
Evaluating Spatiotemporal Reasoning of Vision-Language Models in Atari Gameplay

Mingjia Huo^{1,*} Yao Fu^{2,†} Bo Chang² Yaqing Wang²
Dawei Zhu^{3,*} Dylan Zhang^{4,*} Xinyang Yi² Lichan Hong²
Ed H. Chi² Hao Zhang¹ Jiayi Tang²
¹University of California San Diego ²DeepMind
³Peking University ⁴University of Illinois Urbana-Champaign
mhuo@ucsd.edu jiayit@google.com

Abstract

Vision-Language Models (VLMs) have achieved strong performance on static visual understanding benchmarks, yet their ability to act in dynamic environments remains poorly understood. We introduce AtariBench, a controlled benchmark that evaluates VLMs across 20 Atari 2600 games, requiring models to make sequential decisions from high-temporal-resolution gameplay streams. By decoupling fast in-game dynamics from wall-clock inference speed, AtariBench focuses on whether models can perceive rapid visual state changes, reason over spatiotemporal dynamics, and ground actions in the current observation. Our experiments reveal substantial limitations in state-of-the-art VLMs: models often fail to infer object motion and direction, exhibit brittle sequential decision-making, and over-rely on textual interaction history despite newly appended visual feedback. We further introduce a separability analysis to identify which in-game scores provide reliable signals of model competence, and use the benchmark to study how memory design and prompting strategies affect performance. Overall, AtariBench exposes a persistent gap between passive visual perception and active decision-making, providing a lightweight and rigorous testbed for developing VLMs that can reason and act in dynamic environments.

1 Introduction

Despite the remarkable proficiency of Vision-Language Models (VLMs) in static scene understanding [1], the prevalent evaluation paradigm heavily over-indexes on static or turn-based benchmarks. This static-centric focus obscures a critical capability gap: when scaling to dynamic environments, current models struggle to reliably process continuous visual dynamics into actionable decisions [2–5]. To bridge this gap, VLMs must evolve along three core dimensions: (1) *Spatiotemporal Physics*, the intuitive grasp of physical dynamics such as object trajectory, distance, and velocity; (2) *Sequential Decision-Making*, the capability to synthesize continuous visual streams into coherent, forward-looking strategies; and (3) *Visual Grounding*, the ability to anchor decisions in the present, continuously updating visual state rather than relying on historical text or static priors.

To systematically evaluate these interactive capabilities, we propose AtariBench, which evaluates VLMs across 20 diverse Atari 2600 games.¹ The Atari environment [6] has long served as a canonical testbed for agent-based AI [7–9]; in this work, we repurpose this classic platform as a diagnostic sandbox for modern VLMs. In our setting, models are tasked with making sequential decisions

^{*}Work done as Student Researcher at Google DeepMind.

[†]Work done while at Google DeepMind.

¹Code repository: <https://github.com/mignonjia/AtariBench>.

directly from high-temporal-resolution gameplay streams. By decoupling the fast-paced in-game dynamics from wall-clock model inference speed, we force models to process rapid state transitions and react sequentially, directly challenging their spatiotemporal reasoning. Our approach marks a deliberate departure from static benchmarks [10, 11]. Furthermore, while other vision-to-action frameworks exist, their open-ended strategic and visual complexities can obscure the foundational reasoning failures we aim to isolate [12, 13], or they focus primarily on image-based planning rather than continuous video dynamics [14]. AtariBench, in contrast, leverages constrained 2D physics as controlled environments to specifically measure these foundational abilities.

Our experiments with AtariBench reveal a cascade of critical failures in state-of-the-art models, highlighting the capability gaps identified above. First, we observe a pervasive weakness in motion comprehension: models consistently struggle to infer object movement and direction from sequential visual inputs, leading directly to poor in-game choices. Second, models suffer from long-context failures. Our qualitative analysis reveals that as the interaction history grows, models frequently collapse into pathological action loops, endlessly repeating past sequences regardless of newly appended visual feedback, though this effect is less pronounced in the most recent models. Third, text has mixed effects: textual outcome feedback can help models recover from missed changes in game state, while reliance on stale textual history can also lead to failures. Despite image tokens comprising the vast majority of the prompt, some decisions follow stale textual cues that conflict with the current frames, especially among weaker models.

The AtariBench framework offers several practical advantages for rigorous model evaluation. It provides a lightweight yet robust sandbox for rapid algorithmic iteration. While its simple but diverse game rules enable evaluation across a wide spectrum of cognitive tasks [15, 16], we go beyond raw scores by introducing a rigorous separability analysis. This ensures that the native in-game metrics reliably distinguish the evaluated models rather than reflecting random noise. Crucially, the framework also facilitates nuanced investigations into inference-time behaviors, allowing us to systematically evaluate how context history length and different prompted reasoning strategies impact empirical performance. Ultimately, by systematically exposing the gap between static perception and dynamic execution, AtariBench establishes a rigorous baseline for evolving VLMs from passive observers into active agents.

2 Related Work

Game-Based Benchmarks for Language Models. Recent years have seen a surge of work on game-based benchmarks from diverse genres, including grid-based puzzles [17], simulation and resource-management strategy such as Factorio [18], turn-based role-playing with strategic combat such as Pokémon [19], and narrative reasoning tasks such as murder-mystery games [20]. Others evaluate natural language reasoning through text-based or conversational games [21–24], lacking visual understanding. Multimodal gaming benchmarks such as BALROG [25], LMAct [26], and LMGame-Bench [27] assess planning using image inputs, but their tasks are dominated by grid-based games that do not directly probe fine-grained spatial-temporal reasoning (e.g., movement trajectories, speed, or continuous dynamics) as found in Atari. VideoGameBench [13] evaluates VLMs on AAA games, but most models make little progress due to the extreme complexity of these games, limiting its ability to separate models. Cradle [28] builds computer-use agents for such games but relies on modules like object detectors, making it hard to isolate VLM perception. In contrast, Atari games provide a lightweight testbed with simple rules and visuals, sufficient to probe spatial-temporal reasoning. Atari-GPT [29] explores the use of VLMs on Atari environments, demonstrating initial feasibility. Our work, however, significantly extends this by constructing a systematic and reproducible benchmark designed for rigorous evaluation of VLM spatial-temporal reasoning. This involves a diverse set of games targeting motion perception, a standardized agentic harness, and repeated trials to ensure statistical reliability, thereby moving beyond a proof-of-concept to a robust framework.

Video Understanding, Action Recognition and Decision Making. General video benchmarks [3, 2, 11] and fine-grained action suites [10, 5] evaluate temporal dependencies and human motion comprehension primarily through video-based QA. VideoReasonBench [30] challenges models with vision-centric tracking of altered states over time, while RIVER [31] evaluates timely responses to streaming videos. These works test video understanding without requiring control of the depicted

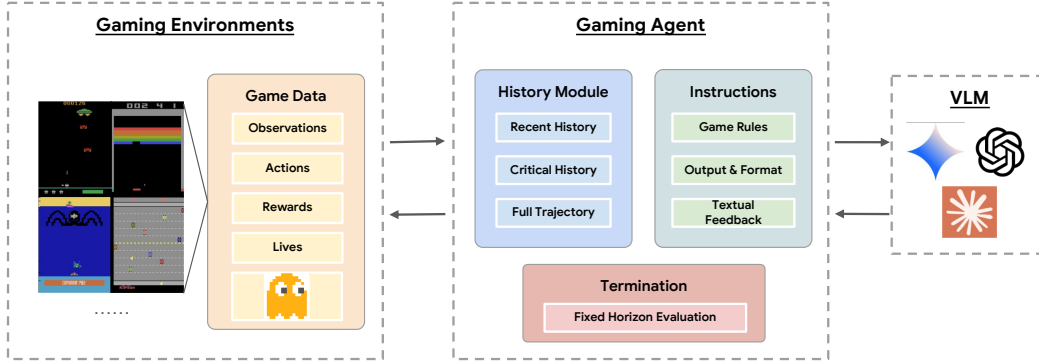


Figure 1: Workflow of AtariBench. At each turn, the harness assembles game instructions, interaction history, and the current frame into a prompt; the VLM returns a thought and an action sequence, which is executed in ALE, and the resulting frames and feedback are added to the history.

environment. Existing action-oriented VLM benchmarks, such as EmbodiedBench [14] and PCA-Bench [4], also focus heavily on static, image-based decision making. AtariBench bridges this gap by requiring active, closed-loop control in dynamic, fast-paced interactive environments.

3 Method

AtariBench is a benchmark for evaluating VLMs as game-playing agents in visually dynamic Atari game environments. We choose 20 fast-paced Atari games that cover diverse genres and goals, thereby probing a broad range of model capabilities. A standardized gaming-agent harness manages instructions, memory, action execution, and termination, enabling closed-loop interaction between VLM decisions and environment feedback.

3.1 Arcade Learning Environment

We implement AtariBench on top of the Arcade Learning Environment (ALE) [6] via Gymnasium [32, 33]. ALE provides a common control interface across Atari 2600 games, defining the action space, reward signal, observations, and termination conditions for each game. Each gaming environment exposes a standardized step API, and a low-level transition is formalized as:

```
obs, reward, terminated, truncated, info = env.step(action_id)
```

Concretely, each interaction step takes an integer `action_id` and returns an RGB observation frame (`obs`), a scalar reward (`reward`), termination flags (`terminated`, `truncated`), and environment metadata (`info`). This unified interface allows AtariBench to evaluate games under a single agent protocol. Appendix F discusses licensing, game assets, and our release policy.

Actions and Scores. ALE defines 18 discrete actions, formed by the Cartesian product of horizontal movement (noop, left, right), vertical movement (noop, up, down), and a firing command (noop, fire). Some games support only a subset of these actions. For example, in *Breakout*, the paddle moves only horizontally, and vertical actions are disabled. Meanwhile, the semantics of fire are game-dependent, such as starting a turn, shooting a fireball, or speeding up the movement. We provide the full action set and game-specific function of each action in the prompt when querying VLMs.

Control and Frame FPS. We render frames at ALE’s native resolution (210×160). While ALE runs at 30 FPS, we execute actions at 10 Hz by repeating each chosen action for three frames. A model query can select several actions, and the environment pauses during in-

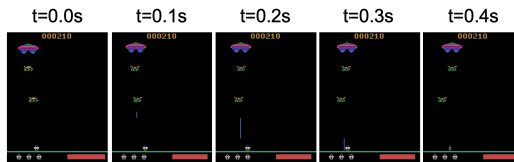


Figure 2: Fast temporal dynamics in *Assault*. The enemy projectile appears at $t = 0.1s$ and already hits the player at $t = 0.3s$, motivating high temporal resolution and 10 Hz control.

ference. This retains fast object motion while keeping the token budget manageable for current VLMs. As illustrated in Fig. 2, Atari dynamics can unfold within only a few tenths of a second: in *Assault*, an enemy projectile appears at $t = 0.1$ s and reaches the player by roughly $t = 0.3$ s. Such examples motivate a sufficiently high observation and control frequency. We disable ALE sticky actions to remove stochasticity.

Fixed-Horizon Evaluation. Each run is evaluated over a fixed 30-second window, with immediate reset after early game termination. Under the original life-based termination, a small survival instruction can change both episode length and reward in games where survival creates more scoring opportunities. The fixed horizon standardizes the interaction budget, reduces variance, and mitigates prompt-sensitive performance variation that makes it more difficult to accurately measure spatial-reasoning capabilities (Appendix A.1).

3.2 Game Choice and Taxonomy

ALE contains 100+ Atari 2600 games, from which we select 20 according to three principles. First, each game should require **motion perception** rather than static scene understanding alone, which rules out math, board, and card games. Second, each game should be **human-interpretable** under the fixed evaluation horizon. After being given the game rules in advance, a human player should be able to infer the controls and scoring from a 30-second gameplay clip. This avoids games with ambiguous controls, sparse feedback, or opaque scoring rules that are difficult to learn within a short horizon. We manually played each selected game to ensure this. Third, we prioritize coverage of **diverse** games, avoiding multiple games that are closely related variants of the same core mechanics and visuals; for example, we include *PacMan* then not *MsPacMan*.

Our selected games span shooters, sports, action games, and maze/navigation games. To make the benchmark analytically useful, we manually classify games along different aspects (Table 1), including the primary skill (e.g., aiming, route planning, interception, scrolling navigation), number of different objectives (e.g., objectives that yield positive or negative points), action topology (horizontal, vertical, both axes, with optional fire/attack actions), reward density, visual load and complexity, and tempo sensitivity. We include 11 games in the *Shooter* category, defined as games in which the player must aim and shoot; despite this shared requirement of shooting, their mechanics and visuals actually vary substantially. Moreover, most games require a high degree of temporal precision to achieve high scores. This makes AtariBench a broad and challenging benchmark for spatiotemporal reasoning.

3.3 Harness Design

The harness defines the benchmark-side protocol through which VLMs interact with Atari environments. We intentionally use a simple yet necessary harness with only the components needed for closed-loop gameplay: task instructions, trajectory memory, environment feedback, output parsing, and termination handling. We do not optimize complex agent designs such as external planners or tool-use modules. While stronger harnesses may improve absolute scores, our goal is to compare VLMs under the same controlled interaction interface.

At each decision point, the harness constructs a multimodal prompt with the following components: (i) a game-specific instruction block describing the objective and action semantics, (ii) historical context and the current frame derived from the current run, and (iii) the VLM output schema. The prompt template is shared across games, while game-specific instructions provide per-game guidelines (see Fig. 6 for an example). We next describe the harness components in detail, including memory, environment feedback, and the VLM output schema.

3.3.1 Memory

To support sequential decision making under partial observability, we augment the current observation with lightweight memory derived from the agent’s own trajectory. In the default `structured_history` mode, each prompt contains the current frame together with two curated

Table 1: AtariBench game taxonomy. #Goals: number of distinct in-game goals. Action: A=fire/attack/throw, H=horizontal, V=vertical, B=both horizontal and vertical. Reward density and visual load are qualitative summaries. Tempo reflects the degree of timing precision required for competent play.

Game	Primary Skill	#Goals	Action	Reward Density	Visual Load	Tempo
<i>Shooter</i>						
AirRaid	Aiming & Avoidance	2	H+A	Dense	Low	High
Assault	Aiming & Avoidance	3	H+A	Dense	Low	High
BeamRider	Aiming & Avoidance	2	H+A	Dense	High	High
DemonAttack	Aiming & Avoidance	2	H+A	Dense	Low	High
LaserGates	Multi-task Navigation	2	B+A	Dense	High	High
NameThisGame	Precision Interception	4	H+A	Dense	Low	High
Phoenix	Aiming & Avoidance	2	H+A	Dense	Low	High
RiverRaid	Scrolling Navigation	3	H+A	Dense	High	High
Seaquest	Multi-task Navigation	3	B+A	Dense	High	High
TimePilot	360-degree Navigation	2	B+A	Sparse	High	High
Robotank	First-person Navigation	2	B+A	Dense	High	High
<i>Sports</i>						
Boxing	Rhythmic Combat	2	B+A	Dense	Low	High
FishingDerby	Depth Targeting	1	B+A	Dense	High	High
Tennis	Predictive Timing	1	B	Sparse	Low	Med/High
<i>Action</i>						
Breakout	Rebound Control	1	H	Dense	Low	Med/High
Freeway	Gap Identification	2	V	Sparse	High	High
Gopher	Reactive Pathing	2	H+A	Dense	High	High
JourneyEscape	Scrolling Navigation	1	B	Dense	High	High
<i>Maze</i>						
PacMan	Route Planning	2	B	Dense	High	High
Qbert	Route Planning	1	B	Dense	High	Low

forms of episodic context: a short recent-history buffer and a sparse archive of reward-relevant events. The benchmark also supports an `append_only` transcript mode, but our main harness uses the structured variant because it keeps context compact while preserving the most decision-relevant events.

Structured-History Mode. Our memory harness contains two components. First, we include a *recent-history buffer* consisting of the most recent K turn-level clips, with $K = 3$ by default. Second, we include a *reward-history buffer* consisting of the most recent R clips that produced non-zero reward, with $R = 3$ by default. This yields around $5k$ input tokens per query. Life-loss events are also inserted into this reward-relevant memory even when the scalar reward is zero, because losing a life is an important gameplay failure that changes gameplay status and models’ decisions.

Each clip corresponds to one completed model turn and stores: (i) the starting frame, (ii) the list of actions proposed by the model, and (iii) the resulting frames in action order, with one frame paired with each executed action. This representation provides the model with compact temporal evidence about what it recently did, what happened next, and which earlier situations were associated with success or failure. The recent-history buffer emphasizes short-horizon continuity, while the reward-history buffer highlights sparse but informative events that help the model infer game mechanics and effective strategies.

Append-Only Mode. AtariBench also supports an alternative `append_only` prompt mode. Rather than organizing past interaction into separate `recent-history` and `reward-history` blocks, this mode keeps the full trajectory just like a typical human-chatbot interaction. The initial user turn contains the game instructions, the time-budget rule, and the first frame. Each later assistant turn is the model’s response, and each following user turn contains the resulting game frames with reward or life-loss feedback. This can lead to more than $100k$ input tokens per query.

We use this `append_only` mode as an optional choice, while `structured_history` remains our default. This is because current models can suffer from long-context issues, and the `append_only` mode can cause meaningless repetition for most models (see Sec. 4.3).

3.3.2 Textual Reward Feedback

In addition to image frames, AtariBench exposes environment rewards through simple textual feedback. After each executed action, the pipeline emits short deterministic annotations whenever a salient event occurs. *Positive or negative rewards* are verbalized as “Feedback: You received a positive/negative score!”, and *life-loss events* are described explicitly as “Feedback: A life was lost and a new life has begun.” If no salient event occurs, no extra feedback text is added.

Textual feedback makes rewards and failure events explicit since it provides clear signals about the consequences of the model’s actions. For instance, in Breakout, after the model loses the ball, the screen can enter a reset state with no visible ball. If the model fails to infer from the frame that it must press `start` to serve the next ball, the game can stall after the initial loss; textual feedback makes this reset explicit, and we discuss its effect in detail in Sec. 4.3.

3.3.3 Model Output Schema

Each game exposes a normalized textual action vocabulary derived from the ALE action set used by that game. The model is required to produce a short free-form *thought* followed by 1-10 *actions*. The harness parses the output, normalizes action names, validates them against the game-specific action map, and executes the resulting sequence in order. Invalid outputs are handled conservatively by falling back to a `no-op` action, which keeps evaluation robust while preserving the raw response for later analysis. Variable-length sequences permit short action sequences at critical frames while balancing evaluation cost (Appendix C.2).

4 Experiments

This section presents the empirical evaluation of AtariBench. We first compare state-of-the-art VLMs and a random-action baseline across AtariBench games. We then analyze game separability to ensure that the retained games can distinguish models. We further present qualitative case studies of recurring failures. Finally, we diagnose perception-to-action skills with controlled QA questions; supplementary gameplay-level failure attribution using VLM-as-a-judge is provided in Appendix D.2.

4.1 Results

Fig. 3 reports the main evaluation across 20 games, comparing Gemini 3.1 Pro, Gemini 3 Flash, Gemini 2.5 Flash, GPT-5.4, GPT-5.4 Mini, Claude Opus 4.6, Claude Sonnet 4.6, DeepSeek V3.1, GLM 5.1, and a random-action baseline. We also vary the thinking effort levels for eligible models. The figure also includes Gemini 3.1 Pro `append-only` variants, marked as “AO”. For each game, we evaluate each model setting over ten 30-second closed-loop runs using ALE seeds 0–9 under the same execution harness, with sticky actions disabled. Overall, Gemini 3.1 Pro and GPT-5.4 occupy the strongest average-rank positions, while the random baseline remains consistently near the bottom, indicating that the benchmark rewards learned visual control rather than arbitrary action sampling. We evaluate PPO-based game-specific RL agents from Sample Factory [34] on the 16 games with matching public checkpoints. They rank first on 14 of these games, highlighting a substantial performance gap between state-of-the-art VLMs and specialized game policies; Appendix B.2 gives the evaluation details.

4.2 Game Separability Analysis

Here we assess how well our selected Atari games distinguish among models. Even with an interpretable objective and a valid reward function, a game can be a weak benchmark if no model beats random play or if scores cluster into the same narrow range. To ensure each game yields informative signal, we apply three complementary criteria that test model separability.

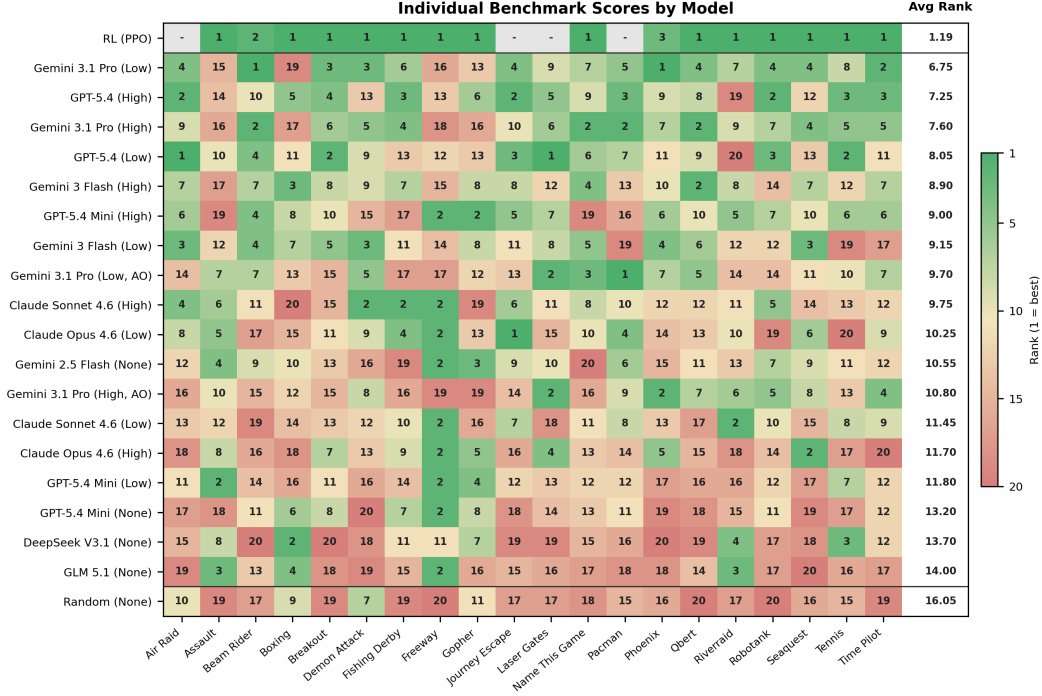


Figure 3: Model performance ranks across the 20 AtariBench games used in the main analysis, including the standard structured-history settings and the Gemini 3.1 Pro append-only (AO) settings. Game-specific RL agents and a random-action baseline are also included. Rows are ordered by average per-game rank; raw reward values are reported in Appendix B.1.

First, the best VLM setting must outperform random play, with a positive mean-reward gap, pooled-SD effect size $d \geq 0.8$, and $q < 0.05$ from a two-group ANOVA [35]. This criterion directly asks whether any evaluated model can exploit the game structure better than random play. Second, the top and bottom quarters of non-random settings (each $\lceil N/4 \rceil$ settings) must have a positive mean gap and $d \geq 0.8$. This top-bottom contrast is useful when only a subset of models discovers the right behavior while many others remain similarly weak. Third, a one-way ANOVA [36] across all settings, including random play, must give $q < 0.05$ and $\eta^2 \geq 0.14$. The ANOVA tests whether mean rewards differ across settings, while η^2 measures the fraction of observed score variance attributable to between-setting differences. For the first and third criteria, we apply Benjamini-Hochberg correction [37] separately across the 20 games.

Figure 4 summarizes the resulting game-level labels using a simple pass-count rule. A game is *strongly separable* when at least two of the three criteria pass, *weakly separable* when exactly one criterion passes, and not separable when none pass. Under this criterion, all 20 games retained in AtariBench are separable: 18 are strongly separable, and the remaining 2 are weakly separable. This result separates benchmark validity from broad model disagreement while keeping the decision rule transparent.

4.3 Qualitative Analysis

Visual Perception. AtariBench exposes failures to detect visible objects. In the controlled diagnostics, perception accuracy ranges from 61.0% to 100.0% across model settings (Table 2). In Name This Game (Fig. 9), GPT-5.4 (high) states that no shark is visible, although a white shark appears in the current frame, and its plan anticipates a shark appearing rather than acknowledging the one already present (Appendix D.1). Models can also misidentify objects, such as mistaking the controlled aircraft for an enemy in Time Pilot (Appendix D.2).

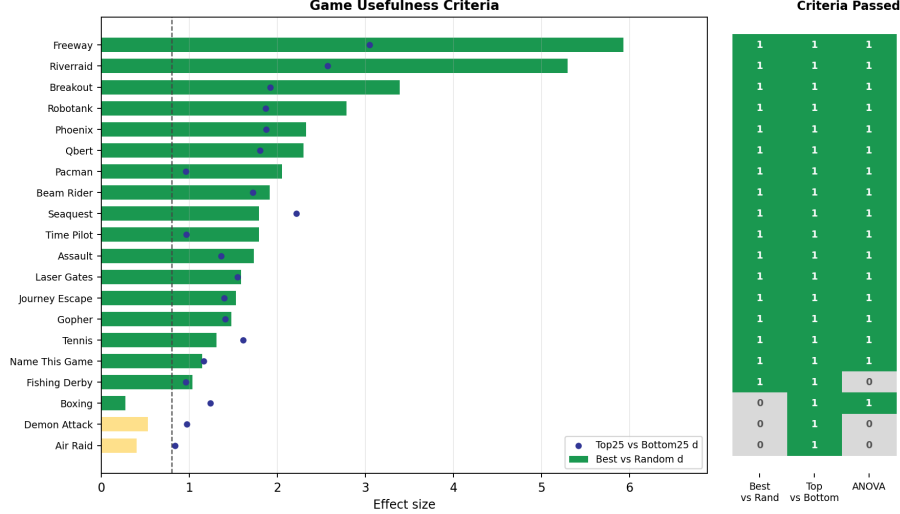


Figure 4: Game separability analysis across the 20 AtariBench games. Bars show the best-model-vs-random effect size, points show the top-quartile-vs-bottom-quartile effect size, and the right-hand matrix marks whether each of the three criteria passes. Games are strongly separable when at least two criteria pass and weakly separable when exactly one passes.

Temporal Misgrounding. Models can describe a historical spatial relation as if it were current. In Freeway, GPT-5.4 (high) describes a blue car as still to the chicken’s right, although it has already crossed to the left (Fig. 10). The claim matches earlier frames provided in the same prompt, consistent with a failure to update the spatial relation at the current time. Appendix D.2 further isolates this failure by comparing the model’s spatial understanding when given the complete history versus only the current frame: on spatial-relation failure cases, spatial errors persist in 39.3% of complete-history re-queries but in only 10.7% of current-frame-only re-queries (exact McNemar $p = 0.0386$). This suggests that interaction history can interfere with grounding the current state, so additional visual context is not necessarily beneficial.

Spatiotemporal Understanding. Motion comprehension is a major challenge, as indicated by the diagnostic questions in Sec. 4.4 and the VLM-judge analysis in Appendix D.2. In a Breakout example (Fig. 11), the inferred movement direction contradicts the ground-truth trajectory. This error can propagate into planning without being repaired by additional reasoning (Appendix D.4), compromising object tracking and action selection in dynamic scenes. This is consistent with the benchmark results: higher reasoning effort does not consistently improve scores and can sometimes reduce performance.

Text-History Overreliance. Textual outcome feedback can both help and hurt model performance. It can help weaker models identify critical gameplay events, but it can also hurt performance when models rely on outdated feedback instead of current visual observations. In Breakout, for example, a life-loss notice helps Gemini 2.5 Flash serve the next ball with `start`, and our ablation analysis finds a significant score benefit for this model on Breakout (Appendix C.1). However, earlier feedback can become stale as the game evolves. In the Assault example in Fig. 12, GPT-5.4 Mini interprets an earlier positive-score notice as evidence that recent upward shots have been hitting, so it continues firing upward without accounting for the current game state.

To measure the prevalence of these failures, we use a VLM judge and find that text-history overreliance accounts for 22.4% of observable failures in the two weaker model settings and 4.9% in the two stronger settings (Appendix D.5). This suggests that the stronger models make more effective use of current visual evidence.

Long-Context Problem. We also evaluate the `append_only` prompting format, which more closely follows the long-context chat interface used by modern LLMs. This makes the input as long as 100k tokens, because the entire 30-second history is included in one call. For non-Gemini-

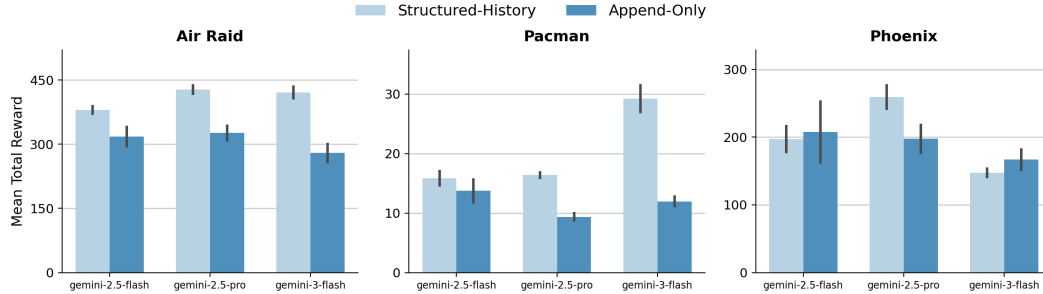


Figure 5: Append-only versus structured-history performance on three representative games for Gemini 2.5 Flash, Gemini 2.5 Pro, and Gemini 3 Flash. Bars show mean total reward, and error bars show standard error over runs. All models use default thinking.

Table 2: Model accuracy on 60 manually curated diagnostic questions, with five queries per question.

Model (reasoning)	Perception	Motion	Decision	Overall
Gemini 2.5 Flash (off)	88.0%	51.0%	48.0%	62.3%
Gemini 2.5 Flash (on)	85.0%	47.0%	65.0%	65.7%
Gemini 2.5 Pro (default)	93.0%	54.0%	79.0%	75.3%
Gemini 3 Flash (low)	98.0%	64.0%	88.0%	83.3%
Gemini 3 Flash (high)	99.0%	63.0%	82.0%	81.3%
Gemini 3.1 Pro (low)	100.0%	69.0%	82.0%	83.7%
Gemini 3.1 Pro (high)	100.0%	60.0%	80.0%	80.0%
GPT-5.4 (low)	91.0%	74.0%	94.0%	86.3%
GPT-5.4 (high)	91.0%	73.0%	83.0%	82.3%
Claude Opus 4.6 (low)	61.0%	65.0%	65.0%	63.7%
Claude Opus 4.6 (high)	65.0%	72.0%	67.0%	68.0%

3.1-Pro models, the comparison on Air Raid, Pacman, and Phoenix shows that appending the full interaction history usually does not improve over the structured-history baseline (Fig. 5). These results suggest that append-only context is not automatically beneficial for weaker or earlier Gemini models, and that the gains from long history appear model-dependent.

One failure mode is action stagnation late in the episode. In the latter half of each run, append-only traces issue noop-only decisions far more often than the matched structured-history setting (48.8% vs. 0.7% of turns). Fig. 13 shows a concrete Breakout trace where the model repeatedly emits only noop actions despite receiving new frames and feedback. By contrast, Gemini 3.1 Pro does not show the same collapse: noop-only decisions account for 25.3% and 23.7% of later-half turns in its high and low settings, respectively. This suggests newer models have significantly better capabilities under long contexts.

4.4 Failure Modes Breakdown

To diagnose why strong VLMs still struggle in closed-loop Atari gameplay, we construct a small, manually labeled diagnostic set. We select ten games and curate 20 perception questions, 20 motion questions, and 10 decision questions: two perception, two motion, and one decision question per game. Each decision question is presented with a short and a long prompt, yielding 60 diagnostic question variants in total.

Each sample consists of a short sequence of frames selected from an actual benchmark trajectory, together with a small set of closed-choice questions. For every question, each model is queried five times and accuracy is computed over the repeated trials. This design isolates specific perception-to-action skills while keeping the answer space easy to parse automatically. Appendix D.3 provides the diagnostic prompts and calibration details.

The diagnostic questions are organized into four fine-grained capability groups:

1. **Perception: detection.** These questions test whether the model can identify salient objects in a single frame, such as counting fish or detecting the object relevant to the next action.
2. **Perception: spatial reasoning.** These questions test whether the model can compare positions of visible objects in a single frame, such as whether the player’s controllable object is currently to the left or right of a target.
3. **Motion comprehension.** These questions use a timestamped frame sequence and ask for object or controller motion, such as whether a target is moving left or right, whether the player’s line is moving up-left, down-left, or still, or which tile changes color after a hop.
4. **Decision making.** These questions ask for the next action implied by the state. We evaluate both a short decision prompt and a long decision prompt. The long version includes the game’s action-planning guidelines before asking the same closed-choice decision question.

Table 2 reports accuracy over five queries per question. Across all VLMs, perception accuracy ranges from 61.0% to 100.0%, motion accuracy from 47.0% to 74.0%, and decision accuracy from 48.0% to 94.0%. Errors in these closed-choice tasks expose weaknesses even when the action space and intended goal are specified. Most VLMs score higher on perception than on motion, while higher reasoning effort does not consistently improve overall accuracy.

VLM-judge. Attributing a failure in a full gameplay turn is more complex than evaluating an isolated diagnostic question: perception, temporal grounding, action selection, and subsequent game dynamics can all contribute to the outcome. For example, an apparent relative-position error may reflect either a spatial error or a temporal misgrounding that relies on a historical frame rather than the current observation. We therefore focus on controlled QA in the main analysis and provide the gameplay-level VLM-judge analysis in Appendix D.2 for reference. Among the judged failures, motion and trajectory-prediction failures are the most frequent, followed by decision and control, spatial relations, and visual perception, while exploration and rule-understanding failures are rare. A follow-up judge analysis further suggests that some spatial-relation errors stem from temporal misgrounding (Appendix D.2). This provides a larger-scale, gameplay-level diagnosis compared with the above QA set.

5 Conclusion

We introduced AtariBench, a spatiotemporal gameplay benchmark built on the Arcade Learning Environment to evaluate the physical reasoning and closed-loop decision-making capabilities of modern VLMs. Our experiments show that even state-of-the-art models struggle to convert fast visual dynamics into reliable actions, often failing to infer object motion, update decisions from current frames, or use long interaction histories effectively. Diagnostic analysis identifies spatiotemporal reasoning as the most frequent failure category in the audited sample, alongside perception and control errors. Overall, AtariBench reveals a persistent gap between passive visual understanding and active control, highlighting the need for future VLMs to be trained and evaluated in dynamic interactive environments.

Acknowledgments and Disclosure of Funding

We thank Google DeepMind for supporting this work. The authors have no additional competing interests to declare beyond the affiliations and employment relationships noted on the first page.

References

- [1] Yue, X., Ni, Y., Zhang, K., Zheng, T., Liu, R., Zhang, G., Stevens, S., Jiang, D., Ren, W., Sun, Y., et al.: Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. (2024) 9556–9567
- [2] Ning, M., Zhu, B., Xie, Y., Lin, B., Cui, J., Yuan, L., Chen, D., Yuan, L.: Video-bench: A comprehensive benchmark and toolkit for evaluating video-based large language models. arXiv preprint arXiv:2311.16103 (2023)

- [3] Li, K., Wang, Y., He, Y., Li, Y., Wang, Y., Liu, Y., Wang, Z., Xu, J., Chen, G., Luo, P., et al.: Mvbench: A comprehensive multi-modal video understanding benchmark. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. (2024) 22195–22206
- [4] Chen, L., Zhang, Y., Ren, S., Zhao, H., Cai, Z., Wang, Y., Wang, P., Meng, X., Liu, T., Chang, B.: Pca-bench: Evaluating multimodal large language models in perception-cognition-action chain. In Findings of the Association for Computational Linguistics ACL 2024. (2024) 1086–1104
- [5] Hong, W., Cheng, Y., Yang, Z., Wang, W., Wang, L., Gu, X., Huang, S., Dong, Y., Tang, J.: Motionbench: Benchmarking and improving fine-grained video motion understanding for vision language models. In Proceedings of the Computer Vision and Pattern Recognition Conference. (2025) 8450–8460
- [6] Bellemare, M.G., Naddaf, Y., Veness, J., Bowling, M.: The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research* **47** (jun 2013) 253–279
- [7] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al.: Human-level control through deep reinforcement learning. *nature* **518**(7540) (2015) 529–533
- [8] Badia, A.P., Piot, B., Kapturowski, S., Sprechmann, P., Vitvitskyi, A., Guo, Z.D., Blundell, C.: Agent57: Outperforming the atari human benchmark. In International conference on machine learning, PMLR (2020) 507–517
- [9] Ecoffet, A., Huizinga, J., Lehman, J., Stanley, K.O., Clune, J.: First return, then explore. *Nature* **590**(7847) (2021) 580–586
- [10] Salehi, M.R., Park, J.S., Kusupati, A., Krishna, R., Choi, Y., Hajishirzi, H., Farhadi, A.: Action-atlas: A videoqa benchmark for domain-specialized action recognition. *Advances in Neural Information Processing Systems* **37** (2024) 137372–137402
- [11] Patraucean, V., Smaira, L., Gupta, A., Recasens, A., Markeeva, L., Banarse, D., Koppula, S., Malinowski, M., Yang, Y., Doersch, C., et al.: Perception test: A diagnostic benchmark for multimodal video models. *Advances in Neural Information Processing Systems* **36** (2023) 42748–42761
- [12] Fan, L., Wang, G., Jiang, Y., Mandlekar, A., Yang, Y., Zhu, H., Tang, A., Huang, D.A., Zhu, Y., Anandkumar, A.: Minedojo: Building open-ended embodied agents with internet-scale knowledge. *Advances in Neural Information Processing Systems* **35** (2022) 18343–18362
- [13] Zhang, A.L., Griffiths, T.L., Narasimhan, K.R., Press, O.: Videogamebench: Can vision-language models complete popular video games? *arXiv preprint arXiv:2505.18134* (2025)
- [14] Yang, R., Chen, H., Zhang, J., Zhao, M., Qian, C., Wang, K., Wang, Q., Koripella, T.V., Movahedi, M., Li, M., et al.: Embodiedbench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents. In Forty-second International Conference on Machine Learning. (2025)
- [15] Machado, M.C., Bellemare, M.G., Talvitie, E., Veness, J., Hausknecht, M., Bowling, M.: Re-visiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research* **61** (2018) 523–562
- [16] Farebrother, J., Castro, P.S.: Cale: Continuous arcade learning environment. *Advances in Neural Information Processing Systems* **37** (2024) 134927–134946
- [17] Nasir, M.U., James, S., Togelius, J.: Gametraversalbenchmark: Evaluating planning abilities of large language models through traversing 2d game maps. In The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track. (2024)
- [18] Hopkins, J., Bakler, M., Khan, A.: Factorio learning environment. *arXiv preprint arXiv:2503.09617* (2025)

- [19] Karten, S., Nguyen, A.L., Jin, C.: Pok\`echamp: an expert-level minimax language agent. arXiv preprint arXiv:2503.04094 (2025)
- [20] Xie, J., Zhang, R., Chen, Z., Wan, X., Li, G.: Whodunitbench: Evaluating large multimodal agents via murder mystery games. In The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track. (2024)
- [21] Costarelli, A., Allen, M., Hauksson, R., Sodunke, G., Hariharan, S., Cheng, C., Li, W., Clymer, J.M., Yadav, A.: Gamebench: Evaluating strategic reasoning abilities of llm agents. In Language Gamification-NeurIPS 2024 Workshop. (2024)
- [22] Hudi, F., Winata, G.I., Zhang, R., Aji, A.F.: Textgames: Learning to self-play text-based puzzle games via language model reasoning. arXiv preprint arXiv:2502.18431 (2025)
- [23] Qiao, D., Wu, C., Liang, Y., Li, J., Duan, N.: Gameeval: Evaluating llms on conversational games. arXiv preprint arXiv:2308.10032 (2023)
- [24] Hu, L., Li, Q., Xie, A., Jiang, N., Stoica, I., Jin, H., Zhang, H.: Gamearena: Evaluating llm reasoning through live computer games. In The Thirteenth International Conference on Learning Representations. (2025)
- [25] Paglieri, D., Cupiał, B., Coward, S., Piterbarg, U., Wolczyk, M., Khan, A., Pignatelli, E., Kuciński, Ł., Pinto, L., Fergus, R., Foerster, J.N., Parker-Holder, J., Rocktäschel, T.: BALROG: Benchmarking agentic LLM and VLM reasoning on games. In The Thirteenth International Conference on Learning Representations. (2025)
- [26] Ruoss, A., Pardo, F., Chan, H., Li, B., Mnih, V., Genewein, T.: LMAct: A benchmark for in-context imitation learning with long multimodal demonstrations. In Forty-second International Conference on Machine Learning. (2025)
- [27] Hu, L., Huo, M., Zhang, Y., Yu, H., Xing, E.P., Stoica, I., Rosing, T., Jin, H., Zhang, H.: lmgames-bench: How good are LLMs at playing games? In The Fourteenth International Conference on Learning Representations. (2026)
- [28] Tan, W., Zhang, W., Xu, X., Xia, H., Ding, Z., Li, B., Zhou, B., Yue, J., Jiang, J., Li, Y., An, R., Qin, M., Zong, C., Zheng, L., Wu, Y., Chai, X., Bi, Y., Xie, T., Gu, P., Li, X., Zhang, C., Tian, L., Wang, C., Wang, X., Karlsson, B.F., An, B., YAN, S., Lu, Z.: Cradle: Empowering foundation agents towards general computer control. In Forty-second International Conference on Machine Learning. (2025)
- [29] Waytowich, N.R., White, D., Sunbeam, M., Goecks, V.G.: Atari-gpt: Benchmarking multimodal large language models as low-level policies in atari games. arXiv preprint arXiv:2408.15950 (2024)
- [30] Liu, Y., Ouyang, K., Wu, H., Liu, Y., Sui, L., Li, X., Zhong, Y., Y.Charles, Zhou, X., Sun, X.: Videoreasonbench: Can MLLMs perform vision-centric complex video reasoning? In The Fourteenth International Conference on Learning Representations. (2026)
- [31] Shi, Y., Zhao, Q., Jiang, T., Zeng, X., Wang, Y., Wang, L.: RIVER: A real-time interaction benchmark for video LLMs. In The Fourteenth International Conference on Learning Representations. (2026)
- [32] Towers, M., Kwiatkowski, A., Terry, J., Balis, J.U., De Cola, G., Deleu, T., Goulão, M., Kallinteris, A., Krimmel, M., KG, A., et al.: Gymnasium: A standard interface for reinforcement learning environments. arXiv preprint arXiv:2407.17032 (2024)
- [33] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., Zaremba, W.: Openai gym. arXiv preprint arXiv:1606.01540 (2016)
- [34] Petrenko, A., Huang, Z., Kumar, T., Sukhatme, G.S., Koltun, V.: Sample factory: Egocentric 3d control from pixels at 100000 FPS with asynchronous reinforcement learning. In Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event. Volume 119 of Proceedings of Machine Learning Research., PMLR (2020) 7652–7662

- [35] Fisher, R.A.: Statistical methods for research workers. (1934)
- [36] Wikipedia contributors: One-way analysis of variance. Wikipedia, The Free Encyclopedia Accessed: 2026-05-07.
- [37] Benjamini, Y., Hochberg, Y.: Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)* **57**(1) (1995) 289–300
- [38] Luo, R., Xue, Z., Dimakis, A., Grauman, K.: When thinking drifts: Evidential grounding for robust video reasoning. In *Advances in Neural Information Processing Systems*. Volume 38. (2025)
- [39] Liang, Z., Li, J., Chen, W., Zhang, Y., Lu, H., Li, G.: OASIS: On-demand hierarchical event memory for streaming video reasoning. *arXiv preprint arXiv:2604.17052* (2026)
- [40] Liu, S., Zhuge, M., Zhao, C., Chen, J., Wu, L., Liu, Z., Zhu, C., Cai, Z., Zhou, C., Liu, H., Chang, E., Suri, S., Xu, H., Qian, Q., Wen, W., Varadarajan, B., Liu, Z., Xu, H., Bordes, F., Krishnamoorthi, R., Ghanem, B., Chandra, V., Xiong, Y.: Videoauto-r1: Video auto reasoning via thinking once, answering twice. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. (June 2026) 32829–32839
- [41] Xu, Z., Liu, C., Wei, Q., Wu, J., Zou, J., Wang, X., Zhou, Y., Liu, S.: More thinking, less seeing? assessing amplified hallucination in multimodal reasoning models. *Advances in Neural Information Processing Systems* **38** (2026) 82878–82905

Appendix Contents

A	Evaluation Protocol and Prompt	15
A.1	Fixed-Horizon Evaluation and Strategy Sensitivity	15
A.2	Evaluation Prompt	15
B	Additional Results	16
B.1	Raw Scores	16
B.2	Game-Specific RL Baseline	16
C	Harness Ablations	17
C.1	Influence of Textual Feedback	17
C.2	Action-Chunk Length	18
D	Failure Analysis and Diagnostics	19
D.1	Qualitative Failure Cases	19
D.2	VLM-Judge Analysis on Gameplay Failures	19
D.3	Diagnostic Questions and VLM-Judge Calibration	23
D.4	Reasoning Effort and Error Propagation	24
D.5	Text-History Overreliance	25
E	Cost Analysis	26
F	Ethics and Release	26

A Evaluation Protocol and Prompt

A.1 Fixed-Horizon Evaluation and Strategy Sensitivity

Original Atari episodes often terminate after a fixed number of lost lives. We do not use this game-over termination as the primary evaluation protocol because the resulting score can be strongly confounded by prompt-induced strategy. In some games, survival itself creates more scoring opportunities, so a small survival instruction can change both behavior and reward.

The old game-over protocol also has high per-run variability because the evaluation length is itself model- and strategy-dependent. Table 3 compares the original non-fixed-horizon Gemini 2.5 Flash runs with the current fixed-horizon runs on DemonAttack and Phoenix, using only the matched old setting: no thinking, structured visual prompting, three history clips, and no reward examples. The old protocol has substantially larger per-run standard deviation on both games.

Table 3: Gemini 2.5 Flash reward variability under original game-over termination versus the current fixed 30-second protocol. We report mean reward $\pm$ standard error and per-run standard deviation (SD).

Game	Protocol	Runs	Mean $\pm$ SE	SD
DemonAttack	Game-over termination	30	70.0 $\pm$ 6.6	36.3
DemonAttack	Fixed 30 seconds	10	41.0 $\pm$ 4.3	13.7
Phoenix	Game-over termination	30	171.3 $\pm$ 19.5	106.6
Phoenix	Fixed 30 seconds	10	198.0 $\pm$ 27.4	86.6

As a diagnostic, we evaluated a Phoenix prompt variant that added the sentence: “Important: You will lose a life if you are hit by the bombs. The game will end if you lose three lives. So avoiding being hit is the top priority!” Under game-over termination, this prompt shifted Gemini 2.5 Flash toward more survival-focused behavior and increased its average reward from 177.9 to 195.4 (see Table 4). This illustrates why AtariBench instead uses a fixed 30-second horizon with immediate reset after early termination: the benchmark measures closed-loop control over a standardized time budget rather than a model’s prompt-sensitive risk preference.

Table 4: Legacy strategy-sensitivity diagnostic under game-over termination. We report the number of reasoning traces classified as reward-seeking or survival-focused out of 100 sampled traces, along with the average reward at termination.

Model	Phoenix (neutral)				Phoenix (survival prompt)			
	Reward seeking	Survival focused	Neutral	Avg Reward	Reward seeking	Survival focused	Neutral	Avg Reward
Gemini 2.5 Flash	76	19	5	177.9	68	25	7	195.4
Gemini 2.5 Pro	59	34	7	239.5	80	16	4	243.8

A.2 Evaluation Prompt

Fig. 6 illustrates the evaluation prompt used in AtariBench, with Breakout as an example. The prompt combines game-specific instructions, recent and reward-relevant history, the current observation, and the required response format.

The Breakout prompt instantiates the harness described in Sec. 3. It begins with the game-specific instruction block, including the objective, available actions, action timing, and game-end condition. The prompt then provides the benchmark-managed memory used in the default `structured_history` mode: non-zero-reward history, recent history, and the current game state. Each historical clip records the model’s previous actions, the resulting frames, and any textual reward or life-loss feedback. Finally, the prompt fixes the output schema by requiring a free-form thought line followed by a move list, which the harness parses into ALE actions for closed-loop execution.

Evaluation Prompt: Breakout

Breakout | Structured-history prompting

You are an intelligent AI player playing a game. Your goal is to make progress and maximize total reward within the fixed time budget.

BREAKOUT Quick Guide:

Goal: Rewards are earned by hitting bricks with a ball controlled by a paddle, while preventing the ball from falling.

Available actions:

- noop: Do nothing. The paddle remains stationary.
- start: At the beginning of the game, generate a ball.
- left: Move the paddle to the left.
- right: Move the paddle to the right.

Your Task: Based on the current state of the game (which will be provided to you), decide on your next actions. Your move can contain 1 to 10 actions. Each action is executed for 0.1 second. As a result, your move is for the next 0.1 to 1 seconds. You are encouraged to plan for less actions if the ball is low and close, as shorter plans (fewer actions) will get feedback sooner. An example of response is:

thought: [your thought here]

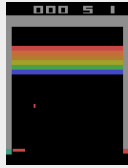
move: [right, right, right]

Time Budget: You are optimizing total reward over a fixed budget of 30 seconds. Losing a life does not directly reduce your score, but it does consume time because the game pauses and restarts the player state.

To make your decision, you will be provided with both historical context from your current game session and the current, up-to-the-moment game state. [Instruction of formats of historical context and current game state.]

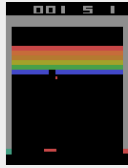
Non-zero reward history:

Action: right



Feedback: positive reward

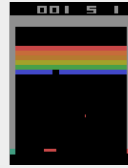
Action: noop



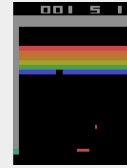
...

Recent history:

Action: right



Action: right | Current



...

Now please predict the next actions.

Figure 6: Example evaluation prompt for Breakout.

Textual feedback is part of the same simple harness design. Fig. 8 shows a Breakout reset case where the feedback makes a life-loss event explicit, allowing the model to choose `start` and resume play rather than stalling on a visually ambiguous reset screen.

B Additional Results

B.1 Raw Scores

Fig. 7 complements the main rank heatmap by preserving each game’s raw reward scale. This view is useful for checking whether rank differences correspond to large reward gaps or to small differences on games with narrow score ranges.

B.2 Game-Specific RL Baseline

We evaluate game-specific RL agents trained with APPO using Sample Factory [34] on 16 games shared with AtariBench. Each agent is evaluated using ten seeds, a 30-second gameplay duration, and no sticky actions. Scores are computed from raw ALE rewards before Sample Factory’s reward clipping, matching the VLM scoring protocol. Figs. 3 and 7 include the RL results. These agents rank first on 14 of the 16 games, with an average rank of 1.19. These results suggest that the state-of-the-art VLMs evaluated in AtariBench still lag behind specialized RL agents.

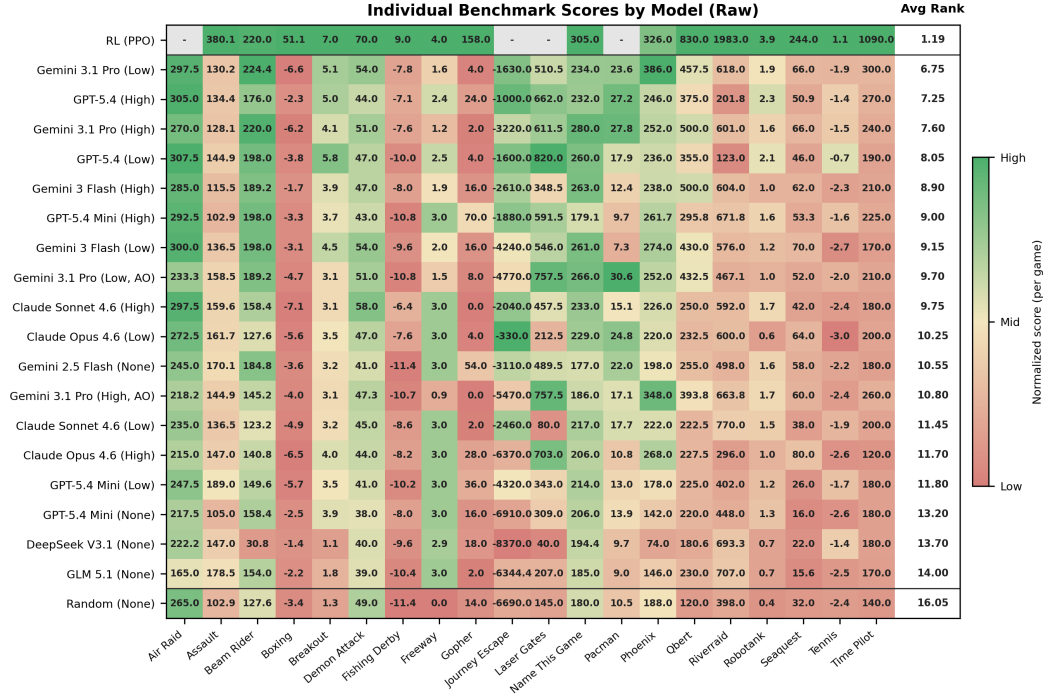


Figure 7: Raw model rewards across the 20 AtariBench games used in the main analysis, including the Gemini 3.1 Pro append-only settings shown in Fig. 3. The first row reports rewards for game-specific RL agents on 16 games, with “-” indicating unavailable results. Cell values preserve each game’s raw reward scale, while colors are normalized within each game. Avg Rank is computed as in Fig. 3.

Why 16 shared games? We selected AtariBench’s 20 games independently using the motion-perception, short-horizon interpretability, and game-diversity criteria in Sec. 3.2. For the RL comparison, we use publicly released Atari checkpoints listed in the Sample Factory documentation.² Matching public checkpoints are available for 16 of our games. The overlap follows checkpoint availability and was not selected based on RL scores; it retains shooters, sports, action games, and maze navigation.

C Harness Ablations

C.1 Influence of Textual Feedback

To test the influence of textual outcome feedback, we compare the standard prompt with a variant that removes reward, life-loss, and reset notices. We retain visual history, action labels, and the history-selection mechanism, including the selection of reward-relevant clips. We select one game from each of the four genres listed in Table 1: Breakout, Air Raid, Pacman, and Fishing Derby. We evaluate two models, Gemini 3.1 Pro (low) and Gemini 2.5 Flash (off), using ten seeds per setting.

Adding textual feedback significantly improves the mean scores of Gemini 2.5 Flash on Breakout and Fishing Derby. Significance is determined by whether the paired bootstrap 95% CI excludes zero (10,000 resamples; no multiple-comparison correction). For Gemini 3.1 Pro, no statistically significant differences are observed across the four games, indicating no detectable benefit from textual feedback for the stronger model.

By inspecting the reasoning traces of Gemini 2.5 Flash, we find that textual feedback helps the weaker model interpret the game state and choose appropriate actions. Without textual feedback, weaker models playing Breakout can stall after losing a ball if they fail to press start to serve

²<https://www.samplefactory.dev/09-environment-integrations/atari/>

Table 5: Textual feedback ablation. Scores are mean $\pm$ SE over ten seeds.

Model	Game	Full feedback	No feedback
Gemini 3.1 Pro	Breakout	4.5 ± 0.2	4.7 ± 0.3
	Air Raid	225.0 ± 23.9	215.0 ± 29.6
	Pacman	31.6 ± 5.4	26.7 ± 3.1
	Fishing Derby	-8.4 ± 1.8	-8.6 ± 1.4
Gemini 2.5 Flash	Breakout	2.3 ± 0.3	1.5 ± 0.3
	Air Raid	255.0 ± 45.3	282.5 ± 30.1
	Pacman	18.0 ± 3.6	18.0 ± 2.6
	Fishing Derby	-9.0 ± 1.3	-11.8 ± 0.2

another (Fig. 8). Such stalls can leave scores at only 0 or 1 over the 30-second window, making performance difficult to distinguish. These observations suggest that textual feedback can help weaker models make progress.

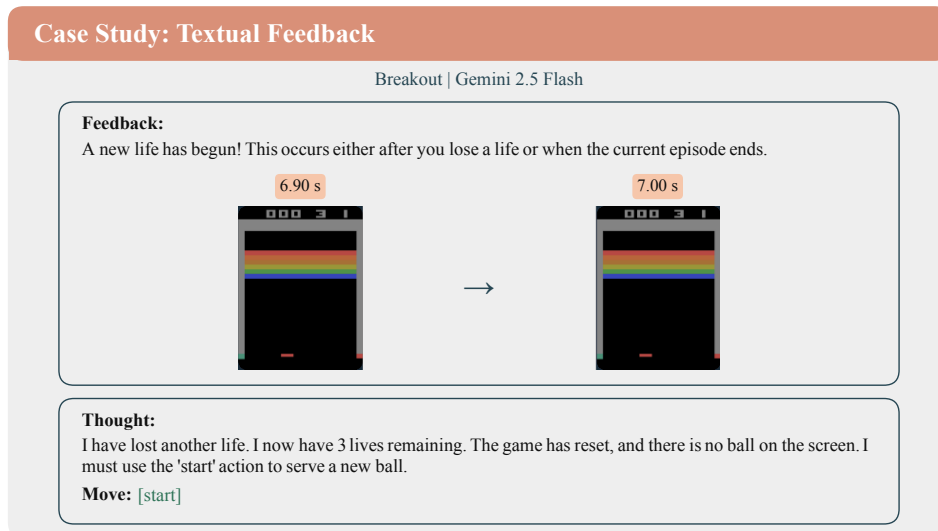


Figure 8: Breakout example with Gemini 2.5 Flash in which textual life-loss feedback makes the reset state explicit. After the harness reports that a new life has begun, the model selects `start`, which serves the next ball and resumes play. Without textual feedback, the model stalls and fails to press `start` to proceed.

C.2 Action-Chunk Length

We vary the permitted number of actions per model query: exactly one, one to five, or the default one to ten. We use the standard textual output format and structured-history prompt for Gemini 3 Flash (low) and GPT-5.4 Mini (none) on the same four games selected above.

Seven of the eight model–game pairs show no significant difference between the default 1–10 range and shorter settings. Significance is determined by whether the bootstrap 95% CI of the score difference excludes zero (10,000 resamples; paired by seed where seeds match, otherwise unpaired; no multiple-comparison correction). The only significant difference is observed for Gemini 3 Flash on Breakout, where one-action control scores higher than the default range (4.6 versus 3.7), suggesting that finer-grained control can help in games that require precise timing. We retain the 1–10 action range because it allows the model to choose short action sequences when needed while balancing evaluation cost.

Table 6: Reward under different allowed action-chunk lengths (mean $\pm$ SE over ten seeds). Variable-length conditions permit the model to choose shorter sequences at each turn.

Model	Game	1 action	1–5 actions	1–10 actions
Gemini 3 Flash	Air Raid	295.0 $\pm$ 33.3	340.0 $\pm$ 36.7	300.0 $\pm$ 31.0
	Breakout	4.6 $\pm$ 0.2	4.5 $\pm$ 0.2	3.7 $\pm$ 0.3
	Fishing Derby	−6.6 $\pm$ 2.1	−8.4 $\pm$ 1.7	−7.2 $\pm$ 1.4
	Pacman	17.4 $\pm$ 4.0	13.6 $\pm$ 3.1	11.9 $\pm$ 2.8
GPT-5.4 Mini	Air Raid	175.0 $\pm$ 18.3	265.0 $\pm$ 39.8	217.5 $\pm$ 41.2
	Breakout	3.4 $\pm$ 0.2	3.8 $\pm$ 0.3	3.9 $\pm$ 0.3
	Fishing Derby	−10.2 $\pm$ 0.8	−9.8 $\pm$ 0.6	−8.0 $\pm$ 0.8
	Pacman	12.4 $\pm$ 1.5	11.9 $\pm$ 1.3	13.9 $\pm$ 1.8

D Failure Analysis and Diagnostics

D.1 Qualitative Failure Cases

We provide examples of the visual-perception, temporal-grounding, motion-reasoning, text-history, and long-context action-selection failures discussed in Sec. 4.3.

- Fig. 9 illustrates object omission in Name This Game: GPT-5.4 (high) denies that a shark is visible despite its presence in the current frame. The selected movement and firing actions remain reasonable; the supported error is the stated absence of the shark.
- Fig. 10 illustrates a historical-state/current-state mismatch in Freeway: the described car position matches earlier input frames but contradicts the current frame.
- Fig. 11 illustrates a motion-reasoning error in Breakout: the relevant objects are visible, but the inferred direction contradicts the ground-truth trajectory.
- Fig. 12 illustrates a text-history conflict in Assault: the model repeats a previously rewarded firing action although the target moves out of alignment.
- Fig. 13 illustrates the long-context failures. Under append-only prompting, the model repeatedly outputs noop actions despite receiving updated frames and feedback. This behavior illustrates how models can fail to update their actions as the game state changes under long-context prompting.

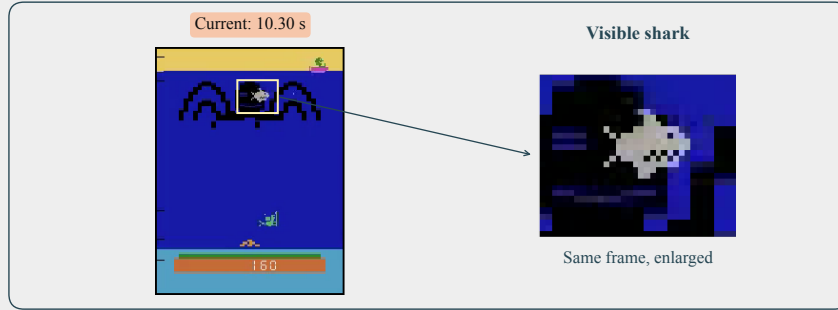
D.2 VLM-Judge Analysis on Gameplay Failures

The controlled QA diagnostics described in Sec. 4.4 provide a breakdown of failures, but they are manually checked and limited in scale. Here we provide a VLM-judge analysis of gameplay failures, complementing the controlled QA diagnostics.

Sampling and adjudication. We study Claude Opus 4.6 (low), GPT-5.4 (high), and Gemini 3.1 Pro (low). For each model–game pair, we consider the benchmark runs with complete logged trajectories, select the run whose score is closest to the median, and randomly sample five decision turns at least five turns apart, yielding 100 turns per model. Turn selection does not condition on reward or life loss. GPT-5.6-Sol with high reasoning receives the model’s input frames and text, stated rationale, selected actions, and subsequent observations and feedback. Model identity and reasoning-setting labels are hidden. The judge first determines whether the evidence supports an observable gameplay failure, then assigns one primary category: spatiotemporal reasoning, decision and control, visual perception, exploration, or task and rule understanding. A penalty alone does not establish a model error. The judge returns a structured judgment containing failure presence, one primary category, a subtype, and supporting evidence. A no-observable-failure label means that the available evidence does not support an error; it does not establish optimal play. As a sanity check, the judge achieves 93.0% accuracy (279/300 responses) on the controlled QA diagnostic set in Sec. 4.4, using a clarified fish-count prompt (Appendix D.3). This diagnostic accuracy is not an estimate of trajectory-attribution accuracy.

Failure Case: Visual Perception

Name This Game | GPT-5.4 (high) | Turn 31



Thought:

No shark or oxygen line is visible and the tentacles are currently retracted. I should shift a bit farther right into the usual central-right lane under the boat/oxygen drop while firing so I'm ready to score on a shark, tentacle, or oxygen line as soon as it appears.

Move: [RIGHTFIRE, RIGHTFIRE, FIRE]

Figure 9: In Name This Game, GPT-5.4 (high) states that no shark is visible despite the white shark in the current frame (10.30 s; enlarged at right).

Failure Case: Temporal Misgrounding

Freeway | GPT-5.4 (high)



Thought:

The left chicken is in a safe gap and should keep climbing. The nearest threat is the blue car in the lane above moving left, but it is **still far enough right** that pushing upward now should make progress before it reaches our column. Use a short aggressive plan and reassess soon.

Move: [UP, UP, UP, UP]

Same region enlarged: the car crosses the chicken's column

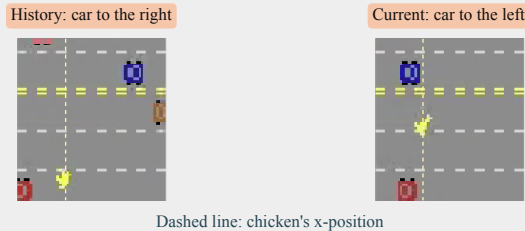


Figure 10: In this Freeway case, GPT-5.4 (high) describes a historical car position as current; with history removed and the remaining input and model settings unchanged, the erroneous relation is no longer stated (see Appendix D.2).

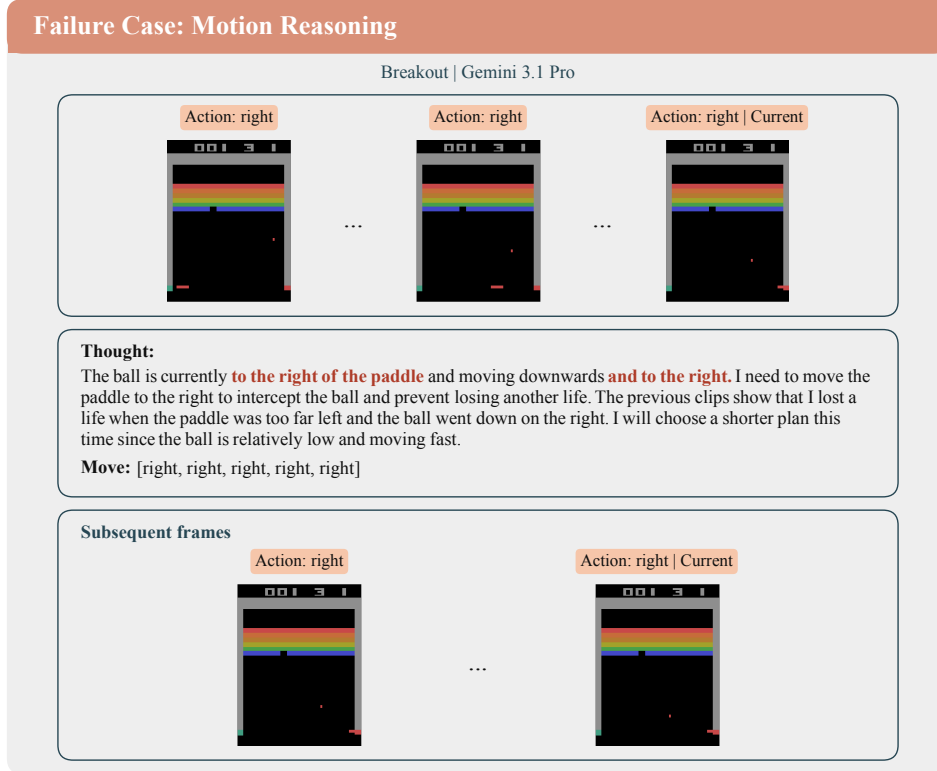


Figure 11: Breakout example of Gemini 3.1 Pro where the model misidentifies the ball’s motion direction.

For example, a Breakout agent describes a ball moving down-left when the frames show it moving up-right, then executes four leftward actions. Such cases connect an incorrect estimate of the evolving visual state to the selected control action. Decision and control errors include cases where a plausible stated plan is not reflected in the executed actions; visual-perception errors include mistaking the controlled aircraft for an enemy in Time Pilot.

Failure categories. The judge identifies 106 failures among 300 sampled turns (35.3%). Motion/state dynamics and trajectory prediction account for 41 failures (38.7%: 32 motion and nine trajectory-prediction errors), decision and control for 27 (25.5%), spatial relations for 19 (17.9%), visual perception for 15 (14.2%), exploration for three (2.8%), and task/rule understanding for one (0.9%). Fig. 14 retains these original labels and separately examines 28 spatial-relation cases from the same source runs.

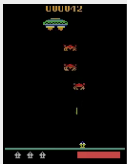
Interpretation and limitations. For reporting, we separate spatial relations from motion/state dynamics and trajectory prediction within the original judge’s spatiotemporal category. Motion/trajectory prediction is the largest reported group, alongside substantial decision/control, spatial-relation, and perception errors. These results do not establish gameplay score as a pure measure of spatiotemporal reasoning or measure the causal effect of each failure on total reward. Attribution remains model-assisted: the judge can misread visual evidence, and an agent’s stated rationale need not fully explain its action. With only five turns from one selected run per model–game pair, the results are descriptive and should not be used to rank models or estimate failure rates across all benchmark trajectories.

Temporal Misgrounding. To examine whether temporal misgrounding contributes to spatial-relation errors, we compare complete-history-input and current-frame-only re-queries for spatial-relation failure cases. Complete input preserves the original prompt and ordered images; current-frame-only input removes history images, associated action/reward text, and the history-format explanation, while retaining the current frame and timestamp, game instructions, action task, and

Failure Case: Text Overreliance

Assault | GPT-5.4 Mini

Action: fire_up



...

Action: fire_up



...

Action: fire_up | Current



Earlier feedback: "You received a positive score!"

Thought:
The cannon is aligned under the remaining vertical stack of aliens, and **recent FIRE_UP actions have been landing hits without overheating**. Staying put and **continuing to fire upward** should maximize the chance of finishing the next ship before it shifts away.

Move: [fire_up, fire_up]

Subsequent frames

Action: fire_up



Action: fire_up | Current



Score remains 63

Figure 12: Assault text-history-overreliance case in GPT-5.4 Mini. After an earlier rewarded upward shot, the model says recent shots are hitting despite updated frames showing the remaining alien left of its firing lane. Two more upward shots yield no reward.

Failure Case: Action Stagnation

Breakout | Append-only prompting

Observation: 6.40 s



Prompt:
Now please choose your actions following the same format in the initial prompt.

Thought (response #58):
The ball is moving upwards and is slightly left of center. I just moved the paddle one step to the right, placing it in a good central position. I will hold this position and wait for the ball to hit the bricks and come back down.

Move: [noop, noop, noop]

Thought (response #60):
The ball is still traveling upwards, approaching the green bricks slightly to the left of center. My paddle is in a good central waiting position. I will continue to hold this position.

Move: [noop, noop, noop]

Figure 13: Example of action stagnation under append-only prompting in Breakout. Despite receiving updated frames and feedback, the model repeatedly outputs only noop actions; we observe this behavior in Gemini 2.5 Flash, Gemini 2.5 Pro, and Gemini 3 Flash. We do not observe the same collapse in Gemini 3.1 Pro, suggesting improved ability to use long interaction histories.

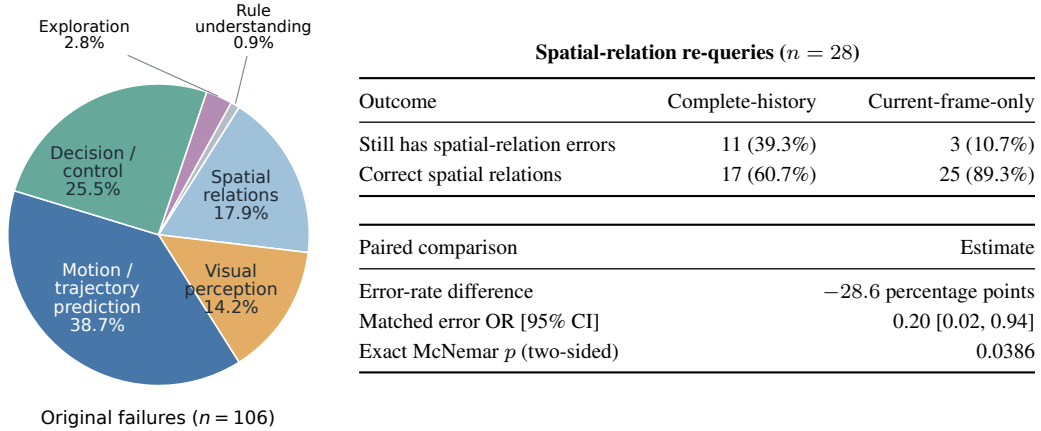


Figure 14: Original VLM-judge failures (left, $n = 106$) and paired spatial-relation re-queries (right, $n = 28$). Current-frame-only inputs remove history images and associated text while retaining the current frame, game instructions, and model settings. Responses without a spatial-relation statement are re-queried with identical input; the first response stating a relation is assessed, whether correct or incorrect. Correct relations need not explicitly repair the original claim. The matched error odds ratio (OR; current only versus complete input) has an exact 95% confidence interval.

model settings. We then check whether a spatial relationship is mentioned in the response. The cases include the 19 judged spatial-relation failures above and 11 additional failures sampled from the same 60 source runs; excluding two cases that lack a spatial-relation statement after re-querying leaves 28 cases for analysis.

We assess the current static spatial relations expressed in each response, including relative position, alignment, distance, and containment. *Still has a spatial-relation error* means at least one stated relation is contradicted by the current frame. *Correct spatial relations* means the stated relations are supported at their expressed level of detail; it does not require explicitly repairing the originally disputed claim. Pure object detection, motion direction, and future action predictions are outside this spatial-only classification.

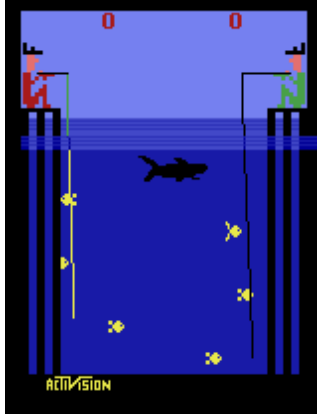
As shown in Fig. 14, complete-history-input re-queries contain spatial errors in 11/28 cases (39.3%), compared with 3/28 (10.7%) for current-frame-only inputs (two-sided exact McNemar $p = 0.0386$). This difference is consistent with temporal misgrounding as one possible source of spatial-relation errors, related to reliance on outdated frames [38] and failures of present-time grounding [39]. Because current-frame-only inputs remove both history images and the associated textual history, this comparison does not isolate the effect of historical frames alone.

D.3 Diagnostic Questions and VLM-Judge Calibration

The controlled QA diagnostics in Sec. 4.4 use 60 manually curated question variants across ten games, with five queries per question. Each game contributes two perception questions, two motion questions, and one decision question with short and long prompts. The games are Assault, Boxing, Breakout, Fishing Derby, Freeway, Gopher, Name This Game, Pac-Man, Q*bert, and River Raid. Table 2 in the main text reports the diagnostic results.

Table 7 gives the prompt text for the multiple-choice diagnostic protocol using Fishing Derby as an example. These closed-choice questions are distinct from the trajectory-level judge protocol in Appendix D.2. The remaining samples use the same structure with game-specific object names and choices. Each query begins with the game-identification sentence; frames are provided as an interleaved timestamped sequence; perception questions use their designated single frame; motion questions use a timestamped sequence, while decision questions use the task-specific frame or sequence.

Table 7: Diagnostic question prompts for each capability group, using Fishing Derby as an example. The long decision prompt additionally prepends the game-specific guidelines.



Capability	Prompt text before interleaved frames
Perception: detection	This is the Atari game Fishing Derby. How many yellow fish are in the game? Choose from 4, 5, 6, or 7.
Perception: spatial reasoning	This is the Atari game Fishing Derby. In the last frame, focus on the deepest fish in the left half of the sea and the left player's fishing line. Is the left player's fishing line currently to the left or to the right of that fish? Choose from left or right.
Motion: target object	This is the Atari game Fishing Derby. Focus on the deepest fish in the left half of the sea. Which direction is that fish moving? Choose from left or right.
Motion: controlled object	This is the Atari game Fishing Derby. Focus on the left player's fishing line. Across the timestamped sequence, what is the fishing line's overall movement direction? Choose only one from upright, upleft, downleft, downright, up, down, left, right, or still.
Decision: short	This is the Atari game Fishing Derby. Focus on the deepest fish in the left half of the sea and the left player's fishing line. To catch that deepest fish, which direction should the left player's fishing line move next? Choose from left or right.
Decision: long	This is the Atari game Fishing Derby. <i>Action Planning Guidelines: [complete Fishing Derby guide included in the model prompt; omitted here, approximately 500 tokens of instruction].</i> Using the frame sequence, focus on the deepest fish in the left half of the sea and the left player's fishing line. First identify that target fish and the current position of the left player's fishing line, then infer their motion. To catch that fish, which direction should the left player's fishing line move next? Choose from left or right.

VLM-Judge Calibration. On the controlled diagnostic QA set, GPT-5.6-Sol (high), our trajectory-level judge, achieves 88.0% perception accuracy, 93.0% motion accuracy, and 98.0% decision accuracy (100 responses per category), with an overall accuracy of 93.0% (279/300), clearly higher than the best evaluated model in Table 2 (GPT-5.4 (low), 86.3%). For trajectory-level failure attribution, the judge additionally receives post-action frames and environment feedback, providing evidence of the selected actions' actual outcomes.

D.4 Reasoning Effort and Error Propagation

Fig. 3 shows that higher reasoning does not consistently improve performance. We investigate whether failures arise from state estimation, planning, or both. For state-estimation failures, we further define a failure as *propagated* when an incorrect state estimate leads to an incorrect or clearly suboptimal action.

We use the VLM judge to identify failures and randomly select 50 failed turns each for GPT-5.4 (high) and GPT-5.4 (low) for further analysis. We ask the judge to classify each failure as a spatiotemporal state-estimation error, a planning error despite a substantially correct state estimate, or another type of failure. For state-estimation errors, we assess whether the incorrect estimate leads to an erroneous action. An incorrect description alone does not count as propagation if the selected action remains reasonable or the link to an action error is unclear.

Table 8: Failure types among 50 failed turns per setting. State estimation, planning with a substantially correct state estimate, and other failures are mutually exclusive. Propagation is measured only among state-estimation failures.

Setting	State-estimation failures	Planning failures with correct state	Other	Propagation
GPT-5.4 (high)	37/50 (74%)	3/50 (6%)	10/50 (20%)	26/37 (70.3%)
GPT-5.4 (low)	29/50 (58%)	12/50 (24%)	9/50 (18%)	20/29 (69.0%)

Spatiotemporal state estimation accounts for 66/100 failures, compared with 15/100 attributed to planning despite a substantially correct state estimate. The remaining 19 cases involve static visual errors or other bottlenecks. Among state-estimation failures, 26/37 (70.3%) in the high-effort setting and 20/29 (69.0%) in the low-effort setting propagate into erroneous actions, for a combined 46/66 (69.7%). One case in each setting has insufficient evidence to establish propagation and is excluded from the numerator.

Under high reasoning effort, the failure composition shifts toward state estimation (74% versus 58% of sampled failures), while planning accounts for a smaller share (6% versus 24%). Because only failed turns are sampled, these shares describe the composition of failures rather than absolute failure rates. In both settings, state-estimation errors propagate into incorrect actions at a similar rate (approximately 70%), suggesting that higher reasoning effort does not reduce the downstream impact of spatiotemporal misestimation.

This is consistent with prior findings: VideoAuto-R1 [40] finds that direct answering often matches or outperforms chain-of-thought on perception-oriented video tasks, while More Thinking, Less Seeing? [41] shows that longer reasoning chains can reduce attention to visual inputs and amplify visual hallucinations.

D.5 Text-History Overreliance

We use the VLM judge to identify failures involving reliance on stale textual history in 240 randomly sampled turns across all 20 games and four model settings: GPT-5.4 (high), Gemini 3.1 Pro (low), GPT-5.4 Mini (low), and Gemini 2.5 Flash (off). We treat GPT-5.4 (high) and Gemini 3.1 Pro (low) as the stronger settings and GPT-5.4 Mini (low) and Gemini 2.5 Flash (off) as the weaker settings. Each setting contributes three turns per game, for 60 turns in total, sampled independently of reward or life loss. We label a failure as text-history overreliance when the model follows a past textual cue that conflicts with current visual evidence instead of updating its decision from the frames. Repeated actions or an incorrect motion estimate alone do not satisfy this definition.

Table 9: Text-history overreliance in sampled gameplay turns. Failures counts the observable failures identified by the judge, and Overreliance Rate is the percentage of these failures attributed to text-history overreliance.

Model setting	Turns	Failures	Text Overreliance Failures	Overreliance Rate
GPT-5.4 (high)	60	21	2	9.5%
Gemini 3.1 Pro (low)	60	20	0	0.0%
GPT-5.4 Mini (low)	60	24	5	20.8%
Gemini 2.5 Flash (off)	60	43	10	23.3%
Two stronger settings	120	41	2	4.9%
Two weaker settings	120	67	15	22.4%

Conditional on observable failure, text-history overreliance occurs in 4.9% of cases for the two stronger settings and 22.4% for the two weaker settings (two-sided Fisher exact test, $p = 0.0154$). Measured per sampled turn, the rates are 1.7% (2/120) and 12.5% (15/120), respectively ($p = 0.0016$). These results suggest that weaker models are more prone to relying on stale textual history, while stronger models exhibit this problem less frequently.

E Cost Analysis

AtariBench does not require model training or local GPU inference in the released harness. The local worker runs ALE simulation, frame capture, prompt assembly, parsing, and artifact writing on CPU; the dominant reproducibility cost is paid VLM API usage. A rollout issues one model query per closed-loop decision turn, so the number of calls depends on the model’s chosen action sequence length rather than on the raw Atari frame count.

We calculate Gemini 3.1 Pro evaluation cost from a measured AtariBench token footprint, covering 20 games with 10 runs per game. The token accounting includes input, output, and thinking-token components, and we price this 200-run closed-loop visual-control footprint with the Gemini 3.1 Pro token prices used for the paper.

Table 10: Estimated Gemini 3.1 Pro API cost for 200 AtariBench runs (20 games $\times$ 10 runs).

Setting	Runs	Calls	Input tok.	Output tok.	Thinking tok.	Est. cost
Ours	200	27,938	140.81M	2.85M	17.18M	\$522

The estimate uses the Gemini 3.1 Pro prices for prompts under 200k tokens: \$2 per million input tokens and \$12 per million output tokens, with thinking tokens counted at the output rate. The resulting formula is

$$\text{cost} = \frac{2N_{\text{input}} + 12(N_{\text{output}} + N_{\text{think}})}{10^6}.$$

The matched setting has an average of 5.0k input tokens per model call under the structured-history memory mode, so the under-200k prompt tier is the appropriate estimate. Context caching is not included in these totals; when cache accounting is used, the price would be lower.

For scale, a one-call-per-task SWE-bench run with 100k input and 20k output tokens per task would cost approximately \$1,009 under the same Gemini 3.1 Pro pricing.

F Ethics and Release

AtariBench uses the Arcade Learning Environment (ALE) as an external GPL-2.0-licensed dependency. We will document the exact ALE version used for evaluation and preserve the applicable copyright and license notices. The required Atari ROMs are provided through the official `ale-py` package, following ALE’s installation instructions.³ Users should follow the applicable dependency and game-asset terms.

The AtariBench code repository is <https://github.com/mignonjia/AtariBench>. Our benchmark release includes code, configurations, prompts, and non-visual metadata needed to reproduce the evaluation. We will not bundle Atari ROM binaries, rendered-frame datasets, or bulk visual trajectories with the benchmark release. Users can generate gameplay trajectories locally through the ALE dependency.

³<https://github.com/Farama-Foundation/Arcade-Learning-Environment>

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope?

Answer: [Yes]

Justification: The abstract and introduction state that AtariBench evaluates VLMs in closed-loop Atari gameplay and that the main findings concern spatiotemporal reasoning, motion comprehension, sequential decision making, and visual grounding. These claims are supported by the benchmark design in Section 3, the main evaluation, the controlled QA diagnostics in Section 4.4, and the supplementary trajectory-level failure analysis in Appendix D.2.

Guidelines:

- The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: Section 3 discusses this limitation in the harness design: AtariBench intentionally uses a simple and necessary harness for closed-loop gameplay, rather than studying complex agent harnesses with external planners, tool-use modules, or other optimized scaffolding. The paper also discusses observed model failure modes and fixed-horizon design tradeoffs in the experimental analysis and appendix.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate “Limitations” section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren’t acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper introduces an empirical benchmark and experimental analysis, and does not claim theoretical results or provide proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Section 3 describes the ALE/Gymnasium environment interface, selected games, action frequency, fixed 30-second horizon, prompt memory design, textual feedback, and output schema. The appendix provides prompt details and raw-score views, and the released code/configuration specifies the executable harness used to reproduce runs.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

- (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
- (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The accompanying code release contains the core AtariBench runner, game registry, model adapters, reviewer-scale configs, and README instructions for environment setup, single runs, batch runs, and tests. The benchmark uses generated trajectories from ALE rather than a static private dataset; generated run artifacts can be reproduced from the code and provider access.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://neurips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://neurips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: There is no model training or optimizer in this work, but the evaluation settings are specified: 20 selected Atari games, 30-second fixed-horizon runs, 10 Hz control by repeating actions for three ALE frames, disabled sticky actions, prompt memory variants, model settings, and repeated closed-loop runs. These details appear in Section 3, Section 4, and the appendix.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: The game separability analysis reports two-group and one-way ANOVA tests with Benjamini–Hochberg correction across games, together with effect-size thresholds. Figures/tables with uncertainty describe standard errors where used (e.g., Appendix A.1 and the append-only analysis figure). The variability captured is across repeated runs or repeated model queries under the same benchmark setting.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Appendix E states that the released harness runs ALE and artifact generation on CPU without local GPU inference, identifies paid VLM API calls as the dominant resource, and reports a 200-run Gemini token-footprint estimate with the corresponding Gemini 3.1 Pro cost calculation.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: The work is an evaluation benchmark built on a standard game environment and public/commercial model APIs. It does not collect personal data, train models on sensitive data, or deploy systems for real-world decision making.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [No]

Justification: The current draft motivates the scientific value of diagnosing VLMs in dynamic settings, but it does not yet contain an explicit broader-impact discussion. A suitable paragraph could mention positive impacts for safer evaluation of interactive agents and potential negative impacts from over-interpreting benchmark performance or improving agents that could later be used in unsafe real-world control settings.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release a new model, scraped dataset, or high-risk generative asset. It releases benchmark code for controlled Atari environments using publicly available ALE environments.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [\[Yes\]](#)

Justification: The paper cites ALE and Gymnasium/OpenAI Gym as the underlying environment interfaces and names the evaluated model providers/settings in the experimental section. Appendix F identifies the ALE dependency license, discusses applicable dependency and game-asset terms, and describes the scope of the benchmark release.

Guidelines:

- The answer [\[N/A\]](#) means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [\[Yes\]](#)

Justification: The new asset is the AtariBench evaluation harness and associated benchmark configuration. Section 3, Appendix A.2, and the accompanying code README document the environment interface, selected games, prompts, memory modes, output schema, and reproduction commands.

Guidelines:

- The answer [\[N/A\]](#) means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not report crowdsourcing experiments or research with human subjects. Manual game selection and diagnostic labeling were performed by the authors, not by recruited participants.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The work does not involve human subjects, participant data, or interventions with people, so IRB approval is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: VLM/LLM usage is central to the paper: Section 3 and Section 4 describe the VLM agents and evaluation protocol. Appendix D.2 describes the GPT-5.6-Sol judge used for model-assisted failure attribution, including its limitations.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.