REALTIME-BENCH: CAN FRONTIER MODELS SOLVE REAL-TIME INTERACTIVE PROBLEMS?

Anonymous authors
Paper under double-blind review

ABSTRACT

Real-time interaction requires agents to perceive evolving environments and act before opportunities disappear, yet existing evaluations for real-time streaming models focus on conversational quality and streaming perception, leaving their ability to act effectively under strict temporal constraints untested. We first formalize real-time capabilities for agents and introduce Realtime-Bench, a suite of dynamic game and web tasks in which delayed actions directly affect success. We evaluate six frontier LLMs spanning streaming models and standard asynchronous models under three clock protocols at low and high reasoning effort. Wall Clock advances environments with actual inference latency, while Paused Clock removes inference delays as an idealized reference. We further introduce Token Clock, which advances simulated time by the generation tokens consumed before action commitment, testing temporal awareness independently of physical serving speed. Best success rates reach only 32% under Wall Clock and 54% under Token Clock, compared with 80% under Pause Clock, revealing a substantial gap between task competence and timely action. Higher reasoning effort yields mixed effects under time pressure, reflecting a trade-off between decision quality and timely execution. Based on these results, we discuss future directions for improving real-time multimodal models, including adaptive reasoning with temporal awareness and continuous perception and action.

1 INTRODUCTION

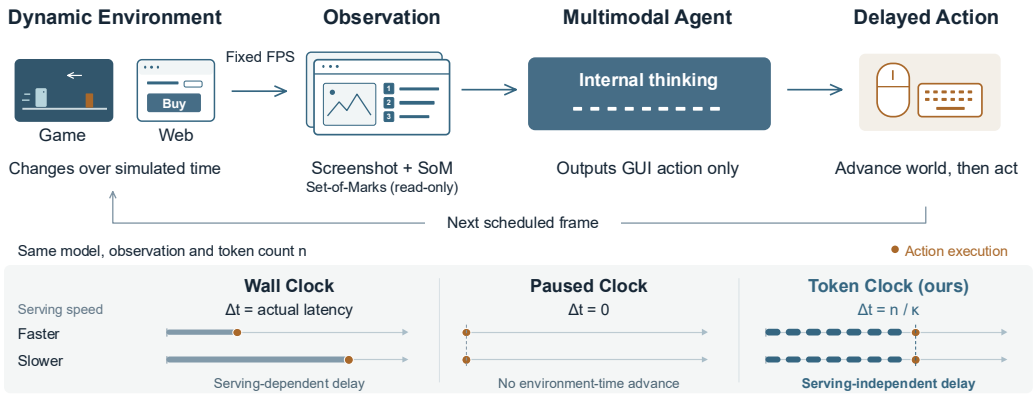


Figure 1: **Realtime-Bench evaluates models on interactive game and web tasks under three clock protocols.** Wall Clock advances environments with actual inference latency, while Paused Clock removes inference delays as an idealized reference. **Token Clock (ours)** advances simulated time according to generation tokens consumed before action commitment, testing temporal awareness of models independently of physical serving speed.

Many real-world interactive scenarios require decisions to be made under strict temporal constraints. For example, when purchasing a highly demanded ticket, detecting availability is not enough: an agent must observe the changing interface and act before the opportunity disappears. Such scenarios require *real-time streaming capabilities*, where a model continuously perceives an evolving

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.

environment and produces timely actions in response to newly arriving information. Recently, a growing family of real-time streaming models, such as GPT-Realtime (OpenAI, 2025) and Gemini Live (Google, 2026a), has begun to make this setting practically possible. In contrast to asynchronous models that process inputs only after they are fully received, these models can consume streaming inputs and react to new information with low latency.

Unlike previous benchmarks that evaluate models in static environments (Zhang et al., 2026; Tan et al., 2026) where the environment does not continuously evolve, evaluating streaming models requires dynamic settings that test whether models can perceive and act effectively under strict temporal constraints. However, existing evaluations of real-time systems largely focus on capabilities such as streaming perception, conversational quality, and interruption handling (Lin et al., 2026; Lu et al., 2026b), with limited coverage of interactive tasks in which actions affect subsequent environment states and delayed execution can change the outcome.

We first characterize *real-time streaming capability* along three complementary dimensions: temporal state awareness, adaptive deliberation, and time-aware action selection. These dimensions capture an agent's ability to track relevant changes and information freshness, adjust deliberation to the available time, and select actions that remain useful when executed. Building on this framework, we introduce **Realtime-Bench**, comprising 50 interactive game and web tasks in which evolving states and limited opportunities make success depend on both action choice and timing.

We evaluate six frontier multimodal models spanning asynchronous and streaming systems under three clock protocols (Figure 1). Following prior interactive-agent evaluations (Kang et al., 2025; Froger et al., 2026; ValsAI, 2026; Parekh et al., 2026), **Wall Clock** advances the environment according to actual inference and serving time, capturing performance under the evaluated deployment conditions. **Paused Clock** freezes the environment during inference, providing an idealized reference in which deliberation incurs no temporal cost (Tan et al., 2026; Zheng et al., 2026; Tan et al., 2025). To retain temporal pressure while controlling for physical serving speed, we introduce **Token Clock**, which advances simulated time according to the number of generation tokens consumed before an action is committed. Models that spend more tokens deliberating therefore act later, preserving the task consequences of delayed decisions without tying action timing to infrastructure-dependent inference latency. Under each clock, we test low and high reasoning effort to examine the trade-off between decision quality and timely execution.

Our evaluation shows that real-time interaction remains challenging for current frontier models, with the best success rate reaching only 32.0% under Wall Clock. Substantial performance gaps relative to Paused Clock persist under Token Clock, highlighting the difficulty of translating task competence into timely action even when temporal costs are standardized independently of serving speed. Higher reasoning effort improves success for all six models under Paused Clock but has mixed effects under temporal pressure, reflecting a tradeoff between decision quality and timely execution. Further experiments show that relaxing the temporal cost of generation generally improves performance but does not ensure task success. These findings motivate future work on adaptive reasoning with temporal awareness and continuous perception and action, enabling agents to coordinate deliberation and execution with the time available.

2 Related Work

2.1 Streaming Multimodal Models and Benchmarks

Video multimodal models have progressively evolved from offline video understanding (Maaz et al., 2024; Lin et al., 2024; Liu et al., 2024b;a; 2023) toward real-time streaming interaction, where models continuously perceive incoming observations and decide when and how to respond as the stream unfolds (Chen et al., 2024; Wang et al., 2025b; Qian et al., 2025; Cui et al., 2026). Existing training and evaluation formulate real-time interaction as responding to a continuously arriving but sparsely sampled audiovisual stream, commonly at around 1–2 FPS (Chen et al., 2024; Qian et al., 2025; Yan et al., 2026; Lu et al., 2026b). Under this setting, existing benchmarks focus on evaluating the abilities of streaming perception (Lin et al., 2026; Huang et al., 2025; Shi et al., 2026), response timing (Wang et al., 2025a; He et al., 2026; Lu et al., 2026b), interruption handling (Lin et al., 2025; Lu et al., 2026b), and conversational quality (He et al., 2026; Lu et al., 2026b) in scenarios such as surveillance anomaly detection (Wang et al., 2025a), question answering during gym exercises and

shopping, and step-by-step task guidance (Lu et al., 2026b), leaving untested whether models can act effectively in closed-loop environments where the timing of their decisions affects subsequent states and task outcomes. In contrast, Realtime-Bench evaluates closed-loop interaction in web environments and games, where task success depends on both what actions models take and when they take them.

2.2 Real-Time Evaluation in Interactive Environments

Existing interactive evaluations assess asynchronous vision-language agents in games that pause during model inference, allowing their perception, reasoning, and planning abilities to be evaluated independently of response speed (Tan et al., 2025; Zhang et al., 2026; Tan et al., 2026; Zheng et al., 2026; Ouyang et al., 2026). To evaluate agents under real-time pressure, VideoGameBench (Zhang et al., 2026), GameWorld-RT (Ouyang et al., 2026), GPTNT (Parekh et al., 2026), and Gaia2 (Froger et al., 2026) allow game or web environments to continue evolving during inference, making task success depend on both decision quality and wall-clock latency. Although such wall-clock capture the consequences of slow responses, they also tie evaluation to deployment-specific inference speed. In Realtime-Bench, we introduce Token Clock, which maps generated tokens to environment time at a shared rate, preserving the temporal cost of deliberation while decoupling evaluation from physical serving speed.

3 Setting Formulation

3.1 Real-Time Streaming Capabilities

We define real-time streaming capability as the ability to perceive, reason, and act effectively while the environment continues to evolve. We characterize this capability along three complementary dimensions. (1) *Temporal state awareness (RT-Perceive)* requires models to use available observations to track relevant changes, information freshness, and opportunity timing; (2) *Adaptive deliberation (RT-Deliberate)* requires models to adjust their reasoning length and depth according to the available time and the expected benefit of further computation; (3)*Time-aware action selection (RT-Act)* requires models to determine which action will remain useful when executed, including whether to wait or act immediately and what action to take.

These dimensions jointly shape how an agent responds to an evolving environment. For example, an agent purchasing a ticket must track the sale status and assess whether observed availability is still current *(RT-Perceive)*, limit further deliberation when the purchase window is closing *(RT-Deliberate)*, and choose between waiting for sales to open and purchasing while tickets remain available *(RT-Act)*.

3.2 Clock Protocols

As shown in Figure 1, we use three clock protocols to distinguish task performance under actual inference latency, performance without inference delays, and performance under a standardized temporal cost of token generation.

- **Wall Clock.** To evaluate performance under real-world inference latency, Wall Clock advances the environment according to the actual time spent on inference. The environment continues to evolve while the model generates a response, so action timing depends on both model computation and serving infrastructure.
- **Paused Clock.** To assess task performance without the temporal cost of inference, Paused Clock pauses the environment while the model generates a response. This provides an idealized reference for Wall Clock, allowing us to compare performance when inference does and does not delay actions in environment time.
- **Token Clock.** To retain the temporal cost of deliberation while removing dependence on serving speed, Token Clock advances the environment at a fixed rate per generation token. An action is applied at the simulated time reached when it is committed, so consuming more tokens delays execution regardless of actual inference latency.

To formalize Token Clock, let B denote the number of generation tokens corresponding to one observation interval and f the observation rate in frames per simulated second. The token-to-time rate is $\kappa = Bf$, measured in generation tokens per simulated second. If the model generates n tokens after receiving an observation and before committing to an action, the action is applied after a simulated-time delay of

$$\Delta t = \frac{n}{\kappa} = \frac{n}{Bf}.$$

In both Token Clock and Wall Clock, when generation spans multiple observation intervals, intermediate frames are skipped, and the model receives the most recent scheduled frame available at the resulting simulated time. This provides a simplified analogue of humans returning attention to the current scene after deliberation, and tests whether the model can respond before relevant events or action opportunities pass. Both protocols thus require the model to balance the benefits of additional deliberation against the cost of delayed action.

4 Benchmark Design

4.1 Design principles.

To evaluate models' real-time streaming capabilities in environments that continue to evolve while they observe, reason, and act, we design tasks around the following three criteria.

Real-world fidelity. To capture practical real-time interaction demands, task interfaces and interaction mechanics should reflect realistic applications and games. For example, web tasks simulate real-world websites such as Amazon, while game tasks feature popular games.

Dynamically evolving environment. Environment states should change over time, independently of the model's actions. These changes may include moving game elements, changing availability, or continuously updated information. For example, available seats change during cinema booking, while obstacles move across the screen in games.



Figure 2: Task distribution in Realtime-Bench. The benchmark comprises 30 web tasks and 20 game tasks across eight categories. Percentages indicate each category's share within its domain.

Temporally constrained interaction. Success should depend on both what actions the model takes and when it takes them. Deadlines, limited action windows, and transient opportunities require timely responses. For example, a model must reserve a seat before it becomes unavailable or dodge an obstacle before a collision.

4.2 Task construction.

Guided by the above criteria, we construct web tasks based on realistic application scenarios and adapt game tasks from popular open-source games. We adjust task goals and temporal constraints so that success requires timely action, while ensuring that tasks remain feasible for humans. This process yields 50 tasks: 30 web tasks and 20 game tasks. Task objectives, temporal mechanics, and success criteria are summarized in Appendix A.

4.3 Task examples and distribution.

Figure 3 illustrates two representative tasks, one from each domain: the web task requires placing a winning bid in a live auction before the bidding window closes, while the game task requires steering a moving snake to eat five apples. Figure 2 shows the distribution of task scenarios.

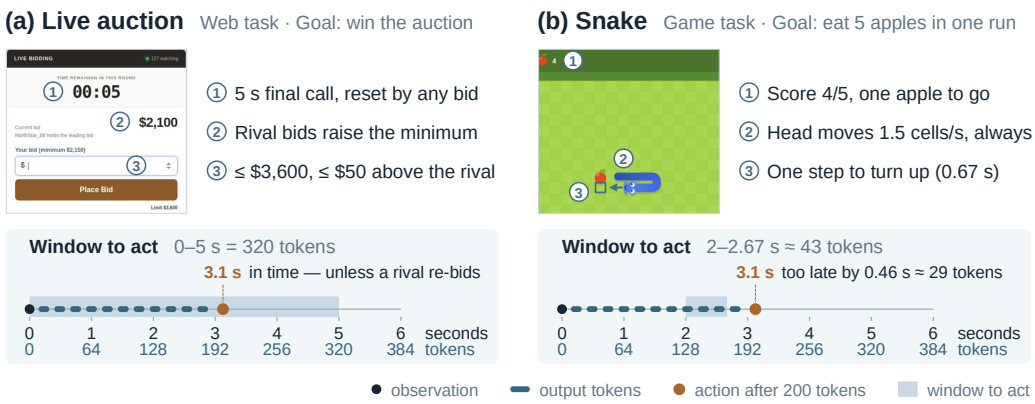


Figure 3: Representative tasks in Realtime-Bench. Each panel shows the interface at the observation, the properties that make the task time-critical (①–③), and the window to act on the token-budget clock ($\kappa = Bf = 64$ tokens per simulated second). A 200-token response lands at 3.1 s: inside the auction's 5 s final call (a), but 0.46 s past the one-cell turn window in Snake (b).

5 Evaluation Protocol

5.1 Observation and Action Interface

The environment provides observations at a fixed rate of f frames per simulated second. At each scheduled observation frame, the model receives the current screenshot together with the retained interaction history. To prevent GUI grounding from becoming a bottleneck for evaluating real-time decision making, we augment the screenshot with a read-only Set-of-Marks (SoM) representation following common GUI-agent practice (Yang et al., 2023), which identifies currently interactable elements and exposes their associated semantics and supported actions. At each decision step, the model generates executable GUI actions from a common GUI action space (e.g., `click`, `type`, `scroll`, and `wait`) (Xie et al., 2024). The detailed implementation are provided in Appendix B.1.

5.2 Context Management

To minimize harness-induced overhead when evaluating models' real-time capabilities, we follow Qwen-CUA (Lu et al., 2026a) in removing older history in blocks rather than at every step. This keeps the history prefix unchanged between folds, allowing models to reuse its KV cache across consecutive steps and reducing repeated prefill computation caused by frequent history truncation. All models follow the same history retention policy. Implementation details are provided in Appendix B.3.

5.3 Episode Execution

Suppose the model receives an observation at environment time t_i and commits an action after a delay Δt_i determined by the clock protocol. Under **Wall Clock**, Δt_i equals the actual inference latency; under **Paused Clock**, $\Delta t_i = 0$; under **Token Clock**, Δt_i is determined by the generation tokens consumed before action commitment. The action timestamp is

$$t_i^{\text{act}} = t_i + \Delta t_i.$$

The environment advances from t_i to t_i^{act} before the action is applied. If generation terminates without a valid action, the harness substitutes `wait` at the timestamp determined by the same clock protocol. GUI execution latency does not advance environment time.

An episode terminates when the environment reaches a task-defined terminal condition (e.g., game over, successful completion of a web transaction, or expiration of a task-specific countdown) or the benchmark-defined time horizon (e.g., five minutes of environment time). The scorer evaluates the terminal environment state against task-specific success criteria, such as completing the first level of a game or securing a ticket.

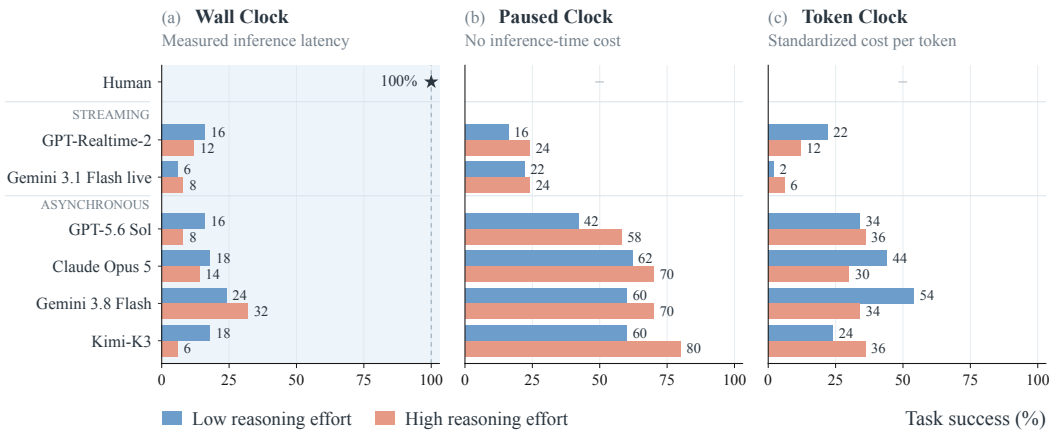


Figure 4: **Task success under three timing protocols.** We evaluate two streaming models and four asynchronous VLMs under (a) Wall Clock, which includes measured inference latency; (b) Paused Clock, which removes inference-time cost; and (c) Token Clock, which applies a standardized time cost per generated token.

6 Experiments

6.1 Experimental Setup

Models. To assess how different types of models perform under real-time constraints, we evaluate six frontier models spanning (1) real-time streaming models, including GPT-Realtime-2 (OpenAI, 2025) and Gemini 3.1 Flash live (Gemini, 2026); and (2) asynchronous vision-language models, including GPT-5.6 Sol (OpenAI, 2026), Claude Opus 5 (Anthropic), Gemini 3.8 Flash (Google, 2026b), and Kimi-K3 (Team et al., 2026).

Evaluation settings. To distinguish task performance under actual inference latency, performance without inference delays, and performance under a standardized temporal cost of token generation, we evaluate models under three clock protocols, Wall Clock, Paused Clock and Token Clock. For each clock protocol, we test both low and high reasoning effort to examine the tradeoff between decision quality and timeliness. Crossing the three clock protocols with the two reasoning levels yields six settings: WL and WH for Wall Clock, PL and PH for Paused Clock, and TL and TH for Token Clock, where L and H denote low and high reasoning effort, respectively.

Temporal configuration. We set the observation rate to $f = 1$ FPS, following the frame-sampling recipes primarily used in current training and evaluation of real-time models (Xia et al., 2026; Qian et al., 2025; Zhang et al., 2025; Team, 2026). For the main experiments, we set $B = 64$, corresponding to $\kappa = Bf = 64$ charged tokens per simulated second, a rate broadly consistent with practical autoregressive decoding speed (Yang et al., 2026; Team, 2025), which also suffice to produce a complete action together with one or two sentences of reasoning within one observation interval based on our pilot experiment. This provides room for brief deliberation that can support successful task execution, while longer reasoning can incur a temporal cost.

6.2 Results

Performance remains limited under Wall Clock. As shown in the Figure 4, under Wall Clock, the environment advances with actual elapsed time, including inference and serving delays, the best success rates are 24.0% in WL and 32.0% in WH, leaving most tasks unsolved even for the strongest evaluated model. This low success rate under realistic Wall Clock suggests that current frontier models remain insufficiently reliable for real-world deployment on real-time tasks.

Temporal costs limit success in evolving environments. With the environment paused during inference, Claude Opus 5 and Gemini 3.8 Flash each complete 70.0% of tasks in PH. At the same

high reasoning effort, their success rates fall to 14.0% and 32.0% in WH, respectively. This decline reflects the difficulty of completing tasks when the environment advances during actual inference and serving. To examine performance independently of physical serving speed, we turn to Token Clock, which assigns a standardized temporal cost to generation. Success remains substantially below Paused Clock at 30.0% for Claude and 34.0% for Gemini in TH. The gap therefore persists when action delay is determined by token consumption rather than actual serving latency. Together, these comparisons highlight a central challenge of real-time capability: sustaining effective interaction when deliberation consumes time and the environment continues to evolve.

Reasoning effort is a tradeoff under time pressure. In Paused Clock, higher reasoning effort improves success for all six models. GPT-5.6 Sol rises from 42.0% in PL to 58.0% in PH, and Claude Opus 5 from 62.0% to 70.0%. However, the effect reverses under Token Clock for Claude, with success falling from 44.0% in TL to 30.0% in TH. Kimi-K3 instead improves from 24.0% to 36.0% under the same clock. Wall Clock shows similarly mixed effects: GPT-5.6 Sol drops from 16.0% in WL to 8.0% in WH, whereas Gemini 3.8 Flash improves from 24.0% to 32.0%. These contrasting outcomes illustrate the trade-off that adaptive deliberation must manage: further reasoning may improve action selection, but its temporal cost can reduce the opportunity to execute that action successfully.

Table 1: Model latency under wall-clock evaluation.

Model	WL (s)	WH (s)
Streaming models		
GPT-Realtime-2	1.77	2.24
Gemini 3.1 Flash live	5.62	5.65
Async VLMs		
GPT-5.6 Sol	5.28	6.51
Claude Opus 5	3.60	4.74
Gemini 3.8 Flash	3.15	3.27
Kimi-K3	4.51	6.23

Relaxing temporal pressure helps but does not ensure success. To examine the role of this temporal cost, we hold reasoning effort at low and vary $B \in \{32, 64, 128, \text{unlimited}\}$ at a fixed observation rate (Figure 5). For a fixed token count, a larger B shortens the simulated delay before an action is applied. The setting of unlimited B corresponds to PL and assigns no simulated time to generation. Success generally improves as temporal pressure is relaxed: Claude Opus 5 rises from 30.0% at $B = 32$ to 62.0% with an unlimited budget, and Kimi-K3 from 22.0% to 60.0%. The improvement is not monotonic for every model, and even the unlimited setting leaves many tasks unsolved. Removing the time cost of inference therefore alleviates one constraint on task performance without resolving the full difficulty of effective interaction.

The advantage of asynchronous models narrows under Wall Clock. Under Token Clock, all four asynchronous VLMs outperform both streaming models in TL and TH. Averaged within each group, success rates are 39.0% versus 12.0% in TL and 34.0% versus 9.0% in TH. Under Wall Clock, the corresponding rates are 19.0% versus 11.0% in WL and 15.0% versus 10.0% in WH. This narrowed gap can reflect differences in measured decision latency, as shown in Table 1. GPT-Realtime-2 has the lowest mean decision latency in both Wall Clock settings, at 1.77 s and 2.24 s, compared with 3.15–5.28 s and 3.27–6.51 s for the asynchronous models. These results suggest that although asynchronous models perform better under standardized token costs, their inference and serving delays may limit how effectively this advantage translates into timely task completion in the real world.

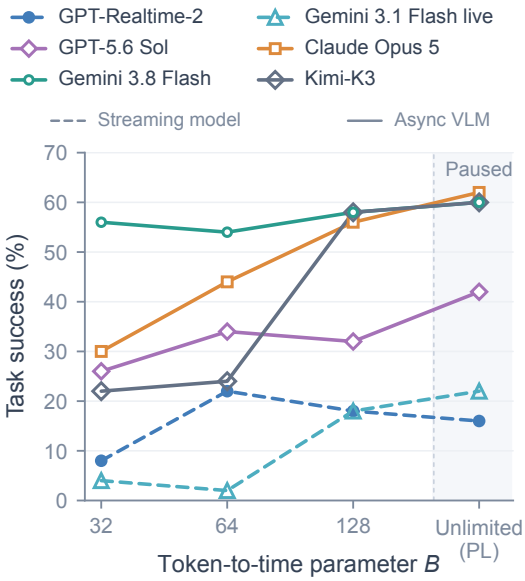


Figure 5: Effect of temporal pressure on task success.

6.3 Qualitative Analysis

By examining recorded videos and action trajectories, we identify three recurring failure modes. Figure 6 shows the distribution of failure modes across models. The annotation codebook and procedure are provided in Appendix D. More cases can be found in Appendix D.3.

Lack of temporal awareness (RT-Deliberate). Models fail to adapt reasoning length to temporal urgency. With only 10 s remaining to make a Gomoku move, Gemini 3.8 Flash in TH generates 1,845 tokens (28.8 simulated seconds), missing the opportunity to move. (Figure 7 (a)). This could stem from an emphasis on allocating reasoning effort by static question difficulty (Han et al., 2025), with insufficient adaptation to time constraints.

Failure to track the evolving state (RT-Perceive). All evaluated models sometimes persist with decisions despite new observations requiring adjustment. In Google Snake under PH, Kimi K3 keeps moving left at 37 s until hitting the wall, despite being one row below the apple (Figure 7 (b)). Such inertia may reflect over-reliance on textual context over visual changes (Leng et al., 2025), or on earlier solutions (Laban et al., 2025), highlighting the need to update decisions as states evolve.

Misjudging whether action is needed (RT-Act). Models act when they should wait, or wait when they should act. Under PH, Claude Opus 5 waits another 2 s after the road begins collapsing at 6 s, instead of steering away. (Figure 7 (c)). These errors may reflect insufficient training on when to act or wait, suggesting a training priority for future streaming models.

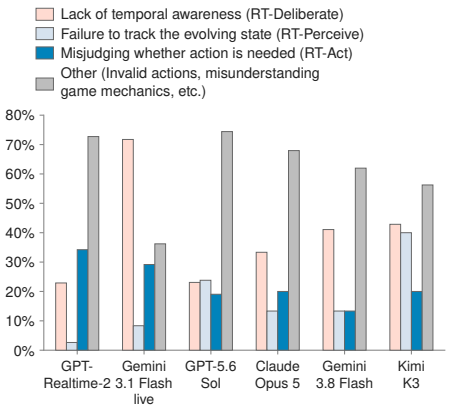
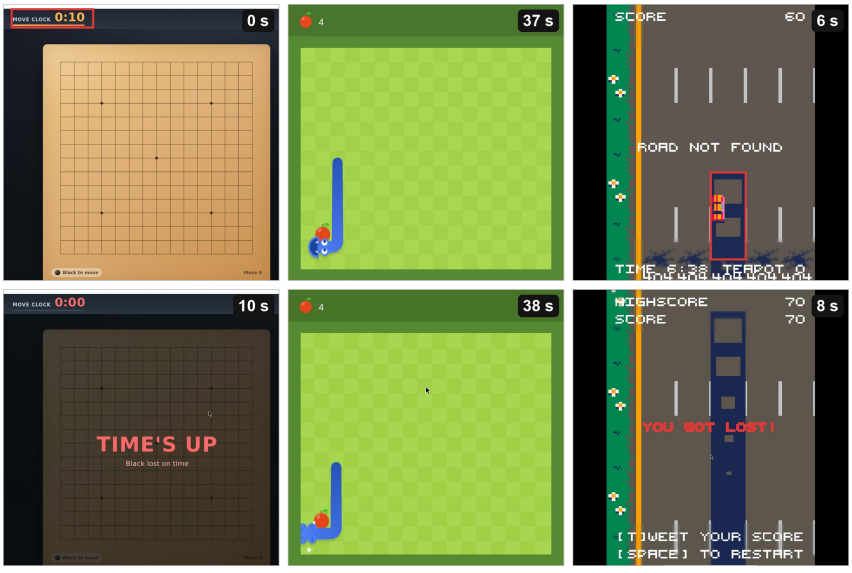


Figure 6: Distribution of failure modes across models.



(a) **RT-Deliberate**
Gomoku
Gemini 3.8 Flash, TH

(b) **RT-Perceive**
Google Snake
Kimi K3, PH

(c) **RT-Act**
Highway 404
Claude Opus 5, PH

Figure 7: **Representative examples of the three failure modes.** The top row shows the states in which models make decisions, and the bottom row shows the resulting outcomes. Timestamps are shown in the upper-right corner of each panel.

7 DISCUSSION

7.1 ADAPTIVE REASONING WITH TEMPORAL AWARENESS

Our results highlight a trade-off between the potential benefits of further reasoning and the delay it introduces. In an evolving environment, additional deliberation may improve a decision while leaving insufficient time to execute it effectively. Adaptive reasoning should therefore account for the time available to act, alongside task difficulty (Anthropic, 2026; Wu et al., 2026). In Tetris, for example, a model may spend more time comparing placements while a falling piece is far above the stack. As the piece approaches the stack, however, it must shorten deliberation to leave enough time to execute the required moves. The appropriate reasoning effort thus changes with the environment, even within the same task. Future work should train models to recognize these changing temporal constraints and allocate deliberation by weighing the expected benefit of further computation against the risk of acting too late.

7.2 CONTINUOUS PERCEPTION AND ACTION

Adapting deliberation also depends on receiving timely observations and updating actions as the environment changes. At the 1 FPS observation rate used in our experiments, brief events or rapid changes may occur between observations, even when inference itself incurs no temporal cost. For example, an enemy may change direction between two frames, preventing the agent from adjusting its action until the next observation. Higher observation rates could provide more timely feedback, but realizing this benefit requires models and harnesses that can process incoming information and revise actions with sufficiently low latency.

(1) Model design. Higher temporal resolution presents two coupled challenges: processing more frames increases inference cost, while compressing visual inputs risks discarding brief changes that determine the next action. Addressing these challenges requires optimizing both (1) processing efficiency, for example by replacing cascaded architectures with an end-to-end single Transformer using block-causal attention (Huang et al., 2026), and (2) the preservation of action-relevant information through compact temporal representations (Ding et al., 2025) or codec-aware visual encoding (Sarkar et al., 2026). Future work should explore these two directions to support higher frame rates with lower latency while preserving important information.

(2) Interaction harnesses. Real-time harnesses must manage context and memory while coordinating observations and actions in a continuously changing environment. At higher frame rates, context sizes can rapidly exceed their budgets, and ongoing inference may become stale as new observations arrive. First, effective history representations or retrieval methods are required to manage context while preserving temporal information (Zeng et al., 2025), keeping inputs within the context budget without sacrificing critical details. Second, robust strategies are needed to address observation backlogs caused by inference delays, which can render planned actions obsolete before execution. These strategies must determine how many frames to retain, whether to employ FIFO or alternative buffer-management policies (Yates et al., 2021; Li et al., 2020), and how to dynamically decide whether to discard or revise planned actions as inference becomes stale (Black et al., 2025). Future work should explore these directions to make real-time systems more robust and efficient.

8 CONCLUSION

We introduced Realtime-Bench, a suite of diverse dynamic game and web tasks for evaluating real-time interactive capability in multimodal agents. Alongside Wall Clock and Paused Clock, we proposed Token Clock to preserve the temporal cost of deliberation independently of physical serving speed. Evaluating six frontier models shows that real-time interaction remains challenging, with the best wall-clock success rate reaching only 32.0%. Higher reasoning effort generally improves performance when inference incurs no temporal cost, but has mixed effects when deliberation delays action execution. These findings highlight the importance of adapting reasoning to the time available and selecting actions that remain useful when executed. They motivate further advances in models and interaction harnesses that support adaptive deliberation, continuous perception, and timely action. Realtime-Bench provides a controlled setting for studying these capabilities and measuring progress toward effective real-time interaction.

AI USE STATEMENT

In this work, we used generative AI tools to generate evaluation environments and code, assist with failure-mode annotation of evaluation trajectories, and edit the manuscript to improve readability. All AI-generated evaluation environments and code, AI-assisted annotations, and AI-assisted text were reviewed by the authors before use. We did not use generative AI tools to generate research ideas or propose or refine hypotheses, design or provide feedback on the research methodology or experiments, or draw scientific conclusions. Beyond the annotation assistance described above, we did not use generative AI tools to interpret results. We did not use generative AI tools for any other tasks requiring disclosure. The research ideas, methods, interpretation of results, and scientific conclusions are the authors' own. We take responsibility for the final content of this work, including text, annotations, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide anonymous access to our code at `https://anonymous.4open.science/r/Streaming-to-action-0FC1/`.

REFERENCES

Anthropic. Introducing Claude Opus 5. https://www.anthropic.com/news/claude-opus-5.

Anthropic. Understand how claude's thinking works. https://platform.claude.com/docs/en/build-with-claude/thinking, 2026.

Kevin Black, Manuel Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 33383–33407. Curran Associates, Inc., 2025. doi: 10.52202/085713-1122.

Joya Chen, Zhaoyang Lv, Shiwei Wu, Kevin Qinghong Lin, Chenan Song, Difei Gao, Jia-Wei Liu, Ziteng Gao, Dongxing Mao, and Mike Zheng Shou. Videollm-online: Online video large language model for streaming video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18407–18418, June 2024.

Junbo Cui, Bokai Xu, Chongyi Wang, Tianyu Yu, Weiyue Sun, Yingjing Xu, Tianran Wang, Zhihui He, Wenshuo Ma, Tianchi Cai, Jiancheng Gui, Luoyuan Zhang, Xian Sun, Fuwei Huang, Moye Chen, Zhuo Lin, Hanyu Liu, Qingxin Gui, Qingzhe Han, Yuyang Wen, Huiping Liu, Rongkang Wang, Yaqi Zhang, Hongliang Wei, Chi Chen, You Li, Kechen Fang, Jie Zhou, Yuxuan Li, Guoyang Zeng, Chaojun Xiao, Yankai Lin, Xu Han, Maosong Sun, Zhiyuan Liu, and Yuan Yao. Minicpm-o 4.5: Towards real-time full-duplex omni-modal interaction, 2026. URL `https://arxiv.org/abs/2604.27393`.

Xin Ding, Hao Wu, Yifan Yang, Shiqi Jiang, Qianxi Zhang, Donglin Bai, Zhibo Chen, and Ting Cao. Streammind: Unlocking full frame rate streaming video dialogue through event-gated cognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 13448–13459, October 2025.

Romain Froger, Pierre Andrews, Matteo Bettini, Amar Budhiraja, Ricardo Silveira Cabral, Virginie Do, Emilien Garreau, Jean-Baptiste Gaya, Hugo Laurençon, Maxime Lecanu, Kunal Malkan, Dheeraj Mekala, Pierre Menard, Gerard Moreno-Torres Bertran, Ulyana Piterbarg, Mikhail Plekhanov, Mathieu Rita, Andrey Rusakov, Vladislav Vorotilov, Mengjue Wang, Ian Yu, Amine Benhalloum, Grégoire Mialon, and Thomas Scialom. Gaia2: Benchmarking LLM agents on dynamic and asynchronous environments. In *The Fourteenth International Conference on Learning Representations*, 2026. URL `https://openreview.net/forum?id=9gw03JpKK4`.

Gemini. Gemini 3.1 Flash Live: Making audio AI more natural and reliable. https://blog.google/innovation-and-ai/models-and-research/gemini-models/gemini-3-1-flash-live/, March 2026.

Google. Introducing Gemini 3.8 Live and 3.8 Live Extended Thinking. https://blog.google/innovation-and-ai/models-and-research/gemini-models/gemini-3-8-live-gemini-3-8-live-extended-thinking/, September 2026a.

Google. Introducing Gemini 3.8 Flash and 3.8 Flash Cyber. https://blog.google/innovation-and-ai/models-and-research/gemini-models/3-8-flash-and-3-8-flash-cyber/, September 2026b.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. Token-budget-aware LLM reasoning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 24842–24855, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.1274. URL https://aclanthology.org/2025.findings-acl.1274/.

Chaoqun He, Mingyang Xiang, Yingjing Xu, Bokai Xu, Junbo Cui, Jie Zhou, Yuan Yao, and Lijie Wen. Omni-duplexeval: Evaluating real-time duplex omni-modal interaction, 2026. URL https://arxiv.org/abs/2605.17360.

Lianghua Huang, Zhi-Fan Wu, Yupeng Shi, Wei Wang, Mengyang Feng, Junjie He, Chen-Wei Xie, Yu Liu, Jingren Zhou, Ang Wang, Bang Zhang, Baole Ai, Chen Liang, Cheng Yu, Chongyang Zhong, Jinwei Qi, Kai Zhu, Pandeng Li, Peng Zhang, Wenyuan Zhang, Xinhua Cheng, Yitong Huang, Yun Zheng, Yuxiang Bao, Yuzheng Wang, and Zoubin Bi. Wan-streamer v0.2: Higher resolution, same latency, 2026. URL https://arxiv.org/abs/2607.04443.

Zhenpeng Huang, Xinhao Li, Jiaqi Li, Jing Wang, Xiangyu Zeng, Cheng Liang, Tao Wu, Xi Chen, Liang Li, and Limin Wang. Online Video Understanding: OVBench and VideoChat-Online . In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3328–3338, Los Alamitos, CA, USA, June 2025. IEEE Computer Society. doi: 10.1109/CVPR52734.2025.00316. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52734.2025.00316.

Lars Benedikt Kaesberg, Tianyu Yang, Florian Valentin Wunderlich, Terry Ruas, Daniel Kurzawe, Jan Philip Wahle, and Bela Gipp. Is visual prompting all you need? studying vlm spatial reasoning under progressive visual scaffolds, 2026. URL https://arxiv.org/abs/2608.21170.

Hao Kang, Qingru Zhang, Han Cai, Weiyuan Xu, Tushar Krishna, Yilun Du, and Tsachy Weissman. Win fast or lose slow: Balancing speed and accuracy in latency-sensitive decisions of llms. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 150862–150884. Curran Associates, Inc., 2025. doi: 10.52202/085713-5044. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/ddaec864ba433e8889ab08dcf5c26e55-Paper-Conference.pdf.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. LLMs get lost in multi-turn conversation. In *Iclr 2026*, May 2025.

Sicong Leng, Yun Xing, Zesen Cheng, Yang Zhou, Hang Zhang, Xin Li, Deli Zhao, Shijian Lu, Chunyan Miao, and Lidong Bing. The curse of multi-modalities: Evaluating hallucinations of large multimodal models across language, visual, and audio. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference. Curran Associates, Inc., 2025. doi: 10.52202/085713-3601.

Mengtian Li, Yuxiong Wang, and Deva Ramanan. Towards streaming perception. In *Proceedings of (ECCV) European Conference on Computer Vision*, pp. 473 – 488, August 2020.

Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-LLaVA: Learning united visual representation by alignment before projection. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 5971–5984, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.342. URL https://aclanthology.org/2024.emnlp-main.342/.

Guan-Ting Lin, Jiachen Lian, Tingle Li, Qirui Wang, Gopala Anumanchipalli, Alexander H. Liu, and Hung-Yi Lee. Full-duplex-bench: A benchmark to evaluate full-duplex spoken dialogue models on turn-taking capabilities. In *2025 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 1–8, 2025. doi: 10.1109/ASRU65441.2025.11433838.

Junming Lin, Zheng Fang, Chi Chen, Haoxuan Cheng, Zihao Wan, Fuwen Luo, Ziyue Wang, Peng Li, Yang Liu, and Maosong Sun. Streamingbench: Assessing the gap for mllms to achieve streaming video understanding. In *ICASSP 2026 - 2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 12147–12151, 2026. doi: 10.1109/ICASSP55912.2026.11463959.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26296–26306, June 2024a.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, January 2024b. URL https://llava-vl.github.io/blog/2024-01-30-llava-next/.

Dunjie Lu, Shuai Bai, Tianyi Bai, Sicheng Fan, Chang Gao, Jian Guan, Feng Hu, Mianqiu Huang, Xingyang Huang, Yizhen Jiang, Yuheng Jing, Dehui Kong, Ning Li, Dayiheng Liu, Shixuan Liu, Zheng Liu, Que Shen, Bowen Wang, Junli Wang, Chencan Wu, Rui Xie, Tianbao Xie, Zhihui Xie, Haiyang Xu, An Yang, Tao Yu, Wenzhen Yuan, Xi Zhang, Zhenru Zhang, Mingkang Zhu, Zhaoqing Zhu, Yizhong Cao, Kai Dang, Binyuan Hui, Kaixin Li, Junyang Lin, Haiquan Wang, Zekun Wang, Yiheng Xu, Fan Yan, Mengqi Yuan, Danyang Zhang, Jiajun Zhang, Zhipeng Zhang, Fan Zhou, and Fan Zhou. Qwen-cua: Native computer use for (almost) everything, 2026a. URL https://arxiv.org/abs/2608.02352.

Xudong Lu, Xueying Li, Annan Wang, Yang Bo, Jinpeng Chen, Zengliang Li, Nianzu Yang, Rui Liu, Xue Yang, Jingwen Hou, and Hongsheng Li. Omniinteract: Benchmarking real-world streaming interaction for real-time omnimodal assistants, 2026b. URL https://arxiv.org/abs/2605.26485.

Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Khan. Video-ChatGPT: Towards detailed video understanding via large vision and language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12585–12602, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.679. URL https://aclanthology.org/2024.acl-long.679/.

OpenAI. Introducing gpt-realtime and realtime api updates for production voice agents. https://openai.com/index/introducing-gpt-realtime/, August 2025.

OpenAI. Previewing GPT-5.6 Sol: A next-generation model. https://openai.com/index/previewing-gpt-5-6-sol/, September 2026.

Mingyu Ouyang, Siyuan Hu, Kevin Qinghong Lin, Hwee Tou Ng, and Mike Zheng Shou. Gameworld: Towards standardized and verifiable evaluation of multimodal game agents, 2026. URL https://arxiv.org/abs/2604.07429.

Amit Parekh, Sabrina McCallum, Kareem Al-Hasan, Malvina Nikandrou, Alessandro Suglia, and Ioannis Konstas. Gptnt: Benchmarking real-time collaboration between multimodal agents on keep talking and nobody explodes, 2026. URL https://arxiv.org/abs/2606.28514.

Rui Qian, Shuangrui Ding, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Yuhang Cao, Dahua Lin, and Jiaqi Wang. Dispider: Enabling video llms with active real-time interaction via disentangled perception, decision, and reaction. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24045–24055, 2025. doi: 10.1109/CVPR52734.2025.02239.

Sayan Deb Sarkar, Rémi Pautrat, Ondrej Miksik, Marc Pollefeys, Iro Armeni, Mahdi Rad, and Mihai Dusmanu. Cope-videolm: Leveraging codec primitives for efficient video language modeling, 2026. URL https://arxiv.org/abs/2602.13191.

Yansong Shi, Qingsong Zhao, Tianxiang Jiang, Xiangyu Zeng, Yi Wang, and Limin Wang. RIVER: A real-time interaction benchmark for video LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=xmtvHH62Ic.

Weihao Tan, Wentao Zhang, Xinrun Xu, Haochong Xia, Ziluo Ding, Boyu Li, Bohan Zhou, Junpeng Yue, Jiechuan Jiang, Yewen Li, Ruyi An, Molei Qin, Chuqiao Zong, Longtao Zheng, Yujie Wu, Xiaoqiang Chai, Yifei Bi, Tianbao Xie, Pengjie Gu, Xiyun Li, Ceyao Zhang, Long Tian, Chaojie Wang, Xinrun Wang, Börje F. Karlsson, Bo An, Shuicheng Yan, and Zongqing Lu. Cradle: empowering foundation agents towards general computer control. In *Proceedings of the 42nd International Conference on Machine Learning*, ICML'25. JMLR.org, 2025.

Weihao Tan, Changjiu Jiang, Yu Duan, Mingcong Lei, Jiageng Li, Yitian Hong, Xinrun Wang, and Bo An. Stardojo: Benchmarking open-ended behaviors of agentic multimodal llms in production-living simulations with stardew valley, 2026. URL https://arxiv.org/abs/2507.07445.

Gemini Team. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities, 2025. URL https://arxiv.org/abs/2507.06261.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y Charles, et al. Kimi k3: Open frontier intelligence. *arXiv preprint arXiv:2607.24653*, 2026.

Qwen Team. Qwen3.5-omni technical report, 2026. URL https://arxiv.org/abs/2604.15804.

ValsAI. Cua-bench. https://vals.ai/benchmarks/cua_bench, 2026.

Yueqian Wang, Xiaojun Meng, Yifan Wang, Huishuai Zhang, and Dongyan Zhao. Proactivevideoqa: A comprehensive benchmark evaluating proactive interactions in video large language models, 2025a. URL https://arxiv.org/abs/2507.09313.

Yueqian Wang, Xiaojun Meng, Yuxuan Wang, Jianxin Liang, Jiansheng Wei, Huishuai Zhang, and Dongyan Zhao. VideoLLM knows when to speak: Enhancing time-sensitive video comprehension with video-text duet interaction format. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 6338–6359, Suzhou, China, November 2025b. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.336. URL https://aclanthology.org/2025.findings-emnlp.336/.

Chao Wu, Baoheng Li, Mingchen Gao, Yu Tian, and Zhenyi Wang. From efficiency to adaptivity: A deeper look at adaptive reasoning in large language models, 2026. URL https://arxiv.org/abs/2511.10788.

Jiaer Xia, Peixian Chen, Mengdan Zhang, Xing Sun, and Kaiyang Zhou. Streaming video instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 31219–31229, June 2026.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 52040–52094. Curran Associates, Inc., 2024. doi: 10.52202/079017-1650.

Weicai Yan, Yuhong Dai, Qi Ran, Haodong Li, Wang Lin, Tao Jin, Xing Xie, Hao Liao, and Jianxun Lian. Proact-VL: A proactive VideoLLM for real-time AI companions. In *Proceedings of the 43rd International Conference on Machine Learning*, Proceedings of Machine Learning Research, 2026.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v, 2023. URL https://arxiv.org/abs/2310.11441.

Penghui Yang, Cunxiao Du, Fengzhuo Zhang, Haonan Wang, Tianyu Pang, Chao Du, and Bo An. LongSpec: Long-context lossless speculative decoding with efficient drafting and verification. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1826–1844, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.83. URL https://aclanthology.org/2026.acl-long.83/.

Roy D. Yates, Yin Sun, D. Richard Brown, Sanjit K. Kaul, Eytan Modiano, and Sennur Ulukus. Age of information: An introduction and survey. *IEEE Journal on Selected Areas in Communications*, 39(5):1183–1210, 2021. doi: 10.1109/JSAC.2021.3065072.

Xiangyu Zeng, Kefan Qiu, Qingyu Zhang, Xinhao Li, Jing Wang, Jiaxin Li, Ziang Yan, Kun Tian, Meng Tian, Xinhai Zhao, Yi Wang, and Limin Wang. StreamForest: Efficient online video understanding with persistent event memory. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 75804–75835. Curran Associates, Inc., 2025. doi: 10.52202/085713-2547.

Alex L. Zhang, Thomas L. Griffiths, Karthik R. Narasimhan, and Ofir Press. Videogamebench: Can vision-language models complete popular video games?, 2026. URL https://arxiv.org/abs/2505.18134.

Haoji Zhang, Yiqin Wang, Yansong Tang, Yong Liu, Jiashi Feng, and Xiaojie Jin. Flash-vstream: Efficient real-time understanding for long video streams. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 21059–21069, 2025. doi: 10.1109/ICCV51701.2025.01957.

Xiangxi Zheng, Linjie Li, Zhengyuan Yang, Ping Yu, Alex Jinpeng Wang, Rui Yan, Yuan Yao, and Lijuan Wang. V-MAGE: A game evaluation framework for assessing vision-centric capabilities in multimodal large language models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 17707–17758, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.878. URL https://aclanthology.org/2026.findings-acl.878/.

A TASK SPECIFICATIONS

Realtime-Bench contains 20 game tasks and 30 web tasks. Tables 2 and 3 summarize their objectives, temporal mechanics, and success criteria. Task-specific deadlines may be shorter than the benchmark's five-minute episode limit. Success is determined by the task scorer from the environment state, rather than by a model's declaration of completion.

A.1 GAME TASKS

Game tasks require agents to control evolving game states or complete objectives under a running clock. The evaluation specifies a concrete target, such as reaching a score, completing a level, or satisfying a survival condition.

Table 2: Game task specifications. The last column states the objective and the condition required for success.

Task	Temporal mechanics	Objective and success criterion
Timed Chess	The player's non-recoverable clock runs during White's turns.	Capture at least four black pieces against the Easy AI before the one-minute clock expires.
Astray	Ball motion and a 90-second countdown continue as the agent acts.	Reach the exit of the first maze before the countdown expires.
Doomsday Kitchen	Customers arrive during a one-minute kitchen countdown; cooking and delivery compete for attention.	Prepare and deliver one order each of ZOMBURGER, DFD, and ZOMBEER before time expires.
Wordle	Each guess has a 10-second limit; expiration locks the board.	Identify the word, producing five green tiles in the current round.
Gomoku	Each Black move has a 10-second limit.	Connect five before the Easy AI while respecting every move deadline.
Tetris	Blocks fall continuously.	Clear at least one full row.
Google Snake	The snake continues moving between decisions.	Eat at least five apples in one run.
Monkey Mart	Harvesting, stocking, and customer service unfold over time.	Increase funds from 35 to at least 60 coins.
Pacman	Ghosts and the player move through the maze.	Reach at least 600 points by eating pellets.
JavaScript Pong	The ball and opposing paddle move continuously.	Score three points before the computer does.
SYNTH*BLAST	Movement and combat continue under a 180-second task countdown.	Activate all yellow target pads in Levels 1 and 2 and reach Level 3 before time expires.
Norman the Necromancer	Villagers advance in waves; attacks and resurrection have recharge times.	Accumulate at least 200 souls before all five hearts are lost.
xx142-b2.exe	A 13-second route must be coordinated with a replay ghost and a timed door.	Open the door and escape.
Evolve	RNA and DNA availability changes over time.	Acquire the four required evolution upgrades.
shapez	Extraction and conveyor transport take time.	Build a factory that completes Level 1.
Mario	Platforming and moving hazards require timely control.	Score at least 200 points in the current level.
Space Invaders	Enemies and projectiles move continuously.	Destroy all 40 enemies and win the round.

Continued on next page

Table 2 (continued)

Task	Temporal mechanics	Objective and success criterion
Elemental One	Platform traversal and obstacle interactions occur under a 120-second task countdown.	Obtain the water and earth elements, clear the corresponding obstacles in Levels 1 and 2, and reach Level 3 before time expires.
Highway 404	Lanes collapse while the vehicle moves.	Collect a teapot and survive 30 native game seconds.
Breakout	The ball moves continuously and must be intercepted by the paddle.	Clear every brick, reaching 100% progress.

A.2 WEB TASKS

Web tasks involve changing availability, asynchronous updates, and actions whose validity depends on execution time. Successful completion requires satisfying both the final objective and the task's ordering, timing, and preservation constraints.

Table 3: Web task specifications. Deadlines expressed as clock times refer to the time displayed in the task interface.

Task	Temporal mechanics	Objective and success criterion
Live auction	Competing bids arrive rapidly; every accepted bid restarts a five-second final call.	Win Lot 184 for at most $3,600, without exceeding the last competing bid by more than $50 or submitting an over-budget bid.
Shared document	Collaborators revise dependencies for 55 seconds, leaving five seconds to save.	After their edits stabilize, save the earliest feasible launch date in the designated cell before 60 seconds, preserving all other cells.
Market dip purchase	The BTC price and confirmation quote change over time.	Submit and confirm a purchase of 0.02 BTC only while the price is below $60,000 and the quote remains valid.
Flight gate assignment	Five flights arrive sequentially and their estimated arrival times change.	Submit a compatible, conflict-free assignment to G12, G16, and G20, with valid turnaround intervals and at most one passenger gate change.
Canary rollback	Error-rate samples arrive over a rolling window.	Roll back only the canary whose error rate exceeds 5% at every sample in the most recent complete 10-second window.
Grocery substitution	Requests expire after approximately 12 seconds; stock and prices change.	Retain milk and bread, choose the cheapest compliant substitutes, ensure peanut-free items of at least 80% of the original size, and keep the total upcharge within $8 through checkout.
Fleet dispatch	Vehicle status, location, and estimated arrival time update each second.	Dispatch the nearest available ALS unit that can arrive within eight minutes, without diverting a unit transporting a patient.
Reservation release	Cancelled tables appear briefly and may be taken by others.	Reserve a table for four at Juniper & Stone on October 16 between 7:00 and 8:30 PM, preferring the earliest dining-room option.
Fallback booking	The preferred driver has a 45-second response window and may confirm late.	Secure exactly one driver for the specified JFK pickup; use the fallback only after the response window expires and reject a conflicting late confirmation.
Flight check-in	Check-in opens during the episode and seat availability changes.	Check in both passengers for AS 482 using the best available pair, prioritizing adjacency, then a window seat, then the forward cabin.

Continued on next page

Table 3 (continued)

Task	Temporal mechanics	Objective and success criterion
Delivery method	Eligible delivery and pickup options appear and disappear during checkout.	Order the specified desk converter with fulfillment costing at most $15 and completion no later than October 16.
Address validation	The service map updates after approximately seven seconds; quotes lock after 15 seconds.	Select the valid address variant for 44 Hudson St Apt 5B that arrives by October 16 at the lowest delivery fee, and place exactly one order.
Flight rebooking	Alternative flights can sell out during rebooking.	Replace the cancelled AS 482 with a compliant itinerary arriving before 10:00 PM that day, with at most one connection.
Hotel booking	Room inventory, prices, cancellation policies, and temporary holds change.	Book a King room at Harbor House Boston for October 16–18 with free cancellation and a total price of at most $620.
Accessible trip dispatch	Capacity changes after a no-show and a newly activated return trip.	Assign Lena to the only suitable vehicle within 90 seconds, cancel the no-show's return by 14:04, assign Nina to the freed capacity, and obtain both pickup confirmations by 14:05.
Restaurant seating	A table becomes available, another is blocked, and a party changes its seating request.	Seat Casey's party of four within approximately 90 seconds and Ellis's party of six within approximately four minutes, preserving reservations and seated diners.
Waitlist booking	A new booking removes availability; client replies and confirmation delivery are delayed.	Contact the longest-waiting compatible client before 12:00, use the next compatible client if necessary, and deliver confirmation of an agreed booking before 12:02:45.
Driver handoff	A driver replacement invalidates an earlier pickup acknowledgment.	Obtain acknowledgment of a step-free entrance within 60 m in the first 90 seconds, then obtain a fresh acknowledgment from the replacement driver before arrival.
Document revision	Colleagues confirm details asynchronously; withdrawal, resend, and completion have separate deadlines.	Withdraw the old agreement before 14:01:30, update confirmed details and signing order, resend by 14:04, and complete the accounts copy by 14:05.
Return and exchange	Two parcels arrive separately; stock handover and payment constrain processing.	Register and restock the first parcel before 16:57, record the damaged unit's rejection in the second parcel, and release only eligible replacements after shipping payment.
Identity recovery	Verification, replacement-device activation, and genuine authentication occur sequentially.	Restore MFA to Active before 10:01:30; activate the confirmed replacement phone and send instructions; retire old personal phones after its first genuine VPN push approval and before 10:04:45, preserving the replacement and shared phone.
Bundle maintenance	A component rename invalidates the bundle; later repricing changes valid totals.	Restore the bundle with only shared colors and combined matching options before 16:02:15; update each price to the component sum minus $5 before 16:04:18, keeping promotion orders valid through 16:04:45.
Cruise cancellation	Customer choices arrive asynchronously; refunds and emails require processing time.	Close both departures before 12:01, settle each order according to the customer's choice, deliver required notices and complete refunds by 12:04:45, and keep cancelled departures closed without changing unrelated bookings.
Shipment regrouping	Packing and carrier deadlines surround a newly imported order.	Void the incorrect label before 14:12, split the service groups, and incorporate the new order by 14:15 so each service has one usable label covering every item exactly once.

Continued on next page

Table 3 (continued)

Task	Temporal mechanics	Objective and success criterion
Inspection correction	A late failed inspection conflicts with an urgent dispatch request.	Move the inspection and fault to Shuttle 43 and hold it before 09:57; keep Shuttle 34 held after its failed release check, create the new issue, and identify the fit vehicle and correct reason before 10:00.
Address correction	Support confirms an address and later corrects it before collection.	Hold fulfillment before 12:01, update and release after confirmation, then repeat after the apartment correction before 12:03:15, preserving paid items and payment.
Room maintenance	The reopening date is extended after the initial room move and block.	Move Maya to a suitable clean Double before 15:01:20 without changing her stay terms; after the extension, rehouse Priya and extend the continuous room block through September 23 before 15:03:50.
Security advisory	Pull-request readiness, release publication, and CVE issuance occur at different times.	Accept the credible report, preserve credit, merge all required patches only when ready, use the released 2.8.5 metadata, obtain the CVE, and publish before 13:09:48 without changing the other report.
Privacy erasure	A legal hold precedes data collection; processor completion arrives asynchronously.	Pause before 09:42:43, resume after release, delete eligible records while preserving required and unrelated data, await the genuine processor response, send the retention notice, and close before 09:45:57.
Promotion replacement	Checkout deployment and a partial then complete catalog sync precede two reviews.	Publish the existing replacement promotion with $15 off orders over $75 and code TRAIL15 for all eligible Trail Bags, then end the old promotion before 13:24 without a coverage gap.

B Evaluation Implementation

B.1 Observations and Actions

Observations. The agent receives a screenshot and a read-only description of currently interactable elements. Screenshots preserve the desktop aspect ratio and are resized to a maximum dimension of 768 pixels, yielding 768×432 images for the 1920×1080 desktop. We adopt this reduced resolution to reduce visual processing overhead and inference latency. Recent studies also suggest that SoM representations preserve visual grounding performance under downsampling with minimal loss (Xie et al., 2024; Kaesberg et al., 2026).The Set-of-Marks representation uses `P#` for discrete target points and `R#` for continuous regions. A region location is specified as `R#@u,v`, where $u, v \in [0, 1000]$ are local coordinates. Marks refer only to the observation in which they appear. Raw screenshot coordinates remain available when no suitable mark exists and are mapped to desktop coordinates before execution.

Actions. Each decision produces a GUI action. Table 4 gives the shared action schema.

B.2 Clock Execution and Token Accounting

Action timing. For a decision based on an observation at time t_i, the token clock assigns an action timestamp $t_i^{\text{act}} = t_i + n_i/(Bf)$, where n_i is the charged output-token count. The harness advances the environment to this timestamp before applying the action. Finishing early does not provide an extra observation: the next observation follows the scheduled frame grid. If generation crosses observation boundaries, the corresponding intermediate observations are skipped. Physical inference and GUI execution latency do not advance simulated time. Browser animation updates use a persistent 60 Hz simulation grid, separate from the agent's observation rate, so splitting an interval around an action does not change the number of animation updates.

Charged tokens. For API-based models, we charge both reasoning tokens and output tokens using the usage reported by the API for the complete response. As Gemini Live natively produces audio rather than text, we configure it to emit GUI actions through native function calls instead of

Action	Arguments	Semantics
`click`	target, button	Click the specified point.
`right_click`	target	Right-click the specified point.
`double_click`, `triple_click`	target	Apply the corresponding click sequence.
`mouse_down`, `mouse_up`	target, duration	Press or release the mouse button.
`move`	target, duration	Move the pointer.
`type`	text	Enter the supplied text.
`key`	key, repeat	Tap a key or key combination.
`hold_key`	key, duration	Hold a key or key combination for the specified time.
`scroll`	target, dx, dy	Scroll horizontally or vertically.
`drag`	start, end, duration	Drag between two points.
`wait`	duration	Advance without issuing another GUI input.

Table 4: Shared GUI action schema. Arguments include optional fields; action durations lie in $[0, 10]$ seconds and `key.repeat` lies in $[1, 100]$.

spoken responses. We charge its reported thought tokens and non-audio response tokens, excluding generated audio.

B.3 Context Management

Visual and textual history are retained separately. When the active visual history reaches 21 observations, the oldest 10 screenshots and their associated interactables are replaced with fixed placeholders, while the corresponding text and action records remain. When detailed interaction history exceeds 50 turns, the oldest 10 complete turns are replaced by a compact executed-action ledger. This block policy retains recent evidence while limiting changes to reusable context prefixes.

Streaming models maintain the retained history within their sessions, while asynchronous models resubmit the retained context. To maximize cache reuse and reduce inference latency, we preserve the order and content of retained messages and remove older history only in blocks when the retention limits are exceeded. This keeps the shared context prefix stable across consecutive decisions, allowing models to reuse cached computation where supported. Both mechanisms implement the same logical history policy.

B.4 Model Configuration

We access GPT-Realtime-2 through OpenAI and Gemini 3.1 Flash Live through the Google Gemini Live API. For asynchronous VLMs, we access GPT-5.6 Sol through OpenRouter, Claude Opus 5 through Anthropic API, Gemini 3.8 Flash through OpenRouter, and Kimi-K3 through the Kimi API. We control reasoning effort directly through the configuration fields exposed by each provider's API, selecting the corresponding low and high settings for our experimental conditions.

C Supplementary Results

C.1 Performance by Domain

Tables 5 and 6 report success rates on the 20 Game and 30 Web tasks, respectively.

C.2 Token-Rate Ablation

Table 7 varies the token-to-time conversion rate B at $f = 1$ frame per second under low reasoning effort.

Model	T_L	T_H	P_L	P_H	W_L	W_H
GPT-Realtime-2	25.0	5.0	10.0	15.0	15.0	10.0
Gemini Live	5.0	5.0	5.0	10.0	0.0	0.0
GPT-5.6 Sol	25.0	20.0	30.0	50.0	15.0	10.0
Claude Opus 5	30.0	20.0	40.0	40.0	10.0	10.0
Gemini 3.8 Flash	45.0	25.0	40.0	50.0	20.0	20.0
Kimi-K3	15.0	15.0	30.0	60.0	10.0	5.0

Table 5: Game success rates (%) over all 20 tasks. T, P, and W denote token-clock, paused, and wall-clock conditions; L and H denote low and high reasoning effort.

Model	T_L	T_H	P_L	P_H	W_L	W_H
GPT-Realtime-2	20.0	16.7	20.0	30.0	16.7	13.3
Gemini Live	0.0	6.7	33.3	33.3	10.0	13.3
GPT-5.6 Sol	40.0	46.7	50.0	63.3	16.7	6.7
Claude Opus 5	53.3	36.7	76.7	90.0	23.3	16.7
Gemini 3.8 Flash	60.0	40.0	73.3	83.3	26.7	40.0
Kimi-K3	30.0	50.0	80.0	93.3	23.3	6.7

Table 6: Web success rates (%) over all 30 tasks. T, P, and W denote token-clock, paused, and wall-clock conditions; L and H denote low and high reasoning effort.

Model	$B = 32$	$B = 64$	$B = 128$	$B = \infty$
GPT-Realtime-2	8.0	22.0	18.0	16.0
Gemini Live	4.0	2.0	18.0	22.0
GPT-5.6 Sol	26.0	34.0	32.0	42.0
Claude Opus 5	30.0	44.0	56.0	62.0
Gemini 3.8 Flash	56.0	54.0	58.0	60.0
Kimi-K3	22.0	24.0	58.0	60.0

Table 7: Token-rate ablation: overall success rates (%) over all 50 tasks under low reasoning effort. The $B = \infty$ condition removes simulated inference delay and uses the paused setting.

D Qualitative Analysis

D.1 Scope and procedure.

By inspecting failed trajectories, the first author identified three failure modes and developed an annotation codebook specifying operational definitions, inclusion criteria, and exclusion rules. For lack of temporal awareness (RT-Deliberate), we annotated failures under TL and TH to examine whether models accounted for the temporal cost of deliberation under standardized token-based timing; Paused Clock removes this cost, whereas Wall Clock additionally reflects deployment-dependent latency. For failure to track the evolving state (RT-Perceive) and misjudging whether action is needed (RT-Act), we annotated failures under PH, which pauses the environment during generation and uses high reasoning effort, allowing us to examine state tracking and action necessity judgments without environment changes during inference. We then prompted Codex to apply the codebook to the recorded action trajectories, timing information, available model reasoning, and observations. All failed trajectories in the selected conditions were screened, with targeted inspection of video frames to verify candidate events rather than exhaustive frame-by-frame review. The second author checked the resulting annotations against the source evidence.

D.2 ANNOTATION CODEBOOK.

Prompt 1: Failure-mode annotation codebook

ANNOTATION SCOPE

Annotate valid failed episodes from the selected runs. Assess RT-Deliberate only in TL and TH, where charged output tokens advance simulated task time. Assess RT-Perceive and RT-Act only in PH. An episode–category pair outside these settings is outside the selected annotation scope, not a negative judgment.

FAILURE MODES

1. **Lack of temporal awareness (RT-Deliberate).** Generation latency is mismatched to the available response window, consuming time needed to observe, act, or complete a required sequence. Qualifying evidence includes generation that exhausts a model-visible or task-stated deadline, leaves insufficient time for necessary subsequent steps, or delays a response to a visibly evolving hazard. Cumulative generation costs may qualify when they contribute before the opportunity becomes irrecoverable.

   A timeout, long generation, or final rejected action alone is insufficient. Do not infer a usable response window solely from hidden backend timing when the visible deadline has already expired. Short generation does not qualify merely because more time remained. Distinguish charged output tokens from reasoning tokens. Generation-related latency does not apply in PH; explicit waiting there is assessed under RT-Act.

2. **Failure to track the evolving state (RT-Perceive).** The model fails to incorporate task-relevant current information and subsequently behaves according to a stale or incorrect state. Examples include overlooking an updated request, reverting to superseded preferences, misidentifying the currently active game object, and incorrectly assuming that a workflow has completed. Both external changes and action-induced interface changes qualify.

   Require evidence of a discrepancy between the available current state and the model's representation or behavior. Generic coordinate errors, repeated actions on an unchanged screen, and incorrect predictions of future states do not qualify by themselves.

3. **Misjudging whether action is needed (RT-Act).** The model executes an action when it should wait, or explicitly waits when intervention is required. Examples include committing before a prerequisite is satisfied, waiting through an actionable opportunity, and unnecessarily repeating a completed action in a way that reverses progress.

   Require evidence about the current readiness state, the executed action or wait, and its consequence. Rejected actions, parser-induced fallback waits, and reasonable waits for pending processing do not establish intentional action-timing errors.

EVIDENCE REQUIREMENTS

For each candidate event, identify the relevant observation and timestamp, proposed and executed behavior, contribution to failure, and plausible alternative explanations. Preserve source references to the trajectory, execution feedback, and supporting frames.

Task instructions establish objectives and known constraints; observations establish information available to the model; execution records establish which actions were applied. Final outcomes may corroborate consequences, but hidden evaluation state alone cannot establish that the model ignored an observable change.

Assign each eligible episode–category judgment an evidence status:

- *Confirmed*: direct evidence satisfies the operational criterion and supports a relevant contribution to failure.
- *Probable*: affirmative evidence supports the category, but an alternative explanation remains or the response window is inferred.
- *Insufficient*: available evidence does not establish the category; this does not establish its absence.

Record out-of-scope episode–category pairs, successful episodes, and invalid runs separately. These labels characterize observable behavior, not verified internal cognitive mechanisms.

AGGREGATION AND INTERPRETATION

The main figure counts confirmed and probable judgments. Retain confirmed-only results as a sensitivity analysis. Count each episode at most once per category.

Prompt 1: Failure-mode annotation codebook (continued)

For each model, divide RT-Deliberate assignments by its valid failed episodes pooled across TL and TH. Divide RT-Perceive and RT-Act assignments by its valid failed PH episodes. Pool episode counts rather than averaging setting-level percentages.

Other denotes valid failed episodes without any selected-category assignment at the reported evidence threshold. Divide this count by the model's total valid failed episodes across TL, TH, and PH. It includes residual errors and unresolved cases; invalid actions and misunderstanding game mechanics are illustrative examples, not separately quantified subcategories.

Categories may overlap in PH, and the plotted denominators differ. Percentages therefore need not sum to 100%; Other is not computed by subtracting the three category percentages from 100%.

Assign labels according to evidence without imposing target frequencies or coverage rates. Interpret the distributions as evidence-supported attributions under the selected settings, rather than an exhaustive causal decomposition of failures.

D.3 Failure Examples

In Figure 8, with only 1 s remaining to bid, Gemini Live in TH generates 544 tokens (8.5 simulated seconds), compared with 88 tokens when 3 s remain, missing the bidding opportunity. Figure 9 shows that Gemini Live keeps moving left from 7 s until hitting the wall at 17 s, despite being one row below the apple. Figure 10 shows Gemini 3.8 Flash in PH waiting through the final 5 s of the Shared document task instead of saving the launch date.

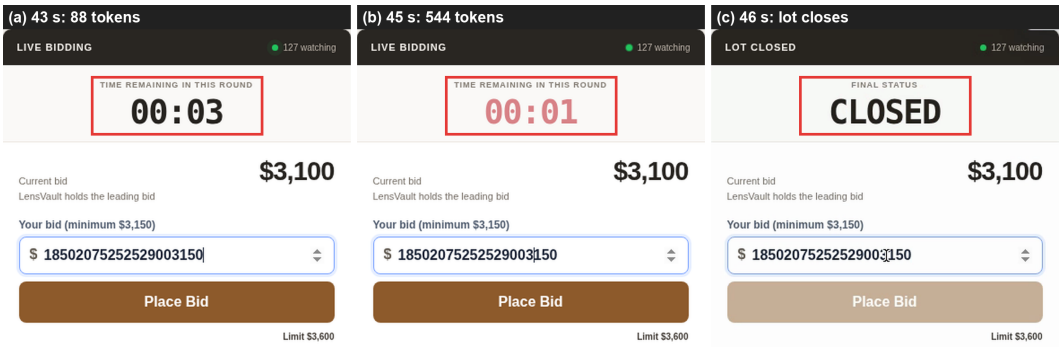


Figure 8: Reasoning longer with less time in Live auction (Gemini Live, TH). (a) With 3 s left, the model generates 88 tokens, and its click lands at 44.4 s. (b) With 1 s left, it generates 544 tokens (8.5 s) for the same click, which would land at 53.5 s. (c) The lot closes at 46 s before the click is applied. The bid field already holds amounts concatenated from earlier inputs.

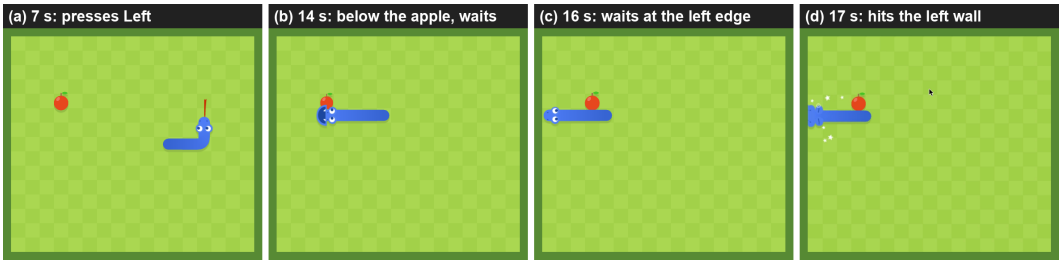


Figure 9: Inertia in Google Snake (Gemini Live, PH). (a) At 7 s the model turns left on the row below the apple. (b) It then waits at every step, even at 14 s, when the head is directly below the apple. (c) At 16 s it has passed the apple and still waits. (d) The snake hits the left wall at 17 s.

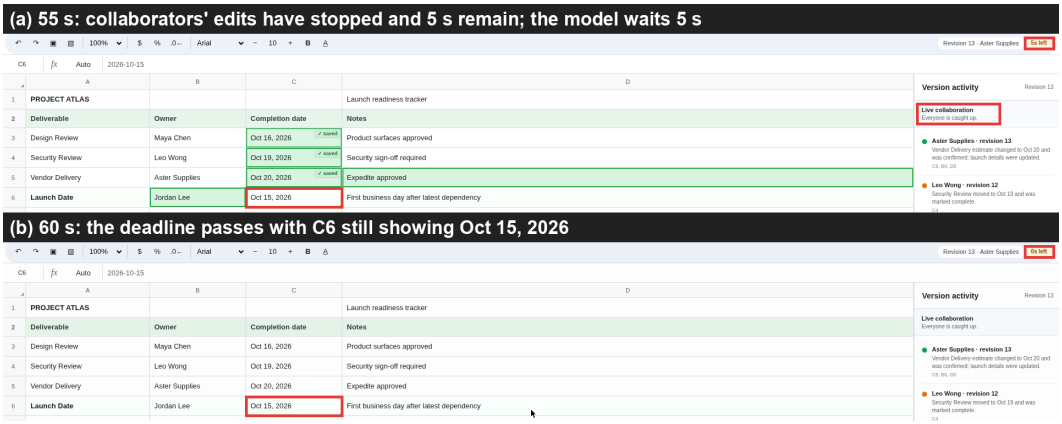


Figure 10: Waiting too long in Shared document (Gemini 3.8 Flash, PH). At the start, the model plans to "calculate the earliest legal date for C6 and verify the save within the remaining five seconds." (a) At 55 s the edits have stopped and 5 s remain, but the model waits 5 s. (b) The deadline passes with C6 unchanged; the correct date is Oct 21, 2026.

E PROMPTS

E.1 PROMPT TEMPLATES.

Prompt 2 specifies the GUI actions, how to use the SoM marks or screenshot coordinates to select an action's location, and the required output format. For each episode, we combine these shared instructions with three components. The *domain instruction* identifies the environment as a game or a web task and specifies how the agent should interact with it, defined in Appendix E.1.1. The *clock instruction* explains the clock protocols, specified in Appendix E.1.2. The *task instruction* states the specific objective and requirements for the current task, corresponding to the success criteria described in Appendix A.

E.1.1 DOMAIN INSTRUCTIONS.

The domain instruction specifies the interaction setting and behavioral constraints. Game tasks use Prompt 3, while web tasks use Prompt 4.

E.1.2 CLOCK INSTRUCTIONS.

The clock instruction specifies how observations, model outputs, and action execution relate to environment time:

- **Token clock** (Prompt 5): charged output tokens determine the simulated delay between observation and action.
- **Paused environment** (Prompt 6): output length does not advance simulated time; each action is applied at the observed tick before the environment advances one frame interval.
- **Wall clock** (Prompt 7): environment time follows physical elapsed time.

Prompt 2: Complete text-action system prompt

<DOMAIN_INSTRUCTION>
AVAILABLE GUI ACTIONS (canonical JSON):
- {"action": "click", "target": "P1"|"R1@u,v", "button": "left"|"middle"|"right"}
- {"action": "right_click"|"double_click"|"triple_click", "target": "P1"|"R1@u,v"}
- {"action": "mouse_down"|"mouse_up"|"move", "target": "P1"|"R1@u,v", "duration": <seconds>}
- {"action": "type", "text": "<string>"}

Prompt 2: Complete text-action system prompt (continued)

- {"action": "key", "key": "<key_or_combo>", "repeat": <int>} (short tap; examples: "enter", "ctrl+s")
- {"action": "hold_key", "key": "<key_or_combo>", "duration": <seconds>} (continuous control)
- {"action": "scroll", "target": "P1"|"R1", "dx": <int>, "dy": <int>} (positive x=right, positive y=up)
- {"action": "drag", "start": "P1"|"R1@u,v", "end": "P2"|"R1@u,v", "duration": <seconds>}
- {"action": "wait", "duration": <seconds>}
- Prefer canonical JSON with double quotes. The adapter can recover one unambiguous action from common wrappers or shorthand, but canonical JSON is shortest and safest.
- P/R marks belong only to the screenshot that supplies them. Pn is an exact target point. Rn@u,v addresses a continuous region using local integers from 0..1000: left/top is 0 and right/bottom is 1000. Never reuse a mark from an earlier screenshot.
- Prefer a supplied mark. If no suitable mark exists, replace target with x/y, start with x1/y1, or end with x2/y2 using the screenshot coordinate space. Reference and coordinate forms may be mixed for drag endpoints, but never specify both forms for the same point.
- Use the coordinate contract supplied with the current screenshot. The harness converts image coordinates to the executor's 1920x1080 desktop space.
- Pointer coordinates may be omitted for click variants, mouse_down/up, and scroll only to act at the current cursor position. When present, x and y must be supplied together.
- duration must be between 0 and 10. For hold_key and wait it is simulated world time under the token clock; hold_key defaults to 1 and wait without duration waits for the next observation. For move, mouse_down/up, and drag it controls executor-side pointer interpolation only; GUI execution time is excluded from simulated time.
- hold_key keeps every key in the '+'-joined chord down for its full declared duration. A later action does not cancel the hold; another hold on the same key can only extend it. Keys are released automatically at expiry and all held input is released when the episode ends. A held key auto-repeats like a physical keyboard (first repeat after 0.5 s, then every 0.05 s), so a game that moves one step per key press keeps stepping for as long as the key is held.
- key.repeat is optional, defaults to 1, and must be an integer from 1 to 100.
- Click variants, scroll, and drag may include "modifier": a '+'-joined combination of "shift", "ctrl", "alt", and "super". Modifiers apply only for that action.
- mouse_down remains active until the matching mouse_up; use drag for an ordinary complete drag. For a gesture that must react to changing frames, use mouse_down, then move actions on later observations, then mouse_up; each step may use that observation's current marks.

ACTION OUTPUT

Use your configured or native reasoning internally. Your entire visible response must be exactly one valid action from the active action space. Output the action and its arguments only: no reasoning, commentary, labels, wrappers, tags, Markdown, ellipses, or placeholders. The action is final and immediately executable. All generated output tokens count. The adapter accepts common formatting wrappers when it can recover exactly one unambiguous valid action. If it finds no valid action or multiple different actions, the harness executes WAIT.

<CLOCK_INSTRUCTION>

TASK: <TASK_INSTRUCTION>

Prompt 3: Game domain instruction

You are playing a live browser game. Use only the visible game controls and keep acting toward the TASK goal. Do not open a terminal, editor, source code, developer tools, or test harness, do not reload the page, and do not modify the game. Prefer hold_key with a short duration for continuous movement instead of many discrete taps. Keep playing until the environment ends the episode.

Prompt 4: Web domain instruction

You are operating a live web task. The page and server state may change while you reason, so use the latest visible state and complete the TASK through the visible UI. Do not inspect or edit source code, call hidden APIs, or use developer tools or terminal shortcuts. The environment determines when the episode ends; use wait when no further UI interaction is needed.

Prompt 5: Token-clock instruction

TOKEN CLOCK: observations are scheduled at f=1 frames per simulated second. Physical inference and GUI execution latency do not advance simulated time. Produce at most one action per observation. B=64 charged tokens/frame; kappa=B*f=64 charged tokens per simulated second. After n provider/native output tokens, an action lands n/kappa simulated seconds after the observation. B is a conversion rate, not a maximum response length. A decision may use more than B tokens, cross scheduled frame boundaries, and miss the corresponding observations. Finishing early does not provide an observation before the next scheduled frame boundary. The task ends at its fixed simulated-time limit T, with at most kappa*T charged tokens over the episode. Executed actions and their outcomes are retained in your context.

Prompt 6: Paused-environment instruction

TOKEN CLOCK: observations are scheduled at f=1 frames per simulated second. Physical inference and GUI execution latency do not advance simulated time. Produce at most one action per observation. B is unlimited: output length does not affect simulated time, so each action lands at the tick it observed. The environment then advances exactly one frame interval (1/f seconds), and the next observation includes the action's effect.

Prompt 7: Wall-clock instruction

WALL CLOCK: observations are scheduled at 1 fps. Environment time follows physical elapsed time. Produce at most one action per observation and use execution feedback to track which actions were applied.