{E}$ of possible events, each specifying an entity, its game mechanic, and its visible change. The states s_{t-1} and s_t determine entity identity, whereas o_t captures changes that appear only in intermediate frames, such as a transient flash. GT is reproducible under these visibility and codebook rules, although reproducibility does not imply perfect accuracy or complete coverage.

3.3 Comparing Actual Changes and Reported Claims

Let $\hat{\Delta}_t$ be the set of change claims extracted from the Reasoning Log ℓ_t. We compare the actual changes Δ_t and reported claims $\hat{\Delta}_t$ in two directions. Starting from each GT event, we ask whether the agent reported it and whether the description was correct. Starting from each change claim in the log, we ask whether any GT event supports it. These are the *event-side* and *claim-side* evaluations, respectively.

For each GT event, `reported` indicates whether the change was mentioned. If so, `entity` and `content` independently indicate whether the object and the change type or attributes match. Figure 2 (right) gives actual examples of each combination. If the event is unreported, the remaining labels are null. Hedging—language that marks uncertainty or alternative interpretations—is recorded separately from correctness.

For each claim, the judge assigns `linked`, `unsupported`, `out_of_scope`, or `indeterminate`. A linked claim has a corresponding GT event but may still contain an entity or content error. Unsupported claims lack support within the evaluation scope; claims outside that scope or with ambiguous correspondence are labeled `out_of_scope` or `indeterminate` and reported separately. The main analysis uses four event-side rates. The fully correct event rate is the fraction of GT events that are reported with the correct entity and content; the unreported rate is the fraction of GT events absent from the report. Among reported events, the entity-match rate

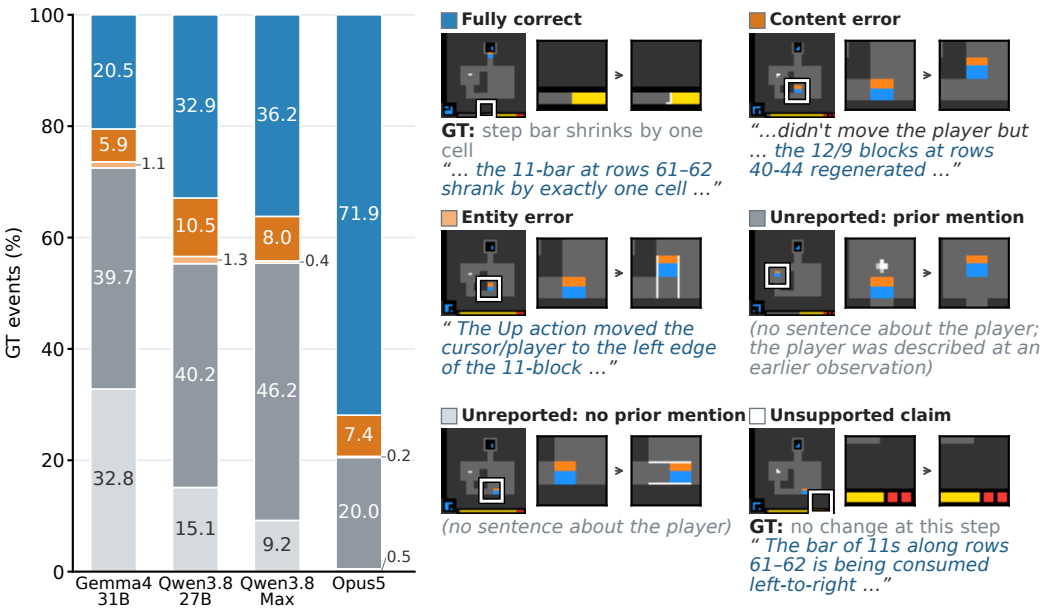


Figure 2: Left: reports by model, as judged by GPT-5.6 Sol, averaged as described in the text; colors follow the panel labels on the right. Right: one actual example per category from a single game under the ordinary condition. Each panel shows the full screen after the agent's action with the changed region outlined, that region before (left) and after (right) the action, and a verbatim excerpt of the report; the phrase in blue is the one the judge matched to the event. In the four player panels the GT event is a five-cell move of the player. Both judges assigned these labels (sources in Appendix G.6); GLM-judged rates and the human comparison are in Figures 20 and 22. "Unreported: prior mention" denotes an unreported change whose game mechanic had already been mentioned earlier in the run; the unsupported claim is a claim-side category with no GT event at that observation.

is the fraction whose object matches (the rest are entity errors), and the content-match rate is the fraction whose change type and attributes match (the rest are content errors).

3.4 Targeted Change Intervention and Task Progress

An unreported or misdescribed change is a candidate bottleneck, but the diagnosis alone does not show that the error held the agent back. We therefore provide the corresponding change information and test whether subsequent progress improves. An improvement supports the diagnosis that the information deficit limited progress at that point; if the report improves but progress does not, correcting the report was not sufficient there. The message identifies the object by its appearance and location and describes what changed. For the scene in Figure 1, it could say "The red square on the left moved one cell down." This anonymized description omits the object's strategic role, game-solving rules, correct actions, and future information. Because the message also points to the target, the comparison estimates the combined effect of supplying the information and directing attention to it, without separating the two.

For the selected event $e_t^\star \in \Delta_t$, the intervention message is $z_t = g(e_t^\star)$, where g maps a GT event to its anonymized textual description by a fixed template. In the intervention condition, the agent receives this message once, immediately before its first resumed Reasoning Log; in the baseline condition, it resumes from the same state and history without the message. Both conditions then continue autonomously for the same action budget. Section 4.3 specifies the rule used to select targets and the number of continuations.

Table 1: Fully correct event rate (%) by type of change, pooled over encountered events (Sol / GLM). Types are assigned from the mechanic descriptions; both aggregations are in Appendix G.5.

Model	Movement	Color or state	Counter or size
Gemma4 31B	32 / 30	6 / 2	22 / 26
Qwen3.8-27B	52 / 45	39 / 44	24 / 23
Qwen3.8-Max	46 / 41	30 / 26	27 / 29
Opus5	80 / 77	55 / 56	74 / 73

We first evaluate whether the resumed report describes the selected event with the correct entity and content. This target-report outcome serves as a manipulation check: it tests whether the supplied information is reflected in the report used for the next action.

Second, we measure level attainment. Let T be the first action at which a new level is attained. Per-run attainment within a budget of N actions is $Z_N = \mathbb{I}[T \leq N]$, where $\mathbb{I}$ is the indicator function; N is set in Section 4. Attainment is determined from all transition records, including runs that reset afterward. Higher attainment across tested points supports a bottleneck diagnosis; report correction can occur without attainment improvement.

4 Empirical Evaluation

4.1 Common Experimental Setup

We use the ARC-AGI-3 Public Demo Set.[1] Screens are presented to the agent as textual integer grids; each run is one episode with an action limit of 50. We evaluate four models: the two self-hosted open-weight models, Gemma4 31B and Qwen3.8-27B, and the two API-served models, Qwen3.8-Max and Claude Opus 5 (hereafter Opus5). We use GPT-5.6 Sol as the primary judge and GLM-5.2, from a different model family, as a second judge on the same inputs.

In RQ1, the agent is asked to write, based on the presented screen, a few sentences of observations useful for its next action. We perform five runs for each of the 20 games and four models, giving 400 runs; each model–game–replicate combination is one evaluation unit.

Because RQ2 requires 20 continuations per game and model, we limit its cost by using only Qwen3.8-Max and Opus5, the two models that reported changes most accurately in RQ1 (Figure 2). We set $N = 15$. Details are in Appendices C, E, and D.

4.2 RQ1: How Accurately Do LLM Agents Perceive Screen Changes after Their Actions?

Weaknesses in reports vary across models and change types Rates are averaged with equal weights over game–mechanic pairs shared by all four models, first across mechanics and then across games. As shown in Figure 2, the unreported rate ranged from 20.5% to 72.5% across models, and the fully correct event rate from 20.5% to 71.9%; Opus5 had the highest fully correct event rate. Aggregate accuracy hides large differences between mechanics of the same game for the same model: the fully correct event rates of the best- and worst-reported mechanics differed by a median of 28 percentage points. Report accuracy also depended on the kind of change (Table 1): pooled over events, movements were reported most accurately for every model under both judges, whereas accuracy on color, state, and counter changes varied by model. Prior mention of a mechanic did not prevent later omissions of its changes (Figure 2), and this pattern persisted for action-specific changes after matching on prior mention and occurrence order. Among reported events (pooled over all encountered events), entity-match rates ranged from 95.5% to 99.7% across models, whereas content-match rates ranged from 79.2% to 93.5%: agents mostly referred to the right object but misdescribed what happened to it. Agents also asserted changes without support in the GT: among

[1] We evaluate 20 of its 25 games. In the other five, changes cannot be reliably attributed to single on-screen entities (overlapping joints, a body and its copy, or water behaving as a continuous region), or the game code is not covered by the shared transition extractor (two games).

claims classified as linked or unsupported, 8.9%–30.4% were unsupported under Sol and 8.0%–20.6% under GLM. Omission and invention were not a trade-off: within each model, games with more unreported changes also had more unsupported claims (Spearman's $\rho = 0.34$–0.80).

Judges and human agreement The two judges agreed on whether an event was fully correct for 92.1% of events, and the model ranking was unchanged across judges and aggregation weights; on 200 items evaluated under blinded conditions, a human evaluator agreed with the judges on the reported/entity/content tuple for 80.0% (Sol) and 83.5% (GLM); Sol had been fixed as the primary judge before this validation (Appendices D.1 and F.3).

Answer to RQ1 RQ1 located where reports fail. Accuracy differed widely across models (20.5%–72.5% of changes unreported) and across mechanics within the same game. Among reported changes, most errors concerned what changed rather than which object changed. These discrepancies are candidates only: reports alone cannot show whether any of them limited progress, which RQ2 tests by intervention.

4.3 RQ2: Are the Detected Perceptual Problems Bottlenecks for Task Execution?

Selecting intervention targets from RQ1 A common rule, fixed before running the comparisons, selects one intervention point per game and model from the RQ1 diagnosis: the mechanic with the lowest fully correct event rate for that model in that game (excluding resource and counter displays), at the earliest observation across the five original runs where it was unreported or misreported. From the saved state and history at that point, we sample 10 baseline and 10 intervention continuations of 15 actions each; the intervention supplies an anonymized description of the selected GT event once, before the first report. We evaluate level attainment within the budget and target-report accuracy, the fully correct rate for the target event in the first resumed report, scored by both judges.

Effect on the target report We first ask whether the intervention repairs the diagnosed reporting error (Figure 4 shows one point). For Qwen3.8-Max, fully correct target reports increased from 33.0% to 73.5% under Sol and from 28.5% to 74.5% under GLM. Accuracy increased in 17/20 and 20/20 games, respectively. Opus5 showed a similar effect: Sol accuracy increased from 48.5% to 74.0%, and GLM from 42.5% to 71.0%, with increases in 13/20 games for each judge. Even so, about a quarter of first resumed reports remained not fully correct, and accuracy decreased in two Opus5 games, so supplying the information did not ensure that it was reflected in the next report. The intervention also changed the claim side of the reports (Table 2): for Opus5, first reports with unsupported claims about the target mechanic fell from 19 to 14 under Sol and from 10 to 3 under GLM (of 200), and for Qwen3.8-Max the share of change claims linked to an actual change, among linked, unsupported, and indeterminate claims, rose from 78% to 90% (Sol) and from 68% to 89% (GLM).

Effect on level attainment We then examine whether the intervention also changes task execution. For Qwen3.8-Max, level attainment increased from 6/200 to 17/200 runs (20 games $\times$ 10 continuations per condition; 3.0% to 8.5%; Cochran–Mantel–Haenszel test stratified by game, $p = 0.011$, common odds ratio $= 3.6$). Six games increased, 13 were unchanged, and one decreased; in 12 games no continuation in either arm reached a new level. The increase is sensitive to individual games: excluding the game with the largest increase gives $p = 0.066$, and a game-level sign-flip test gives $p = 0.078$. For Opus5, total attainment was 66/200 in both conditions (33.0%), masking game-level changes: five games increased, nine were unchanged, and six decreased. Decreases (e.g., from 10/10 to 7/10) are not explained by ceiling effects.

For Opus5, the reporting failure identified in the original trajectory often did not recur: at eight checkpoints, baseline target reports were already correct in at least nine of ten continuations, leaving little room to test report correction at those points.

Reports and attainment Correct target reports were more common among attaining runs than among all runs for Qwen3.8-Max; the corresponding proportions were similar for Opus5 (Table 2). Among the 23 attaining Qwen3.8-Max runs, 87% (20 runs, under both judges) had reported the target correctly, against 53% (Sol) and 52% (GLM) of all its runs; among the 132 attaining Opus5

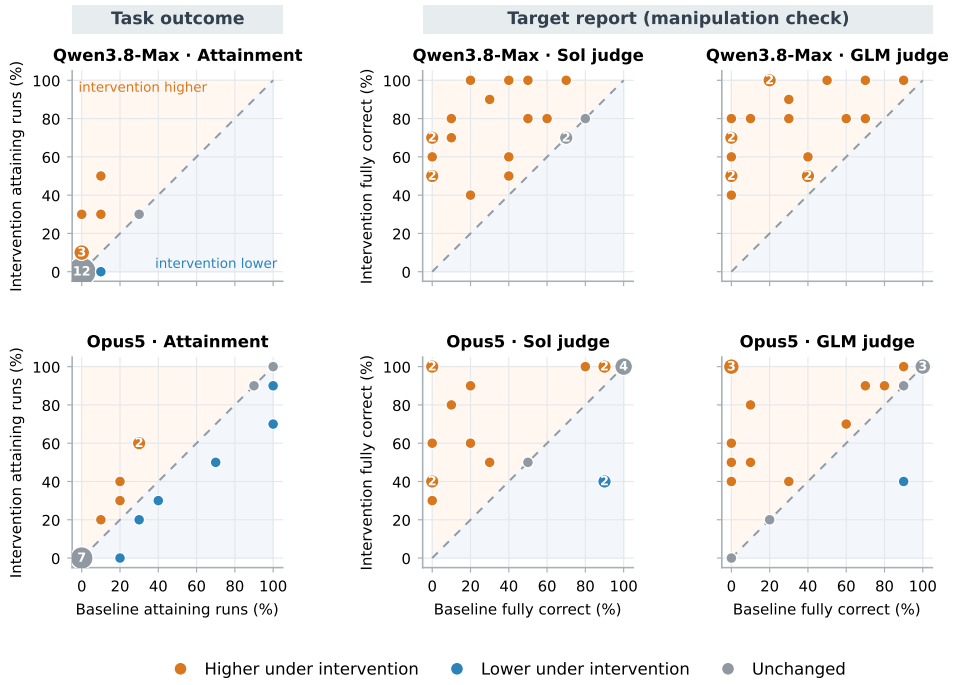


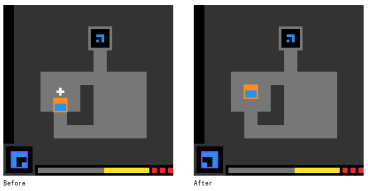
Figure 3: Game-level effects of targeted change supplementation. The horizontal axis shows the baseline condition and the vertical axis the intervention condition. Points above the diagonal therefore indicate a higher value under intervention, points below the diagonal a lower value, and points on the diagonal no observed difference. Orange, blue, and gray encode these three cases; when several games share a coordinate, the point is enlarged and the number inside it gives the number of games. The left column shows level attainment within 15 actions; the middle and right columns show fully correct reporting of the targeted change according to Sol and GLM.

Table 2: Effects of supplying the diagnosed change, baseline → intervention (200 runs per condition). Unsupported target claims: first reports containing an unsupported claim about the target mechanic; linked share: claims linked to a GT event among linked, unsupported, and indeterminate claims. The last two columns give the fully correct target-report rate among attaining runs (23 for Qwen3.8-Max, 132 for Opus5, both conditions) and among all runs.

						Correct target report (%)	
Model	Judge	Target report (%)	Attainment (%)	Unsupported target claims	Linked share (%)	Among attaining	Among all runs
Qwen3.8-Max	Sol	33.0 → 73.5	3.0 → 8.5	15 → 4	78 → 90	87	53
Qwen3.8-Max	GLM	28.5 → 74.5	3.0 → 8.5	1 → 3	68 → 89	87	52
Opus5	Sol	48.5 → 74.0	33.0 → 33.0	19 → 14	79 → 79	64	61
Opus5	GLM	42.5 → 71.0	33.0 → 33.0	10 → 3	81 → 92	56	57

runs, 64% (Sol) and 56% (GLM) had, matching the 61% and 57% of all its runs. Of Opus5's 66 attaining baseline runs, 30 (Sol) and 34 (GLM) began with an inaccurate target report. Details are in Appendix I.

Answer to RQ2 The intervention separated points where supplying the diagnosed change improved progress from points where it did not. For Qwen3.8-Max, supplying the diagnosed change improved both the report and, in aggregate, attainment, consistent with the diagnosed error having limited progress at some of the tested points. For Opus5, report improvements and attainment gains rarely coincided: in the 13 games where the report improved, aggregate attainment did not rise (Sol:



Baseline report (excerpt): "The Up move succeeded: the piece now sits at rows 30–34, cols 19–23 … both side panels flashed with 0 borders (confirmation) and the bottom-left mini-map added a filled 9 cell at rows 59–60, cols 7–8 …" (the rotation is not stated, although a change in the bottom-left display is described).

Supplied change: "The displayed shape at (3,55) changed orientation from 270° to 0° (90° clockwise)."

Intervention report (excerpt): "… it rotated the shape in the bottom-left panel (event confirms 270°→0°) … an exact match for the target pattern … the next move should be to confirm/submit rather than rotate further."

Attainment at this point: 3/10 → 6/10.

Figure 4: One common-rule intervention point on real screens (Opus5). Left: before and after the action; the player moves up and the piece display at the bottom left rotates. Right: first resumed reports from one baseline and one intervention continuation, and the supplied message.

42/130 to 37/130; GLM: 38/130 to 35/130), although it rose in one (Sol) and two (GLM) of these games. There, improving target-report accuracy was not sufficient.

5 Discussion and Limitations

The intervention shows that correcting a report and improving task progress are distinct outcomes. For Qwen3.8-Max, more accurate target reports were accompanied by higher attainment. For Opus5, the two rarely occurred together, and at many selected points the baseline agent no longer made the reporting error. Because the RQ1 scores measure what the agent writes rather than how it uses the information, these results do not show what limited Opus5. They show that whether a reporting error limits progress depends on the model and the decision point, and must be tested by following execution after the correction.

The diagnosis is based on the agent's written reports, which may omit information that the model represents internally (Turpin et al., 2023; Lanham et al., 2023). It also does not measure the understanding of a single static screen, nor does it separate errors in reading a state, comparing states, or retaining past information. Correcting one report therefore provides only part of what execution needs; the agent must still infer the goal and rules, choose actions, and process later observations, and when the report improves but attainment does not, the results do not identify which of these steps limits execution. Additional comparisons at other decision points, reported in the appendix, show the same pattern.

The message both provides information and directs attention to the named object; separating the two requires controls that vary its content. In two Opus5 games, attainment increased (3/10 to 6/10 and 2/10 to 4/10) although baseline reports were already correct; the message remained in the agent's history for the first steps, so its effect need not act through the first report alone.

Finally, the evaluation depends on GT and on the judges' decisions. Agreement between the two judges and with a human evaluator supports the judges' decisions, and the GT rules were verified against rendered screens for each mechanic; GT covers the evaluated mechanics only. Our evidence is also limited to the studied games, observation format, models, and continuation budget.

6 Conclusion

We developed a framework for evaluating LLM agents' perception of screen changes in interactive games with unknown rules, instantiated on ARC-AGI-3. Comparing actual changes with reports used for action selection identified unreported changes, entity or content errors, and unsupported claims, whose patterns differed across models and game mechanics. Supplying the diagnosed change improved target reports in both tested models. Level attainment rose for Qwen3.8-Max (3.0% to 8.5%), consistent with the diagnosed error having limited progress at some of the tested points. For Opus5, attainment rose in some games and fell in others; in the games where target-report accuracy improved, aggregate attainment did not rise, so improving it was not sufficient for progress there. Measuring change reports separately from task outcomes, and then supplying the diagnosed change, tests which diagnosed errors limit progress; task outcomes alone cannot. Extending the framework to image inputs, other environments, and additional models is future work.

AI USE STATEMENT

LLMs were used to assist with manuscript editing, organization of materials, and code development. The authors are responsible for research decisions and the final content. Generated text, code, and citations are checked by the authors before being used to support research claims.

ETHICS STATEMENT

This study primarily evaluates public game environments and model responses. Audits of GT and judgments serve to verify the consistency of research data. For event-side semantic judgments, we aggregate 200 eligible blinded judgments by one evaluator who did not author the evaluation rules and report agreement with the judges (Appendix F.3).

REPRODUCIBILITY STATEMENT

The appendices provide scoring definitions, exclusion reasons, execution settings, evaluation-prompt structure and applied versions, model outputs, denominators, and additional analyses. The change-extraction rules, the evaluation prompts actually used, the intervention messages, the scoring, intervention, and plotting code, and the execution logs will be provided to reviewers and area chairs through an anonymized repository during the discussion phase, and released publicly upon acceptance. The supplementary material submitted with this paper contains the structured results and traces referenced in the appendices. Correspondence between RQ1 data versions and original records is given in Appendix C.2; the common-rule RQ2 protocol and completion status are given in Appendices I.1 and I.2, while exploratory comparisons, their intervention messages, and execution logs are given in Appendix I.3 and the following subsections.

REFERENCES

Alibaba Cloud. Qwen3.8-Max api documentation. Model Studio documentation, 2026. URL https://www.alibabacloud.com/help/en/model-studio/qwen3-8-max.

Anthropic. Claude Opus 5. Anthropic Research, 2026. URL https://www.anthropic.com/research/claude-opus-5.

ARC Prize Foundation. ARC-AGI-3-Agents: Official agent implementation. https://github.com/arcprize/ARC-AGI-3-Agents, 2026a. Accessed September 21, 2026.

ARC Prize Foundation. ARC-AGI-3: A New Challenge for Frontier Agentic Intelligence. arXiv preprint arXiv:2603.24621, 2026b. URL https://arxiv.org/abs/2603.24621.

Mohsen Arjmandi. Sensi: Learn One Thing at a Time—Curriculum-Based Test-Time Learning for LLM Game Agents. arXiv preprint arXiv:2603.17683, 2026. URL https://arxiv.org/abs/2603.17683.

Seoyoung Choi, Minseok Ko, Hyunseok Lee, Kunwoong Kim, Woomin Song, Chanseok Jeon, and Jinwoo Shin. Naive Visual Memory is Not Enough: A Failure-Mode Study of GUI Agents. arXiv preprint arXiv:2606.14106, 2026. URL https://arxiv.org/abs/2606.14106.

David Courtis, Wenhao Li, and Scott Sanner. OPINE-World: Programmatic world modeling with ontology-error-prioritized interactive exploration for ARC-AGI-3, 2026. URL https://arxiv.org/abs/2607.01531.

Google DeepMind. Gemma 4 model card. Google AI for Developers, 2026. URL https://ai.google.dev/gemma/docs/core/model_card_4.

Lanxiang Hu, Mingjia Huo, Yuxuan Zhang, Haoyang Yu, Eric P. Xing, Ion Stoica, Tajana Rosing, Haojian Jin, and Hao Zhang. lmgame-Bench: How Good are LLMs at Playing Games? In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/83a4ea71b13bc86308a2bd0b5e07fb61-Abstract-Conference.html.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998. doi: 10.1016/S0004-3702(98)00023-X. URL https://www.sciencedirect.com/science/article/pii/S000437029800023X.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, et al. Measuring faithfulness in chain-of-thought reasoning, 2023. URL https://arxiv.org/abs/2307.13702.

Jens Lehmann, Andrei Aioanei, and Sahar Vahdati. Tycho: Active abstraction with programmatic world models for ARC-AGI-3, 2026. URL https://arxiv.org/abs/2607.28287.

Emmy Liu, Varun Gangal, Michael Yu, Zhuofu Tao, Karan Singh, Sachin Kumar, and Steven Y. Feng. HalluWorld: A Controlled Benchmark for Hallucination via Reference World Models. arXiv preprint arXiv:2605.19341, 2026. URL https://arxiv.org/abs/2605.19341.

Davide Paglieri, Bartłomiej Cupiał, Samuel Coward, Ulyana Piterbarg, Maciej Wołczyk, Akbir Khan, Eduardo Pignatelli, Łukasz Kuciński, Lerrel Pinto, Rob Fergus, Jakob Foerster, Jack Parker-Holder, and Tim Rocktaeschel. BALROG: Benchmarking Agentic LLM and VLM Reasoning On Games. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/f0b1515be276f6ba82b4f2b25e50bef0-Abstract-Conference.html.

Qwen Team. Qwen3.8-27B-FP8 model card. Hugging Face model documentation, 2026. URL https://huggingface.co/Qwen/Qwen3.8-27B-FP8.

Ruchit Rawal, Reza Shirkavand, Heng Huang, Gowthami Somepalli, and Tom Goldstein. ARGUS: Hallucination and Omission Evaluation in Video-LLMs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20280–20290, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Rawal_ARGUS_Hallucination_and_Omission_Evaluation_in_Video-LLMs_ICCV_2025_paper.html.

Ronald A. Rensink. Change detection. *Annual Review of Psychology*, 53:245–277, 2002. doi: 10.1146/annurev.psych.53.100901.135125.

Evgenii Rudakov, Jonathan Shock, and Benjamin Ultan Cowley. Graph-based exploration for ARC-AGI-3 interactive reasoning tasks, 2026. URL https://arxiv.org/abs/2512.24156.

Alexy Skoutnev, Kirill Acharya, Gaston Longhitano, Madeleine Udell, Kevin Ellis, and Iddo Drori. Twin: Playing an unknown game with a test-time digital twin, 2026. URL https://arxiv.org/abs/2608.14490.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-3275. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/ed3fea9033a80fea1376299fa7863f4a-Abstract.html.

Kangrui Wang, Pingyue Zhang, Zihan Wang, Yaning Gao, Linjie Li, Qineng Wang, Hanyang Chen, Yiping Lu, Zhengyuan Yang, Lijuan Wang, Ranjay Krishna, Jiajun Wu, Fei-Fei Li, Yejin Choi, and Manling Li. VAGEN: Reinforcing World Model Reasoning for Multi-Turn VLM Agents. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-5754. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/fc6688d75adde86b9df910769c1d02e3-Abstract-Conference.html.

Xinhe Wang, Jin Huang, Xingjian Zhang, Tianhao Wang, and Jiaqi W. Ma. Your Reasoning Benchmark May Not Test Reasoning: Revealing Perception Bottleneck in Abstract Reasoning Benchmarks. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 18113–18124. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.826. URL https://aclanthology.org/2026.acl-long.826/.

Bo Wen, Chen Wang, and Erhan Bilal. How Modality Shapes Perception and Reasoning: A Study of Error Propagation in ARC-AGI. arXiv preprint arXiv:2511.15717, 2025. URL https://arxiv.org/abs/2511.15717.

Rui Yang, Hanyang Chen, Junyu Zhang, Mark Zhao, Cheng Qian, Kangrui Wang, Qineng Wang, Teja Venkat Koripella, Marziyeh Movahedi, Manling Li, Heng Ji, Huan Zhang, and Tong Zhang. EmbodiedBench: Comprehensive Benchmarking Multi-modal Large Language Models for Vision-Driven Embodied Agents. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 70576–70631. PMLR, 2025. URL https://proceedings.mlr.press/v267/yang25f.html.

Lance Ying, Katherine M. Collins, Prafull Sharma, Cedric Colas, Kaiya Ivy Zhao, Adrian Weller, Zenna Tavares, Phillip Isola, Samuel J. Gershman, Jacob D. Andreas, Thomas L. Griffiths, Francois Chollet, Kelsey R. Allen, and Joshua B. Tenenbaum. Assessing Adaptive World Models in Machines with Novel Games. arXiv preprint arXiv:2507.12821, 2025. URL https://arxiv.org/abs/2507.12821. arXiv:2507.12821.

Oussama Zenkri and Oliver Brock. Probing Embodied LLMs: When Higher Observation Fidelity Hurts Problem Solving. arXiv preprint arXiv:2605.20072, 2026. URL https://arxiv.org/abs/2605.20072.

Alex L. Zhang, Thomas L. Griffiths, Karthik R. Narasimhan, and Ofir Press. VideoGameBench: Can Vision-Language Models complete popular video games? arXiv preprint arXiv:2505.18134, 2025a. URL https://arxiv.org/abs/2505.18134.

Pingyue Zhang, Zihan Huang, Yue Wang, Jieyu Zhang, Letian Xue, Zihan Wang, Qineng Wang, Keshigeyan Chandrasegaran, Ruohan Zhang, Yejin Choi, Ranjay Krishna, Jiajun Wu, Li Fei-Fei, and Manling Li. Theory of Space: Can Foundation Models Construct Spatial Beliefs through Active Exploration? In *International Conference on Learning Representations*, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/dd9c5ce8803e1898d438e636fbae0236-Abstract-Conference.html.

Weichen Zhang, Yiyou Sun, Pohao Huang, Jiayue Pu, Heyue Lin, and Dawn Song. MIRAGE-Bench: LLM agent is hallucinating and where to find them. *arXiv preprint arXiv:2507.21017*, 2025b.

Weijia Zhang, Kunlun Zhu, Zeyi Liu, Yinting Chen, Tianyi Ma, Jiateng Liu, Jiaxun Zhang, Bingxuan Li, Xiangru Tang, Heng Ji, and Jiaxuan You. CUADebug: Diagnosing and Repairing Computer-Use Agent Failures. arXiv preprint arXiv:2608.02643, 2026b. URL https://arxiv.org/abs/2608.02643.

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*, volume 36, pp. 46595–46623, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html.

Kunlun Zhu, Zijia Liu, Bingxuan Li, Muxin Tian, Yingxuan Yang, Jiaxun Zhang, Pengrui Han, Qipeng Xie, Fuyang Cui, Weijia Zhang, Xiaoteng Ma, Xiaodong Yu, Gowtham Ramesh, Jialian Wu, Zicheng Liu, Pan Lu, James Zou, and Jiaxuan You. Where LLM Agents Fail and How They can Learn From Failures. arXiv preprint arXiv:2509.25370, 2025. URL https://arxiv.org/abs/2509.25370.

Guide to the appendix. The appendix is ordered for reading: the evaluated games and log notation (Appendix A); examples of agent responses that report changes correctly or incorrectly (Appendix B); construction of the reference records (Appendix C); execution settings (Appendix D); scoring rules, the evaluation prompt, and the illustrated guide to evaluated entities (Appendix E); scoring reliability, including agreement between judges and human validation (Appendix F); RQ1 results (Appendix G) and sensitivity analyses (Appendix H); RQ2 interventions, results, and interpretation, covering both the common-rule protocol (Appendices I.1 and I.2) and the exploratory

comparisons (Appendix I.3); representative LS20 post-intervention traces (Appendix J); and the relationship to prior measurement approaches and the electronic evidence index. Full visual atlases and original reports are retained as supporting evidence.

A Evaluated Games and Reading the Logs

Game IDs are shown in this font throughout the appendix (e.g., LS20). Figure 5 shows the initial screen of each of the 20 evaluated games of the ARC-AGI-3 Public Demo Set; the five games not evaluated are listed in Table 10.



Figure 5: Initial screens of the 20 evaluated games, rendered from the integer grids presented to the agents.

A.1 Integer Observations and Reading the Logs

The board presented to the model is a two-dimensional array in which integers from 0 to 15 represent cell colors. These values are neither object identifiers nor numerals drawn on the screen. Figures render the same array using the colors in Table 3. For example, `12/9 stacks` in the LS20 log in Appendix B refers to stacked regions made of orange (12) and blue (9) cells. The slash is part of the model's original wording and does not indicate division.

Reading positions, time points, and actions Coordinates in GT and this paper are $(x, y) = (\text{column}, \text{row})$, with $(0, 0)$ at the upper left; x increases to the right and y downward. Lengths

Table 3: Mapping between the integer grid and figure colors. The environment's 16-color palette is shown. Color names are given in English to match the logs.

Integer	Swatch	Color name
0		white
1		off-white
2		light gray
3		gray
4		dark gray
5		black
6		magenta
7		pink
8		red
9		blue
10		light blue
11		yellow
12		orange
13		maroon
14		green
15		purple

and displacement magnitudes are given in grid cells. When logs specify positions using `rows` and `cols`, these are read as rows and columns, respectively. Even when the model confuses coordinates or entities, quotations are left unchanged and compared with GT. When one action returns multiple frames, they form a temporal sequence within the same observation, not separate actions.

Table 4 lists run-reference notation and action IDs.

Table 4: Notation in execution records and logs. Directions indicate the action type; what moves depends on the game.

Notation	Meaning
Original trial	Trial number of the run producing the pre-intervention trajectory.
Observation	Observation number within the original trial's trajectory.
Turn (post-intervention)	Action order after resumption, starting at 1.
`ACTION1`	Up action.
`ACTION2`	Down action.
`ACTION3`	Left action.
`ACTION4`	Right action.
`ACTION5`	Simple action.
`ACTION6`	Click at specified coordinates.
`ACTION7`	Undo.
`RESET`	Reset.

Distinguishing reports from reference records A Reasoning Log contains the agent's observations, hypotheses, and action policy, not a reference explanation. Text inside quotation boxes is reproduced verbatim. Before-and-after boards and GT provide the basis for checking colors, positions, and changes. In GT, `position` denotes location, `bbox` denotes region width and height, and `level` denotes the level to which the event belongs. Merely mentioning an entity in the present tense does not report a change from the preceding state. To make this distinction clear, each example explains the actual change and the relevant log passage outside the quotation box.

B EXAMPLES OF AGENT RESPONSES: CORRECT AND INCORRECT CHANGE REPORTS

Unlike the constructed illustration in Figure 1, the examples in this appendix reproduce actual screens and reports. Appendix B.1 shows an observation in which the model correctly reports the movement. The following subsections show cases in which an entity is specifically described while its change is explicitly denied: first a representative LS20 execution, showing in detail how entity descriptions and denial of changes coexist in the same log, and then examples from other games aligned with before-and-after screens and Reasoning Logs. Appendix B.12 shows a case in which the current state is described while changes are omitted. Each box reproduces the model's full output verbatim; explanations and the actual next action appear outside the box.

B.1 EXACT-INPUT OBSERVATION EXAMPLE: A CORRECTLY REPORTED MOVEMENT

The following input example corresponds to the 15th observation of replicate s1 for Opus5 on AR25 under ordinary observation instructions. The saved observation request contains the previous and current screens. We retain that request and response here as a supplementary AR25 example. The full response to the observation is shown below. In this case, the model correctly reports the movement.

AR25, Opus5, natural original trial 1, observation 15

```
The block moved another 3 rows: the 4 now spans rows 30-38 with the
mirrored 5 template at rows
30-38, columns unchanged. The right-edge counter reads 14 consumed (5s
rows 0-13), confirming one
tick per action with plenty of margin remaining.

The 11 target still sits at rows 45-53, so 15 rows separate the 4's
top edge from its landing spot --
exactly 5 more ACTION2 presses. The descent lane stays clear of any 10
pillar (columns 30-32) or
other obstruction.

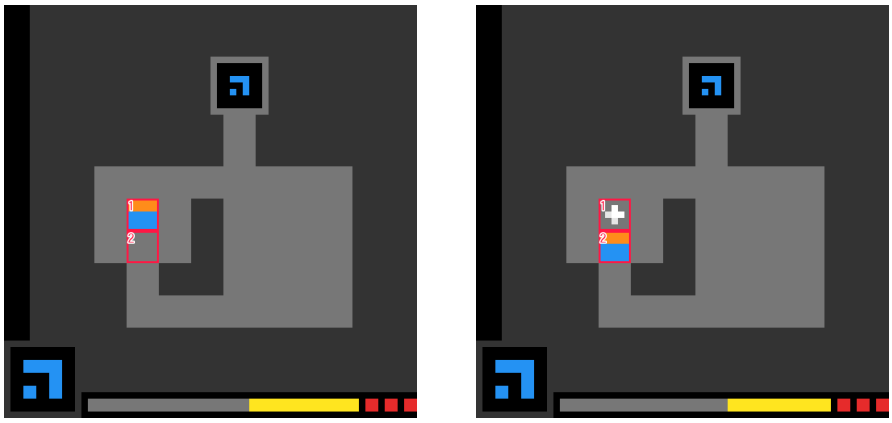
Next: ACTION2.
```

B.2 LS20: A REPRESENTATIVE CASE OF DESCRIBING AN ENTITY WHILE DENYING ITS MOVEMENT

The square object in Figure 6 moved downward by five cells in response to the preceding ACTION2. The log specifically describes the post-movement positions of the orange and blue block while stating that the block is unchanged. From this interpretation, it infers that the block is a palette or progress meter and plans to treat a separate cursor as the controllable entity.

LS20, Opus5, original trial 2, observation 36: full Reasoning Log

```
The board changed meaningfully: a new small cursor/piece appeared at
rows 31-33, cols 20-22 (a plus
shape of 0s with 1s marking orientation), and the bottom 11-bar shrank
by one cell (now starting at
col 39), which looks like a move/energy counter ticking down. The
target panel at top-right (rows
9-15, cols 33-39) still shows the goal pattern with 9s forming a hook
shape, and the 12/9 stacks at
rows 35-39, cols 19-23 are unchanged, so those likely represent the
palette or progress meter. My
ACTION2 moved/created this cursor downward, so directional keys steer
it inside the large 3-field;
```



Before action After action (screen presented to the model)

Figure 6: LS20, Opus5, original trial 2, observation 36. The 5×5 object in the red box moved from $(19, 30)$ to $(19, 35)$. The log should be read by distinguishing the white cross at the previous position from the orange and blue block after movement.

```
next I should keep guiding it toward the 12/9 target region below
while watching whether the 11-bar
keeps shrinking, since that limits how many actions I can spend.
```

Preceding action: ACTION2. **Recorded next action:** ACTION2.

In the saved scores for this scene, both judges matched the description of the cursor to the player's movement and marked reporting, entity, and content correct. However, the log describes the moved orange and blue block itself as unchanged. This discrepancy shows the need to check which entity is being described against before-and-after screens, in addition to whether a change is mentioned.

B.3 Games and scenes exhibiting the same discrepancy

In RQ1, all 20 games contained GT changes that both GLM and Sol judged unreported. Inspection of saved logs identified at least 15 scenes across eight games in which an entity was specifically described while its change was explicitly denied. Table 5 lists the confirmed scene counts and representative examples.

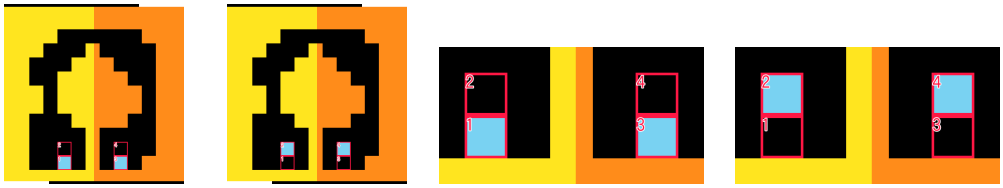
Table 5: Confirmed examples of specific entity descriptions co-occurring with denial of changes to the same entity. Counts are observation scenes. One example per game is shown with images and a full log.

Game	Confirmed count	Representative model	Original trial / observation	Actual entity change
LS20	1	Opus5	Trial 2 / obs. 36	Downward movement of the square object
M0R0	5	Qwen3.8-Max	Trial 5 / obs. 39	Upward movement of the left and right blocks
DC22	1	Qwen3.8-27B	Trial 5 / obs. 33	Upward movement of the small object
SC25	1	Opus5	Trial 1 / obs. 30	Upward movement of the block
WA30	4	Qwen3.8-Max	Trial 3 / obs. 37	Downward movement of the target object
FT09	1	Qwen3.8-27B	Trial 2 / obs. 37	Shortening of the bottom bar
VC33	1	Qwen3.8-Max	Trial 3 / obs. 28	Shortening of the top bar
RE86	1	Gemma4 31B	Trial 5 / obs. 2	Upward movement of the cross-shaped piece
Total	15	8 games, 8 representative examples		

Verification checked that a log for the same observation contained both a specific entity description and an explicit denial of its change, while the corresponding before-and-after screens and GT showed an actual change. Each example below pairs enlarged views of the entity with full screens. Coordinates are columns and rows with the origin at the upper left; coordinate expressions in logs are retained verbatim. Original text and judgments for the 15 confirmed scenes, and integer grids, all presented frames, and execution records for the eight representative examples, are included in the electronic supplement.

B.4 M0R0: DENYING MOVEMENT OF BOTH LEFT AND RIGHT ENTITIES

Both the left and right 5×5 blocks moved upward by five cells. The log describes the rows and columns of both blocks after movement while stating that they did not move in response to the Up action, and plans to try moving down.



Full screen: before | Full screen: after | Entity: before | Entity: after

Figure 7: M0R0, Qwen3.8-Max, original trial 5, observation 39. The left and right blocks moved from $(19, 54) \rightarrow (19, 49)$ and $(39, 54) \rightarrow (39, 49)$, respectively.

M0R0, Qwen3.8-Max, original trial 5, observation 39: full Reasoning Log

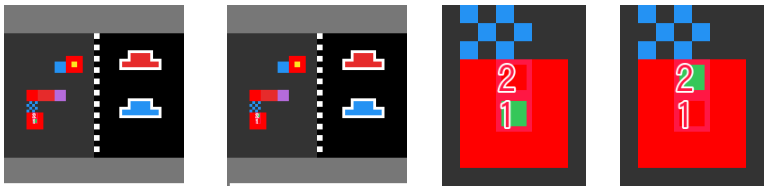
```
The Up action (ACTION1) did not move the 10-blocks vertically - they
remain at rows 49-53, with the
left block at cols 19-23 and right block at cols 39-43. The
bottom-left exit (row 63, cols 0-15) and
top-right exit (row 0, cols 48-63) are the targets. Since Up didn't
work, I'll try Down to see if
the blocks can move vertically toward the bottom exit.

{"action": "ACTION2", "x": 0, "y": 0}
```

Preceding action: ACTION1. **Recorded next action:** ACTION1 (different from the proposal in the log).

B.5 DC22: DENYING MOVEMENT AND JUDGING THE ACTION BLOCKED

The 2×2 entity moved upward by two cells. The log describes its values, rows, and columns while denying movement, concludes that directional actions are blocked, and proposes another action.



Full screen: before | Full screen: after | Entity: before | Entity: after

Figure 8: DC22, Qwen3.8-27B, original trial 5, observation 33. The entity moved from $(10, 40) \rightarrow (10, 38)$.

DC22, Qwen3.8-27B, original trial 5, observation 33: full Reasoning Log

```
The player (14) at rows 39-40, cols 10-11 is completely surrounded by
2-blocks on all four sides and
```

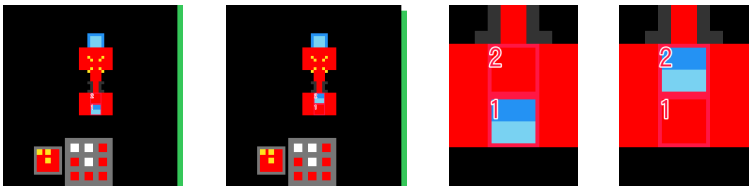
```
hasn't moved despite the Up command. The frame is essentially
identical to before. The only change
is the bottom-left corner cell shifting from 0 to 3, suggesting a very
slow environmental tick.
Since directional movement is blocked, I'll try pressing Enter/Space
(ACTION5) to see if it triggers
an interaction, start, or escape mechanic.

{"action": "ACTION5", "x": 10, "y": 39}
```

Preceding action: ACTION1. **Recorded next action:** ACTION3 (different from the proposal in the log).

B.6 SC25: DENYING UPWARD MOVEMENT AND TRYING HORIZONTAL MOVEMENT

The 4×4 block of 9/10 cells moved upward by four cells. The log locates the block in its post-movement region while concluding that nothing moved, then proceeds to try a horizontal action.



Full screen: before Full screen: after Entity: before Entity: after

Figure 9: SC25, Opus5, original trial 1, observation 30. The entity moved from $(31, 35) \rightarrow (31, 31)$.

SC25, Opus5, original trial 1, observation 30: full Reasoning Log

```
The up press cost time (the gauge now shows 2 rows of 0 consumed at
the top) but nothing moved -- the
9/10 block is still parked at rows 32-35, cols 31-34, and the target
chamber is unchanged at rows
10-14.

So vertical input isn't driving this piece, or the pinch at rows 27-30
(walls of 4 leaving only cols
32-33 open) is flatly blocking upward travel. Last level the piece
traveled horizontally with
ACTION3, so lateral movement is likely the active axis here; I may
need to shift the block sideways
out from under the 4-pinch before any ascent is possible.

I'll try ACTION3 to test whether the block responds to left movement.
```

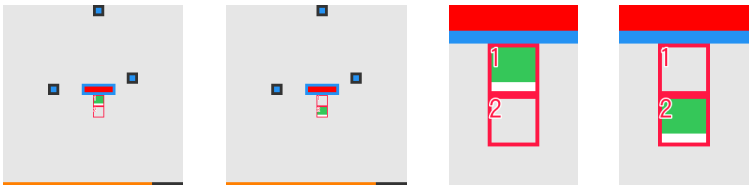
Preceding action: ACTION1. **Recorded next action:** ACTION3.

B.7 WA30: DESCRIBING ORIENTATION WHILE DENYING DOWNWARD MOVEMENT

The 4×4 entity moved downward by four cells. The log describes its rows, columns, and downward-facing configuration while denying downward movement, considers whether movement was blocked or depended on orientation, and tries a Left action.

WA30, Qwen3.8-Max, original trial 3, observation 37: full Reasoning Log

```
The piece (rows 31-34, cols 32-35, with the 0-row at bottom) appears
unchanged after the second
ACTION2 -- it didn't move further down. This suggests either downward
movement is blocked, the piece
needs a different input to continue, or movement is
orientation-dependent (the 0-row faces down).
```



Full screen: before Full screen: after Entity: before Entity: after

Figure 10: WA30, Qwen3.8-Max, original trial 3, observation 37. The entity moved from $(32, 32) \rightarrow (32, 36)$.

```
I'll test horizontal responsiveness with ACTION3 (left) to see if the
piece can still be steered,
and to probe whether the goal might be to navigate it toward one of
the boxes or the 4-region on the
floor's right side.

{"action": "ACTION3", "x": 0, "y": 0}
```

Preceding action: ACTION2. **Recorded next action:** ACTION3.

B.8 WA30: A NATURAL ONLINE MISMATCH, COMPLETE RESPONSE

The following is the full response body to the second observation in the first replicate (original trial 1) of Opus5 on WA30 under the ordinary observation instructions. The corresponding preceding action is ACTION1, and GT indicates that a 4×4 entity moved from $(32, 48)$ to $(32, 44)$. The preceding action and response were aligned using the temporal order of the saved observations.

WA30, Opus5, original trial 1, observation 2

```
ACTION1 (Up) produced an identical frame -- nothing moved, so either
the directional keys are not the
control scheme, or the controlled entity was blocked/needs prior
selection. The three 4×4 blocks
(rows 24-27 @ cols 44-47, rows 28-31 @ cols 16-19, rows 36-39 @ cols
32-35), the wide 9/2 rectangle
(rows 28-31, cols 28-39), the 0/14 stack (rows 48-51, cols 32-35), and
the 7 floor row are all
unchanged.

Since a plain keypress had zero effect, the most informative next
probe is a click (ACTION6) to
select an object -- the 0/14 stack at cols 32-35 looks like a
color/palette or slot indicator, and
the block at cols 32-35 (rows 36-39) is directly aligned above it,
suggesting a select-then-place
mechanic. I'll click on that aligned 4×4 block's center to test
whether clicking selects/moves
objects; if the frame changes there, I'll know the interaction model
and can then place blocks into
the 32-35 target column band.
```

The log states that no movement occurred and, based on that interpretation, proposes exploration through clicks. Thus, a description inconsistent with the actual change is also reflected in the policy for what to try next. This example motivates examining the correspondence between change reports and subsequent decisions and proceeding to RQ2, which supplies the same change information.

B.9 FT09: COUNTING A BAR'S COMPOSITION WHILE DENYING ITS SHORTENING

The bottom bar shortened by two cells from its right end. The log counts the values and column ranges composing the bar while denying any change, concludes that directional keys have no effect, and proposes a click.



Full screen: before | Full screen: after | Entity: before | Entity: after

Figure 11: FT09, Qwen3.8-27B, original trial 2, observation 37. The bottom bar's length changed from 12 to 10 cells.

FT09, Qwen3.8-27B, original trial 2, observation 37: full Reasoning Log

```
Looking at the frame carefully, I notice that the bottom bar is
unchanged (14 "12" cells at columns
0-13, followed by 46 "11" cells), and the entire grid pattern is
identical to the previous frame.
This means the A/Left key (ACTION3) had no visible effect, just as
W/Up (ACTION1) apparently also
had no effect -- my earlier observation about a cursor shift was
incorrect. Both directional keys
produced no change. Since simple key presses aren't triggering any
game response, I should try
clicking directly on the grid content -- perhaps on the bottom bar's
"12" region or on one of the
pattern blocks -- to see if a click interaction is required to
progress the game.

{"action": "ACTION6", "x": 7, "y": 59}
```

Preceding action: ACTION3. **Recorded next action:** ACTION6 (7, 59).

B.10 VC33: DESCRIBING A BAR'S BOUNDARY WHILE DENYING ITS CHANGE

The top bar shortened by two cells from its right end. The log describes the bar's boundary position while judging it identical to the previous screen, considers whether the previously clicked location has stopped working, and selects a different location.



Full screen: before | Full screen: after | Entity: before | Entity: after

Figure 12: VC33, Qwen3.8-Max, original trial 3, observation 28. The top bar's length changed from 40 to 38 cells.

VC33, Qwen3.8-Max, original trial 3, observation 28: full Reasoning Log

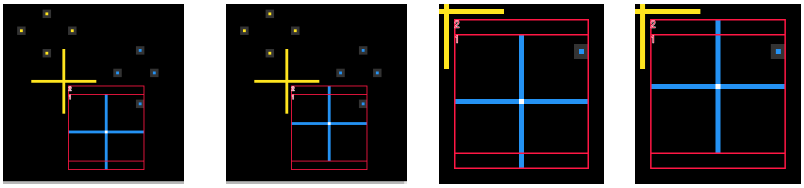
```
The progress bar appears to have stopped advancing -- the boundary
between 7s and 4s remains at
column 38, unchanged from the previous frame despite clicking the same
gate position. The rest of
```

```
the grid is identical. This suggests the bar reached a checkpoint or
the gate at (38,29) is no
longer effective. I'll try clicking on the 11,11 elements in the lower
structure around (50,43),
which may be the next interactive target to progress further.
```

Preceding action: ACTION6. **Recorded next action:** ACTION6 (50, 43).

B.11 RE86: DESCRIBING A CROSS-SHAPED ENTITY'S POSITION WHILE DENYING UPWARD MOVEMENT

The cross-shaped piece moved upward by three cells. The log gives coordinates for the horizontal and vertical intersection of value-9 cells while stating that the preceding Up action did not change its position, and plans to repeat the Up action with different input parameters.



Full screen: before | Full screen: after | Entity: before | Entity: after

Figure 13: RE86, Gemma4 31B, original trial 5, observation 2. The upper-left position of the cross-shaped piece moved from $(23, 32) \rightarrow (23, 29)$.

RE86, Gemma4 31B, original trial 5, observation 2: full Reasoning Log

```
The agent is currently at the intersection of a horizontal and
vertical track of value 9s,
specifically at coordinate (44, 36). The previous attempt to move UP
using ACTION1 did not change
the agent's position. I will attempt to move UP again, ensuring the
action is called without
unnecessary parameters, to proceed along the vertical track toward the
upper section of the map.

{"action": "ACTION1"}
```

Preceding action: ACTION1. **Recorded next action:** ACTION1.

B.12 CURRENT-STATE DESCRIPTIONS WITH OMITTED CHANGES

In connection with the preceding examples, we present a saved example in which mentions of the current state coexist with unreported changes to the entity.

In the following natural Reasoning Log, a detailed description of the current state coexists with unreported actual movements. At observation 23 of original trial 4 for Qwen3.8-Max on AR25, the preceding Up action moved two 9×9 objects upward by three cells each, and the step bar at the right edge decreased by one cell. The model specifically described the central pillar, several L-shaped objects, the floor, the right wall, and the rows of 0s in the current screen, but reported only the step-bar change and neither object movement. GLM-5.2 and GPT-5.6 Sol returned identical judgments for these three events.

AR25, Qwen3.8-Max, original trial 4, observation 23

```
The grid is nearly static: a full-height column of 10s splits the
field in half, with an L-shaped
"5" block (upper-left, containing 0 dots), a mirrored "4" block
(upper-right), and an "11" L-shape
```

Table 6: GT events and reporting judgments for AR25, Qwen3.8-Max, original trial 4, observation 23.

Game mechanic	Actual change	Both judges
piece	$(18, 15) \rightarrow (18, 12)$: a 9×9 object moves upward by three cells	Unreported
reflection	$(36, 15) \rightarrow (36, 12)$: the reflected 9×9 object moves upward by three cells	Unreported
step_bar	The display at the right edge decreases by one cell, from 64 to 63	Fully correct

```
(lower-right), all resting above a floor of 5s and beside a right wall
of 11s. The only detectable
change after my last Up input is that the top-right corner cell (row
0, col 63) flipped from 11 to
5, which suggests inputs are being registered but producing very
subtle effects--possibly a hidden
cursor, selector, or something moving along the wall. The 0 dots
inside the 5-structure form a
diagonal trail (across its top row, then down its right column) and
remain the most likely
interactive targets or indicators. Next I'll press Down to A/B test
whether the corner cell changes
again, confirming input responsiveness before committing to clicks on
the exact 0-dot cells (e.g.,
x=19, y=13).
```

In this example, object movements are omitted while the decrease in the display is reported. A detailed description of the current structure is not equivalent to reporting the changes to each object caused by the preceding action. Evaluation must therefore assess which entities' changes are reflected in the report, in addition to the amount of description of the screen as a whole.

C Ground Truth Construction

C.1 GT Construction Policy and Reproducible Decisions

Table 7: Examples of GT events. Illustrative, not experimental observations.

Change	Identifying the entity	Recorded change content
Movement	Red square on the left	One cell down
Rotation	Shape at the lower left	Orientation from 270° to 0°
Color change	A specified screen region	Post-change color
Flashing	A specified screen region	Occurrence within the presented frames

GT in this study is generated deterministically from environment state transitions by extraction code implemented by the authors, producing reference records of entities and their changes. Following shared extraction rules and game-specific codebooks, the process outputs changes visible on the screens presented to the agent. Judgments are based on the environment implementation and the actual presented screens, independently of the Reasoning Log. The basic policy is to fix visibility, event aggregation, entity identity, and vocabulary consistency so that the same occurrence maps to the same entities, attributes, and events regardless of who processes it.

C.1.1 From sprite definitions to visible events

Sprites in a game have internal attributes such as position, rotation, pixel arrays, and visibility. After rendering accounts for camera position, magnification, transparent regions, and overlap with other objects, the visible regions and attributes become the targets of GT. Internal identifiers are used to track entity continuity, while the output describes changes that appear on screen.

Events are defined in the environment's transition processing, and the frame sequence returned by each action is saved. Continuous movement or display switching of the same entity is aggregated into the event units specified in the codebook. For extraction implementation and rule version v and game-specific codebook $\mathcal{C}$, this correspondence is expressed by the deterministic mapping

$$\Delta_t = \Phi_{v,\mathcal{C}}(s_{t-1}, a_{t-1}, \mathcal{F}_t)$$

where s_{t-1} is the environment state before the action, a_{t-1} is the action, and $\mathcal{F}_t$ is the sequence of frames actually presented. Given the same state, action, frame sequence, and rules, the occurrence of events, their entities, and their attribute values are identical. The mapping $\Phi_{v,\mathcal{C}}$ is the implemented realization of the π/δ decomposition in Section 3 under rule version v and codebook $\mathcal{C}$.

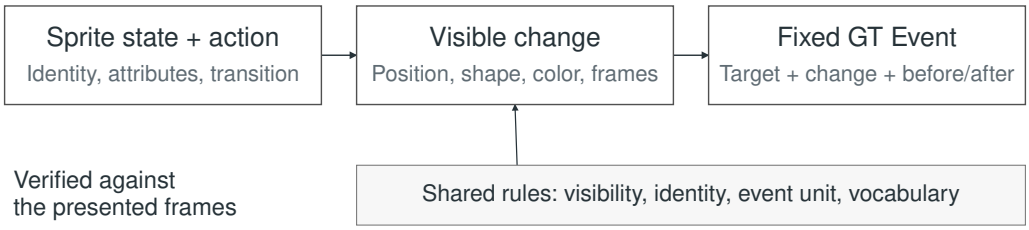


Figure 14: GT generation process. Sprites and state transitions are mapped to entities and attributes appearing in the actual presented frames. GT is determined by applying shared rules for visibility, tracking the same entity, event units, and vocabulary. The Reasoning Log is not an input to this generation process.

C.1.2 Observability and independence of judgments

A fair evaluation of the agent must use reference information that can be verified from the observations presented to it. We therefore define correspondences between internal entities and visible objects, pixel differences and object changes, and frame sequences and individual events. For example, disappearance and appearance before and after movement are aggregated into movement of the same object, while flashing that occurs only between endpoints is recorded from the full presented frame sequence. These correspondences are fixed using the implementation and screens and applied as criteria independent of the logs being scored.

C.1.3 GT construction policies

Table 8 lists the rules for each judgment and their purposes. The set of evaluated entities is specified in the game-specific codebook before scoring, and visible events within that scope are recorded using shared rules. Small changes are handled according to whether a defined attribute of the entity changes, rather than by pixel count alone.

Table 8: Shared policies for constructing GT of screen changes. Entities, time points, and attributes are fixed so that the same reference information can be regenerated from the implementation and presented screens.

Aspect	Rule	Purpose
Visibility and entity inclusion		
Entity set	Fix the evaluated game mechanics and the entities to which events are attributed in the codebook. Within that set, record all object changes that exist in the implementation and can be verified in the observations.	Establish shared criteria for entity selection and coverage.

Continued on the next page

Table 8: GT construction policies (continued)

Aspect	Rule	Purpose
Observation scope	Record changes appearing on screen, excluding changes limited to internal counters, invisible sprites, or identifiers. Associate an occluded state with the time at which it becomes visible and verifiable again.	Limit evaluation to information presented to the model.
Outcome states	Record reachable win, miss, and gameover states once per causal action. Separate outcome-state recording from scoring of visible changes, retaining success or failure without screen changes in the former. Also distinguish non-attainment from missing records.	Avoid conflating task success or failure with changes discernible from the screen.
Event and temporal aggregation		
Event unit	Count the same change to the same entity caused by the same action as one event, and check for omissions and duplicates. Record distinct entities that change simultaneously separately.	Count changes to each entity in consistent units.
Frame sequence	Aggregate continuous movement, rotation, or flashing within an action into event units, also consulting intermediate frames. Merge rendering differences accompanying the same event, but retain changes to defined attributes such as bar length.	Prevent duplication due to frame count and omission of transient or small attribute changes.
Spatial information and attributes		
Entity continuity	Track the same object before and after movement as one entity. Retain its identity through reappearance or changes in color or shape, and distinguish internal identifiers from visible attributes at each time point.	Preserve attribution of changes without splitting movement of one object into disappearance and appearance.
Rendered region	Use values corresponding to actual rendering for position, bounding box, color, shape, angle, and displayed amount. Base bounding boxes on nontransparent regions and clip them at screen edges. Exclude transparent padding and collision regions that are not rendered.	Refer to the size and attributes of the object the model actually saw.
Coordinates and change magnitudes	Use the upper-left corner of the screen as the origin, with columns as x and rows as y. Use coordinates after camera transformations, and assess rotation from the object's center and rendering at each time point. Retain the defined before-and-after values and change magnitudes.	Enable comparison of positions, directions, and magnitudes in a common coordinate system.
Temporal alignment	Fix conventions for completed-level counts, GT level identifiers, observation numbers, and action numbers, and link corresponding actions, presented screens, and events.	Align the action that caused a change with the time the model received it.
Consistency of vocabulary and descriptions		
Naming	Use shared verbs and attribute names across games for the same types of changes. Match color names and display names to current rendering, and use internal sprite names only for tracking.	Reduce wording variation and align names with visible attributes.
Reference descriptions	Fix RQ1 reference descriptions for the same events and attributes, revising them only on the basis of factual discrepancies with the implementation or screens. Map English and Japanese outputs to the same events.	Keep reference content independent of model answers.
Intervention messages	In RQ2, withhold roles, functions, win/loss information, correct actions, and hidden states, and present the entity's visible attributes and changes. Fix retained attributes, before-and-after values, and wording by condition, and save the actual original notification.	Hold assistance about visible changes constant and make comparisons reproducible.

Continued on the next page

Table 8: GT construction policies (continued)

Aspect	Rule	Purpose

C.1.4 Checking reproducibility and correspondence to screens

The shared rules keep entity delineation, time-point selection, and attribute-value reading consistent. Regeneration applies the same rules to the same state, action, and presented frames and checks for identical GT. Comparison with actual screens aligns frame-level pixel differences, entity positions and appearances, and before-and-after event values. These checks separately establish that the same output can be reproduced and that the output corresponds to observable changes.

Table 9: Aspects checked to establish correspondence between extraction rules and presented screens.

Property checked	Correspondence among implementation, records, and screens
Event units	Each entity's visible change is recorded once, and transient changes also correspond to the presented sequence.
Agreement with rendering	Display coordinates, bounding boxes, rotation, colors and display names, and handling of screen edges correspond to actual rendering.
Outcome-state records	Reachable terminal branches correspond to actual attainment records, allowing outcomes to be distinguished from visible changes.
Output consistency	The same entities and attributes are referenced across languages and ordinary versus anonymized descriptions; anonymized descriptions are limited to the specified visible information.
Regeneration	The same conditions produce the same GT, and the implementation and codebook used can be identified.

New types of events are incorporated into the shared rules on the basis of the implementation and presented screens. Revised rules are applied to analogous entities, levels, and branches, while retaining the applied version and the basis for the judgments. Corrections are thus managed as updates to reusable extraction criteria rather than adjustments to individual answers.

C.1.5 Matching reports against independently constructed GT

GT generation and revision are based on implementation code, presented screens, and differences between those frames. Scoring Reasoning Logs against the finalized GT separates reference generation from evaluation of model answers. If GT agrees with the screens but semantic matching to a report is problematic, the matching rules and judge are examined. When GT is revised, the discrepancy with the implementation or screens is recorded as the basis for the revision.

RQ1 reference descriptions and RQ2 anonymized descriptions use the same entities and changes for different purposes. RQ1 matches reports after a run. RQ2 adds the visible change of the target event once to the first observation request after resumption, then tracks subsequent reports, actions, screens, and attainment.

C.2 GT Construction and Game Exclusions

The 25 candidate games are AR25, BP35, CD82, CN04, DC22, FT09, G50T, KA59, LF52, LP85, LS20, M0R0, R11L, RE86, S5I5, SB26, SC25, SK48, SP80, SU15, TN36, TR87, TU93, VC33, and WA30. The five games in Table 10 were excluded because their visible changes could not be reliably converted into entity-specific GT. The remaining 20 games are evaluated.

GT generation extracts candidates from the game server's transition records, then checks the entity and its visibility against screen differences for the same action.

Data and version correspondence The correspondence among observations, GT, applied evaluation prompts, and original judge outputs for the 20 games will be provided as a manifest in the electronic supplement. Counts of scored items and the shared set of games and game mechanics used for model comparisons are given in Appendix G.2.

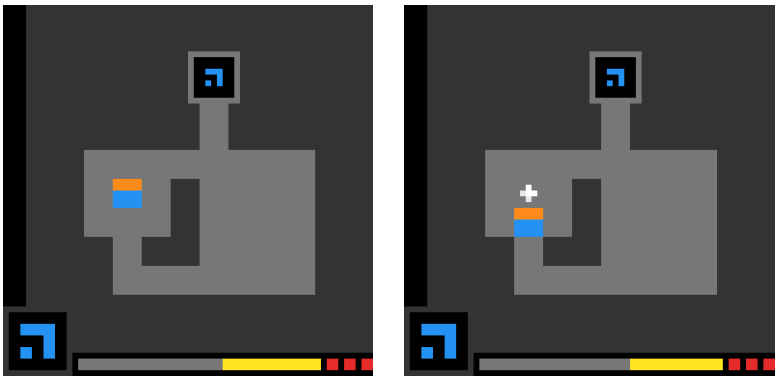
Table 10: The five games excluded from evaluation and the reasons related to GT construction.

Game	Obstacle to GT construction
BP35	The game code differs substantially in structure from the other 23 games, preventing reliable construction of visible events using the shared transition extractor.
LF52	As with BP35, substantial differences in code structure make it difficult to construct GT that aligns screen changes with transition records using the shared procedure.
SP80	Dynamic water behavior prevents reliable tracking of the correspondence between changes and individual entities.
CN04	When three or more joints overlap, on-screen identity and attribution of changes become ambiguous.
G50T	When the main body and its copy overlap, the entity to which a change should be attributed cannot be determined.

Game mechanic selection and version records Screen differences are aggregated into meaningful changes at the granularity of game mechanics defined in the main text. For example, a screen difference in which an object disappears from its previous region and appears in its new region is evaluated as one movement. Candidates are identified at this granularity from game code and visible transitions, and decisions are made to retain, merge, or split them. For each game mechanic, entity identity, required attributes, and visibility conditions are retained in the game-specific evaluation rules. All categories actually encountered in the main analysis and their denominators are retained in the game $\times$ game mechanic atlas; the reasons for excluding the five games are also fixed in Table 10.

C.3 Examples Connecting Screens to Rules: LS20

In the LS20 movement example, mapping the internal entity to the on-screen 5×5 block gives a position change from $(19, 30)$ to $(19, 35)$. The disappearance and appearance in two regions are counted as one downward movement of the same object, rather than as separate events. The actual before-and-after screens and mentions of the entity are shown in Appendix B.2.



Before action After action

Figure 15: Example of fixing entity continuity and rendered attributes. The 5×5 block in LS20 moves downward by five cells as the same entity. GT retains the display coordinates before and after movement without splitting the movement into disappearance and appearance. The model's description is not an input to determining this GT.

In a rotation scene from the same LS20 game, the six presented frames include transient flashing. All presented frames are checked to record this transient event, rather than missing it by examining only the final frame. Rotation is described by its before-and-after angles and is not combined with flashing in the same row. All corresponding frames are shown in Figure 17.

C.3.1 LINKING GAME MECHANICS TO VISIBLE REGIONS

The evaluated LS20 entities are player, piece rotation, and steps bar. Table 11 links them to their visible changes. Role names are explanatory labels for readers and are not included verbatim in intervention messages. For example, player is described in the intervention as a square object based on its appearance. The colored grids shown in the screens visualize the stored integer grids.

Table 11: Correspondence between LS20 evaluation rows and visible changes.

Game mechanic	Change verified on screen
player	The 5×5 square object moves within the board.
piece rotation	The blue shape at the lower left changes orientation.
steps bar	The yellow display along the bottom edge becomes shorter.

C.3.2 OBSERVATION 34: ROTATION

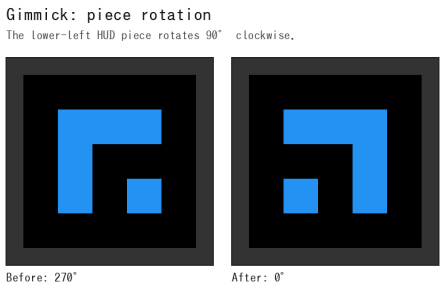
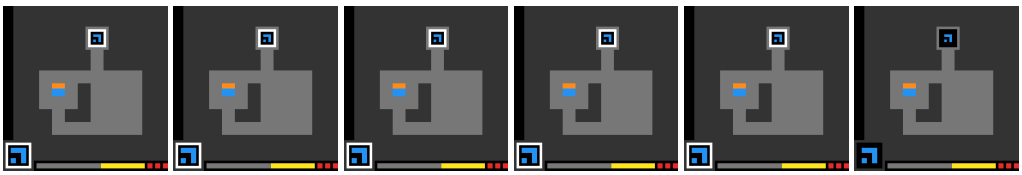


Figure 16: piece rotation. The lower-left shape rotates 90° clockwise. GT describes this as 270° → 0°.

C.3.3 ALL FRAMES RETURNED BY THE ROTATION ACTION

Figure 17 shows all six frames returned at observation 34 in presentation order. No temporal spacing between frames is assumed here; only their order is shown.



Frames 1–6, left to right

Figure 17: All six frames at observation 34 from the same original LS20 run with Opus5. Numbers indicate presentation order, not separate agent actions.

C.3.4 OBSERVATION 36: MOVEMENT AND BAR UPDATE

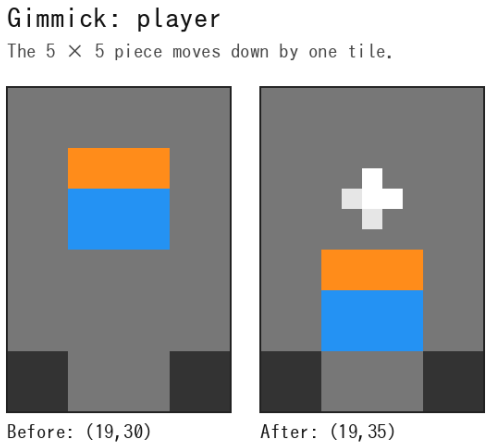


Figure 18: player. The square object moves down by one tile, or five cells. Its position changes from $(19, 30)$ to $(19, 35)$, and a white mark appears at its previous position. This corresponds to observation 36.

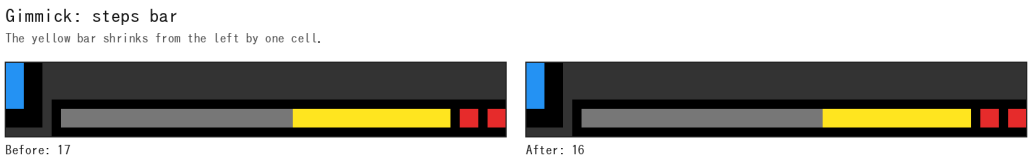


Figure 19: steps bar. At observation 36, the yellow display along the bottom edge shortens by one cell from its left end, and the GT value changes from 17 to 16. This is a separate event caused by the same action as Figure 18.

D Execution Settings

D.1 Models, Generation, and Context

The evaluated models and judges are listed in Tables 12 and 13. The same model is used for observation and action requests, with a retained history window of 10 messages and a harness context budget of 250k tokens. Generation parameters and submitted inputs for observation and action calls are included in the configuration records for each run in the electronic supplement.

Table 12: Evaluated models and access routes. Each model is represented by a single identifier.

Model	API model ID	Access route
Gemma4 31B	`gemma-4-31b-long-fp8`	Self-hosted
Qwen3.8-27B	`Qwen/Qwen3.8-27B`	Self-hosted
Qwen3.8-Max	`qwen/qwen3.8-max`	OpenRouter
Opus5	`anthropic/claude-opus-5`	OpenRouter

Table 13: Judge settings used for automated scoring. A fixed seed does not guarantee identical model outputs across repeated API calls.

Judge	Access route	temperature	seed	reasoning effort
GPT-5.6 Sol	OpenRouter (OpenAI)	not supported	0	xhigh
GLM-5.2	OpenRouter (z-ai fixed)	0	0	high

GPT-5.6 Sol is the prespecified primary judge so that RQ1 scoring, target selection, and the main reported results follow one scoring path; GLM-5.2, from a different model family, serves as a sensitivity judge. Results are aggregated separately for each. For Sol calls through the OpenRouter OpenAI route, the client sends seed 0 and does not send temperature because that parameter is unsupported. Fixing a seed does not guarantee identical outputs across repeated requests. Correspondence among API requests, applied evaluation prompts, and scoring outputs is recorded using run IDs and hashes.

D.2 Observation and Action Requests

Observation requests elicit Reasoning Logs, and action requests ask the model to select one action from the history containing that description. The full observation, action, and evaluation prompts and their applied versions are included in the electronic supplement and will be provided during review and released upon acceptance.

D.3 Message History Passed to the Agent

The Reasoning Log returned in response to an observation request is appended to the history as an assistant message, followed by the action request as a user message. The action call receives the retained screens, responses, and action requests as input. Table 14 shows the 15th observation call of replicate s1 for Opus5 on `AR25`. As a concrete example after the initial history window has filled, it lists all 10 messages retained at this point, including earlier history, in order.

Table 14: Input history for an observation call. Indices are zero-based, and the content column summarizes the role of each message. The original text corresponds to the input records in the public data.

Index	API role	Content
0	assistant	Earlier action response: ACTION2
1	user	Earlier screen, state, score, and observation instructions
2	assistant	Reasoning Log for that screen
3	user	Shared game context, available actions, and action request
4	assistant	Action response: ACTION2
5	user	Immediately preceding screen, state, score, and observation instructions
6	assistant	Reasoning Log for the immediately preceding screen
7	user	Shared game context, available actions, and action request
8	assistant	Immediately preceding action response: ACTION2
9	user	Current screen, state, score, and observation instructions

This request contains three grids spanning past to present; the previous and current screens in the supplementary `AR25` example (Appendix B.1) correspond to indices 5 and 9. The original text of all messages and its correspondence to each run are included in the electronic supplement.

E Scoring Rules and Evaluated Entities

E.1 Event-Side Rules

`reported` indicates whether the change corresponding to the GT event is mentioned. Paraphrases and references through demonstratives, coordinates, appearance, or local context are evaluated semantically. A general change report that does not identify an entity is recorded as a generic report. Descriptions of the current state are distinguished from mentions of changes, and claims of no change are also checked against GT.

When a change is reported, `entity` and `content` are independent. The entity label evaluates grounding to the entity that changed, allowing alternative names and different levels of abstraction. The content label evaluates the type of change and its required attributes. An incorrect entity alone does not make content 0. Conversely, even if the entity is correct, content is 0 when the direction, color, or before-and-after states are incorrect. When multiple entities are described together, semantic correspondence and the scope of the attributes are checked separately for each entity.

Table 15 provides constructed examples of the scoring rules. Suppose that only square A in the left region moves one cell to the right. After linking the description to the event through its reference to the left region, whether the entity is mistaken for circle B and whether the movement direction is incorrect are assessed separately. Claims whose correspondence to the event is itself ambiguous are classified as indeterminate.

Table 15: Retained event-side scoring states. The short statements below are constructed examples illustrating the definitions, not actual model outputs. The GT is "Square A moved one cell to the right."

reported	entity	content	hedge	Constructed example and interpretation
0	null	null	null	No mention of the change
1	0	0	0/1	"Circle B in the left region moved left": both entity and direction mismatch
1	0	1	0/1	"Circle B in the left region moved one cell to the right": entity mismatch, content match
1	1	0	0/1	"Square A moved left": only the entity matches
1	1	1	0/1	"Square A moved one cell to the right": both entity and content match

The auxiliary variable hedge records qualifications such as "probably." It is retained independently of entity and content correctness and is null for unreported events. The fully correct event rate uses events with reported, entity, and content all equal to 1 as its numerator.

E.2 Claim-Side Rules and Different Denominators

Specific change claims are split into minimal semantic units. A claim is `linked` if it can be matched to GT within the scoring scope, `unsupported` if it is within scope but lacks corresponding evidence, `out_of_scope` if it falls outside the current codebook, and `indeterminate` if insufficient evidence or ambiguity prevents a definite match. Linked indicates the existence of a corresponding event; incorrect attributes are assessed through entity and content on the event side. When distinguishing contradiction from lack of evidence within unsupported claims, the original claim and the reason for the judgment are retained.

The denominators of the unreported rate and fully correct event rate are GT events. The entity-match and content-match rates use only events with reported=1 as their denominator. The claim-side linked-claim proportion uses linked+unsupported as its denominator; counts and proportions of out_of_scope and indeterminate claims are disclosed separately. The unsupported-claim turn rate uses eligible turns with zero GT events for that game mechanic as its denominator. Even if a turn contains multiple unsupported claims, it counts as 1 in this turn-level rate.

E.3 Metric Definitions and Denominators

Let $\mathcal{E}$ be the set of GT event identifiers in an aggregation and $\mathcal{E}_{\text{rep}}$ its reported subset. For reported event i, $e_i, c_i \in \{0, 1\}$ are the entity and content match judgments. The notation $|\cdot|$ denotes set cardinality. Rates with a zero denominator are treated as missing.

E.4 Units, Alignment, and Eligibility

Each GT event is identified by the run ID, the index of the preceding action, the GT event ID, and the GT version. The Reasoning Log scored is the response body from the next observation call, which is the same string passed to action selection. Screens, GT, and responses are joined after checking the correspondence between observation and action indices. Internal reasoning in a separate API field is excluded from scoring.

GT covers changes that can be verified in the frames presented to the model. Updates confined to internal state, intermediate frames that were not displayed, and success or failure appearing only in transition records are excluded. For animations, transient events are checked against the sequence of frames actually presented.

Table 16: Metric definitions, denominators, and locations of the results.

Metric	Definition and interpretation	Results
Unreported rate	$1 - \frac{\lvert\mathcal{E}_{\text{rep}}\rvert}{\lvert\mathcal{E}\rvert}$: proportion of actual changes not mentioned.	Appendix Table 45
Entity-match rate	$\frac{\sum_{i\in\mathcal{E}_{\text{rep}}} e_i}{\lvert\mathcal{E}_{\text{rep}}\rvert}$: proportion of reported events whose entity matches (the rest are entity errors).	Appendix Table 45
Content-match rate	$\frac{\sum_{i\in\mathcal{E}_{\text{rep}}} c_i}{\lvert\mathcal{E}_{\text{rep}}\rvert}$: proportion of reported events whose content matches (the rest are content errors).	Appendix Table 45
Fully correct event rate	$\frac{\sum_{i\in\mathcal{E}_{\text{rep}}} e_i c_i}{\lvert\mathcal{E}\rvert}$: proportion with reporting, entity, and content all correct.	Figure 2, Appendix Table 45
Linked-claim proportion	Proportion of linked and unsupported claims that have corresponding GT.	Appendix Table 44
Unsupported-claim turn rate by game mechanic	Proportion of scored turns without GT for that mechanic that contain at least one unsupported claim of change.	Appendix Table 46

Missing inputs, undisplayed frames, unmatched GT, empty judge responses, invalid JSON, truncated outputs, and missing required labels are recorded as unscorable in the denominator ledger. Specific change claims are scored on the claim side even on eligible turns with zero GT events.

E.5 Inputs, History, and Structured Change Events

Inputs, history, and intervention The same evaluated model generates the Reasoning Log ℓ_t from the retained history $\mathcal{H}_t$ and the current observation o_t, and selects action a_t from $(\mathcal{H}_t, o_t, \ell_t)$. The history is updated as

$$\mathcal{H}_{t+1} = \mathcal{W}(\mathcal{H}_t \oplus o_t \oplus \ell_t \oplus a_t),$$

where $\oplus$ denotes appending in order and $\mathcal{W}$ denotes history retention and truncation by the harness. The intervention adds one description $z_t = g(e_t^\star)$ of the selected event's identifying appearance, location, and visible change to the first resumed observation request. Equation (1) in the main text defines the GT event set Δ_t from visible attributes $m_t = \pi(s_t)$ and the presented frame sequence. Below, $\mathcal{F}_t = o_t = (f_{t,1}, \ldots, f_{t,K_t})$ denotes this frame sequence; it contains the K_t frames presented after the single action a_{t-1} and before report ℓ_t; it is not the history across actions. The common-rule and exploratory procedures are given in Appendices I.1 and I.3.

Before-and-after values and transient events The quantity $m_t = \pi(s_t)$ comprises the visible attributes selected for evaluation, such as position, orientation, color, and display state. A GT event is described using before-and-after attribute values, a post-change value, or an event such as disappearance or flashing. Table 17 gives examples of attribute fields in the stored GT. Evaluation checks the attributes included in the reference description of each event.

Table 17: Forms of GT event descriptions. Examples are excerpts from attribute fields in the stored GT. The content evaluated follows each event's reference description and the game-specific scoring rules.

Form	Example GT attribute field	Described content
Before-and-after attribute values	`angle(270)_(0)`	The orientation changes from 270° to 0°.
Post-change attribute	`color(purple)`	The target region is recolored purple. The reference attribute is the post-change color.
State-switch event	`state(off)`	A state switch in which the highlight turns off.
Transient event	`state(flash)`	Flashing occurs. Its occurrence may not be identifiable from the final screen state alone.

For transient events, flashing or similar changes can occur between identical initial and final states. Equation (1) therefore also takes the sequence of frames actually presented, $\mathcal{F}_t$, as input to extraction. The quantity Δ_t is a structured description of changes, including their progression within the presented frames. Unlike numerical subtraction, it retains object correspondence and the occurrence of discrete events.

Distinguishing GT from intervention messages GT events are reference information for evaluation, whereas z_t is information presented to the model. Depending on the condition, z_t provides notification only that a change occurred, a change description with before-and-after values, or assistance that also includes the current state. Assistance using the current screen and pixel differences is likewise distinguished by its actual content. The quantity m_t covers only some visible attributes; task completion also requires using this information, inferring rules and goals, and selecting actions. An intervention adds information; whether that information is incorporated into the internal state depends on the model's processing.

E.6 Evaluation Prompt Structure and Provenance

RQ1 semantic judgments use prompts combining a shared judging procedure with game-specific evaluation rules. This paper presents the judgment structure, applied game-specific versions, and input/output specifications needed to interpret the results. Byte-exact production prompts, composition and validation code, prompt hashes, and raw execution traces will be provided in an anonymized repository during review and released in a public repository upon acceptance.

Shared judging procedure The judge receives numbered GT events and the Reasoning Log for the current turn. On the GT event side, it sequentially determines (1) whether the change is mentioned (reported), (2) whether the mentioned entity matches (entity), and (3) whether the change content and attributes match (content). Hedge is recorded independently when needed. On the claim side, specific change claims in the Reasoning Log are classified as linked, unsupported, out_of_scope, or indeterminate. For events with reported=0, entity, content, and hedge are null. The fully correct event rate is computed from reported, entity, and content; auxiliary fields used for earlier aggregations are not used.

Game-specific rules For the 20 evaluated games, the original judging rules specify entity categories, visible positions, synonymous expressions, priorities for entity matching, change attributes, and evaluation scope. Entity matching uses rules for identity based on visible attributes, distinguishing center coordinates from upper-left coordinates, movement described as disappearance and appearance. For example, `AR25` treats piece, reflection, and step bar as distinct entities; `CD82` distinguishes container movement from changes in the color inside. Game-specific rules are codebooks specifying what is matched as a GT event in each game, not postprocessing to alter the meaning of judgments.

Execution and validation The evaluation prompt is passed as system input, and GT events and the Reasoning Log as user input. Output JSON is checked mechanically for ID ranges and duplicates, label values, and null conditions. These are structural checks; semantic judgments are assessed by comparing the adopted judgments from GLM-5.2 and GPT-5.6 Sol. Comparisons with human judgments are reported in Appendix F.3.

Correspondence between evaluation rules and results The evaluation prompt and judgment applied to each observation will be traceable through the manifest of adopted judgments and the execution records in the electronic supplement, which will include both the complete rules and the aggregate data.

E.7 Aggregation and Selective Reporting

We distinguish rates with equal event weights, equal run weights, and equal weights assigned successively to game mechanics and games. In the primary aggregation, events are pooled across runs within each model $\times$ condition $\times$ game $\times$ game mechanic, and the resulting rates are then averaged over game mechanics and games, in that order. The set of games and game mechanics shared by

all four models and the full set encountered by each model are aggregated separately; units with zero denominators are treated as missing. Alternative aggregations check whether games with more evaluated entities dominate the overall value.

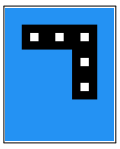
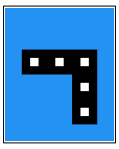
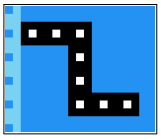
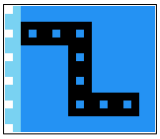
Intervals are estimated with 1,000 bootstrap resamples using runs as clusters. The game set is fixed, and runs are resampled independently within each model × game. The same resampling weights are used for game mechanics within a game to preserve within-run dependence. For units with few runs or small denominators, raw points and numerators/denominators are also shown. The comprehensive model × game × game mechanic plots provide descriptive diagnostics of the changes encountered in each run.

Unreported events are split by whether the game mechanic was mentioned previously. Together with the 00, 01, 10, and 11 combinations of entity and content for reported events, this gives six retained categories. Figure 2 combines 00 and 01 as entity errors and displays five categories. Prior mention indicates whether the same game mechanic was mentioned in an earlier report from the same trajectory, excluding the current turn. Occurrence order, first encounter, and whether the change is a repeated display or action-specific are separate variables.

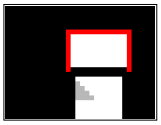
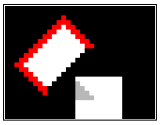
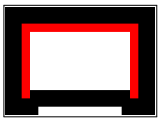
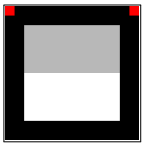
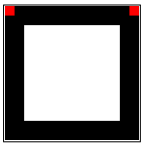
E.8 Illustrated Guide to Evaluated Entities and Changes

The following guide illustrates entities observed during execution using their original before-and-after grids and brief descriptions. The left and right images in each row show the same region before and after a change. Rows marked "intermediate use frames selected from the presented animation where the relevant change is visible. Bars are enlarged around the end where their length changes. Game mechanic names retain the English labels in the evaluation data to match the scoring plots in Appendices G.3 and G.4. Changes include object movement and rotation, changes in color or pattern, flashing and display switching, and bar shortening. Correspondence among original screens, all frames, GT, and logs is available through the illustrated-guide index in the electronic supplement.

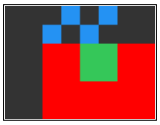
AR25: entities and changes Five game mechanics cover object and mirror-image movement, selection-display switching, and bar shortening.

Game mechanic	Before	After	Observed change
`piece`			The controllable object changes position while retaining its shape.
`reflection`			The mirror image drawn on the opposite side changes position.
`mirror_axis`			The selection display of the band crossing the board turns on.
`selection`			The marker of the selected object turns off.
`step_bar`			The band at the right edge shortens from the top.

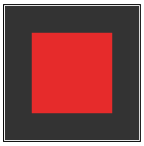
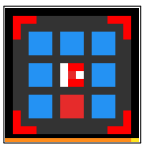
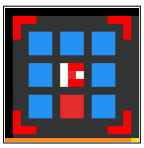
CD82: entities and changes Four retained game mechanics cover container movement, tilt and color, filled regions, and bar shortening.

Game mechanic	Before	After	Observed change
`energy_bar`			The band at the bottom edge shortens from the right.
`container_tilt`			The container moves around the periphery, changing position and orientation.
`reservoir_recolor`			The color inside the container changes.
`stack_fill`			The upper half of the lower square is recolored.

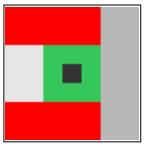
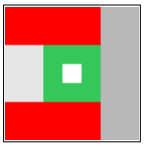
DC22: entities and changes Three game mechanics cover small-object movement, region patterns, and bar shortening.

Game mechanic	Before	After	Observed change
`budget_bar`			The band at the bottom edge shortens from the left.
`course_gimmick`			A small region changes from a solid fill to a checkerboard pattern.
`player`			The small controllable object moves horizontally.

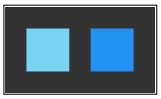
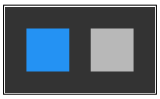
FT09: entities and changes Three game mechanics cover tile colors, border flashing, and bar shortening.

Game mechanic	Before	After	Observed change
`color_tile`			The clicked tile changes color.
`click_budget`			The band at the bottom edge shortens from the right.
`border_flash`			The panel border flashes briefly. (intermediate frame)

KA59: entities and changes Three retained game mechanics cover small-object movement, selection, and bar shortening.

Game mechanic	Before	After	Observed change
budget_bar			The band at the bottom edge shortens from the right.
player			Another small object is pushed and moves.
player_center_selection			The center of the selected small object changes color.

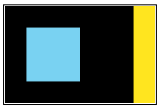
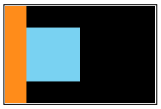
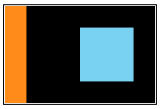
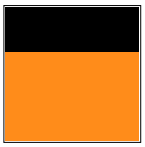
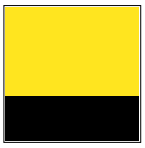
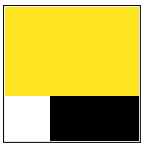
LP85: entities and changes Three game mechanics cover tile movement, progress dots turning on, and bar shortening.

Game mechanic	Before	After	Observed change
tiles			The arranged colored tiles move to adjacent positions.
life_bar			The band at the left edge shortens from the top.
level_pips			The progress dots near the top turn on.

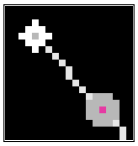
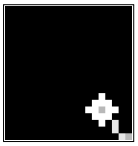
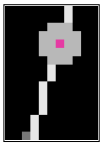
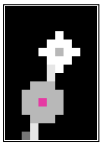
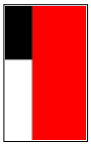
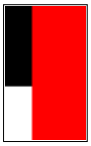
LS20: entities and changes Three retained game mechanics cover block movement, shape rotation, and bar shortening.

Game mechanic	Before	After	Observed change
player			The block within the board moves horizontally.
piece_rotation			The lower-left shape rotates clockwise.
steps_bar			The yellow band at the bottom edge shortens from the left.

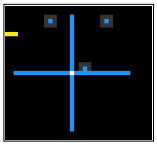
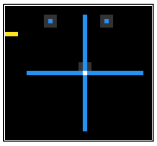
M0R0: entities and changes Four game mechanics cover movement of the left and right objects and shortening of two bars.

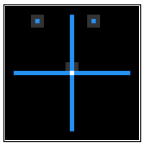
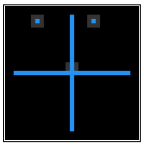
Game mechanic	Before	After	Observed change
`player`			The object in the left half moves horizontally.
`mirror_player`			The object in the right half moves in the mirrored direction.
`life_bar`			The band at the top edge shortens from the right.
`mirror_life_bar`			The band at the bottom edge shortens from the left.

`R11L`: entities and changes Three retained game mechanics cover movement of the body and arm, and bar shortening.

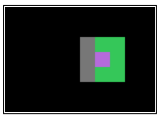
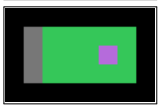
Game mechanic	Before	After	Observed change
`arm`			The white arm connected to the body moves.
`body`			The purple body moves.
`budget_bar`			The band at the left edge shortens from the top.

`RE86`: entities and changes Three retained game mechanics cover object movement, selection displays, and bar shortening.

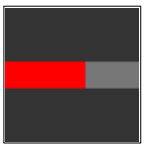
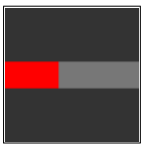
Game mechanic	Before	After	Observed change
`budget_bar`			The band at the bottom edge shortens from the right.
`piece`			The large cross-shaped object moves horizontally.

RE86 (continued)	Before	After	Observed change
piece_selection			The selection display at the center of the object turns off.

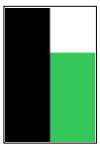
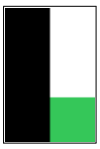
S5I5: entities and changes Three game mechanics cover arm length, dot position, and bar shortening.

Game mechanic	Before	After	Observed change
budget_bar			The band at the bottom edge shortens from the right.
extending_arm			The arm shortens while its fixed endpoint stays in place.
moving_dot			The small dot changes position.

SB26: entities and changes Three game mechanics cover tile movement, selection borders turning on, and bar shortening.

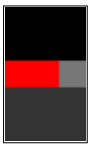
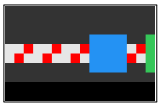
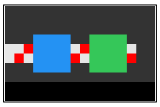
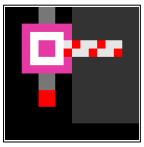
Game mechanic	Before	After	Observed change
budget_bar			The band near the bottom of the screen shortens from the right.
checkbox_selection			The lower selection border lights up white.
tile			The colored tile near the bottom moves into the board.

SC25: entities and changes Two retained game mechanics cover player changes and the action-budget display.

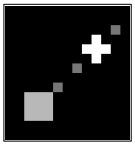
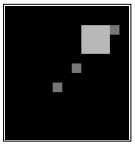
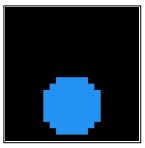
Game mechanic	Before	After	Observed change
budget_bar			The band at the right edge shortens from the top.

SC25 (continued)	Before	After	Observed change
player			The small controllable object changes position.

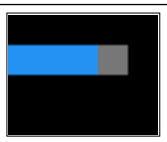
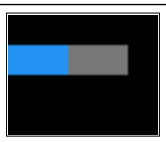
SK48: entities and changes Three game mechanics cover movement of blocks and elongated objects, and bar shortening.

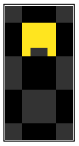
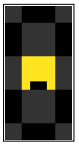
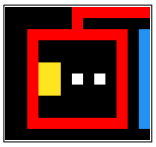
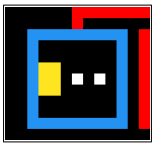
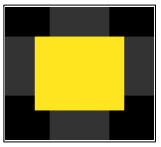
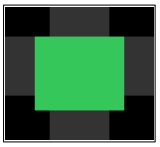
Game mechanic	Before	After	Observed change
budget_bar			The band near the bottom of the screen shortens from the right.
color_block			The colored block moves horizontally.
skewer			The colored elongated object moves upward.

SU15: entities and changes Three retained game mechanics cover block movement, appearance of a circular ring, and bar shortening.

Game mechanic	Before	After	Observed change
block			The small block moves diagonally upward.
budget_bar			The band at the bottom edge shortens from the right.
vacuum			A circular ring appears. (intermediate frame)

TN36: entities and changes Seven game mechanics cover movement of the controllable and reference objects, button colors, target flashing, and bar shortening.

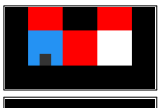
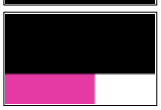
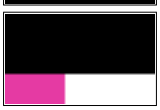
Game mechanic	Before	After	Observed change
budget_bar			The band at the top edge shortens from the right.

TN36 (continued)	Before	After	Observed change
instruction_button			The short button display changes color.
player			The controllable object moves downward. (intermediate frame)
reference			The reference object moves horizontally. (between intermediate frames)
run_button			The lower run button changes color. (intermediate frame)
selector_button			The lower selector button changes color.
target_flash			The target flashes briefly. (between intermediate frames)

TR87: entities and changes Three game mechanics cover bracket movement, symbols within slots, and bar shortening.

Game mechanic	Before	After	Observed change
bottom_timer_bar			The band at the bottom edge shortens from the right.
selector_bracket_pair			The brackets above and below the symbol shift horizontally.
slot1_glyph_cycle			The symbol in the selected slot changes shape.

TU93: entities and changes Three game mechanics cover movement of an object on a path and a ball, and bar shortening.

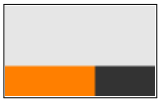
Game mechanic	Before	After	Observed change
player_block_on_path			The small object on the path moves rightward.
bottom_energy_bar			The band at the bottom edge shortens from the right.

TU93 (continued)	Before	After	Observed change
enemy_ball			Another ball-like object moves leftward.

VC33: entities and changes Two game mechanics cover object movement in response to actions and bar shortening.

Game mechanic	Before	After	Observed change
top_timer_bar			The band at the top edge shortens from the right.
yellow_bird_blob			The composite object moves in response to an action. Its color and shape vary by level.

WA30: entities and changes Three retained game mechanics cover movement of the controllable object, border-color changes, and bar shortening.

Game mechanic	Before	After	Observed change
moving_green_block			The controllable object moves upward.
bottom_timer_bar			The band at the bottom edge shortens from the right.
yellow_button_toggle			The small block changes border color.

F Scoring Reliability

F.1 RQ1: Agreement Between Judges

Table 38 reports run counts for the 20 games and item-wise agreement between GLM and Sol. Events are matched when their run IDs, observation numbers, and GT event IDs agree and their GT text is identical across judges. In addition to the 23,310 matched events, 103 appear only on the GLM side and 139 only on the Sol side; these are kept separate from the agreement denominator.

This comparison evaluates agreement between two judges on the same GT. For the binary fully correct judgment, both judges marked 9,489 events correct, only GLM marked 816, only Sol marked 1,018, and neither marked 11,987. Cohen's κ is 0.841. This value indicates the extent to which the two adopted sets of judgments agree on the same GT and Reasoning Logs. GLM cost less for the same inputs, while the different model families produced similar fully correct judgments. The game mechanic labels differed for 2 matched events; these differences are recorded separately to avoid conflating them with event correctness. Agreement with human judgments on the same 200 items is reported in Appendix F.3. The precision and coverage of GT itself require separate validation.

Table 38: Comparison between judges and verification of data coverage. Proportions use items that can be matched across both judges as their denominators.

Audit	Result
Run counts	400 evaluated and 400 scored.
GLM vs. Sol: reported	Agreement on 21,596 of 23,310 matched events (92.6%).
GLM vs. Sol: fully correct yes/no	Agreement on 21,476 of 23,310 matched events (92.1%).
GLM vs. Sol: $(reported, entity, content)$	Agreement on 20,714 of 23,310 matched events (88.9%).
Both judges assign reported=1	$n = 10,865$. Entity agreement: 97.5% (10,597/10,865); content agreement: 93.7% (10,181/10,865).
Game $\times$ game mechanic $\times$ model combinations	Across 238 combinations, Pearson $r = 0.927$, Spearman $\rho = 0.912$, and mean absolute difference 0.063.
API cost for the same four inputs	Sol: $0.162; GLM: $0.107 (measured totals on OpenRouter).

F.2 Judge Robustness and Sensitivity to Aggregation

GPT-5.6 Sol is used for the main results, and agreement with GLM-5.2 is examined (Table 39). Across the 238 game $\times$ game mechanic $\times$ model combinations with positive denominators for both judges, fully correct event rates had Pearson correlation $r = 0.927$, Spearman rank correlation $\rho = 0.912$, and mean absolute difference 0.063 (Figure 20). Item-wise agreement on the same events is reported in Table 38. We compare both aggregate values and individual event judgments.

Table 39: Fully correct event rates by judge. Each judge uses the 50 game and game mechanic pairs (20 games) shared by all four models, averaging with equal weights over game mechanics and then games.

Model	Sol (primary judge)	GLM
Gemma4 31B	20.5%	19.7%
Qwen3.8-27B	32.9%	29.2%
Qwen3.8-Max	36.2%	33.6%
Opus5	71.9%	71.2%

To examine sensitivity to aggregation weights, we compared equal weighting over game mechanics and then games in the common set, the same aggregation over all encountered pairs, and equal event weighting in the common set (Figure 21). Although fully correct event rates differed by several percentage points, the broad patterns across models were consistent across aggregations.

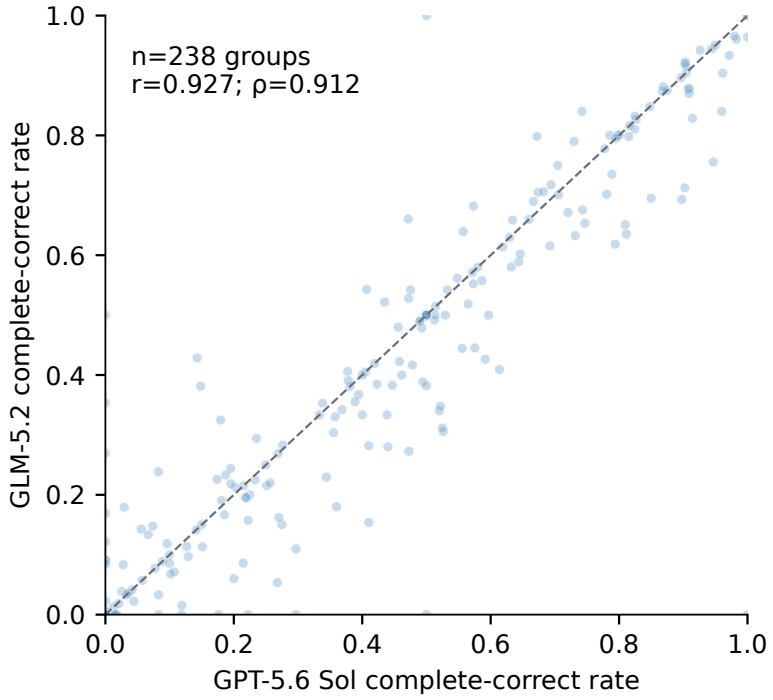


Figure 20: Comparison of fully correct event rates between judges. Each point is a game, game mechanic, and model combination evaluable by both judges. The dashed line indicates equality. GPT-5.6 Sol is the primary judge and GLM-5.2 the comparison judge.

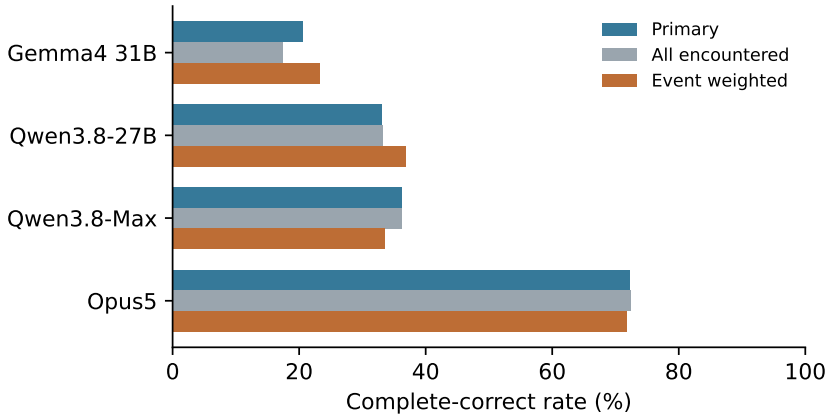


Figure 21: Sensitivity of RQ1 fully correct event rates to aggregation. Primary uses the game and game mechanic set shared by all four models, weighting game mechanics and then games equally. All encountered uses all games and mechanics encountered by each model. Event weighted directly pools GT events within the common game and game mechanic set.

F.3 Human Validation of Event-Side Semantic Judgments

Purpose, sample, and materials presented We assessed how closely automated event-side scoring in RQ1 aligned with human semantic judgments. The evaluator was one person who had not authored the evaluation rules. For 200 items, comprising 10 from each of 20 games, we presented the shared instructions, game-specific rules, all GT events for the observation, the target GT event, and the Reasoning Log. GLM and Sol judgments, information about their agreement, aggregate results, and agent model names were withheld. Game IDs were provided because they were needed to apply the rules.

The sampling design was random sampling without replacement of 10 items per game, stratified by game; judge correctness and agreement or disagreement were not selection criteria. For all 200 items, the target GT, the full GT set within the observation, and the Reasoning Log were checked against the original records and matched one-to-one to the adopted Sol and GLM judgments. The analysis set therefore contains 200 items from 143 runs and 199 observations, 10 per game.

Human and judge classifications When reported=0, entity and content are null; when reported=1, each is independently judged as 0 or 1. There are five valid tuples: $(0, \text{null}, \text{null})$, $(1, 0, 0)$, $(1, 0, 1)$, $(1, 1, 0)$, and $(1, 1, 1)$. Fully correct is derived from $(1, 1, 1)$. Figure 22 shows how the human and each judge classified the same 200 items. The human classified 101 as unreported, 9 as reported with an entity or content error, and 90 as fully correct. The corresponding GLM counts were 106, 11, and 83; Sol's were 92, 19, and 89. Similarity of aggregate distributions and event-wise agreement are evaluated separately.

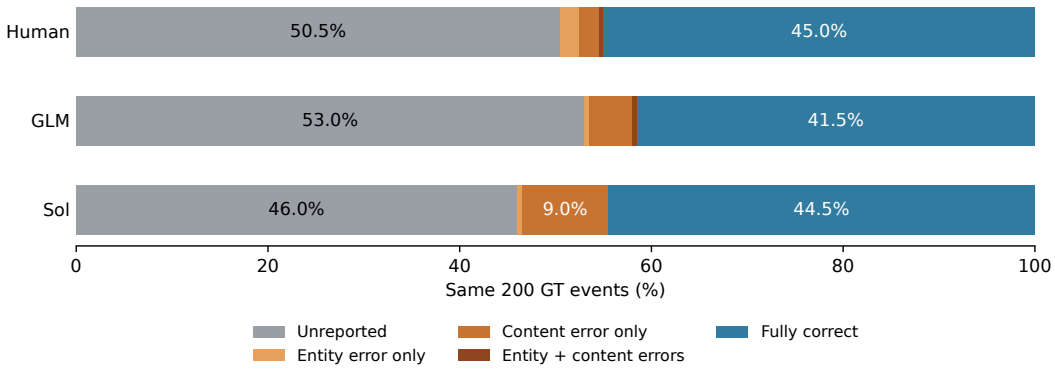


Figure 22: Human, GLM, and Sol judgment distributions for the same 200 items, pooled over items. Counts for unreported / entity-only error / content-only error / both errors / fully correct are 101/4/4/1/90 for the human, 106/1/9/1/83 for GLM, 92/1/18/0/89 for Sol. Entity and content errors are limited to events with reported=1.

Agreement and correction for chance agreement We calculate agreement P_o and Cohen's κ for binary reported, the complete label tuple (five classes), and binary fully correct yes/no. We use unweighted κ, assuming no ordering among label tuples. Let p_{Ak}, p_{Bk} denote the marginal proportions of category k for raters A, B. Index games by $j = 1, \ldots, 20$ and events within game j by $i = 1, \ldots, n_j$, and let y_{Aji}, y_{Bji} be their labels for the same event. Using the indicator function $\mathbb{I}[\cdot]$, which equals 1 when its condition holds and 0 otherwise,

$$P_o = \frac{1}{20} \sum_{j=1}^{20} \frac{1}{n_j} \sum_{i=1}^{n_j} \mathbb{I}[y_{Aji} = y_{Bji}], \qquad P_e = \sum_k p_{Ak} p_{Bk}, \qquad \kappa = \frac{P_o - P_e}{1 - P_e}.$$

P_e is the expected agreement under independent judgments based on each rater's label frequencies and varies with those frequencies. A value of $\kappa = 0$ indicates agreement comparable to this expectation, and $\kappa = 1$ indicates perfect agreement. Each event in game j has weight $1/(20n_j)$. We use these weights for agreement and the marginal label frequencies in κ, retaining equal game weights despite unequal remaining sample sizes. Pooled counts are reported separately.

Confidence intervals and dependence The primary analysis fixes the 20 games and uses a stratified bootstrap, resampling the n_j events within each game with replacement. The human, GLM, and Sol judgments are resampled together as a triplet, and agreement, marginal proportions, κ, and differences between judges are recalculated at each iteration. The 2.5th and 97.5th percentiles of 10,000 iterations (seed=20260918) form the 95% intervals. This uses an approximation that events are exchangeable within games.

As a sensitivity analysis accounting for correlations among events from the same run, we also performed 10,000 iterations of a cluster bootstrap with runs as units within games. Let R_j be the number of runs represented in the sample for game j. We sampled R_j runs with replacement and retained together all sampled events belonging to each run. Within each game, weights were normalized by the number of events after resampling, giving each game a total weight of $1/20$. This preserves game weights even when event counts vary across resamples. Intervals from both methods represent sampling variation for the fixed 20 games and this evaluator.

Table 40: Agreement and chance-corrected κ for 200 items. Point estimates use equal game weights. Brackets show 95% intervals from the event-level stratified bootstrap (Event) and from within-game resampling with runs as units (Run). P_e is expected agreement based on label frequencies.

			P_o 95% interval				κ 95% interval	
Metric	Comparison	P_o (%)	Event	Run	P_e (%)	κ	Event	Run
Reported	GLM–Sol	92.0	[88.5, 95.5]	[88.6, 95.4]	49.8	0.841	[0.771, 0.910]	[0.773, 0.908]
Reported	Sol–Human	89.5	[85.5, 93.5]	[84.5, 93.9]	50.0	0.790	[0.707, 0.869]	[0.691, 0.878]
Reported	GLM–Human	89.5	[85.5, 93.5]	[85.4, 93.5]	50.0	0.790	[0.709, 0.869]	[0.707, 0.869]
Exact tuple	GLM–Sol	88.0	[83.5, 92.0]	[83.9, 92.2]	43.3	0.789	[0.717, 0.857]	[0.722, 0.860]
Exact tuple	Sol–Human	80.0	[75.0, 85.0]	[74.7, 85.8]	43.4	0.646	[0.563, 0.729]	[0.560, 0.741]
Exact tuple	GLM–Human	83.5	[79.0, 88.0]	[79.2, 88.2]	45.5	0.697	[0.616, 0.776]	[0.618, 0.778]
Fully correct	GLM–Sol	93.0	[89.5, 96.0]	[90.3, 96.1]	50.9	0.857	[0.785, 0.919]	[0.800, 0.921]
Fully correct	Sol–Human	83.5	[78.5, 88.5]	[79.0, 88.5]	50.6	0.666	[0.565, 0.760]	[0.573, 0.766]
Fully correct	GLM–Human	86.5	[82.0, 91.0]	[82.0, 90.8]	50.9	0.725	[0.632, 0.811]	[0.633, 0.811]

Extent of agreement With equal game weights, reported agreement was 89.5% for both Sol and GLM. Complete-tuple agreement was 80.0% and 83.5%, respectively; fully correct yes/no agreement was 83.5% and 86.5%. For complete tuples, pooled agreement counts were 160/200 for Sol and 167/200 for GLM. The corresponding game-weighted κ values were 0.646 and 0.697, with intervals above zero.

Complete-tuple disagreements remained for 40/200 items for Sol and 33/200 for GLM. Game-weighted complete agreement between GLM and Sol was 88.0%. Scoring reliability is assessed using agreement, κ, and disagreement counts together.

With run-level resampling, complete-tuple κ intervals against the human were [0.560, 0.741] for Sol and [0.618, 0.778] for GLM, leaving the interpretation of agreement beyond chance unchanged.

Conditional agreement on entity and content Entity and content are restricted to events that both raters in each comparison labeled reported=1. Eligible counts are 93 for GLM–Sol, 93 for Sol–Human, and 86 for GLM–Human. Disagreements at the reported gate number 16, 21, and 21, respectively, and are excluded from the conditional comparisons. Entity/content rates alone therefore do not characterize overall scoring agreement. Table 41 reports both pooled rates across eligible events and equally weighted means of game-specific conditional rates. In the sample, all 20 games have eligible events in every comparison. Games with zero eligible events in a bootstrap replicate are omitted from that replicate's mean, not assigned 0. Means were defined in every replicate.

Which judge agrees more closely with the human evaluator? Using agreement with the human on the same events, let G_{ji}, S_{ji}, H_{ji} denote the GLM, Sol, and human labels for event i in game j, respectively. For each metric, we compute $D_{ji} = \mathbb{I}[G_{ji} = H_{ji}] - \mathbb{I}[S_{ji} = H_{ji}]$ and average with equal weights within games and then across games. The quantity D_{ji} is the difference in agreement judgments on the same event and preserves the correspondence among both judges and the human. Positive values indicate higher agreement between GLM and the human evaluator than between Sol and the human evaluator. For complete tuples, GLM was 3.5 percentage points higher, with a

Table 41: Conditional agreement on entity and content, and paired differences in agreement with the human. (A) Conditional agreement on events for which both raters assign reported=1; intervals are from the event-level stratified bootstrap for game means. (B) Difference in agreement with the human (GLM minus Sol), in percentage points; the 95% intervals for the differences preserve correspondence among both judges and the human.

(A) Conditional agreement on entity and content

Metric	Comparison	Agreements/eligible	Pooled rate (%)	Game mean (%) [95% interval]
Entity	GLM–Sol	91/93	97.8	98.6 [96.5, 100.0]
Entity	Sol–Human	89/93	95.7	97.3 [94.3, 99.5]
Entity	GLM–Human	84/86	97.7	98.2 [95.0, 100.0]
Content	GLM–Sol	86/93	92.5	91.7 [84.7, 97.5]
Content	Sol–Human	78/93	83.9	82.9 [74.6, 89.7]
Content	GLM–Human	75/86	87.2	87.9 [82.2, 92.3]

(B) Difference in agreement with the human (GLM minus Sol)

Metric	Difference	Event-level 95% interval	Run-level 95% interval
Reported	0.0	[-3.5, 3.5]	[-3.5, 3.7]
Exact tuple	3.5	[0.0, 7.0]	[-0.2, 6.9]
Fully correct	3.0	[-0.5, 6.5]	[-0.2, 5.6]

primary-analysis 95% interval of [0.0, 7.0] percentage points (Table 41). The reported difference was 0.0, and the interval for the fully correct difference included 0. This comparison is exploratory and presents metric-specific 95% intervals before correction for multiple comparisons.

The directions of disagreement also differed. Sol labeled 15 events reported=1 when the human labeled them unreported, versus 8 for GLM; the reverse occurred in 6 cases for Sol and 13 for GLM. For fully correct, the human labeled an event not fully correct while the judge labeled it fully correct in 16 cases for Sol and 10 for GLM; the reverse occurred in 17 cases for each judge. These counts reflect differences in judgment tendencies between one evaluator and each judge.

Among the 24 items on which GLM and Sol assigned different complete tuples, the human agreed with GLM on 11, with Sol on 4, and with neither on 9. This breakdown describes disagreements naturally included in the retained sample.

Estimand and validation scope This section compares 200 items with equal weights across 20 games, distinct from agreement across all 23,310 items in Appendix F.1. The original sampling frame contained 22,379 unique events matched across GLM and Sol by run ID, step, GT event ID, and exact GT text. The sampling code reproduces all 200 sampled identities using seed 20260918 for within-game sampling and a separate generator with the same seed for row shuffling; game-specific population sizes and inclusion probabilities are recorded in the sampling manifest. The original sampling frame is distinct from the full RQ1 evaluation set. The estimand remains the equally weighted mean across the 20 games, rather than a population-pooled agreement rate. Each interval is an approximation conditional on the fixed games, evaluator, and sample composition.

This validation covers event-side reported, entity, and content judgments by one evaluator. Reasoning Log length is not controlled in this validation, so differences in report verbosity may affect what is available for semantic scoring. Reproducibility between human evaluators; claim-side linked, unsupported, out_of_scope, and indeterminate labels; hedge; and the precision and coverage of GT itself each require corresponding additional evaluation. GT precision is the proportion of correct events among inspected GT events. Coverage is the proportion included in GT among eligible events enumerated by humans from screen sequences; its denominator also includes changes absent from GT.

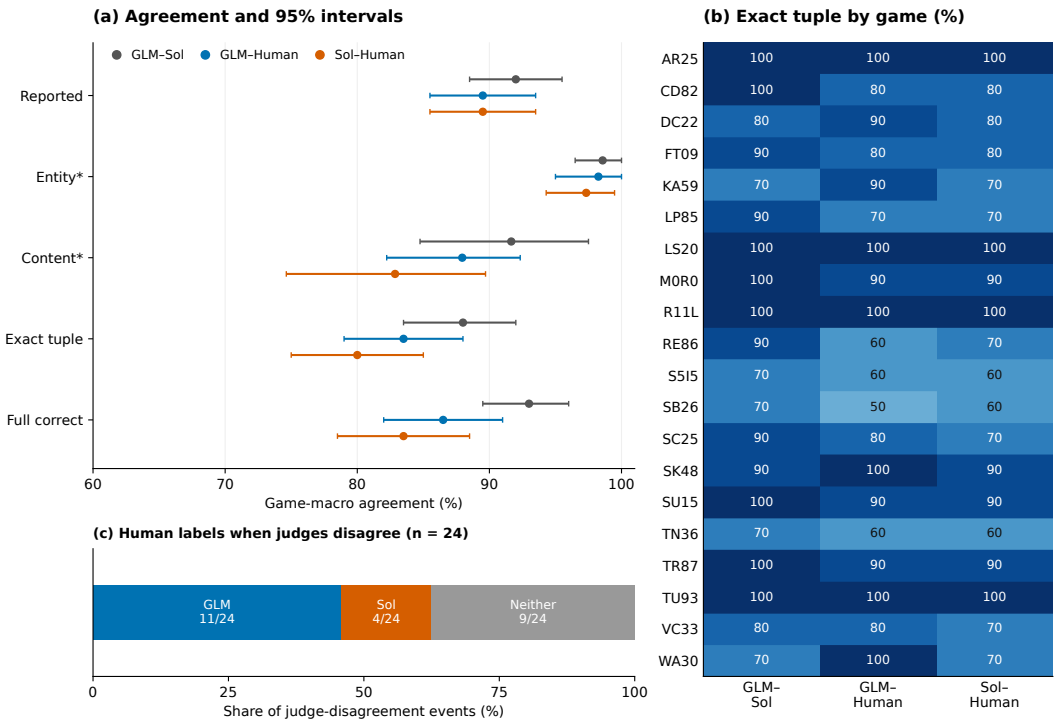


Figure 23: Human validation of event-side semantic judgments. (a) Equally weighted game agreement rates and 95% intervals for the same 200 items. Entity and Content (*) are restricted to events for which both raters assign reported=1. (b) Complete agreement by game ID. Labels give the sample size for each game. (c) Proportions of the 24 GLM–Sol disagreements where the human agreed with GLM, Sol, or neither. The human scored without seeing judge outputs.

G RQ1: Auditable Results

G.1 Scoring After Level Completion

RQ1 scoring does not stop at attainment of the first level. If an agent advances to later levels within the same run's action budget, eligible events observed in those levels are also matched against their corresponding Reasoning Logs. Mapping scored events to GT level information shows that events from levels 2 and 3 are included (Table 42). Validation covers the levels actually reached within each run's budget. Table counts are each judge's scored set, distinct from the count of events shared by both judges.

Table 42: Scored GT event counts in RQ1 and their levels. The original GT level was checked for every scored event.

Event level	Sol (primary judge)	GLM
Level 1	20,738	20,691
Level 2	2,542	2,547
Level 3	169	175
Total	23,449	23,413

G.2 Evaluation Scope and Denominators

Table 43 lists the RQ1 analysis set and the denominator of each metric. All 400 runs were scored.

The evaluation covers 20 games, including CD82 and WA30. Resource and counter displays remain in this evaluation.

The primary comparison across models uses game × game mechanic pairs with GT events for all four models; this common set covers all 20 games. This common set of games and game mechanics aligns the change categories while comparing events encountered through each model's own actions. Table 43 also reports counts for the full encountered set and the common set. For SK48, changes to the budget display, colored blocks, and elongated objects are evaluated; claim-side aggregation uses the same scope.

Table 43: RQ1 analysis set and denominators.

Item	Count
Evaluated runs	400
Scored runs	400
Scored observation steps	17,657
GT Event	23,449
Specific change claims	12,420
All encountered game × game mechanic pairs	66
Game × game mechanic pairs shared by all four models	50 (20 games)

Table 44: Classification of change claims by GPT-5.6 Sol. The denominator of the linked-claim proportion is the sum of linked and unsupported claims.

Model	linked	unsupported	out-of-scope	indeterminate	linked/(linked+unsupported)
Gemma4 31B	1,295	565	102	281	69.6%
Qwen3.8-27B	1,592	459	193	291	77.6%
Qwen3.8-Max	1,435	346	111	190	80.6%
Opus5	4,592	447	334	187	91.1%
Total	8,914	1,817	740	949	83.1%

Table 45: Counts corresponding to the four event-side metrics (GPT-5.6 Sol). Entries are numerators/denominators pooled over all encountered events. Entity and content matching use reported events as their denominators. The main-text comparison with equal game and game mechanic weights is shown in Figure 2 and Table 46.

Model	Reported / GT events	Entity match / reported	Content match / reported	Fully correct / GT events
Gemma4 31B	1,812/5,916	1,731/1,812	1,436/1,812	1,370/5,916
Qwen3.8-27B	2,343/5,168	2,275/2,343	1,975/2,343	1,933/5,168
Qwen3.8-Max	1,783/4,239	1,758/1,783	1,460/1,783	1,438/4,239
Opus5	6,265/8,126	6,244/6,265	5,859/6,265	5,843/8,126

Table 46: Primary RQ1 metrics by model (GPT-5.6 Sol), with run-based 95% intervals for the unreported and fully correct event rates. Unreported and fully correct event rates use the game and game mechanic set shared by all four models; unsupported-claim turn rates use eligible turns with no GT events for each game mechanic. All rates are averaged with equal weights over game mechanics and then games.

Model	Unreported [95% interval]	Fully correct [95% interval]	Unsupported-claim turn rate
Gemma4 31B	72.5% [67.9, 75.0]	20.5% [18.7, 23.4]	4.85%
Qwen3.8-27B	55.3% [52.4, 58.2]	32.9% [30.5, 36.0]	3.20%
Qwen3.8-Max	55.4% [51.7, 58.9]	36.2% [32.7, 39.6]	3.27%
Opus5	20.5% [19.1, 22.0]	71.9% [70.4, 73.4]	4.39%

G.3 Fully Correct Event Rates by Game and Game Mechanic

Within the same model and game, the fully correct event rates of the best- and worst-reported mechanics differed by a median of 28 percentage points over the 72 model–game pairs with at least two mechanics of at least 10 events each, and by 50 points or more in 13 of these pairs.

The fully correct event rate in each plot cell is the proportion of events that simultaneously satisfy reported=1, entity=1, and content=1, pooling GT events from the five runs for that model $\times$ game $\times$ game mechanic. Each cell shows the rate and GT event count. Rows list categories with GT events actually observed in the scoring set; a dash indicates a zero denominator for that model. Entity names correspond to the illustrated guide in Appendix E.8. Figures 24–28 display the 20 games on a common scale. These are descriptive diagnostic plots showing the pattern and denominator in each cell.

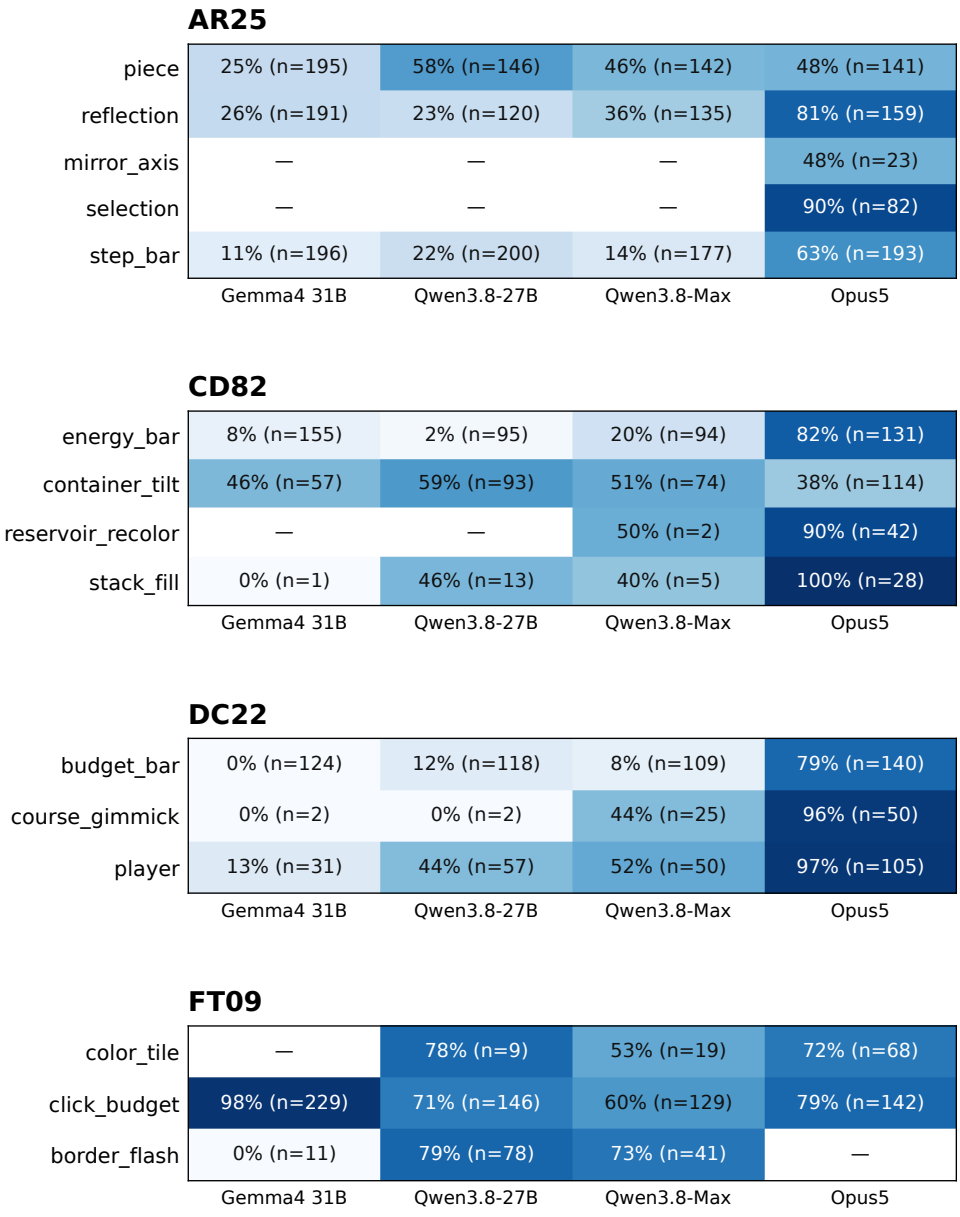


Figure 24: Fully correct event rate atlas (1/5): AR25, CD82, DC22, FT09. Colors use a common 0–100% scale. Cells show rates and GT event counts.

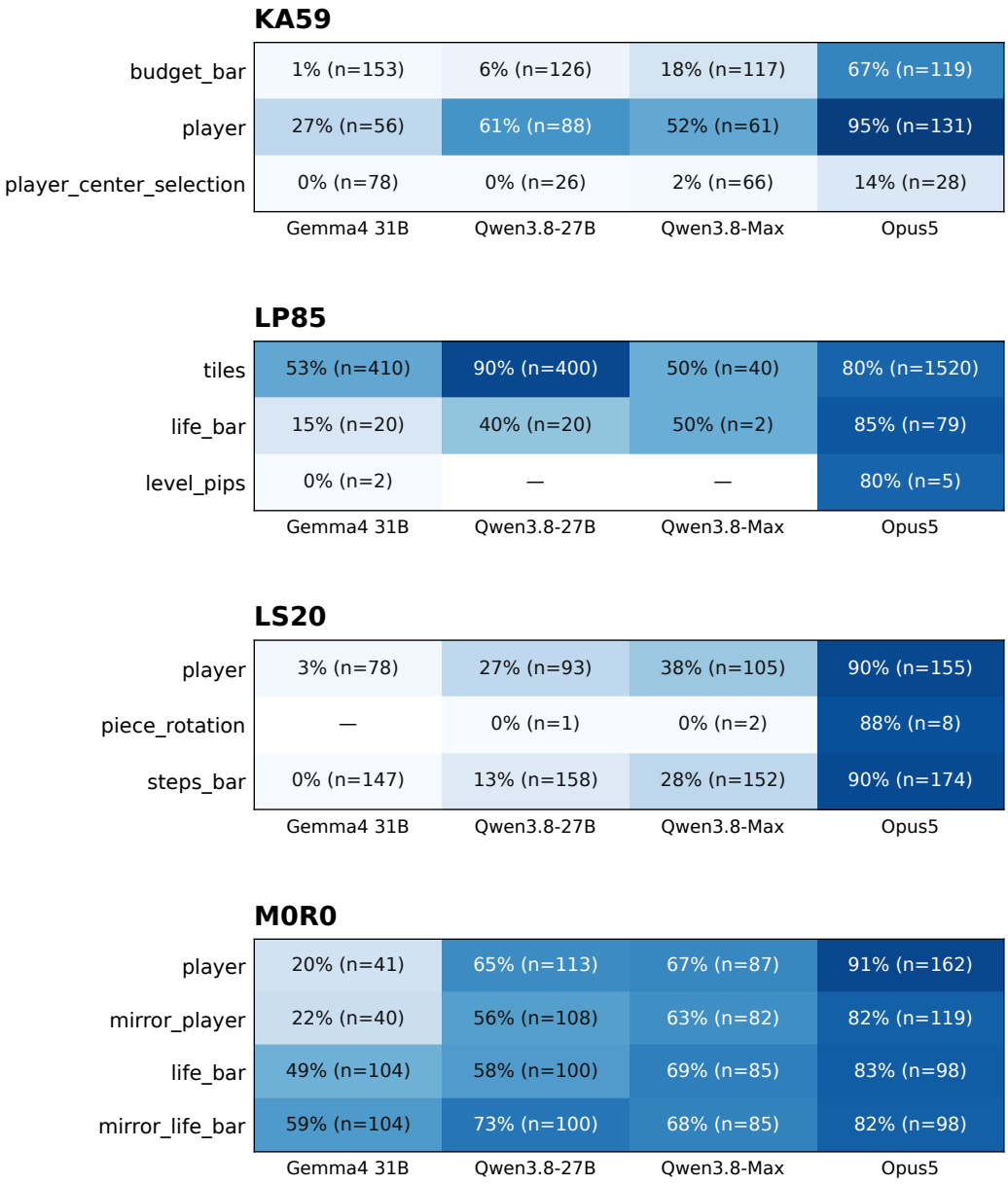


Figure 25: Fully correct event rate atlas (2/5): KA59, LP85, LS20, M0R0. Presentation and scale follow Figure 24.

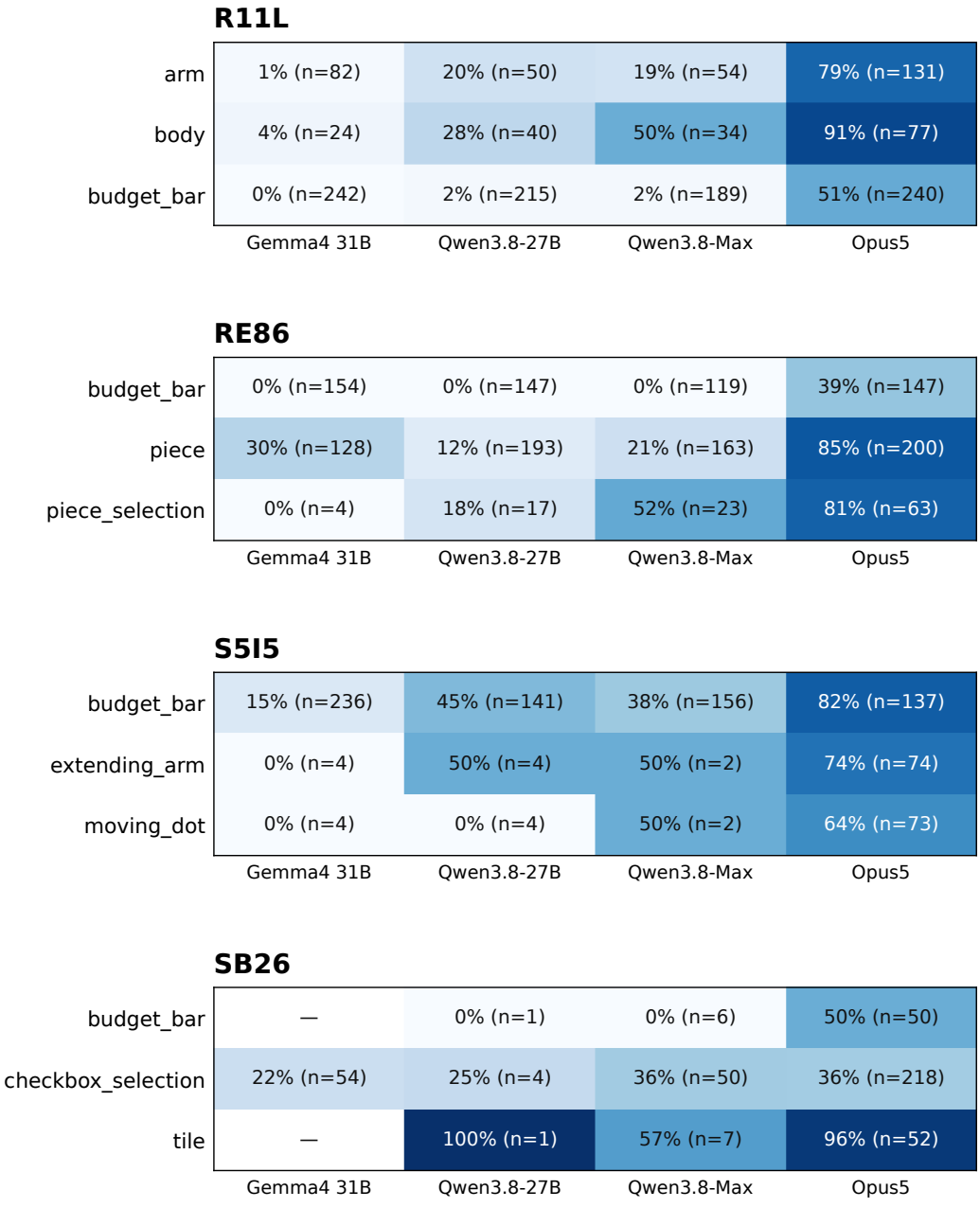


Figure 26: Fully correct event rate atlas (3/5): R11L, RE86, S5I5, SB26.

SC25

	Gemma4 31B	Qwen3.8-27B	Qwen3.8-Max	Opus5
budget_bar	22% (n=108)	30% (n=91)	15% (n=53)	57% (n=96)
player	8% (n=121)	18% (n=105)	27% (n=37)	70% (n=44)

SK48

	Gemma4 31B	Qwen3.8-27B	Qwen3.8-Max	Opus5
budget_bar	0% (n=17)	0% (n=45)	3% (n=36)	47% (n=53)
color_block	0% (n=3)	34% (n=61)	37% (n=38)	67% (n=95)
skewer	10% (n=40)	41% (n=117)	47% (n=55)	90% (n=127)

SU15

	Gemma4 31B	Qwen3.8-27B	Qwen3.8-Max	Opus5
block	8% (n=36)	0% (n=14)	4% (n=28)	78% (n=114)
budget_bar	0% (n=156)	10% (n=59)	22% (n=87)	98% (n=146)
vacuum	0% (n=468)	10% (n=177)	20% (n=261)	19% (n=438)

TN36

	Gemma4 31B	Qwen3.8-27B	Qwen3.8-Max	Opus5
budget_bar	3% (n=240)	7% (n=149)	17% (n=115)	75% (n=150)
instruction_button	0% (n=22)	0% (n=78)	0% (n=33)	0% (n=99)
player	0% (n=2)	—	63% (n=27)	91% (n=35)
reference	—	—	33% (n=9)	43% (n=23)
run_button	0% (n=2)	—	0% (n=14)	7% (n=30)
selector_button	—	—	0% (n=6)	0% (n=8)
target_flash	0% (n=1)	—	0% (n=1)	0% (n=3)

Figure 27: Fully correct event rate atlas (4/5): SC25, SK48, SU15, TN36.

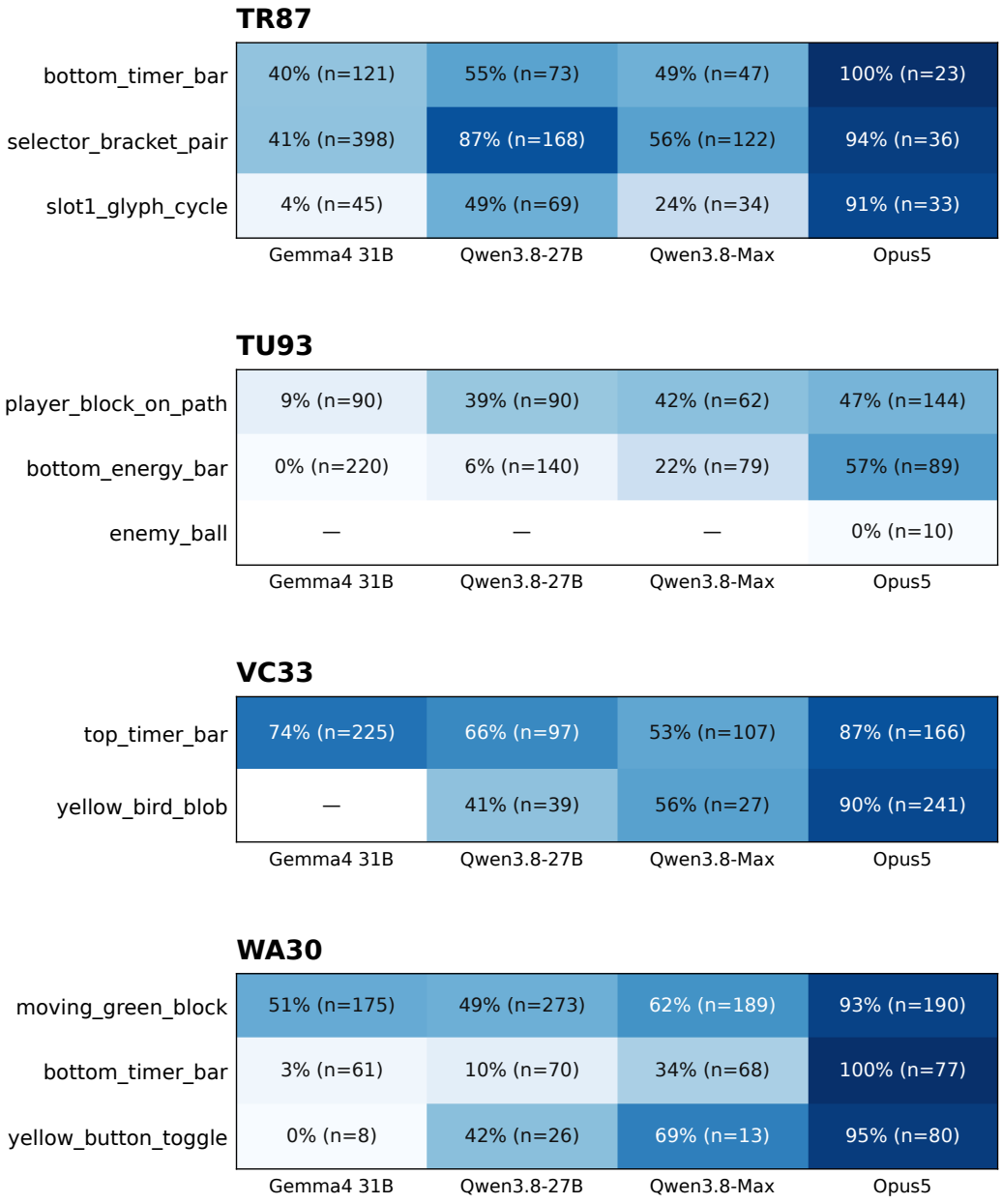


Figure 28: Fully correct event rate atlas (5/5): TR87, TU93, VC33, WA30.

G.4 Unsupported-Claim Atlas by Game Mechanic

Figures 29–33 show unsupported-claim turn rates by game mechanic and the numbers of eligible turns without GT for that mechanic. Rates follow the definitions in Table 16. The color scale is capped at 25% to make low rates distinguishable; actual rates are printed in the cells. Entity names follow the illustrated guide in Appendix E.8, and the same set of categories is plotted. Numerators and denominators for all evaluated categories are retained in the electronic supplement. Among turns with zero GT events across all game mechanics, the overall proportion containing unsupported claims was 9.57% (567/5,922).

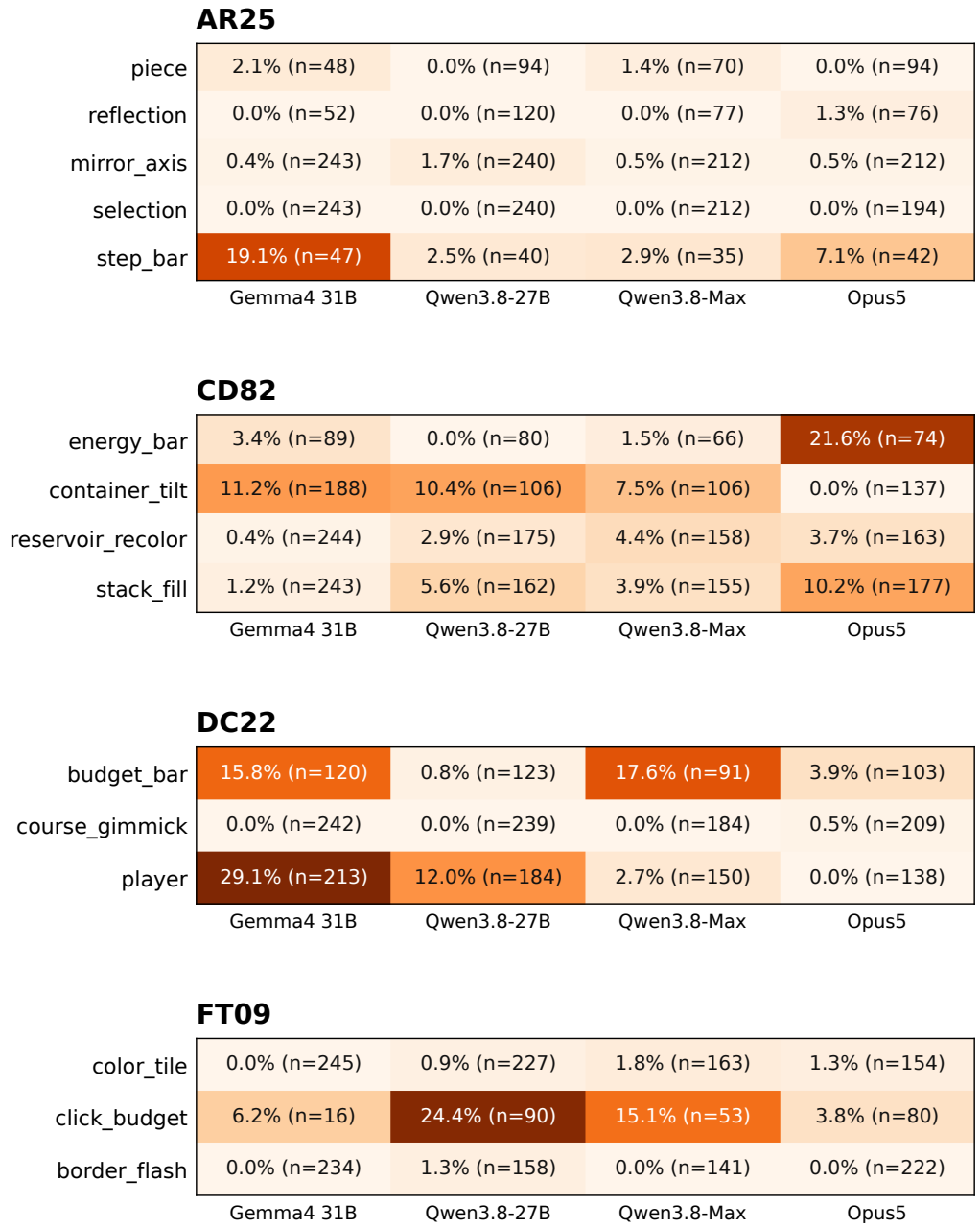


Figure 29: Unsupported-claim turn-rate atlas by game mechanic (1/5): AR25, CD82, DC22, FT09. Cells show the rate and number of eligible turns without GT for the corresponding mechanic.

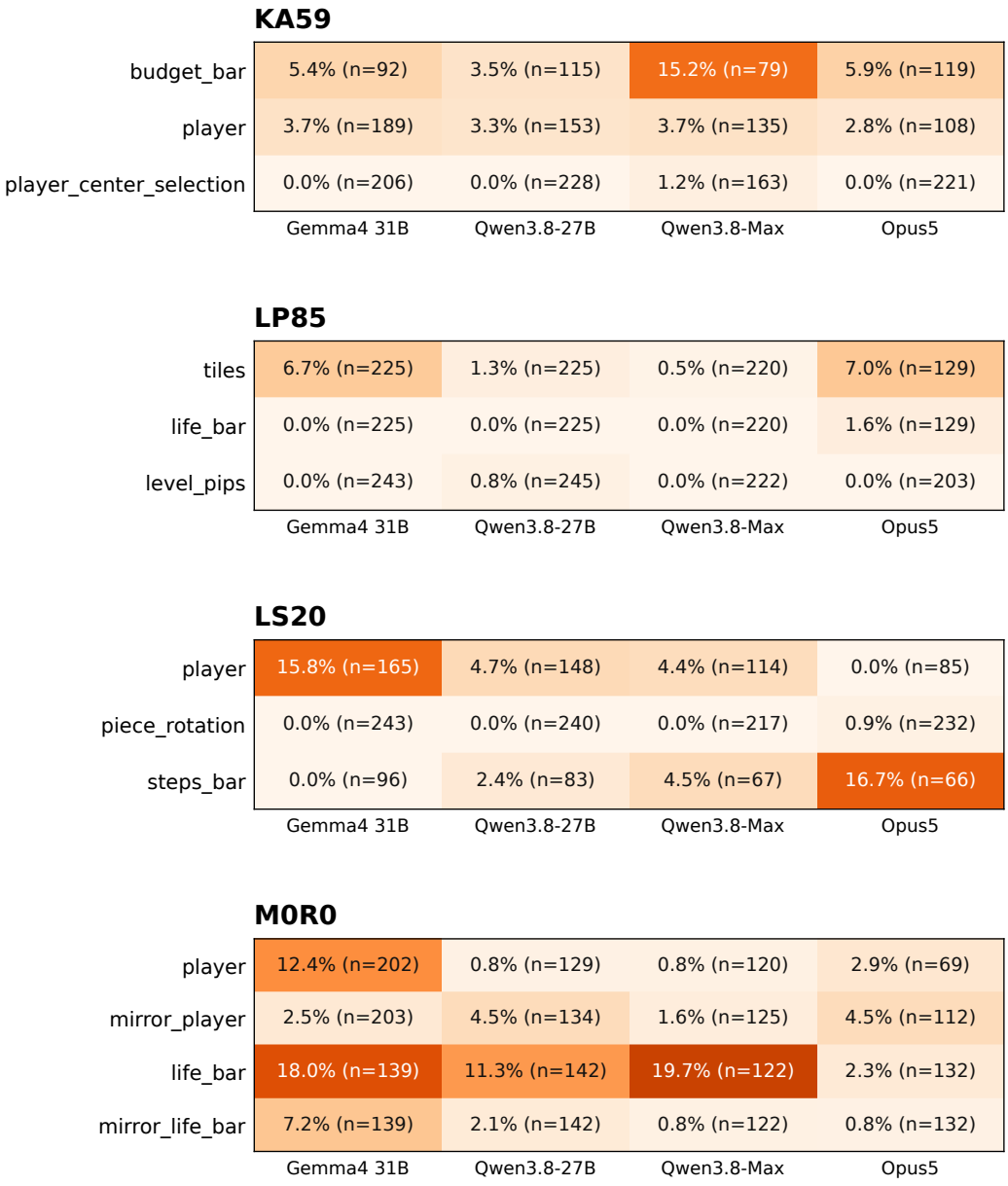


Figure 30: Unsupported-claim turn-rate atlas by game mechanic (2/5): KA59, LP85, LS20, M0R0.

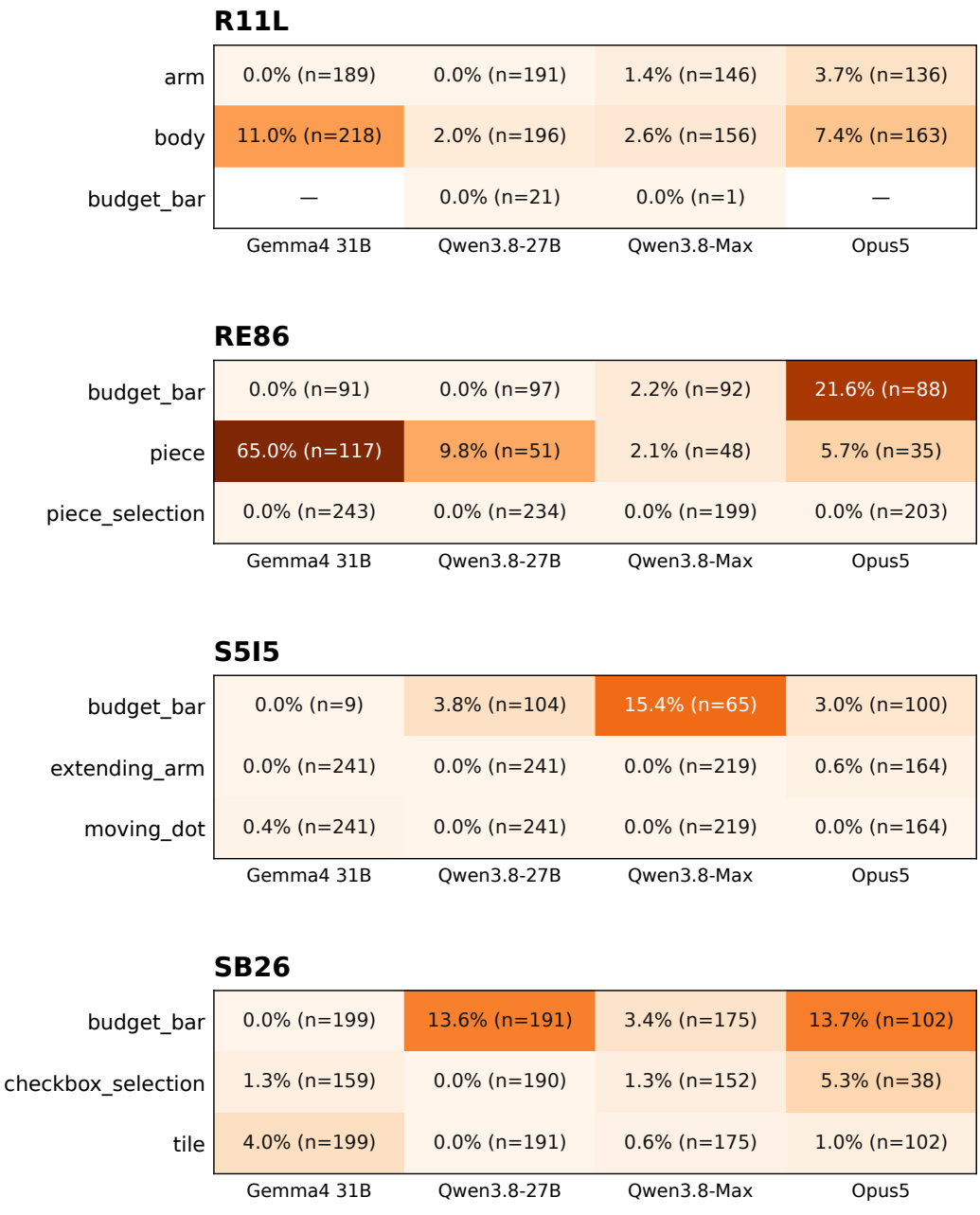


Figure 31: Unsupported-claim turn-rate atlas by game mechanic (3/5): R11L, RE86, S5I5, SB26.

SC25

	Gemma4 31B	Qwen3.8-27B	Qwen3.8-Max	Opus5
budget_bar	6.2% (n=80)	6.8% (n=103)	1.4% (n=71)	15.9% (n=113)
player	0.9% (n=106)	1.8% (n=112)	4.3% (n=94)	5.1% (n=175)

SK48

	Gemma4 31B	Qwen3.8-27B	Qwen3.8-Max	Opus5
budget_bar	0.0% (n=228)	1.1% (n=190)	1.2% (n=164)	4.3% (n=188)
color_block	0.8% (n=244)	6.4% (n=188)	1.2% (n=173)	0.6% (n=170)
skewer	1.0% (n=205)	4.2% (n=118)	4.8% (n=145)	0.0% (n=114)

SU15

	Gemma4 31B	Qwen3.8-27B	Qwen3.8-Max	Opus5
block	12.1% (n=206)	15.6% (n=224)	5.7% (n=157)	2.2% (n=138)
budget_bar	0.0% (n=86)	0.6% (n=179)	10.2% (n=98)	6.8% (n=88)
vacuum	0.0% (n=86)	0.0% (n=179)	0.0% (n=98)	1.1% (n=88)

TN36

	Gemma4 31B	Qwen3.8-27B	Qwen3.8-Max	Opus5
budget_bar	0.0% (n=5)	0.0% (n=95)	6.1% (n=82)	6.1% (n=82)
instruction_button	0.0% (n=223)	0.6% (n=166)	0.0% (n=173)	0.7% (n=145)
player	0.0% (n=244)	0.0% (n=244)	0.0% (n=190)	0.0% (n=220)
reference	0.0% (n=245)	0.0% (n=244)	0.5% (n=194)	0.0% (n=223)
run_button	0.0% (n=244)	0.0% (n=244)	0.0% (n=190)	0.9% (n=217)
selector_button	0.0% (n=245)	0.0% (n=244)	0.0% (n=194)	0.0% (n=228)
target_flash	0.4% (n=244)	0.0% (n=244)	0.0% (n=196)	0.0% (n=229)

Figure 32: Unsupported-claim turn-rate atlas by game mechanic (4/5): SC25, SK48, SU15, TN36.

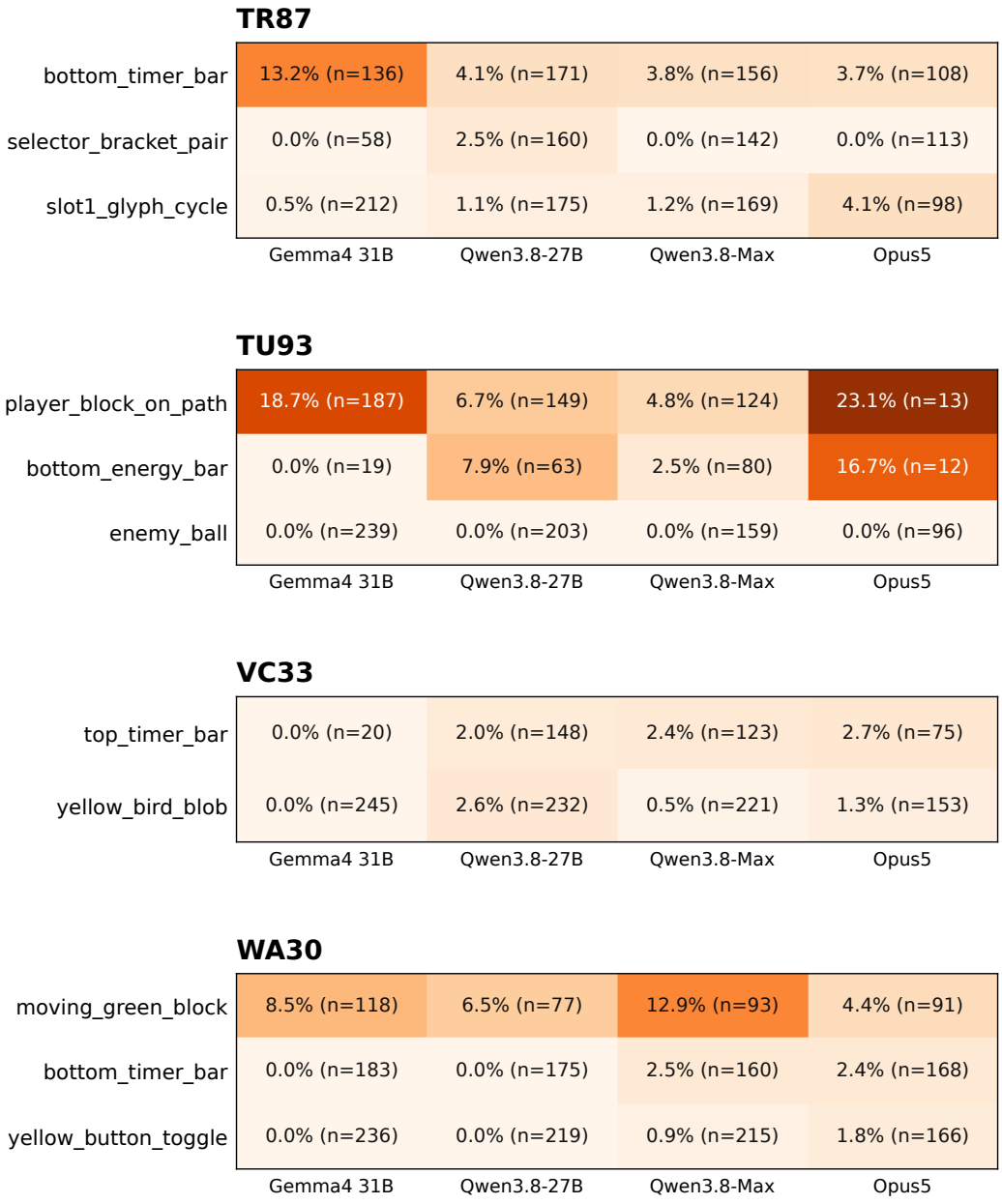


Figure 33: Unsupported-claim turn-rate atlas by game mechanic (5/5): TR87, TU93, VC33, WA30.

G.5 Accuracy by Type of Change

Each mechanic is assigned to one type by keywords in its description in Appendix E.8: movement (moves, shifts, rotates, falls, pushes, changes position), color/state (color, turns on/off, lights up, flashes, becomes, changes shape), counter/size (shortens, grows, fills, empties, counts, by one), and appear/disappear (appears, disappears, added, removed). Keywords are matched in this order. The appear/disappear type has one mechanic and is omitted from Table 47.

Movement (26): AR25 piece, AR25 reflection, CD82 container_tilt, DC22 player, KA59 player, LP85 tiles, LS20 player, LS20 piece_rotation, M0R0 player, M0R0 mirror_player, R11L arm, R11L body, RE86 piece, S5I5 moving_dot, SB26 tile, SC25 player, SK48 color_block, SK48 skewer, SU15 block, TN36 player, TN36 reference, TR87 selector_bracket_pair, TU93 player_block_on_path, TU93 enemy_ball, VC33 yellow_bird_blob, WA30 moving_green_block.

Color/state (16): AR25 mirror_axis, AR25 selection, CD82 reservoir_recolor, CD82 stack_fill, FT09 color_tile, FT09 border_flash, KA59 player_center_selection, LP85 level_pips, RE86 piece_selection, SB26 checkbox_selection, TN36 instruction_button, TN36 run_button, TN36 selector_button, TN36 target_flash, TR87 slot1_glyph_cycle, WA30 yellow_button_toggle.

Counter/size (23): AR25 step_bar, CD82 energy_bar, DC22 budget_bar, DC22 course_gimmick, FT09 click_budget, KA59 budget_bar, LP85 life_bar, LS20 steps_bar, M0R0 life_bar, M0R0 mirror_life_bar, R11L budget_bar, RE86 budget_bar, S5I5 budget_bar, S5I5 extending_arm, SB26 budget_bar, SC25 budget_bar, SK48 budget_bar, SU15 budget_bar, TN36 budget_bar, TR87 bottom_timer_bar, TU93 bottom_energy_bar, VC33 top_timer_bar, WA30 bottom_timer_bar.

Appear/disappear (1): SU15 vacuum.

Table 47: Fully correct event rate (%) by type of change, using the mechanic descriptions in Appendix E.8. Each cell lists movement / color-state / counter-size. "Pooled" weights every event equally (10414 / 1665 / 10026 events); "per mechanic" averages game–mechanic pairs with at least 10 events with equal weights (25 / 12 / 23 pairs). Movements are reported most accurately for every model under both judges and both aggregations, except Gemma4 31B under GLM per mechanic; the ordering of the two non-movement types depends on the aggregation.

	Sol		GLM	
Model	Pooled	Per mechanic	Pooled	Per mechanic
Gemma4 31B	31.6 / 6.1 / 21.8	22.0 / 5.3 / 19.9	30.0 / 1.8 / 25.6	18.7 / 4.1 / 22.5
Qwen3.8-27B	51.9 / 38.8 / 24.4	42.6 / 33.6 / 26.3	44.9 / 44.0 / 23.2	34.3 / 33.2 / 25.2
Qwen3.8-Max	45.5 / 29.6 / 27.1	44.7 / 34.3 / 29.2	41.2 / 25.5 / 29.3	39.5 / 27.7 / 30.5
Opus5	79.6 / 54.9 / 73.7	75.3 / 60.3 / 74.9	76.7 / 56.3 / 73.0	71.5 / 61.6 / 74.8

G.6 Sources of the Examples in Figure 2

All six examples are from LS20 under the ordinary condition; labels are the adopted judgments of both judges. Screens are the 64×64 grids presented to the agent at the preceding and current observations, rendered with the official palette; the outlined region is the crop window centred on the GT-derived changed cells (for the unsupported claim, on the right end of the step bar). The full Reasoning Logs are in the released data.

Table 48: Sources of the six examples in Figure 2. Sol/GLM give reported/entity/content for events and the claim label for the claim-side example.

Row	Category	Run	Obs.	GT record (verbatim)	Report sentence (verbatim)	Sol	GLM
1	Fully correct	`opus5_s1`	36	The yellow step bar at the bottom shrinks from the left (21→20 px).	Important new signal: the 11-bar at rows 61–62 shrank by exactly one cell (11s used to begin at col 34, now col 35).	1/1/1	1/1/1
2	Content error	`qwen27b_s1`	6	The player moves from (34,45) to (34,40).	Clicking on the "1" at (20,32) didn't move the player but caused environmental changes: the 12/9 blocks at rows 40-44 regenerated, and the 3-flood advanced 2 cells into the 11-strip at the bottom (cols 14-15 turned to 3).	1/1/0	1/1/0
3	Entity error	`qwen27b_s5`	4	The player moves from (34,45) to (34,40).	The Up action moved the cursor/player to the left edge of the 11-block at approximately (12, 48-49), converting those 11s to 3s.	1/0/1	1/0/1
4	Unreported: prior mention	`opus5_s2`	34	The player moves from (19,35) to (19,30).	No sentence about the player at this observation. Prior mention at observation 5: "Big breakthrough: ACTION4 moved the 5×5 block of 12s/9s from cols 34–38 to cols 39–43 . . . "	0/–/–	0/–/–
5	Unreported: no prior mention	`gemma_s1`	2	The player moves from (34,45) to (39,45).	No sentence about the player at this observation (first observation of the run).	0/–/–	0/–/–
6	Unsupported claim	`opus5_s1`	5	No GT event at this observation.	The bar of 11s along rows 61–62 is being consumed left-to-right (now two cells of 3 at cols 13–14), so it looks like a limited action/energy meter — I should stop wasting moves.	unsupported	unsupported

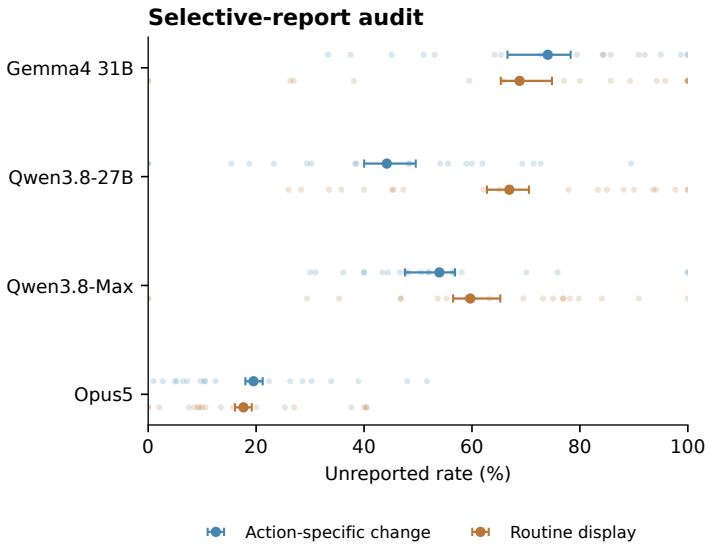
H RQ1: Sensitivity Analyses

H.1 Selective-Report Audit

For Opus5, 20.0 of the 20.5 percentage points of its aggregate unreported rate came from events whose game mechanic had already been mentioned earlier in the same run; for Gemma4 31B, the corresponding contribution was 39.7 of 72.5 percentage points (Figure 2). Reasoning Logs are free-form descriptions for selecting the next action, and models may not repeat predictable displays that they have already described. To examine this alternative explanation, we used codebook semantics to classify counters and bars that change in fixed screen regions at specified update opportunities as **routine displays**, and all other changes as **action-specific changes**. Classification used game mechanic names and codebook definitions. Table 49 lists the categories treated as routine displays. Other evaluated mechanics in those games are classified as action-specific changes.

For the comparison, we retained only strata within the same model and game that matched on (i) whether the mechanic had already been mentioned in earlier Reasoning Logs from the same trajectory and (ii) the occurrence index of the GT event within that mechanic. Within each model and game, events were pooled across runs separately for routine displays and action-specific changes to compute unreported rates, which were then averaged with equal game weights. Intervals were obtained from 1,000 run resamples within each model × game while holding the game set fixed. Small points in Figure 34 represent model × game values; the table in the same figure lists the game means and intervals.

This analysis compares reporting frequencies for routine displays and action-specific changes at the same prior-mention status and occurrence order. It measures external descriptions; their relationship to internal representations is discussed in Appendix B.12.



Model	Routine displays	Action-specific changes
Gemma4 31B	68.8% [65.3, 74.8]	74.0% [66.6, 78.3]
Qwen3.8-27B	66.9% [62.8, 70.6]	44.2% [40.0, 49.6]
Qwen3.8-Max	59.7% [56.5, 65.3]	54.0% [47.6, 56.9]
Opus5	17.7% [16.1, 19.2]	19.5% [18.0, 21.2]

Figure 34: Selective-reporting audit: unreported rates for routine displays and action-specific changes, matched on prior mention and within-mechanic occurrence order. Top: small points represent model × game values, large points are equally weighted game means, and horizontal lines are run-cluster 95% intervals. Bottom: the same equally weighted game means with their 95% intervals from run-cluster resampling.

Table 49: Game mechanics treated as routine displays. Other evaluated mechanics in each game are classified as action-specific changes.

Game	Routine display	Game	Routine display
AR25	step_bar	S5I5	budget_bar
CD82	energy_bar	SB26	budget_bar
DC22	budget_bar	SC25	budget_bar
FT09	click_budget	SK48	budget_bar
KA59	budget_bar	SU15	budget_bar
LP85	life_bar	TN36	budget_bar
LS20	steps_bar	TR87	bottom_timer_bar
M0R0	life_bar, mirror_life_bar	TU93	bottom_energy_bar
R11L	budget_bar	VC33	top_timer_bar
RE86	budget_bar	WA30	bottom_timer_bar

H.2 Hedged Reports and Perception Errors

Hedge is a label attached to change reports containing qualifications such as "probably" or "appears to." Table 50 splits reported GT events by the presence of hedging and aggregates entity and content errors. Each denominator is the number of reported events in that group; unreported events are excluded. Hedging here is an expression in the output, not a numerical confidence score.

Table 50: Correspondence between hedging and errors (GPT-5.6 Sol). Values are errors/reported events (rate). Events with both entity and content errors are included in both columns.

Model	Hedge	Entity errors	Content errors
Gemma4 31B	Absent	79/1762 (4.5%)	373/1762 (21.2%)
Gemma4 31B	Present	2/50 (4.0%)	3/50 (6.0%)
Qwen3.8-27B	Absent	67/2274 (2.9%)	354/2274 (15.6%)
Qwen3.8-27B	Present	1/69 (1.4%)	14/69 (20.3%)
Qwen3.8-Max	Absent	25/1608 (1.6%)	275/1608 (17.1%)
Qwen3.8-Max	Present	0/175 (0.0%)	48/175 (27.4%)
Opus5	Absent	21/6041 (0.3%)	390/6041 (6.5%)
Opus5	Present	0/224 (0.0%)	16/224 (7.1%)

Under Sol's judgments, content-error rates were higher for hedged reports in the two Qwen models, lower in Gemma4 31B, and similar in Opus5. Under GLM, they were higher in all four models, so this relationship also varied by judge. Errors remained without hedging, and models that rarely hedged had small denominators in the "present" group.

Unsupported claims have no independent hedge label. Table 51 is therefore a supplementary aggregation split by **whether the same Reasoning Log contains a hedged event report**. It includes logs with at least one reported event whose hedge label can be verified. The denominator is the number of linked and unsupported claims in those logs. This table does not measure whether unsupported claims themselves are hedged.

Table 51: Unsupported claims in the same log, grouped by the presence of hedged reports. Values are unsupported/(linked+unsupported) (rate).

Model	No hedged report	Hedged report present
Gemma4 31B	99/1373 (7.2%)	3/24 (12.5%)
Qwen3.8-27B	211/1734 (12.2%)	15/84 (17.9%)
Qwen3.8-Max	113/1377 (8.2%)	48/219 (21.9%)
Opus5	260/4455 (5.8%)	33/427 (7.7%)

These are descriptive comparisons pooling events and claims encountered within each model. Because the entities and scenes reported may differ with the presence of hedging, the comparisons should not be interpreted as causal effects of hedging on errors. Complete judge-specific counts and aggregation sets are included in the electronic supplement.

I RQ2: Interventions, Results, and Interpretation

I.1 Common-Rule Target Selection and Intervention

The selection rule, including the exclusion of resource and counter displays and the tie-breaking order, was fixed before any new run for the common-rule cohort was launched; saved runs that already matched the protocol were reused (Appendix I.2). The rule was not revised after observing its outcomes. The main-text RQ2 analysis uses the same selection procedure for both models across 20 games. For each model and game, we first compute the Sol fully correct event rate for each candidate mechanic separately within each original run that encountered it. We average these rates with equal weight across the encountered runs and select the mechanic with the lowest mean, breaking tied means by mechanic name. For that mechanic, we select the earliest observation with a reporting failure across the five original runs. Ties at the same observation number are resolved by

seed number and then GT event ID. We use this earliest failure as the intervention starting point; the experiment does not compare alternative starting-point rules.

Fully correct events and reporting failures follow the definitions in Section 3 and Appendix C. Missing judgments are not treated as failures; an event explicitly judged unreported remains a failure even when the conditional entity and content judgments do not apply. Selection uses no minimum event-count threshold and no screening by game progress. Each model is evaluated on its own selected target; the rule does not require the two models to select the same mechanic or starting point.

Candidate set After applying the RQ1 eligibility criteria, 99 game–mechanic pairs remain. RQ2 additionally excludes 21 resource and counter displays: three life/durability displays, two energy displays, five step/time displays, and eleven action-budget/click-count displays. These remain in RQ1, which evaluates reporting of visible changes broadly. RQ2 instead tests supplementation of changes to objects and their relations on the board, rather than notifications about remaining resources, time, or action budget. The same RQ2 exclusion rule applies to both models, leaving 78 eligible pairs (Table 52). Membership in this catalog does not imply that a mechanic was encountered or selected.

Matched continuations Each selected point has 10 baseline runs and 10 intervention runs from the same board state and history, each continued for 15 actions. Continuations are independent samples: no generation seed is specified, since seeds do not guarantee reproducible outputs from API-served models. Qwen3.8-Max uses temperature 1.0 and top_p 0.95; Opus5 uses the provider defaults. These settings are the same for observation and action generation, for both arms, and for all replicates. Immediately before generation of the first resumed observation report, the intervention supplies the anonymized GT of the selected event once. The message retains the entity's visible appearance and its before-and-after change. It supplies no strategic role, game-solving rule, correct action, or future information. Both conditions then continue autonomously. Existing runs are reused when their conditions match this protocol; reused runs are not counted again as independent repetitions.

Table 52: The 78 eligible game–mechanic pairs in the common RQ2 catalog. This is the selection range, including unencountered catalog entries, not a list of 78 tested interventions. Each model selects one target per game from its own RQ1 scores.

Game	Count	Eligible mechanics
AR25	4	mirror_axis, piece, reflection, selection
CD82	3	container_tilt, reservoir_recolor, stack_fill
DC22	8	bridge, carried_object, color_tile_visibility, course_gimmick, crane, ghost, movable_floor, player
FT09	2	border_flash, color_tile
KA59	5	bomb, cross_object, explosion_blast, player, player_center_selection
LP85	2	level_pips, tiles
LS20	2	piece_rotation, player
M0R0	2	mirror_player, player
R11L	4	arm, body, goal_flash, paint
RE86	2	piece, piece_selection
S5I5	4	chained_arm, extending_arm, fixed_arm, moving_dot
SB26	8	background_flash, bracket, checkbox_selection, nested_tile, outline_frame, reference_cursor, target_box, tile
SC25	6	door, energy_pack, fireball, player, switch, teleport_indicator
SK48	5	block_flash, color_block, match_indicator, skewer, skewer_selection
SU15	6	block, block_flash, goal_9x9, moving_ufo, reference_display, vacuum
TN36	8	barrier, ghost_overlay, instruction_button, player, reference, run_button, selector_button, target_flash
TR87	2	selector_bracket_pair, slot1_glyph_cycle
TU93	2	enemy_ball, player_block_on_path

Game	Count	Eligible mechanics (continued)
VC33	1	yellow_bird_blob
WA30	2	moving_green_block, yellow_button_toggle
Total	78	Common candidate set for both models.

I.2 Complete Common-Rule Results

The common-rule cohort contains 40 current conditions, one per model and game. All 20 game comparisons are complete for each model: every baseline and intervention arm has 10 runs, and both Sol and GLM judgments are available for every first target report. Each game therefore has equal weight in the aggregate summaries.

Table 53: Automatic-selection results. All 20 games are complete 10-vs-10 comparisons for both models. Each game has equal weight. Differences are percentage points, not significance tests.

A. Fully correct target reports

Model	Judge	Games	Baseline	Intervention	Difference
Qwen3.8-Max	Sol	20/20	66/200 (33.0%)	147/200 (73.5%)	+40.5
Qwen3.8-Max	GLM	20/20	57/200 (28.5%)	149/200 (74.5%)	+46.0
Opus5	Sol	20/20	97/200 (48.5%)	148/200 (74.0%)	+25.5
Opus5	GLM	20/20	85/200 (42.5%)	142/200 (71.0%)	+28.5

B. New-level attainment

Model	Games	Baseline	Intervention	Difference
Qwen3.8-Max	20/20	6/200 (3.0%)	17/200 (8.5%)	+5.5
Opus5	20/20	66/200 (33.0%)	66/200 (33.0%)	+0.0

Selected targets and starting points differ between models; the rows do not isolate a model effect. Both judges cover every run in each complete comparison.

Table 54: Directions of observed changes in complete 10-vs-10 comparisons. Improvement and decrease describe rate differences, not significance.

Model	Measure	Completed	Improved	Unchanged	Decreased
Qwen3.8-Max	Target report (Sol)	20	17	3	0
Qwen3.8-Max	Target report (GLM)	20	20	0	0
Opus5	Target report (Sol)	20	13	5	2
Opus5	Target report (GLM)	20	13	6	1
Qwen3.8-Max	Attainment	20	6	13	1
Opus5	Attainment	20	5	9	6

Inference for Qwen3.8-Max attainment For the primary attainment comparison, we form a 2×2 table (intervention/baseline by attained/not attained) within each game and apply a Cochran–Mantel–Haenszel test across the 20 game strata. The common odds ratio is 3.56 (95% interval 1.28–9.92, Robins–Breslow–Greenland) and the uncorrected two-sided test gives $p = 0.011$ (continuity-corrected $p = 0.021$). Strata with no attainment in either arm contribute no information to this statistic; they are retained in the reported 20-game cohort rather than treated as exclusions. As a game-level sensitivity analysis, we sign-flip the observed per-game differences in attaining-run counts. The exact two-sided p-value is 0.078 (one-sided 0.039). Leaving out one informative game at a time gives p between 0.004 and 0.066; the largest value arises when LP85 is removed. Thus the run-level stratified test supports an aggregate increase, whereas the game-level sensitivity test is not significant at 0.05.

Judge agreement at selected targets Agreement between judges is lower for some individual target events than for RQ1 aggregates. For example, the Opus5 DC22 baseline target is fully correct in 9/10 reports under Sol but 0/10 under GLM. We therefore report both judges for every condition.

Variation across selected points For Opus5, attainment increases occur in KA59 (1/10 to 2/10), LS20 (3/10 to 6/10), R11L (2/10 to 4/10), TN36 (2/10 to 3/10), and TU93 (3/10 to 6/10). The six decreases are AR25 (3/10 to 2/10), LP85 (10/10 to 9/10), RE86 (7/10 to 5/10), S5I5 (10/10 to 7/10), SU15 (2/10 to 0/10), and VC33 (4/10 to 3/10); nine are unchanged. Two decreases, LP85 and S5I5, start from 10/10 baseline attainment, so those starting points have no room for an increase, although this does not explain the observed decreases. Tables 56 and 57 list all selected conditions.

Relationship between the two outcomes For Qwen3.8-Max, five games improve in both attainment and the Sol target-report rate; six do so under GLM. For Opus5, one game improves in both outcomes under Sol and two under GLM. These counts concern changes in game-level rates, not paired outcomes of individual runs. In LS20, for example, the Opus5 Sol target-report rate is already 10/10 in the resumed baseline and remains 10/10 under intervention, while attainment rises from 3/10 to 6/10. Selection identifies a failure in the original RQ1 trajectory; repeated baseline reports need not reproduce that failure. Target-report improvement therefore verifies an effect on the intended report, whereas attainment tests whether supplementation also changes task execution; one does not imply the other.

All selected targets and observed results Table 55 lists every selected mechanic. Tables 56 and 57 give the corresponding source run, step, GT event ID, attainment, and both target-report judgments. Here, the source-run column identifies the original RQ1 trajectory used to locate the checkpoint; it is not a generation parameter. Each model has one current condition per game. Historical comparisons at other starting points are not added to this cohort.

Table 55: Selected mechanic for every current model–game pair. Source run and step are given in Tables 56 and 57. Catalog identifiers are retained to support comparison with the source.

Game	Qwen3.8-Max	Opus5
AR25	reflection	mirror_axis
CD82	container_tilt	container_tilt
DC22	course_gimmick	course_gimmick
FT09	color_tile	color_tile
KA59	player_center_selection	player_center_selection
LP85	tiles	tiles
LS20	piece_rotation	piece_rotation
M0R0	mirror_player	mirror_player
R11L	arm	arm
RE86	piece	piece_selection
S5I5	extending_arm	moving_dot
SB26	checkbox_selection	checkbox_selection
SC25	player	player
SK48	skewer	color_block
SU15	block	vacuum
TN36	instruction_button	instruction_button
TR87	slot1_glyph_cycle	selector_bracket_pair
TU93	player_block_on_path	enemy_ball
VC33	yellow_bird_blob	yellow_bird_blob
WA30	moving_green_block	moving_green_block

Table 56: All 20 current Qwen3.8-Max conditions. Results are baseline → intervention, with observed denominators. GT denotes the selected event ID. All rows are complete 10-vs-10 comparisons.

Game	Source run	Step	GT	Attainment	Target report (Sol)	Target report (GLM)
AR25	2	2	2	0/10 → 0/10	0/10 → 7/10	0/10 → 7/10
CD82	4	3	1	0/10 → 3/10	3/10 → 9/10	3/10 → 9/10
DC22	4	20	1	0/10 → 0/10	2/10 → 4/10	0/10 → 6/10
FT09	5	23	1	0/10 → 1/10	4/10 → 5/10	4/10 → 5/10
KA59	1	3	1	0/10 → 0/10	0/10 → 5/10	0/10 → 5/10
LP85	2	6	1	1/10 → 5/10	6/10 → 8/10	6/10 → 8/10
LS20	3	26	4	1/10 → 0/10	0/10 → 5/10	0/10 → 5/10
M0R0	4	2	1	0/10 → 0/10	1/10 → 8/10	1/10 → 8/10
R11L	4	5	3	0/10 → 1/10	7/10 → 7/10	4/10 → 5/10
RE86	3	3	1	0/10 → 0/10	0/10 → 7/10	0/10 → 7/10
S5I5	4	15	2	0/10 → 0/10	4/10 → 10/10	2/10 → 10/10
SB26	3	7	1	0/10 → 1/10	4/10 → 6/10	4/10 → 6/10
SC25	1	3	1	0/10 → 0/10	1/10 → 7/10	0/10 → 8/10
SK48	3	3	1	0/10 → 0/10	5/10 → 8/10	3/10 → 8/10
SU15	1	2	1	0/10 → 0/10	2/10 → 10/10	2/10 → 10/10
TN36	3	5	1	3/10 → 3/10	7/10 → 7/10	7/10 → 8/10
TR87	3	2	1	0/10 → 0/10	8/10 → 8/10	7/10 → 10/10
TU93	2	6	2	0/10 → 0/10	0/10 → 6/10	0/10 → 4/10
VC33	2	19	1	1/10 → 3/10	5/10 → 10/10	5/10 → 10/10
WA30	2	2	2	0/10 → 0/10	7/10 → 10/10	9/10 → 10/10

Table 57: All 20 current Opus5 conditions. Results are baseline → intervention, with observed denominators. GT denotes the selected event ID. All rows are complete 10-vs-10 comparisons.

Game	Source run	Step	GT	Attainment	Target report (Sol)	Target report (GLM)
AR25	2	19	1	3/10 → 2/10	0/10 → 4/10	0/10 → 4/10
CD82	1	2	1	0/10 → 0/10	0/10 → 4/10	0/10 → 6/10
DC22	4	10	1	0/10 → 0/10	9/10 → 10/10	0/10 → 10/10
FT09	1	5	1	9/10 → 9/10	10/10 → 10/10	10/10 → 10/10
KA59	1	12	2	1/10 → 2/10	5/10 → 5/10	2/10 → 2/10
LP85	1	5	1	10/10 → 9/10	9/10 → 10/10	8/10 → 9/10
LS20	2	34	4	3/10 → 6/10	10/10 → 10/10	9/10 → 10/10
M0R0	4	7	1	0/10 → 0/10	1/10 → 8/10	1/10 → 8/10
R11L	2	5	1	2/10 → 4/10	10/10 → 10/10	10/10 → 10/10
RE86	2	8	1	7/10 → 5/10	9/10 → 4/10	3/10 → 4/10
S5I5	3	12	1	10/10 → 7/10	3/10 → 5/10	1/10 → 5/10
SB26	2	3	2	10/10 → 10/10	0/10 → 3/10	0/10 → 0/10
SC25	1	3	1	0/10 → 0/10	0/10 → 10/10	0/10 → 10/10
SK48	2	8	1	0/10 → 0/10	8/10 → 10/10	7/10 → 9/10
SU15	2	4	1	2/10 → 0/10	2/10 → 6/10	6/10 → 7/10
TN36	4	4	1	2/10 → 3/10	9/10 → 4/10	9/10 → 4/10
TR87	4	27	1	0/10 → 0/10	10/10 → 10/10	10/10 → 10/10
TU93	3	36	2	3/10 → 6/10	0/10 → 6/10	0/10 → 5/10
VC33	3	12	3	4/10 → 3/10	2/10 → 9/10	9/10 → 9/10
WA30	1	2	1	0/10 → 0/10	0/10 → 10/10	0/10 → 10/10

Reused runs Five conditions (Opus5 AR25 and LS20; Qwen3.8-Max LS20, SB26, and VC33) use saved runs that match this protocol; each run is counted once. Empty continuation reports are retained in all outcome counts.

Interpretation Improved reports alone do not establish a benefit to task performance, and improved attainment alone does not establish that report correction caused the benefit; the two outcomes are therefore reported separately.

Reproducibility The electronic supplement will provide structured results for all 40 selected conditions, the candidate catalog with eligibility flags, and scripts that regenerate the aggregate tables and figures. Each condition records the target mechanic, seed, observation number, GT event ID, outcome counts, and observed denominators.

Claim-side effects and attainment versus target reports Table 58 scores the first resumed reports on the claim side, and Table 59 relates attainment within 15 actions to the correctness of that first report.

Table 58: Claim-side scoring of the first resumed report in the common-rule comparisons (200 runs per condition). "Any": reports containing at least one unsupported claim; "target": reports containing an unsupported claim about the target mechanic; "linked share": claims linked to a GT event among linked, unsupported, and indeterminate claims. The same judge prompts as in RQ1 were used.

		Baseline			Intervention		
Model	Judge	Any	Target	Linked share	Any	Target	Linked share
Qwen3.8-Max	Sol	24/200	15/200	78%	9/200	4/200	90%
Qwen3.8-Max	GLM	4/200	1/200	68%	6/200	3/200	89%
Opus5	Sol	46/200	19/200	79%	47/200	14/200	79%
Opus5	GLM	20/200	10/200	81%	20/200	3/200	92%

Table 59: Attaining runs whose first resumed report was fully correct on the target, against the fully correct rate among all runs of the same condition (200 runs per condition and model).

		Baseline		Intervention		Both	
Model	Judge	Attaining	All runs	Attaining	All runs	Attaining	All runs
Qwen3.8-Max	Sol	4/6	33%	16/17	74%	20/23	53%
Qwen3.8-Max	GLM	4/6	28%	16/17	74%	20/23	52%
Opus5	Sol	36/66	48%	48/66	74%	84/132	61%
Opus5	GLM	32/66	42%	42/66	71%	74/132	57%

I.3 Exploratory Interventions: Overview and Selection

The comparisons in this and the following subsections are exploratory: their targets were identified from RQ1 scores, reports, and screens rather than by the common rule in Section 4.3. Three of them (Opus5 LS20 observation 34; Qwen3.8-Max VC33 observation 19 and SB26 observation 7) use the same runs as common-rule conditions and are counted only once. We retain the selection rationale, observed attainment increases, and comparisons without increased attainment, together with the corresponding reports and actions. The immediate reporting rates recorded for these comparisons cover all GT events at the shared observation; they are not the target-only rates used in the main analysis.

Selection and comparison conditions Using RQ1 scores and the original reports and screens, we identified errors in change information that could impede subsequent decisions and supplied the corresponding information once. The games were LS20, where directional actions move a block; AR25, where an L-shaped object moves; VC33, where regions expand or contract and objects move; SB26, where blocks and selection borders appear; and TN36, where clicks change region colors. We

compared LS20, AR25, and VC33 with Opus5, and VC33, SB26, and TN36 with Qwen3.8-Max. At each decision point, we restarted 10 runs per condition for the baseline and intervention from the same state and history and measured level attainment within 15 actions. The AR25 comparisons use the same shortened restart context in both conditions. Appendix I.5 details intervention targets, messages, and restart contexts.

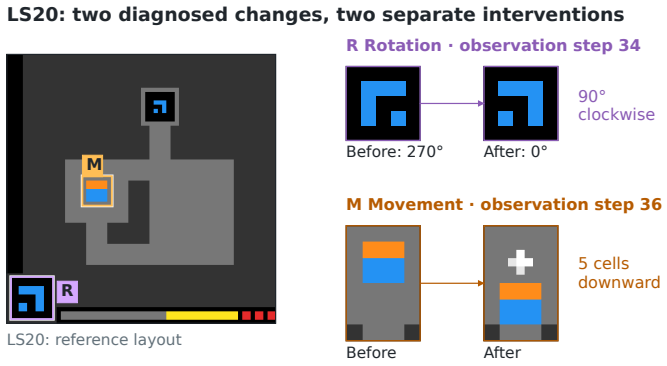
As concrete intervention examples, Figure 35 shows the two changes supplied in LS20. At the point where rotation was unreported, the shape's rotation was added once to the first observation request; at the other point, the object's downward movement was added instead.

I.4 Exploratory Interventions: Diagnosis and Target GT

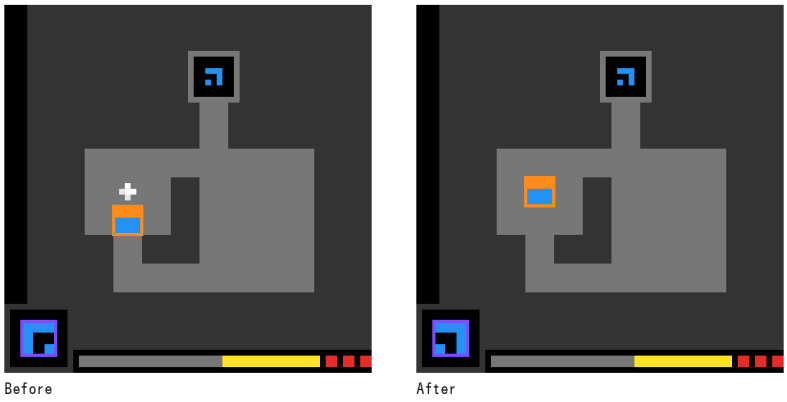
Target selection aligns RQ1 judgments, original reports, and actual screen changes. Table 60 lists the missing information and intervention hypotheses. Attainment rates for every scene and condition are shown in Figure 37 in Appendix I.7.

Table 60: Hypotheses and the corresponding target GT for each scene. For Opus LS20 observation 36 and Opus VC33 observations 22 and 27, the target event was fully correct in the original RQ1 report under Sol; the hypothesis concerns how the reported change was interpreted, not a reporting failure. Opus LS20 observation 34 and Qwen LS20 observation 26 are also common-rule starting points.

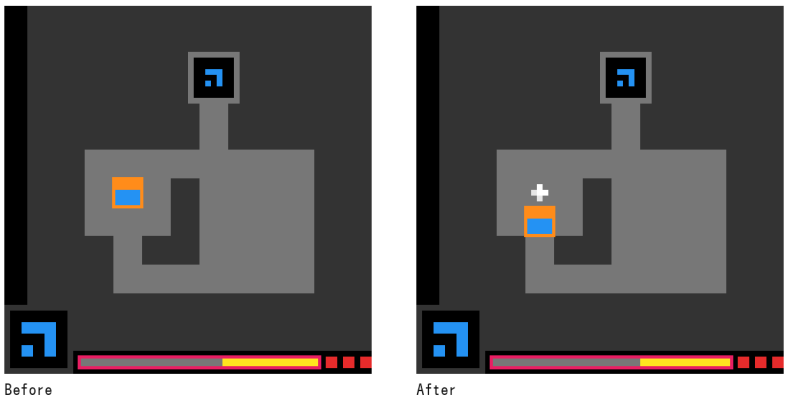
Scene	Diagnosis and hypothesis	Change supplied once
Opus LS20 original trial 2, observation 34	Rotation is unreported, and the next decision based on the displayed shape is inadequate	Shape rotation $270^\circ \to 0^\circ$
Opus LS20 original trial 2, observation 36	The moving entity is confused with another display, leading to incorrect tracking of the controllable object	Square object $(19, 30) \to (19, 35)$
Opus VC33 original trial 1, observation 27	The region change is treated as a superficial switch, and exploration of the buttons is abandoned	White-region tip $(0, 24) \to (4, 24)$
Opus VC33 original trial 1, observation 22	Region shortening is interpreted as time passing, losing its correspondence with the action	White-region tip $(4, 24) \to (0, 24)$
Qwen VC33 original trial 2, observation 19	Movement of the yellow object is not tracked, and interaction shifts to another stationary display	Yellow object $(46, 44) \to (50, 44)$
Qwen LS20 original trial 3, observation 26	Both judges mark rotation unreported. Supplying rotation is hypothesized to aid shape-based decisions	Shape rotation $270^\circ \to 0^\circ$



(a) Locations of the two targets and the supplied changes



(b) Observation 34: before and after the Up action



(c) Observation 36: before and after the Down action

Figure 35: Examples of targeted intervention: rotation and movement in LS20 (Opus5, original trial 2). (a) The left shows the locations of the targets; the right shows the actual inputs before and after the action at each intervention point. **R (purple)** denotes rotation of the lower-left shape, and **M (orange)** denotes downward movement of the square object. One corresponding change is supplied at each point. The reference board on the left is the last frame of observation 34. (b) Before-and-after images at observation 34. The Up action caused the object to move upward and the lower-left shape to rotate 90° clockwise. Orange boxes indicate movement of the square object; purple boxes indicate rotation of the lower-left shape. Only rotation information was supplied at this scene in RQ2. (c) Before-and-after images at observation 36. The Down action moved the square object from $(19, 30)$ to $(19, 35)$. Orange boxes indicate movement of the square object; pink boxes indicate the decrease in the display along the bottom edge. Only movement information was supplied at this scene in RQ2. Observations 34 and 36 indicate the order of observations in original trial 2. Unannotated before-and-after images, all presented frames, and their correspondence to the original records are retained in the electronic supplement.

I.5 Exploratory Interventions: Protocol and Records

This validation resumes from a specified observation on an original execution trajectory and performs 10 replicate trials per condition from the same starting point. In the tables, "original trial" identifies the run that produced the pre-intervention trajectory, and "observation" identifies the observation number within that trajectory. In these exploratory comparisons, change information for the target GT event $e_t^\star$ is supplied once as $z_t = g(e_t^\star)$. The mapping $g(\cdot)$ retains information identifying the entity on screen and its actual change, while excluding strategic role names and correct actions. The baseline and intervention conditions branch from the same state and history, and level attainment is compared over 15 actions after resumption. Figure 36 shows where the intervention is added and the subsequent observations that are aligned.

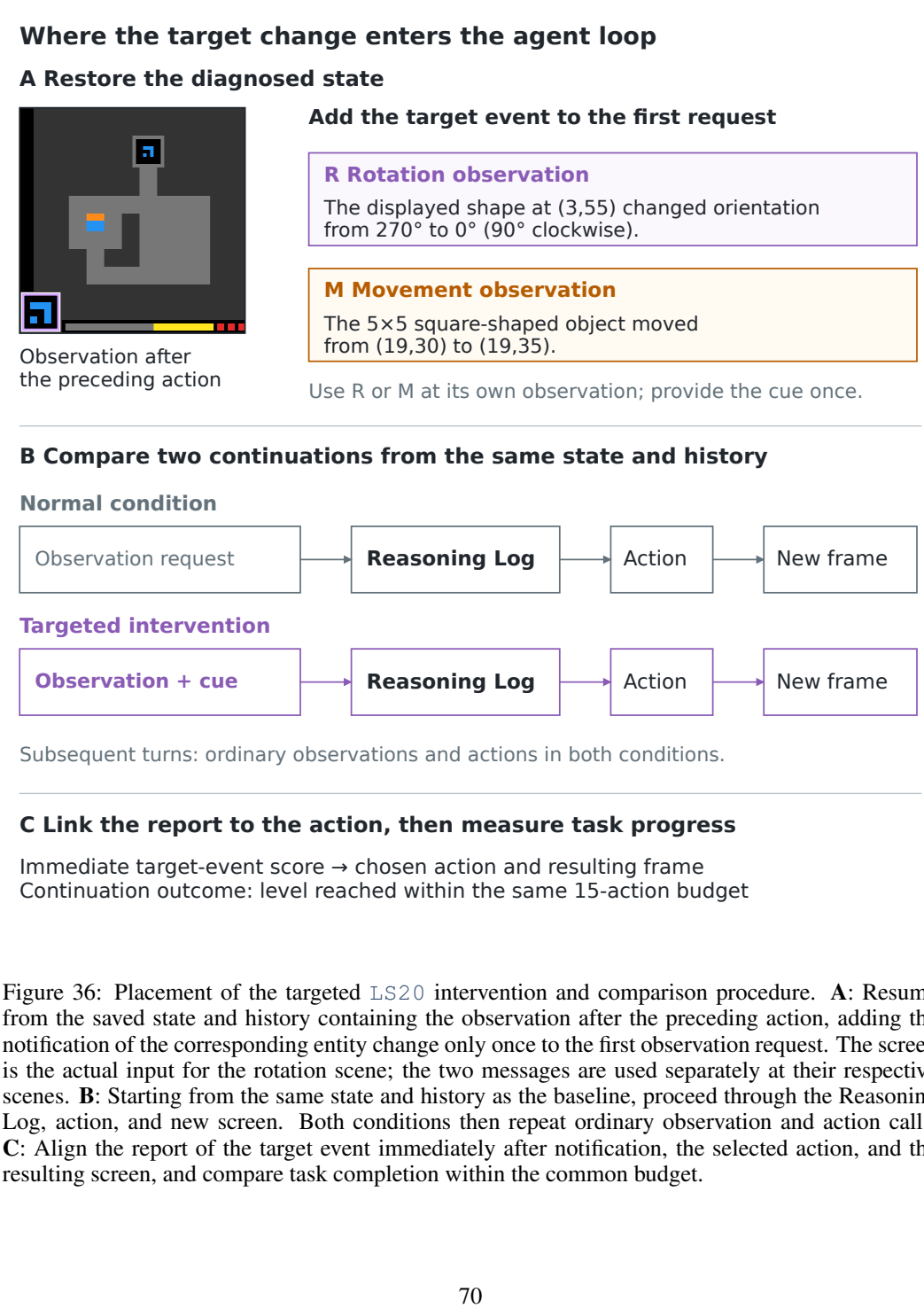


Figure 36: Placement of the targeted `LS20` intervention and comparison procedure. **A**: Resume from the saved state and history containing the observation after the preceding action, adding the notification of the corresponding entity change only once to the first observation request. The screen is the actual input for the rotation scene; the two messages are used separately at their respective scenes. **B**: Starting from the same state and history as the baseline, proceed through the Reasoning Log, action, and new screen. Both conditions then repeat ordinary observation and action calls. **C**: Align the report of the target event immediately after notification, the selected action, and the resulting screen, and compare task completion within the common budget.

Evaluation budget and speed of attainment A common evaluation budget of 15 actions after resumption is used. Continuation extends beyond the immediate notification to examine whether the supplied change information is retained and used in later decisions after the initial notification leaves the 10-message history window. Messages count observation requests and responses and action requests and responses; one action involves multiple messages. For a budget of N actions, per-run attainment is

$$Z_N = \mathbb{I}[T \leq N],$$

where T is the first action after resumption at which the next level is reached, with $T = N+1$ when it is not reached, and $\mathbb{I}[\cdot]$ equals 1 when its condition holds and 0 otherwise. The primary experiment sets $N = 15$, so non-attainment has $T = 16$. First attainment is checked against all transition records, including runs that reset afterward. We define an auxiliary score for speed of attainment as

$$P = 16 - T = \sum_{h=1}^{15} \mathbb{I}[T \leq h]$$

where h is the number of actions after resumption and $\mathbb{I}[\cdot]$ is the indicator function. The score P is 0 if the level is not attained and 15 if it is attained on the first action; earlier attainment gives a larger value. The number of attaining runs is the sum of Z_{15} within a condition, and the attainment rate is its within-condition mean.

I.6 Targeted Notification Text

For rotation and movement in LS20, we use the messages in Figure 36. For VC33, the following entity changes are supplied once at the corresponding scenes.

> **Opus original trial 1, observation 27:** The white region extended rightward: its tip moved from (0,24) to (4,24), while its anchor remained at (0,24).
>
> **Opus original trial 1, observation 22:** The white region shortened leftward: its tip moved from (4,24) to (0,24), while its anchor remained at (0,24).
>
> **Qwen original trial 2, observation 19:** The 6×6 yellow object moved from (46,44) to (50,44).

The notification is added to the first observation request as a description of the visible change caused by the preceding action. Each VC33 condition has 10 completed runs, and Qwen's observation-output limit during continuation is 32,768 tokens. Run-level completion records and attainment counts are included in CSV files in the electronic supplement.

I.7 Exploratory Interventions: Results and Interpretation

Attainment at exploratory points Increased attainment counts were observed at exploratory decision points spanning five games and two models (Figure 37; immediate fully correct rates for every point are provided in the electronic supplement). For example, at a TN36 point where the target change was unreported in RQ1, attainment increased from 3/10 to 6/10. Points that re-present a change the original RQ1 report had already described fully correctly do not test correction of an RQ1-diagnosed reporting error. The supplement pairs attainment counts at each point with the immediate fully correct event rate over all GT events.

Report correction and effects on task success For Qwen3.8-Max in VC33, SB26, and TN36, increased attainment counts were accompanied by higher immediate fully correct event rates over all GT events according to both judges. For example, at the VC33 point where movement of the yellow object was supplied, the rate increased from 45.0% to 73.7% for GLM and from 47.5% to 75.0% for Sol, while attainment increased from 1/10 to 3/10. For Opus5 in LS20 and VC33, however, the direction of change in the immediate fully correct event rate differed across judges or decision points even when attainment increased. The rate over all GT events includes reports of changes other than the target.

At some points, reports were corrected without a change in attainment. For Qwen3.8-Max in AR25, all three runs receiving information about downward movement reported the change, but attainment

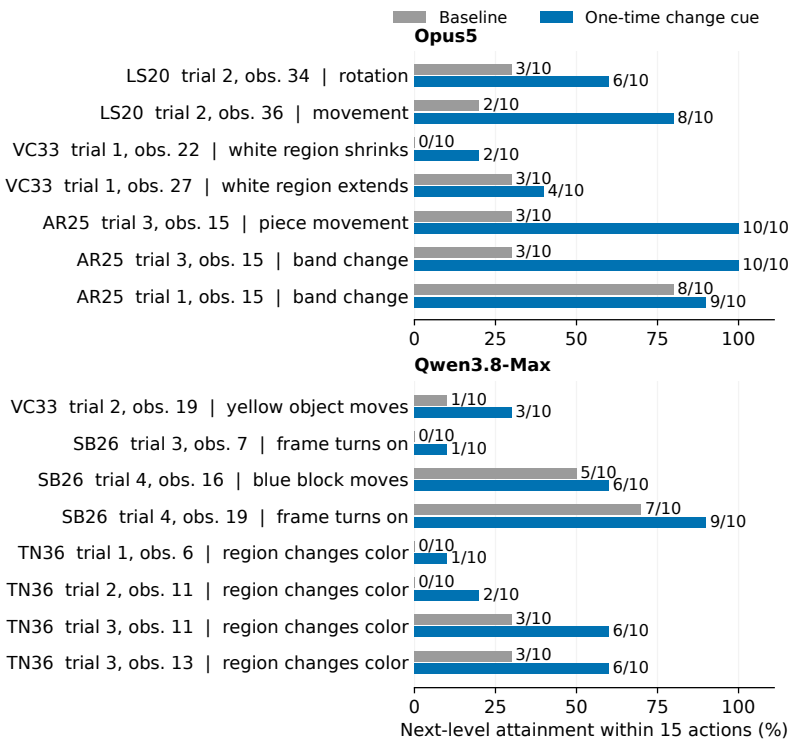


Figure 37: Level attainment under baseline and intervention conditions starting from the same state and history. Horizontal bars show attainment rates, and numbers indicate attaining runs out of 10. The piece and band conditions for AR25 original trial 3, observation 15 share a baseline.

remained 0/3 in both the baseline and intervention conditions (Appendix I.9). These results motivate evaluating the effect of supplied information separately on report correction, subsequent task success, and selectivity for the supplied information.

Linking reports to subsequent actions In representative trajectories, descriptions linking the supplied change to an operation appeared before subsequent actions. For example, in the LS20 movement case, reports shifted from describing the object's arrangement to identifying the displacement caused by an operation, after which subsequent actions and level attainment could be traced. In AR25 and VC33, reports also linked the notified change to operations. Appendix J provides the corresponding LS20 logs, actions, and screens; the AR25 and VC33 records are in the electronic supplement.

Interpretation of the exploratory comparisons Exploratory comparisons at RQ1-identified reporting failures provide condition-specific evidence that supplementing the corresponding change can affect subsequent execution. Comparisons at already-reported events, with altered restart contexts, or with broader assistance are reported separately and do not establish correction of an RQ1-diagnosed reporting error.

I.8 Continuation-Window Correctness in LS20 and AR25

For the single targeted-change intervention, we distinguish the same target event immediately after intervention from the set of events encountered by each run after branching. Table 61 aggregates the latter, using saved scores for 10 runs and 15 actions per condition. Because actions and encountered events diverge across conditions, denominators also differ. Original immediate reports, actions, and screens for LS20 are shown in Appendix J.

Fully correct event rates for all GT events in the single immediate observation are provided in the electronic supplement. Reported, entity, content, and related labels for intervention-target events

Table 61: Fully correct events/scored events (rate) over complete trajectories after single GT supplementation. The K at the end of each parenthesis is the number of valid turn-level scorings. Scores for the single immediate observation in VC33 are provided in the electronic supplement.

Scene	Condition	GLM-5.2	GPT-5.6 Sol
LS20 original trial 2, observation 34	Baseline	219/280 (78.2%; $K = 150$)	226/280 (80.7%; $K = 150$)
LS20 original trial 2, observation 34	Single supplement	190/248 (76.6%; $K = 150$)	189/248 (76.2%; $K = 150$)
LS20 original trial 2, observation 36	Baseline	205/257 (79.8%; $K = 150$)	210/257 (81.7%; $K = 150$)
LS20 original trial 2, observation 36	Single supplement	222/274 (81.0%; $K = 150$)	219/276 (79.3%; $K = 150$)
AR25 original trial 3, observation 15	Baseline	41/357 (11.5%; $K = 150$)	53/360 (14.7%; $K = 150$)
AR25 original trial 3, observation 15	Single supplement	83/358 (23.2%; $K = 149$)	103/360 (28.6%; $K = 150$)
AR25 original trial 1, observation 15	Baseline	78/348 (22.4%; $K = 149$)	91/350 (26.0%; $K = 150$)
AR25 original trial 1, observation 15	Single supplement	89/350 (25.4%; $K = 150$)	107/350 (30.6%; $K = 150$)

alone require a separate aggregation matched to judge-specific event outputs. Rates across the 15-action continuation measure reporting accuracy along each trajectory; rates for all GT events immediately afterward measure reporting for the first observation as a whole. Target-only judgments measure the immediate report of the supplied change. The original records in Appendix J show the observable reports, actions, and attainment.

I.9 Comparisons Without Improved Level Progress

For mirror-axis assistance at AR25 original trial 1, observation 21, both the baseline and assistance conditions attained the level in 2/9 runs. At R11L original trial 1, observation 29, we compared three runs each of the baseline and conditions supplying, once, the upper arm turning off, the lower arm turning on, or both changes. Level increases were 1/3, 1/3, 0/3, and 0/3, respectively. No improvement in progress from a single information supplement was observed under these conditions.

Immediate scoring of target events In R11L, a game involving selection of arms connected to a purple body, we supplied the white arm at $(27, 49)$ turning gray (turning off) and the gray arm at $(26, 54)$ turning white (turning on). For single GT supplementation, both judges scored the two target events immediately after the intervention; the per-condition counts are provided in the electronic supplement. Results are aggregated by the same runs to align report correction with level attainment.

For Qwen at AR25 original trial 1, observation 17, an exploratory comparison supplying downward-movement GT was conducted with three runs per condition; both baseline and intervention attained the level in 0/3 runs. All three intervention runs reported the downward movement immediately afterward and selected ACTION2. This case is used to examine separately whether the immediate change description is corrected and whether that correction persists through subsequent tracking and task completion.

J Response Logs: Representative Post-Intervention Traces in LS20

To illustrate differences in reports and actions with and without change information, this section shows key turns from representative baseline and single-supplement trajectories for LS20 rotation and movement. We align the initial decision in each trajectory with its first level-attainment turn or its final non-attaining turn at the budget limit. For selected turns, we retain presented screens,

notification messages, the full Reasoning Log, selected actions, post-action screens and GT, and attainment states. Original logs appear inside boxes and explanations outside.

The illustrative examples use the first replicate number for which the baseline failed to attain the level and the intervention attained it at the same scene. Complete trajectories for these comparisons will be included in the electronic supplement. All presented frames, notifications, logs, selected actions, GT, and state transitions are accessible through the run index and accompanying CSV files.

Table 62: First attainment action T for all 40 LS20 runs. A dash indicates no attainment within 15 actions. Replicate numbers are identifiers within each condition.

	Observation 34: rotation		Observation 36: movement	
Replicate	Baseline	Single supplement	Baseline	Single supplement
1	—	—	—	—
2	10	11	—	15
3	—	—	—	12
4	—	—	14	12
5	—	13	—	13
6	11	12	—	13
7	8	—	—	12
8	—	9	—	13
9	—	9	13	13
10	—	9	—	—

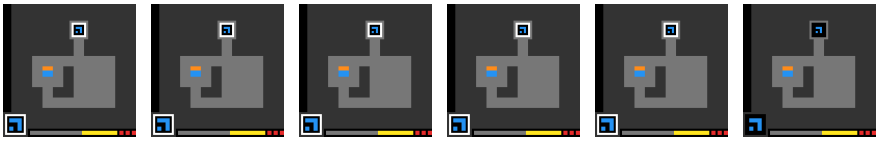
J.1 LS20 OPUS ORIGINAL TRIAL 2, OBSERVATION 34 (ROTATION)

The first report in the baseline example mentions upward movement and border flashing and describes a change in the bottom-left display ("added a filled 9 cell"), but does not state the rotation of the lower-left shape. The intervention example with supplied rotation information explicitly states $270° \rightarrow 0°$, considers the match to the reference shape, and initially selects ACTION5. The baseline example does not attain the level within 15 actions; the intervention example attains it at action 13.

J.2 BASELINE, REPLICATE 5

Turn 1 of 15 (original observation 34)

Presented screen



Frame 1 of 6 | Frame 2 of 6 | Frame 3 of 6 | Frame 4 of 6 | Frame 5 of 6 | Frame 6 of 6

GT up to the presented screen (verbatim)

```
[move]:[player]:[position(19_35)_(19_30);bbox(5_5)]:player moves up
from (19,35) to (19,30).
[bright]:[lock_1]:[state(flash);position(34_10);bbox(5_5)]:Lock 1 at
(34,10) flashes white to
indicate a match.
[match]:[lock_1]:[position(34_10);bbox(5_5)]:The piece attributes
match lock 1 at (34,10).
[bright]:[goal_highlight_border]:[state(flash);position(1_53);bbox(10_
10)]:goal_highlight_border
at
(1,53) flashes white to indicate all locks are cleared.
```

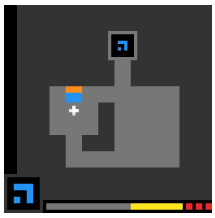
```
[rotate]:[hud_piece]:[angle(270)_(0);position(3_55);bbox(6_6)]:hud_
piece at (3,55) rotates from
270°
to 0°.
```

Reasoning Log (complete, verbatim)

```
The Up move succeeded: the piece now sits at rows 30-34, cols 19-23,
and the 0/1 goal marker there
is gone -- both side panels flashed with 0 borders (confirmation) and
the bottom-left mini-map added
a filled 9 cell at rows 59-60, cols 7-8, so a sub-goal was scored even
though the displayed score is
still 0. No new 0/1 marker is visible in the current grid yet, so the
next target may appear
elsewhere, possibly in the right-hand chamber (cols 34-53), which is
reachable only through the open
rows 25-29 above or rows 40-49 below the wall gap at cols 29-33. The
11-meter is at col 38, so I
still have time; a small probing move (e.g., Up into the open rows
25-29 corridor) would set up
access to either chamber.
```

Selected action: {"name": "ACTION1"}

After the selected action



Frame 1 of 1

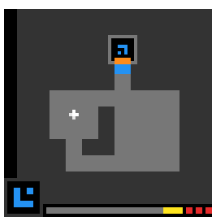
GT after the selected action (verbatim)

```
[move]:[player]:[position(19_30)_(19_25);bbox(5_5)]:player moves up
from (19,30) to (19,25).
[change]:[steps]:[count(17)_(16)]:The yellow step bar at the bottom
shrinks from the left (17→16
steps remain).
```

Attainment state: levels completed 0→0; NOT_FINISHED.

Turn 15 of 15 (original observation 48)

Presented screen



Frame 1 of 1

GT up to the presented screen (verbatim)

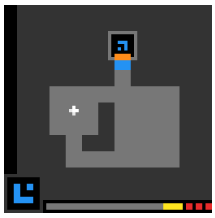
```
[change]:[steps]:[count(7)_(6)]:The yellow step bar at the bottom
shrinks from the left (7→6 steps
remain).
```

Reasoning Log (complete, verbatim)

```
Left also drained fuel with zero visual change -- the 11-bar lost
another segment (now cols 49-54, ~6
left) and the 12-block is still pinned at rows 15-16, cols 34-38.
Three different simple inputs (Up,
Enter, Left) have all been no-ops that only cost fuel, so the
directional keys clearly aren't the
interface for this puzzle; ACTION6 clicks must be. The real content
looks like a pattern-matching
task: the small 9-glyph inside the room (rows 11-13, cols 35-37)
mirrors the enlarged 9-glyph in the
bottom-left panel, with the 12-strip at rows 15-16 as the
editable/answer region. I should stop
burning fuel on keys and start clicking -- targeting the 12-strip or
the glyph cells -- to find what
actually responds.
```

Selected action: `{"name": "ACTION6", "x": 36, "y": 15}`

After the selected action



Frame 1 of 1

No GT record is available for this action.

Attainment state: levels completed 0→0; `NOT_FINISHED`.

J.3 SINGLE-SUPPLEMENT CONDITION, REPLICATE 5

Turn 1 of 15 (original observation 34)

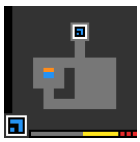
Added notification (verbatim; this turn only)

```
The displayed shape at (3,55) changed orientation from 270° to 0° (90°
clockwise).
```

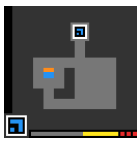
Presented screen



Frame 1 of 6



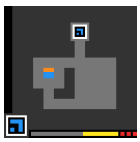
Frame 2 of 6



Frame 3 of 6



Frame 4 of 6



Frame 5 of 6



Frame 6 of 6

GT up to the presented screen (verbatim)

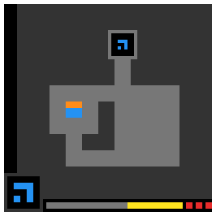
```
[move]:[player]:[position(19_35)_(19_30);bbox(5_5)]:player moves up
from (19,35) to (19,30).
[bright]:[lock_1]:[state(flash);position(34_10);bbox(5_5)]:Lock 1 at
(34,10) flashes white to
indicate a match.
[match]:[lock_1]:[position(34_10);bbox(5_5)]:The piece attributes
match lock 1 at (34,10).
[bright]:[goal_highlight_border]:[state(flash);position(1_53);bbox(10_
10)]:goal_highlight_border
at
(1,53) flashes white to indicate all locks are cleared.
[rotate]:[hud_piece]:[angle(270)_(0);position(3_55);bbox(6_6)]:hud_
piece at (3,55) rotates from
270°
to 0°.
```

Reasoning Log (complete, verbatim)

```
The Up input didn't move anything--it rotated the shape in the
bottom-left panel (event confirms
270°→0°). Decoding that panel as a 2x-scaled 5×5, its 9-cells now
read: top row 9,9,9 / middle .,.,9
/ bottom 9,.,9. That is an exact match for the target pattern shown in
the boxed reference at rows
9-15, cols 33-39 (9s at (11,35-37), (12,37), (13,35),(13,37)). Since
orientation now aligns with the
goal, the next move should be to confirm/submit rather than rotate
further, or I risk spinning past
the correct orientation.
```

Selected action: {"name": "ACTION5"}

After the selected action



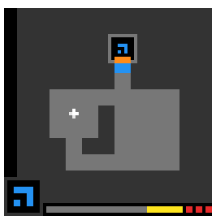
Frame 1 of 1

GT after the selected action: no recorded events.

Attainment state: levels completed 0→0; NOT_FINISHED.

Turn 13 of 15 (original observation 46)

Presented screen



Frame 1 of 1

GT up to the presented screen (verbatim)

```
[move]:[player]:[position(34_20)_(34_15);bbox(5_5)]:player moves up
from (34,20) to (34,15).
[change]:[steps]:[count(12)_(11)]:The yellow step bar at the bottom
shrinks from the left (12→11
steps remain).
```

Reasoning Log (complete, verbatim)

```
Big progress: the block now straddles the room's bottom edge --
12-rows at 15-16 and 9-rows at 17-19,
and the room's bottom wall has opened at cols 34-38 to admit it. The
room interior spans rows 9-15,
so one more upward push should slam the block against the top wall at
row 8, placing the 12-rows at
9-10 and the 9-rows at 11-13 -- exactly the rows where the target 9
marks sit (row 11 cols 35-37, row
12 col 37, row 13 cols 35 & 37). The meter shrank once more (11s begin
at col 44), so ~12 ticks
remain; I should commit now. Next action: ACTION1.
```

Selected action: {"name": "ACTION1"}

After the selected action



Frame 1 of 2



Frame 2 of 2

GT after the selected action (verbatim)

```
[move]:[player]:[position(34_15)_(34_10);bbox(5_5)]:player moves up
from (34,15) to (34,10).
[unlock]:[lock_1]:[position(34_10);bbox(5_5)]:Lock 1 at (34,10) is
unlocked.
[change]:[steps]:[count(11)_(10)]:The yellow step bar at the bottom
shrinks from the left (11→10
steps remain).
[win]:[game]:[]:The level is cleared.
```

Attainment state: levels completed 0→1; NOT_FINISHED.

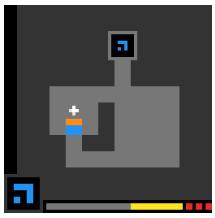
J.4 LS20 OPUS ORIGINAL TRIAL 2, OBSERVATION 36 (MOVEMENT)

The baseline example describes the arrangement of the 5×5 block and another small mark, then selects ACTION2. The intervention example with supplied movement information states that the block moved downward by five cells and relates the action to the displacement. The baseline example does not attain the level within 15 actions; the intervention example attains it at action 15.

J.5 BASELINE, REPLICATE 2

Turn 1 of 15 (original observation 36)

Presented screen



Frame 1 of 1

GT up to the presented screen (verbatim)

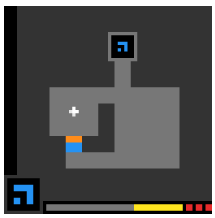
```
[move]:[player]:[position(19_30)_(19_35);bbox(5_5)]:player moves down
from (19,30) to (19,35).
[change]:[steps]:[count(17)_(16)]:The yellow step bar at the bottom
shrinks from the left (17→16
steps remain).
```

Reasoning Log (complete, verbatim)

```
A small avatar/cursor made of 0s and 1s has appeared at rows 31-33,
cols 20-22, sitting directly
above the 12/12/9/9/9 striped block (rows 35-39, cols 19-23) inside
the large 3-field. Meanwhile the
bottom progress bar of 11s has shrunk by one cell (now starting at col
39 instead of 38), which
suggests every action consumes a limited resource/timer, so I should
avoid wasted inputs. The mini
pattern boxes (bottom-left and top-right) both lost their 0-borders,
implying the previous match was
registered and a new phase started. Moving down should push the avatar
toward the 12/9 stack, which
looks like the interactive target.
```

Selected action: {"name": "ACTION2"}

After the selected action



Frame 1 of 1

GT after the selected action (verbatim)

```
[move]:[player]:[position(19_35)_(19_40);bbox(5_5)]:player moves down
from (19,35) to (19,40).
[change]:[steps]:[count(16)_(15)]:The yellow step bar at the bottom
shrinks from the left (16→15
steps remain).
```

Attainment state: levels completed 0→0; NOT_FINISHED.

Turn 15 of 15 (original observation 50)

Presented screen


Frame 1 of 6


Frame 2 of 6


Frame 3 of 6


Frame 4 of 6


Frame 5 of 6


Frame 6 of 6

GT up to the presented screen (verbatim)

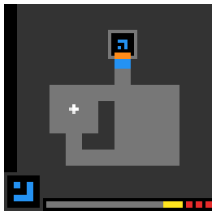
```
[bright]:[goal_highlight_frame]:[state(flash);position(1_53);bbox(10_
10)]:goal_highlight_frame
at
(1,53) flashes to indicate a rejected move.
```

Reasoning Log (complete, verbatim)

```
Up produced no movement this time -- the 12-band is still at rows
15-16 and the 9-band at 17-19 -- but
the animation frames showed the bottom-left 10x10 pattern panel flash
from 5 to 0 and back, which
reads as a rejection/mismatch signal rather than progress. The budget
bar ticked down again (now 6
cells, cols 49-54), so the block is wedged at the box's bottom border
and further up presses will
just burn budget. The likely lock is a shape/pattern mismatch: the box
interior wants the sparse 9
glyph (rows 11-13) while my block presents a solid 12 band plus 9
band, and the bottom-left panel's
9 pattern looks like the reference template. Rather than repeat a
blocked move, I should try ACTION5
to commit/transform the piece, or reconsider whether the 0/1 marker
cluster at rows 30-33, cols
20-22 is the actual controllable cursor.
```

Selected action: {"name": "ACTION5"}

After the selected action


Frame 1 of 1

No GT record is available for this action.

Attainment state: levels completed 0→0; NOT_FINISHED.

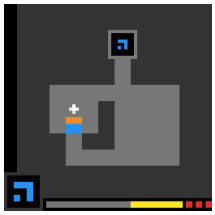
J.6 SINGLE-SUPPLEMENT CONDITION, REPLICATE 2

Turn 1 of 15 (original observation 36)

Added notification (verbatim; this turn only)

```
The 5×5 square-shaped object moved from (19,30) to (19,35).
```

Presented screen



Frame 1 of 1

GT up to the presented screen (verbatim)

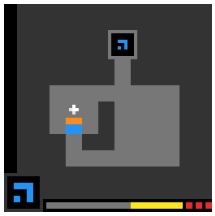
```
[move]:[player]:[position(19_30)_(19_35);bbox(5_5)]:player moves down
from (19,30) to (19,35).
[change]:[steps]:[count(17)_(16)]:The yellow step bar at the bottom
shrinks from the left (17→16
steps remain).
```

Reasoning Log (complete, verbatim)

```
ACTION2 moved the 5×5 block down five rows (from y=30 to y=35), so
each press shifts the active
piece by one 5-cell cell-unit -- the field is a coarse grid of 5×5
cells. A new small 0/1 marker
sprite appeared at rows 31-33, cols 20-22, sitting just above the 12/9
stack in the column band
19-23, which likely indicates the next drop target or the piece's
orientation handle. Both side
panels have lost their 0 borders (now all-5), suggesting those goal
boxes were consumed/completed
and the top-right box (rows 9-15) is the active reference pattern to
replicate. Next I should
continue nudging the piece toward the empty 3-region and only press
ACTION5 when the shape aligns
with the reference.
```

Selected action: `{"name": "ACTION5"}`

After the selected action



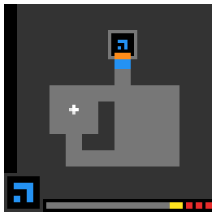
Frame 1 of 1

GT after the selected action: no recorded events.

Attainment state: levels completed 0→0; `NOT_FINISHED`.

Turn 15 of 15 (original observation 50)

Presented screen



Frame 1 of 1

GT up to the presented screen (verbatim)

```
[move]:[player]:[position(34_20)_(34_15);bbox(5_5)]:player moves up
from (34,20) to (34,15).
[change]:[steps]:[count(5)_(4)]:The yellow step bar at the bottom
shrinks from the left (5→4 steps
remain).
```

Reasoning Log (complete, verbatim)

```
The block's leading two 12-rows are now inside the box interior (rows
15-16) while its three 9-rows
trail at rows 17-19; each press still advances exactly 5 rows with no
drift. One more press should
place the whole 5-row block at rows 10-14, fully within the interior
(rows 9-15) and overlapping the
box's 9-glyph rows 11-13. The 11-bar is down to 4 units, so I have
margin for that final press plus
a spare.
```

Selected action: {"name": "ACTION1"}

After the selected action



Frame 1 of 2



Frame 2 of 2

No GT record is available for this action.

Attainment state: levels completed 0→1; NOT_FINISHED.

K Relationship to Prior Measurement Approaches

This appendix corresponds to all subsections of Related Work in the main text and organizes the information and processes directly examined by prior studies. Even when studies share the term perception, descriptions of objects in a single screen, changes between screens, history-based state updates, and rule inference measure different quantities.

Table 63: Measurement targets in prior work and this study. The final column lists observations aligned in this study.

Study	Input/reference	Process directly examined	Connection to this study
ARC-AGI-3 (ARC Prize Foundation, 2026b)	Interactive environments with unknown rules	Task performance, including harness and context management	Match individual screen changes to the next report and supply the same changes.
Sensi (Arjmandi, 2026)	Before-and-after screens, difference descriptions, and accumulated knowledge	Observer errors and subsequent knowledge updates	Separate unreported events, entity errors, and content errors against actual events, and track subsequent actions.
Rensink (Rensink, 2002)	Before-and-after visual stimuli presented to humans	Relationship between change detection and attention	Provides a conceptual reference for treating change identification separately.
ARGUS (Rawal et al., 2025)	Free-form video descriptions and human references	Omissions and unsupported content in descriptions	Match free-form descriptions during action bidirectionally against actual changes.
HalluWorld (Liu et al., 2026)	Reference world states and visible scope	Direct observation, state tracking, and temporal change	Align state changes and reports during natural execution.
MIRAGE-Bench (Zhang et al., 2025b)	Observations, history, and instructions at fixed decision points	Consistency of actions with those inputs	Score change reports and measure progress through continued execution after supplementation.
Zenkri and Brock (Zenkri & Brock, 2026)	RGB, RGB-D, and correct symbolic observations	Differences in sequential task performance across observation types	Supply a specific diagnosed change once and compare from the same scene.
AgentDebug (Zhu et al., 2025)	Execution trajectories and module-wise error classifications	Diagnosis of critical errors and re-execution with feedback	Identify mismatches between visible changes and reports by entity and content, and supply the corresponding change information.

An entity appearing in an observation is not equivalent to its change being reflected in a report. Even when a report is corrected, whether that information can be used for rule inference or action planning must be checked separately. This study examines this distinction through diagnosis and intervention involving the same screen changes.

Table 64: Processing boundaries and observables in this study. Distinctions among processes are separate from the scoring labels for reporting, entity, and content.

Process	Concrete question	Role in this study
Entity description	What is where?	Check descriptions of the same entity as context for examples denying its change.
Change identification	Which attribute of which entity changed?	The central target of matching GT and reports in RQ1.
History-based state updating	How are the preceding action and past observations reflected in the current understanding?	Examine reports under natural input histories and supply change information in RQ2.
Rule inference and action planning	Why did it change, and what should be done next?	Distinguish these from GT change descriptions and examine subsequent actions and task completion in RQ2.
Error recognition and correction	Was an earlier claim corrected in the next observation?	An adjacent process traceable in full-text trajectories; not aggregated as a separate primary metric.

L Electronic Evidence Index

The electronic supplement, to be released with the paper, will include: the observation-level RQ1 judgments of both judges and the aggregation inputs for all 400 runs; the run records of the common-rule RQ2 comparisons (Appendix I.2), each with its target mechanic, seed, observation number, GT event ID, and outcomes; the exploratory RQ2 comparisons with their target scenes, supplied change descriptions, and attainment counts; complete 15-action traces (presented frames, messages, reports, actions, GT, and attainment states for each turn) for the exploratory comparisons whose traces are shown in this appendix; and a mapping from original trials and observation numbers to the source records. Reused runs keep their original identifiers. The supplement also holds the material removed from this appendix: the bar chart of new-level attainment for all 40 common-rule conditions, the per-point attainment counts and immediate fully correct rates of the exploratory comparisons, the per-condition counts for the comparisons without improved progress (including the `R11L` target-event judgments), and the `AR25` shortened-restart observation and action responses.