StarBench: A Turn-Based RPG Benchmark for Agentic Multimodal Decision-Making and Information Seeking

Haoran Zhang*
University of Michigan
Ann Arbor, United States
haoranwh@umich.edu

Hanzhe Guo University of Michigan Ann Arbor, United States hanzheg@umich.edu Chenhao Zhu* Stanford University Stanford, United States chenhzhu@stanford.edu

Haiming Li University of Michigan Ann Arbor, United States haiming@umich.edu Sicong Guo University of Michigan Ann Arbor, United States stevengu@umich.edu

Donglin Yu University of Illinois Urbana–Champaign Urbana–Champaign, United States donglin5@illinois.edu

ABSTRACT

Human players do more than press buttons: they ground what they see on screen into precise keyboard-mouse actions and, when stuck, they seek information before trying again. We ask whether current vision-language models (VLMs) can do the same. Despite encouraging results under simplified control or tool scaffolds, humanlike play in a real client—mapping raw screenshots to temporally coherent low-level actions while deciding when to ask for guidance-remains an open challenge. We introduce StarBench, a turnbased RPG benchmark derived from Honkai: Star Rail that targets these two human-like competencies: multimodal decision-making from pixels to actions and agentic information seeking. StarBench standardizes evaluation across eight combat tasks and two regimes with shared tasks and metrics: (i) direct control, where agents receive only screenshots and must emit low-level primitives (click and keypress) with no semantic hints; and (ii) tool-assisted control, where higher-level intents can be mapped to primitives by detectors and OCR outputs provide optional textualized observations to ease UI grounding. To mirror human practice, StarBench also includes an ask-or-act diagnostic that measures whether and when agents choose to request brief guidance before proceeding, and how that choice affects subsequent performance. We report reference baselines for contemporary VLMs and a human reference. Results expose sizable gaps in perception-to-control fidelity in the direct regime, while showing that judicious information seeking correlates with improved success-establishing StarBench as a reproducible yardstick for agentic information seeking and multimodal decision-making in real-client play.

KEYWORDS

VLMs, Multimodal Benchmark, Decision-making, UI grounding

1 INTRODUCTION

With the emergence of large language models (LLMs) such as ChatGPT [27], research on intelligent agents has expanded from language-centric tasks (e.g., code generation [17] and conversational systems [9]) to interaction with dynamic environments [43]. This shift raises a direct question: *can current VLMs play games like humans do*—perceiving raw screens, deciding, and issuing precise

keyboard-mouse actions—without relying on bespoke macros or scripted APIs?

Games offer a compelling testbed for this question: they are structured yet diverse, repeatable yet challenging. Prior work spans traditional board games and modern role-playing games (RPGs), from *Pokémon* [16] to classic titles like *Go* and *Chess* [8, 44]. Early evaluations often targeted unimodal agents—purely visual [25] or purely textual [4]. More recent efforts incorporate multimodal inputs, but many rely on simplified simulators, scripted APIs, or macros, which blur whether a model can truly ground pixels into low-level control.

Answering the human-like play question requires a benchmark that *separates* two factors yet evaluates them under the *same* tasks and metrics. First, **direct screenshot-to-action control**: the agent sees only the screenshot and must output low-level primitives (click and keypress) at precise coordinates. Second, **information access**: humans routinely decide whether to look up information after setbacks; thus, beyond raw control, a benchmark should measure *when* an agent chooses to seek information versus proceeding to act.

Honkai: Star Rail (HSR) is a turn-based RPG whose squad-based combats naturally intertwine visual cues (enemy states, turn order) and textual descriptions (skills, mechanics, character specs). Its complexity and evolving UI patterns make it a strong platform for studying perception-to-control links in realistic clients. Importantly, HSR is a continually updated live-service game: periodic balance changes, new enemies, and UI adjustments reduce the utility of static prior knowledge embedded in pretrained models. This dynamic setting naturally pressures agents to decide when to seek task-relevant information (e.g., clarifying mechanics or target priorities) versus proceeding without it.

We introduce **StarBench**, a benchmark derived from HSR that evaluates multimodal agents in a *real game client* across eight combat tasks. StarBench standardizes both **direct control** (screenshot \rightarrow keyboard-mouse) and **tool-assisted control** (computer-operation primitives with textualized observations) under identical task definitions, success criteria, and configurations for reproducibility. To analyze the decision of seeking information, StarBench further includes an **ask-or-act** diagnostic experiment: before each battle, agents may either ASK a targeted question to obtain bounded textual guidance or CONTINUE without assistance; we measure ask

^{*}Equal contribution

Table 1: Comparison of game-centric evaluation settings. StarBench uniquely pairs a *direct screenshot-to-action* track with a matched *tool-assisted* track under identical tasks/metrics, and adds an *ask-or-act* decision diagnostic.

	Enviro	Control Interface	Scaffolds	Tracks & Diagnostics		
Benchmark / Setting	Real Client	Observation	Action Interface	Tool Assist	Direct Ctrl. Track	Ask-or-Act
Atari ALE [25]	Emulator	Pixels	Low	No	Yes	No
StarCraft II (RL) [23]	Engine API	Pixels+Text	High	No	No	No
MineDojo / Voyager [6, 42]	Real Client + Engine API	Pixels+Text	High	Yes	No	No
PokéLLMon [16]	Engine API	Text	High	No	No	No
BALROG [28]	Emulator	Pixels+Text	High	Yes	No	No
CRADLE (RDR2) [37]	Real Client	Pixels+Text	High	Yes	No	No
MiniHack [32]	Engine API	Text	High	No	No	No
NetHack (NLE) [19]	Engine API	Text	High	No	No	No
StarBench (ours)	Real Client	Pixels (& optional Text)	Low/High	Optional	Yes	Yes

Real Client: whether actions are sent to a real game window via OS events (vs. emulator/engine APIs). Observation: what the agent perceives (raw screenshots "Pixels"; Engine/API returned state information/OCR summaries "Text"). Action Interface: Low-level OS primitives like click/keys vs. High-level scripted/engine APIs or macros). Tool Assist: availability of helpers that simplify perception or action selection (e.g., textualized UI summaries and detector/OCR JSON). Direct Ctrl. Track: presence of a pure screenshot→primitive setting with no semantic aids. Ask-or-Act: whether the setting includes a controlled decision point to ask for concise guidance or proceed without it.

rate, timing, benefit, and downstream performance. Our contribution could be summarized as following:

- Benchmark specification. We present StarBench, a singletitle yet multi-task benchmark in HSR with a precise environment interface, eight tasks spanning distinct combat competencies, and standardized metrics and configurations for reproducibility.
- (2) Two evaluation regimes. We formalize protocols for both direct screenshot-to-action control and tool-assisted control (computer-operation primitives with textualized observations).
- (3) Ask-or-act decision diagnostic. We provide a methodagnostic diagnostic that quantifies whether and when agents decide to seek information (ask) versus proceed (act), and how these choices affect subsequent gameplay.
- (4) **Reference baselines.** We report reference results for contemporary VLMs and a human reference to establish starting points rather than state-of-the-art claims.

2 RELATED WORK

2.1 Vision-Language Models

Vision-language models (VLMs) have emerged as a powerful paradigm for integrating visual and textual modalities, allowing for a variety of applications, including image captioning [41, 45], visual question answering [2], and cross-modal retrieval [21]. Early approaches, such as CLIP [29], leverage contrastive learning to align visual and language representations, thereby achieving robust zeroshot performance in various classification tasks. More recent work extends these methods to generate textual output conditioned on visual inputs. For instance, models such as GPT-4V [26], Gemini [38], and Claude-3.5-Sonnet [1] employ cross-attention mechanisms [22] to fuse features from vision transformers [5, 40] with large language models. This architectural integration not only facilitates the generation of detailed image descriptions, but also enhances contextual understanding in downstream tasks.

2.2 Games as Benchmarks

Since Arthur Samuel's seminal work in 1959 on checkers-playing programs [31], games have served as fundamental benchmarks for evaluating artificial intelligence systems [15, 35]. The evolution of game-based benchmarks mirrors advances in AI capabilities, from rule-based board games like Chess that challenged early symbolic systems [33], to real-time video games like Atari games and StarCraft II that tested reinforcement learning approaches [23, 25]. Modern interactive environments such as *Minecraft* [6, 18, 42] and RDR2 [37] now push the boundaries of multimodal understanding and long-horizon planning. These gaming environments offer three key advantages as AI benchmarks: (1) structured rule systems that enable quantitative evaluation, (2) complexity that mirrors real-world challenges, and (3) multimodal interfaces that require human-like perception and reasoning. While these settings span a broad spectrum of observations and action interfaces, they often conflate perception, control, and assistance. Table 1 contrasts representative evaluations and highlights how StarBench different from existing game benchmarks.

2.3 VLMs as Game Agents

Traditional game agents often assume privileged interfaces—direct APIs to game state [16, 42] or unimodal inputs [46]-that bypass many challenges human players face. Humans, by contrast, must ground actions in raw pixels, resolve UI affordances, and combine on-screen perception with textual knowledge, all under partial observability. Recent VLM-based agents push toward this human-like setting by reasoning over screenshots and language, but common evaluations still blur key difficulties. Some operate in simplified simulators or expose high-level action APIs that hide low-level control; others lean on large, static knowledge stores or extensive demonstrations [3, 28]. This makes it difficult to disentangle (i) perception-to-control fidelity from (ii) the benefits of tool scaffolding or textual hints. Our benchmark makes this concrete by standardizing two regimes under identical tasks and metrics: direct control, where the agent sees only screenshots and must emit low-level keyboard-mouse primitives (no semantic aids), and toolassisted control, where the agent may get help from tool for easier



Figure 1: Annotated HSR combat UI. Key affordances include: (1) the action order track governing future turns; (2) Ultimate indicators (glow = ready); (3) team skill points; (4) Basic Attack and (5) Skill buttons with availability reflecting (3); (6) damage and status text; (7) the auto-battle toggle. Top-center, the boss frame shows a white toughness bar above a red HP bar. In the example, the team has 0 skill points, disabling (5) while (4) remains usable; the glow at (2) indicates Seele's Ultimate can be cast, potentially off-turn.

perception and manipulation while playing the game (like previous benchmarks). We also include an *ask-or-act* diagnostic that measures whether and when an agent seeks concise guidance before proceeding. This design isolates core capabilities for human-like play without relying on privileged game APIs.

3 BACKGROUND

3.1 Problem Formalization

We model *Honkai: Star Rail* (HSR) combat as a partially observable Markov decision process (POMDP) $\mathcal{M} = (S, \mathcal{A}, \mathcal{T}, \mathcal{R}, O)$. The latent state S comprises complete combat information—ally and enemy health, energy, toughness, speed, buffs/debuffs, and internal cooldowns—while the observation space O exposes only a subset of this information through the on-screen user interface (UI), making perception inherently partial and noisy (see Figure 1 for an annotated overview of the combat screen.). The action space \mathcal{A} contains all feasible player commands, and the transition kernel $\mathcal{T}(s_{t+1} \mid s_t, a_t)$ captures both deterministic rules (e.g., turn scheduling) and stochastic effects (e.g., critical hits, status proc chances). The reward function $\mathcal{R}(s_t, a_t)$ is task-dependent (e.g., boss defeat

under step limits, score accumulation under action-value budgets) and is used to evaluate outcomes rather than to define additional supervision.

3.2 HSR Combat Mechanics

HSR features squad-based, turn-driven combat in which a player controls up to four characters against one or more enemies. Turn order is governed by speed-related statistics and evolves over time due to buffs, debuffs, and "Weakness Break" effects. On a character's turn, the player typically chooses among a Basic Attack, a Skill, or (when available) an Ultimate. Skills consume a shared team resource—skill points—while Basic Attacks replenish it, creating a tight economy of actions that trades short-term power for future flexibility. Ultimates are gated by a separate energy meter and may be cast off-turn as an interrupt when fully charged, adding an additional layer of timing and coordination.

Enemy resilience is mediated by a *toughness* bar and *elemental* weaknesses. When attacks of an appropriate element reduce an enemy's toughness to zero, the enemy enters Weakness Break, experiencing delayed turns and reduced resistances. Managing break

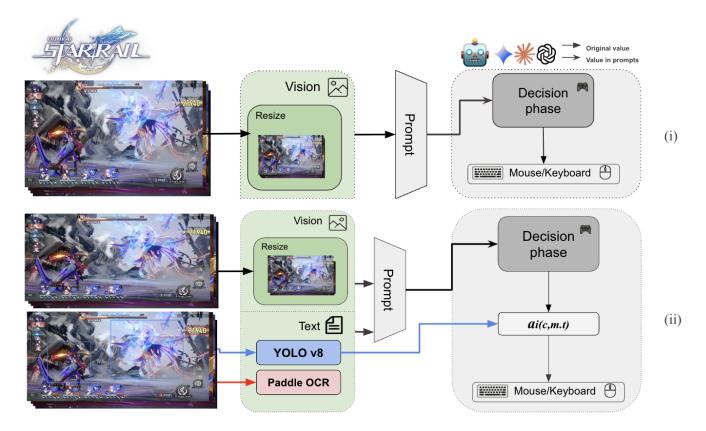


Figure 2: Interaction Protocol: (i) DC (direct control) and (ii) TA (tool-assisted). Inputs are resized to fit each VLM's native resolution, and outputs are resized back to the original scale for evaluation.

windows—while maintaining team survivability—requires fusing visual cues (e.g., turn order track, HP/energy bars, toughness and weakness icons, ability readiness) with textual information (e.g., skill and status descriptions). Because only a subset of internal variables is directly visible, agents must infer latent quantities from these on-screen signals.

Throughout this work we consider a canonical Quantum-aligned team to keep evaluation controlled and reproducible. In our configuration, a primary damage dealer (e.g., Seele) can chain eliminations via an extra-turn mechanic; a support (e.g., Sparkle) advances ally turns and modulates the team's damage and skill-point economy; a debuffer (e.g., Silver Wolf) implants weaknesses and reduces defenses; and a defender (e.g., Fu Xuan) mitigates incoming damage and provides team-wide sustain. This composition illustrates the interplay among resource management (skill points and energy), turn manipulation, and weakness exploitation that underlies strategic decision making in HSR.

4 STARBENCH

4.1 Interaction Protocol

StarBench evaluates real-client play under two matched regimes that share the same game client, tasks, and metrics, and differ only in how actions are expressed and how much UI grounding assistance is provided.(see Figure. 2)

4.1.1 Direct Control (DC). The agent receives the raw screenshot at a fixed resolution and UI scale (1920 \times 1080) and must act purely by spatial pointing. At each decision step it must (i) return an *explicit click location* as *pixel coordinates* (x,y) with $x \in [0,1919], y \in [0,1079]$ for select target and (ii) a keypress for confirming the action. The environment then issues OS primitives via pyautogui library [36]. Out-of-bounds values are clipped to the valid range; empty or non-numeric outputs are treated as no-ops. No YOLO/OCR summaries or other semantic aids are provided—performance reflects pixel-level UI parsing and localization.

Component	Description
Character (c)	0: Seele; 1: Sparkle; 2: Silver Wolf; 3: Fu Xuan
Move (m)	0: Basic Attack; 1: Skill; 2: Release Ultimate; 3: Hold Ultimate
Target (t)	0-3: Teammates; 4-8: Enemies; 9: Select all

Table 2: Action space in StarBench under TA Mode. Teammates organized based on characters position in team (e.g., Seele is the left most character so it is at position 0). Hold Ultimate represents not release the Ultimate in off-turn.

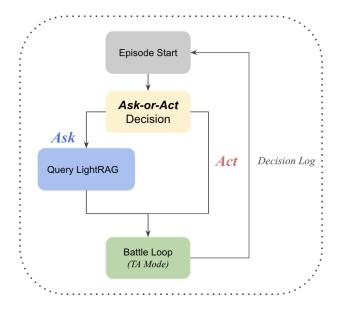


Figure 3: Ask-or-Act diagnostic pipeline. At episode start (TA regime), the agent chooses Ask (query LightRAG once for a short hint, persisted across the episode) or Act (proceed without guidance). Both paths run the same battle loop with the same action interface $a_t = (c, m, t)$; they differ only in prompt context. A decision log records concise summaries (actions sequence for previous battle and battle result) to support learning-to-ask.

4.1.2 Tool-Assisted (TA). The same screenshot is provided, but the agent may express actions as a high-level triple $a_t = (c, m, t)$, where c indexes the controlled character, $m \in \{Basic, Skill, Ultimate, Hold\}$ specifies the move type, and t designates the target (e.g., an enemy identifier)[see Table 2] A static UI map supplies fixed anchors for character icons and skill buttons. For targets that refer to on-screen entities, we attach a lightweight perception stack that runs on the current frame: a finetuned YOLOv8 detector (per task; ~400 labeled battle images) returns, for each entity, a semantic label and a tight bounding box (x, y, w, h) [39], and PaddleOCR (zero-shot) extracts short texts such as HP%, buff/debuff names, and remaining skill points [20]. The detector outputs serve two roles simultaneously: (i) they "textify" the UI into compact structured tokens (enemy information, total damage dealt this turn, current HP of the four playable characters) that can be appended to the policy prompt to ease grounding; and (ii) they parameterize manipulation by providing the actionable region for t. Action execution then maps (c, m, t)to concrete OS events: the environment clicks the center of the detector's bounding box for the referenced target. We will release the detection checkpoints and label schema for the structured TA observations to support reproducibility.

4.2 Ask-or-Act Diagnostic

Humans seek help *selectively*: they ask when expected benefit exceeds the (time/attention) cost and skip otherwise. Our diagnostic evaluates (i) *efficiency*—how much performance is gained per unit

cost of asking; and (ii) effect—how much obtained guidance changes downstream performance (see §4.4).

StarBench models the choice to seek guidance as a single preepisode decision available *only* in the tool-assisted (TA) regime, isolating decision making from low-level control. At the beginning of each battle, the agent chooses between ACT (proceed immediately) or ASK (obtain one short textual hint, then proceed). We implement ASK via LightRAG [11], instantiated with GPT-40-mini [26], over a fixed, read-only corpus of public HSR resources (wikis, guides, FAQs)[see Figure4 for corpus composition]. We will also release the LightRAG checkpoint for better reproduction of baseline results.

If Ask is selected, the agent issues a *single*, targeted question before the episode starts. LightRAG returns one bounded textual answer, which is then *persisted* and appended to the model prompt at *every* subsequent decision step during the battle. No additional queries are permitted within the same episode. Execution is otherwise identical to TA (same screenshot, textualized observations, and action interface $a_t = (c, m, t)$). To support learning-to-ask, we also provide a decision log for each episode, enabling policies that calibrate the Ask threshold from past performance.

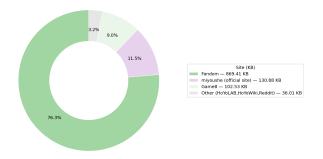


Figure 4: Corpus composition by site. Forums: HoYoLAB[13], Reddit[30]; Wiki: Fandom[7], Game8[10], HoYoWiki[14]; Official site: Miyoushe[24]. Total: 1.14 MB (1138.39 KB).

4.3 Task Suite

StarBench comprises eight battles that cover the four principal HSR combat families (Table 3). We adopt the game's native objectives and report family-specific metrics as formalized in §4.4; here we summarize what each family tests, avoiding repetition of those definitions. Upper Stage denotes activities with two stages (Upper/Lower) that require distinct teams; for reproducibility we standardize on the Upper Stage. No Assistance follows the in-game rule (e.g., End of the Eternal Freeze disables the Engine of Creation), so outcomes reflect the agent's own control.

4.3.1 Echo of War. Single-boss encounters that emphasize efficient finish. Agents must parse the time skills/ultimates, and manage survivability while minimizing decision steps.

¹The corpus is public-only and frozen for all runs. No coordinates, macros, or action strings are returned; responses are brief textual guidance (mechanics, priorities) only.

- 4.3.2 Memory of Chaos. Two-wave battles governed by limited time. The core challenge is *tempo management*: planning damage and healing so both waves complete within a limited number of cycles.
- 4.3.3 Pure Fiction. A score-chasing mode under a global time budget. Unlike the above, the focus is *throughput*: rapidly clearing adds and chaining AOEs to maximize the in-game score within fixed time.
- 4.3.4 Apocalyptic Shadow. Boss fights scored by a composite of *HP* depletion and remaining time. This family stresses burst windows, mitigation, and end-of-fight economy.

Task ID	Task Name	Family	Notes
1	cocolia	Echo of War	No Assistance
2	phantylia	Echo of War	No Assistance
3	swarm king	Echo of War	-
4	theater	Echo of War	No Assistance
5	feixiao	Echo of War	-
*6	xianzhou	Forgotten Hall	Upper Stage
*7	cliched	Pure Fiction	Upper Stage
*8	stardevourer	Apocalyptic Shadow	Upper Stage

Table 3: Task specifications for *StarBench*. Starred tasks include multi-wave or scoring-oriented constraints

4.4 Evaluation Metrics

In HSR, while many combat challenges only require characters to remain undefeated at the end, some combat challenges with high difficulties have additional requirements, including total "time" − in terms of the action values (AVs) − consumption and scores earned by defeating enemies. We adopt the game's native scoring/timing for each activity and standardize only logging and aggregation. Preference is indicated with monotone arrows: ↓ (lower preferred), ↑ (higher preferred).

(1) Echo of War. We report two metrics.

(i) Completion steps (↓):

$$t_{\text{steps}} \in \mathbb{N} \cup \{+\infty\},\$$

the number of agent decisions until victory. If the boss is not defeated, set $t_{\rm steps} = +\infty$.

(ii) Episode reward (↑):

$$R_{\text{EoW}} = \sum_{t=1}^{T} r_t, \qquad r_t = \frac{1}{2} r_t^{\text{HP}} + \frac{1}{2} r_t^{\text{DMG}},$$

the total assigned reward by summing up the reward per action. For every action at t, the reward is assigned upon the HP condition of all characters $(r_t^{\rm HP})$ and the damage dealt by the active character $(r_t^{\rm DMG})$. For $r_t^{\rm HP}$, rewards are assigned on effective and timely heal, particularly when characters are at risk of not surviving the next round of enemy attacks. We combine

$$\begin{split} \hat{\epsilon}_t^{\mathrm{HP}} \coloneqq \hat{\pmb{h}}_t + \Delta \hat{\pmb{h}}_{t-1}, \quad r_t^{\mathrm{safety}} \coloneqq \big(\operatorname{sign}(\hat{\epsilon}_t^{\mathrm{HP}}) \ge \pmb{0} \big), \\ r_t^{\mathrm{healing}} \coloneqq \neg \big(\operatorname{sign}(\Delta \hat{\pmb{h}}_t) \ge \pmb{0} \big). \end{split}$$

into a boolean variable

$$r_t^h \coloneqq r_t^{\text{safety}} + r_t^{\text{healing}} - (r_t^{\text{safety}} \cdot r_t^{\text{healing}}).$$

Using r_t^h , r_t^{HP} is computed as

$$\begin{array}{lll} \hat{\delta}_t^{\mathrm{HP}} & \coloneqq \hat{\pmb{h}}_t - \frac{1}{2} \; \pmb{1}, & \hat{\sigma}_t^{\mathrm{HP}} & \coloneqq -\operatorname{sign}(\hat{\epsilon}_t^{\mathrm{HP}}) \odot \Delta \hat{\pmb{h}}_t, \\ \hat{\Sigma}_t^{\mathrm{HP}} & \coloneqq (\neg r_t^h) \cdot \hat{\delta}_t^{\mathrm{HP}} + (r_t^h) \cdot \hat{\sigma}_t^{\mathrm{HP}}, & r_t^{\mathrm{HP}} & = -\operatorname{sign}(\hat{\epsilon}_t^{\mathrm{HP}}) \cdot \hat{\Sigma}_t^{\mathrm{HP}}. \\ \end{array}$$

The damage term is

$$r_t^{\text{DMG}} = \hat{d}_t$$
.

Notation: \hat{h}_t : normalized HP vector at step t, with all characters concatenated; $\Delta \hat{h}_t$: change in h from the previous timestep; \hat{d}_t : normalized damage at t; 0 and 1 are zero and one vectors, respectively.

(2) Memory of Chaos. Cycles advance by the game's AV schedule (first cycle: 150 AV; subsequent: 100 AV). With total AV consumed AV_{used},

$$C_{\text{used}} = 1\{AV_{\text{used}} \ge 150\} + \max\left(0, \left\lfloor \frac{AV_{\text{used}} - 150}{100} \right\rfloor\right),$$
 (1)

and with cycle budget C_{max} , we report remaining cycles

$$S_{\text{MoC}} = \max(0, C_{\text{max}} - C_{\text{used}}) \qquad (\uparrow). \tag{2}$$

(3) **Pure Fiction**. With global AV budget $AV_{max} = 450$ and in-game incremental score s_t from enemy eliminations at step t,

$$S_{\text{PF}} = \sum_{t=1}^{T} s_t \cdot \mathbf{1} \{ AV_t \le AV_{\text{max}} \} \qquad (\uparrow). \tag{3}$$

(4) **Apocalyptic Shadow**. Let $\mathcal{S}_{AS}(HP\%_{\downarrow}, AV_{rem})$ be the game's native composite score (boss HP depletion + remaining AV bonus). We report

$$S_{AS} = S_{AS}(HP\%_{\parallel}, AV_{rem}) \qquad (\uparrow).$$
 (4)

(5) **Ask-or-Act Metrics**. Let S be the task score; for Echo of War use S = -Steps. Episodes are ordered within each task by time: $k = 1, \ldots, n_t$. Let $A_{t,k} \in \{0,1\}$ indicate Act (0) vs. Ask (1). Define the ask rate AR $= \frac{1}{N} \sum_{t,k} A_{t,k}$, where N is the number of evaluated episodes, and let $M = \sum_{t,k} 1\{A_{t,k} = 1, k > 1\}$ be the count of asked episodes that have a predecessor. (i) Effect (temporal uplift per ask; \uparrow). For every asked episode with a predecessor on the same task ($A_{t,k} = 1$ and k > 1), set

$$\Delta_{t,k} = S_{t,k} - S_{t,k-1}.$$

Report the mean per-ask effect:

$$\widehat{\text{Effect}} = \frac{1}{M} \sum_{t} \sum_{k: A_{t,k}=1, k>1} \Delta_{t,k}.$$

(ii) Efficiency (normalized by number of asks; \uparrow). Each evaluation allows at most T asks. The expected number of asks is $(T \cdot AR/100)$. We therefore define

Efficiency =
$$\frac{\widehat{\text{Effect}}}{(T \cdot AR/100)}$$

Table 4: Echo of War per-task results. We report SR% (\uparrow), Steps (\downarrow), and R_{EoW} (\uparrow); rewards use a unified 0–100 scale. VLMs are grouped by regime; baselines appear separately. Coloring: best VLM per boss in *yellow*, best Baseline per boss in *blue*; the winner's SR/Steps/Reward cells are all highlighted. Tie-break: higher SR \rightarrow lower Steps \rightarrow higher Reward. ∞ stands for infinite steps for failure case and would not take in count during comparison.

		Cocolia			Phantyl	ia		Swarm Ki	ng		Theater			Feixiao	
Model	SR%	Steps	$R_{\rm EoW}$	SR%	Steps	$R_{ m EoW}$	SR%	Steps	$R_{ m EoW}$	SR%	Steps	$R_{ m EoW}$	SR%	Steps	$R_{ m EoW}$
VLMs - Direct Control (DC: screens	VLMs − Direct Control (DC: screenshot→OS primitives, no text aids)														
GPT-4o-mini (DC)	0.0	∞	0.14 ± 0.0	0.0	∞	0.06 ± 0.0	0.0	00	0.12 ± 0.0	0.0	∞	0.10 ± 0.0	0.0	00	0.11 ± 0.0
Claude 3.5 Sonnet (DC)	0.0	00	0.08 ± 0.0	0.0	∞	0.07 ± 0.0	0.0	000	0.11 ± 0.0	0.0	00	0.09 ± 0.0	0.0	00	0.10 ± 0.0
Gemini 1.5 Flash (DC)	0.0	00	0.0 ± 0.0	0.0	00	$0.0\!\pm\!0.0$	0.0	00	0.0 ± 0.0	0.0	00	0.0 ± 0.0	0.0	00	0.0 ± 0.0
$VLMs - Tool$ -Assisted (TA: $a_t = (c, t)$	m, t) w	ith YOLO/ opti	onal OCR te	ct helps)										
GPT-4o-mini (TA-NO-OCR)	62.5	44.3 ± 6.8	63.2 ± 4.1	37.5	92.0 ± 12.3	50.8 ± 3.7	12.5	128.0 ± 18.5	46.3 ± 4.9	25.0	132.7 ± 15.4	45.1 ± 3.2	25.0	141.5 ± 20.6	44.2 ± 3.1
Claude 3.5 Sonnet (TA-NO-OCR)	62.5	43.8 ± 7.2	62.7 ± 4.9	25.0	98.4 ± 13.6	49.6 ± 3.5	12.5	134.2 ± 17.1	45.5 ± 3.8	37.5	124.9 ± 14.0	46.2 ± 2.9	25.0	138.7 ± 18.3	45.0 ± 3.0
Gemini 1.5 Flash (TA-NO-OCR)	12.5	36.7 ± 0.0	67.1 ± 0.0	0	∞	0.13 ± 0.02	0.0	000	0.0 ± 0.4	0.0	∞	0.0 ± 0.2	0	00	0.1 ± 0.3
GPT-4o-mini (TA)	100.0	31.40 ± 2.60	71.1 ± 1.8	100.0	50.8 ± 6.54	59.7 ± 2.6	50.0	70.8 ± 3.20	54.1 ± 5.1	100.0	84.6 ± 11.02	52.0 ± 1.3	100.0	83.1±7.26	52.1 ± 1.0
Claude 3.5 Sonnet (TA)	100.0	31.50 ± 2.00	71.5 ± 5.6	87.5	47.0 ± 6.0	56.5 ± 3.0	37.5	69.3 ± 4.70	52.4±3.9	87.5	74.0 ± 8.0	51.0 ± 3.0	62.5	100.0 ± 10.0	54.0 ± 3.0
Gemini 1.5 Flash (TA)	12.5	37.0 ± 0.0	67.1 ± 0.0	12.5	54.0 ± 9.0	58.0 ± 2.0	0.0	00	-0.2 ± 0.4	0.0	00	3.0 ± 1.0	12.5	132.0 ± 10.0	47.0 ± 2.0
GPT-4o-mini (ReAct-TA)	75.0	30.20 ± 1.20	69.8 ± 5.6	50.0	51.8 ± 5.60	56.8 ± 4.5	25.0	70.5 ± 3.50	50.6 ± 2.9	62.5	76.3 ± 3.90	53.1 ± 0.7	75.0	76.40 ± 7.50	53.3 ± 1.2
GPT-4o-mini (Reflexion-TA)	62.5	30.4 ± 3.4	68.4 ± 7.2	50.0	52.4 ± 7.8	54.3 ± 3.5	62.5	72 ± 10.2	49.2 ± 2.8	62.5	75.3 ± 4.25	52.8 ± 1.2	62.5	84.4±11.5	52.4 ± 2.4
GPT-4o-mini (TA-Ask)	87.5	29.4 ± 4.2	69.2±5.7	62.5	55.0 ± 8.3	58.2 ± 2.3	50.0	69.3±9.8	54.5±2.1	100.0	74.1 ± 9.0	51.4 ± 4.8	100.0	119.4 ± 10.3	47.1 ± 4.7
Claude 3.5 Sonnet (TA-Ask)	100.0	30.6 ± 4.8	69.1±8.0	100.0	44.1 ± 3.8	58.5 ± 6.0	50.0	65.7 ± 10.3	55.6 ± 2.0	100.0	70.8 ± 9.7	52.9 ± 3.7	75.0	96.5 ± 14.0	55.9 ± 2.0
Gemini 1.5 Flash (TA-Ask)	12.5	39.0±0.0	65.5±0.0	25.0	51.5±9.2	59.3±3.4	0.0	00	0.1 ± 0.0	0.0	00	4.1 ± 0.0	12.5	127.0 ± 0.0	48.2 ± 0.0
Baselines (no DC/TA separation; Human participates, reward missing)															
Random (TA)	25.0	45.3 ± 11.3	60.5 ± 9.7	0.0	∞	-0.4 ± 0.3	0.0	000	0.0 ± 0.1	0.0	00	-0.4 ± 0.4	0.0	00	-0.2 ± 0.2
Human (reference)	100.0	30.3 ± 6.6	-	90.0	43.8 ± 11.6	-	90.0	52.4 ± 12.4	-	90.0	77.0 ± 14.4	-	100.0	75.9 ± 9.2	-

Table 5: MoC / PF / AS family metrics. Models shown with DC, TA, and TA-Ask subrows; baselines listed separately. - ∞ stands for model failed to accomplished the MoC task

Model	Regime	MoC C_{rem} ↑	PF Score ↑	AS Score ↑
	DC	-∞	0	0
GPT-40-mini	TA	16 ± 0.5	7246 ± 128	2804 ± 137
	TA-Ask	18 ± 0.5	10280 ± 1348	2810 ± 150
	DC	-∞	0	0
Claude 3.5 Sonnet	TA	16 ± 0.5	7235 ± 837	2833 ± 168
	TA-Ask	16 ± 0.5	8000 ± 246	2825 ± 125
	DC	-∞	0	0
Gemini 1.5 Flash	TA	-00	1780 ± 374	476 ± 236
	TA-Ask	-∞	2000 ± 120	380 ± 202
Human (reference)	-	17 ± 1.5	7560 ± 1749	3240 ± 170
Auto-battle	-	18 ± 0.0	7120 ± 0.0	3202 ± 0.0

5 EXPERIMENT

5.1 Baselines

We build several baselines in StarBench: a random policy, human expert performance, AutoBattle, a multimodal RL policy with VLMs baselines (GPT4O-mini,Claude-3.5-Sonnet, Gemini-1.5-Flash,). with two prompt engineering method: ReAct and Reflexion

- 5.1.1 Random Policy. This policy selects a valid action uniformly at random from the set of permissible actions indicated by the action mask in the observation with tool-assisted mode. It does not exploit any visual or textual cues.
- 5.1.2 Human Expert. We recruited 10 participants with different levels of experience in HSR gameplay, ranging from novices with no previous exposure to veterans with over 200 hours of gameplay, to engage in controlled battle sessions. Their performance was recorded and averaged to establish a human performance benchmark.

Table 6: Ask-or-Act diagnostic (TA only). Effect is the mean per-ask uplift vs. the immediately previous episode on the same task; Efficiency uses the normalization $Effect/[(AR/100)\times 8].$ AR shown as percent. Best effect in yellow, while best efficiency in blue.

Model	Task	AR (%)	Effect	Efficiency
GPT-40-mini	Echo of War	52.5	82.8	19.7
Claude 3.5 Sonnet	Echo of War	22.5	49.4	27.4
Gemini 1.5 Flash	Echo of War	97.5	47.4	6.1
GPT-40-mini	Memory of Chaos	62.5	17	3.4
Claude 3.5 Sonnet	Memory of Chaos	100	16	2.0
Gemini 1.5 Flash	Memory of Chaos	100	0	0.0
GPT-40-mini	Pure Fiction	100	10800	1350.0
Claude 3.5 Sonnet	Pure Fiction	100	6400	800.0
Gemini 1.5 Flash	Pure Fiction	100	2000	250.0
GPT-40-mini	Apocalyptic Shadow	100	3002	375.3
Claude 3.5 Sonnet	Apocalyptic Shadow	100	2817	352.1
Gemini 1.5 Flash	Apocalyptic Shadow	100	380	47.5

 \overline{T} = 8 ask opportunities per evaluation. Efficiency is set to 0 when AR = 0.

- 5.1.3 AutoBattle. AutoBattle is a function in Honkai: Star Rail that automatically controls the player's team during combat. The usage and targeting of abilities are determined by many factors, such as skill points, enemy weakness, enemy HP, status effects, etc. Auto-Battle is more like a state machine that follows simple logic. For example, it will release the Ultimate right when it is ready, except such Ultimates with healing behaviors [12].
- 5.1.4 Prompting Variants: ReAct and Reflexion. In addition to evaluating the VLMs directly, we also assess two lightweight decoding-time prompting strategies: ReAct (Reasoning+Acting) [47] and Reflexion (Self-critique) [34].

5.2 Experiment setup

For each evaluation, we run each agent **eight** trials per task using the same standardized team configuration. All experiments execute on a single RTX 4070 Ti GPU. The HSR client runs windowed at 1920×1080 with UI scale 100%, VSync on, and a fixed graphics preset. Input events are issued via OS-level primitives with a fixed 0.5 s inter-event delay.

Each task emphasizes turn-by-turn decision complexity, including optimal skill-point usage and effective exploitation of enemy weakness breaks. An episode terminates on victory, failure, or invalid-output conditions. For \mathbf{DC} , if the agent fails to produce a valid OS primitives on a required UI widget for 10 consecutive attempts (e.g., off-target region, or empty emission), we stop the episode and record a failure. For \mathbf{TA} , if the agent emits an invalid tuple (c, m, t) that cannot be mapped to a legal action or violates preconditions (e.g., attempting SKILL with zero skill points) for K=10 consecutive decisions, we also stop and record a failure. Timeout decisions are treated as no-ops and count toward step/AV budgets.

5.3 Results

StarBench evaluates two human-like competencies introduced in the abstract and introduction: (i) pixel-to-action grounding—mapping raw screenshots to precise OS primitives; and (ii) agentic information seeking—deciding when brief guidance is worth the cost. By holding tasks and metrics fixed across regimes, Tables 4, 5, and 6 isolate where capability breaks.

(1) DC exposes a fundamental grounding gap. In Table 4 (DC blocks), GPT–40-mini, Claude 3.5 Sonnet, and Gemini 1.5 Flash all record SR = 0% and near-zero $R_{\rm EoW}$ on every Echo of War boss; Table 5 shows the same collapse for MoC ($-\infty$ cycles), PF (0), and AS (0). With tasks/metrics unchanged, these failures specifically indicate an end-to-end screenshot—primitive localization deficit rather than task difficulty. For DC, most of cases failed because those VLMs selected incorrect space for targets (e.g., selecting the non-UI elements in the battle scene). As results, most of actions are selected by VLMs under DC mode are choosing the default target and finally lead to bad performance (e.g., keep using the basic attacks to attack left most enemies and never use skill).

(2) Minimal UI grounding unlocks competent play under the same tasks. Switching only the interface to TA yields large gains. In Table 4 (TA blocks), GPT-40-mini reaches 100% SR on Cocolia, Phantylia, Theater, and Feixiao with finite steps on Cocolia, on Phantylia), and Claude shows a similar pattern. These improvements carry to other families in Table 5. The key reason for this performance jump is the abstraction provided by the TA regime: instead of issuing low-level OS primitives (e.g., pixel-level clicks), VLMs in TA mode operate over structured action tuples (character, move type, target). The model can focus on higher-level decisionmaking-such as when to cast a Skill or Ultimate-without worrying about where to click or whether the UI has changed slightly. For example, under TA, models reliably trigger the correct abilities because the action space is cleanly defined. In contrast, in the DC regime, even if a model correctly reasons that an Ultimate should be used, it may fail to locate and click the correct UI element-resulting in a no-op or illegal action. Thus, TA removes a major bottleneck: the challenge of grounding visual perception to actionable control,

allowing evaluation to better isolate strategic competence from UI manipulation skill.

(3) Why OCR matters beyond boxes. Removing OCR while keeping YOLO bounding boxes consistently hurts both success and efficiency (Table 4, TA–NO–OCR). For GPT–4o-mini, SR drops from 100% to 62.5% on *Cocolia*, to 37.5% on *Phantylia*, to 25.0% on *Theater* and *Feixiao*; Steps inflate (e.g., *Cocolia* 31.4 \rightarrow 44.3, *Phantylia* 50.8 \rightarrow 92.0) and $R_{\rm EoW}$ falls accordingly. Claude shows parallel degradations (e.g., *Cocolia* 100% \rightarrow 62.5%, *Phantylia* 87.5% \rightarrow 25.0%). These gaps indicate that textualized UI (HP%, skill points, readiness/status strings) reduces decision friction—especially for legality and timing—beyond what spatial boxes provide. Practically, this also surfaces a limitation of native VLM text recognition on dense, stylized UIs: without OCR, models misread or miss small texts(e.g., characters' HP), leading to illegal or mistimed actions and higher Steps (e.g., forgot to use Healing Ultimate when characters' HP are low).

(4) Asking is beneficial when calibrated. Table 6 shows that brief guidance produces measurable uplifts when used judiciously. GPT-40-mini attains the largest *Effect* across Echo of War (82.8), PF (10800), and AS (3002), while Claude achieves the best *Efficiency* on Echo of War (27.4) despite a lower ask rate. Gemini asks frequently (AR 97.5%) but gains little (Effect 47.4, Efficiency 6.1). We observe that all VLMs tend to ask questions during the first trial of a task (out of eight total), likely due to uncertainty about unfamiliar mechanics or objectives. For tasks with explicit goals—such as Echo of War (defeating a boss)—models typically delay asking until after a failure, suggesting a reactive asking policy. However, for implicit-goal tasks, such as maximizing scores in Pure Fiction or managing time in Memory of Chaos, models ask more proactively and frequently, likely in an attempt to iteratively refine strategy across episodes.

6 CONCLUSION

We introduced **StarBench**, a real-client benchmark for testing whether vision—language models can (i) ground raw pixels into coherent keyboard—mouse actions and (ii) decide when to seek guidance via an *ask-or-act* diagnostic. By fixing tasks and metrics and varying only the interaction regime—*direct* screenshot-to-action versus *tool-assisted* control—StarBench cleanly separates perception-to-control grounding from higher-level decision making. Current VLMs fail almost entirely in direct control, revealing a core deficiency in pixel-to-primitive localization and UI manipulation. Tool assistance, especially with textualized UI, markedly improves success beyond that. The ask-or-act results further demonstrate that brief, targeted guidance can yield measurable uplifts. These findings support that competent real-client play today hinges on lightweight tools and calibrated information seeking, while end-to-end pixel-to-primitive control remains the key challenge.

7 LIMITATIONS AND FUTURE WORK

StarBench runs on a *real* live-service client, which introduces practical constraints (frame-timing variance, and OS-level input flakiness) that may affect exact reproducibility. To mitigate these issues, we are developing a companion *emulator* that replays recorded traces. This will allow frozen UI states, deterministic screenshots, and offline evaluation while keeping the same tasks and metrics.

REFERENCES

- Anthropic. 2024. Introducing Claude 3.5 Sonnet. https://www.anthropic.com/ news/claude-3-5-sonnet
- [2] Stanislaw Antol, Aishwarya Agrawal, Jing Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. VQA: Visual Question Answering. In Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2425–2433
- [3] Peng Chen, Pi Bu, Jun Song, Yuan Gao, and Bo Zheng. 2024. Can VLMs Play Action Role-Playing Games? Take Black Myth Wukong as a Study Case. arXiv preprint arXiv:2409.12889 (2024). https://doi.org/10.48550/arXiv.2409.12889
- [4] Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Ruo Yu Tao, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. 2018. TextWorld: A Learning Environment for Text-based Games. arXiv preprint arXiv:1806.11532 (2018).
- [5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2020. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. arXiv preprint arXiv:2010.11929 (2020).
- [6] L. Fan, G. Wang, Y. Jiang, A. Mandlekar, Y. Yang, H. Zhu, A. Tang, D.-A. Huang, Y. Zhu, and A. Anandkumar. 2022. MineDojo: Building Open-Ended Embodied Agents with Internet-Scale Knowledge. arXiv preprint arXiv:2206.08853 (2022).
- [7] Fandom Inc. 2025. Fandom Wiki. https://www.fandom.com Accessed: 2025-02-07
- [8] Xidong Feng, Yicheng Luo, Ziyan Wang, Hongrui Tang, Mengyue Yang, Kun Shao, David Mguni, Yali Du, and Jun Wang. 2023. ChessGPT: Bridging Policy Learning and Language Modeling. arXiv preprint arXiv:2306.09200 (2023).
- [9] Luke Friedman, Sameer Ahuja, David Allen, Zhenning Tan, Hakim Sidahmed, Changbo Long, Jun Xie, Gabriel Schubiner, Ajay Patel, Harsh Lara, et al. 2023. Leveraging Large Language Models in Conversational Recommender Systems. arXiv preprint arXiv:2305.07961 (2023).
- [10] Game8 Co., Ltd. 2025. Game8 Gaming Wiki. https://game8.co Accessed: 2025-02-11.
- [11] Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2024. LightRAG: Simple and Fast Retrieval-Augmented Generation. arXiv preprint arXiv:2410.05779 (2024).
- [12] Honkai Star Rail Wiki. [n.d.]. Auto Battle. https://honkai-star-rail.fandom.com/ wiki/Auto-Battle Accessed: 2025-02-08.
- [13] HoYoverse. 2025. HoYoLAB: Official miHoYo Community. https://www.hoyolab. com Accessed: 2025-02-11.
- [14] HoYoverse. 2025. HoYoWiki: Official Game Encyclopedia. https://wiki.hoyolab. com Accessed: 2025-02-11.
- [15] Sihao Hu, Tiansheng Huang, Fatih Ilhan, Selim Tekin, Gaowen Liu, Ramana Kompella, and Ling Liu. 2024. A Survey on Large Language Model-Based Game Agents. arXiv preprint arXiv:2404.02039 (2024).
- [16] Sihao Hu, Tiansheng Huang, and Ling Liu. 2024. PokéLLMon: A Human-Parity Agent for Pokémon Battles with Large Language Models. arXiv preprint arXiv:2402.01118 (2024).
- [17] Juyong Jiang, Fan Wang, Jiasi Shen, Sungju Kim, and Sunghun Kim. 2024. A Survey on Large Language Models for Code Generation. arXiv preprint arXiv:2406.00515 (2024).
- [18] Matthew Johnson, Katja Hofmann, Tim Hutton, and David Bignell. 2016. The Malmo Platform for Artificial Intelligence Experimentation. In Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI). 4246–4247.
- [19] Heinrich Küttler, Nantas Nardelli, Alexander H. Miller, Roberta Raileanu, Marco Selvatici, Edward Grefenstette, and Tim Rocktäschel. 2020. The NetHack Learning Environment. In Advances in Neural Information Processing Systems (NeurIPS).
- [20] Chenxia Li, Weiwei Liu, Ruoyu Guo, Xiaoting Yin, Kaitao Jiang, Yongkun Du, Yuning Du, Lingfeng Zhu, Baohua Lai, Xiaoguang Hu, Dianhai Yu, and Yanjun Ma. 2022. PP-OCRv3: More Attempts for the Improvement of Ultra Lightweight OCR System. arXiv:2206.03001 [cs.CV] https://arxiv.org/abs/2206.03001
- [21] Fengling Li, Lei Zhu, Tianshi Wang, Jingjing Li, Zheng Zhang, and Heng Tao Shen. 2023. Cross-Modal Retrieval: A Systematic Review of Methods and Future Directions. arXiv preprint arXiv:2308.14263 (2023).
- [22] Hezheng Lin, Xing Cheng, Xiangyu Wu, Fan Yang, Dong Shen, Zhongyuan Wang, Qing Song, and Wei Yuan. 2021. CAT: Cross Attention in Vision Transformer. arXiv preprint arXiv:2106.05786 (2021). https://doi.org/10.48550/arXiv.2106.05786
- [23] Ruo-Ze Liu, Zhen-Jia Pang, Zhou-Yu Meng, Wenhai Wang, Yang Yu, and Tong Lu. 2022. On Efficient Reinforcement Learning for Full-length Game of StarCraft II. arXiv preprint arXiv:2209.11553 (2022).
- [24] miHoYo / HoYoVerse. 2025. Miyoushe miHoYo Official Community. https://www.miyoushe.com/ Accessed: 2025-02-04.
- [25] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing Atari with Deep Reinforcement Learning. arXiv preprint arXiv:1312.5602 (2013).
- [26] OpenAI. 2023. GPT-4 Technical Report. Technical Report. OpenAI. https://cdn.openai.com/papers/gpt-4.pdf

- [27] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. arXiv preprint arXiv:2203.02155 (2022).
- [28] Davide Paglieri, Bartlomiej Cupiał, Samuel Coward, Ulyana Piterbarg, Maciej Wolczyk, Akbir Khan, Eduardo Pignatelli, Łukasz Kuciński, Lerrel Pinto, Rob Fergus, Jakob Nicolaus Foerster, Jack Parker-Holder, and Tim Rocktäschel. 2024. BALROG: Benchmarking Agentic LLM and VLM Reasoning On Games. arXiv preprint arXiv:2411.13543 (2024).
- [29] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models from Natural Language Supervision. arXiv preprint arXiv:2103.00020 (2021).
- [30] Reddit Inc. 2025. Reddit. https://www.reddit.com Accessed: 2025-02-06.
- [31] A. L. Samuel. 1959. Some Studies in Machine Learning Using the Game of Checkers. IBM Journal of Research and Development 3, 3 (May 1959), 210–229.
- [32] Mikayel Samvelyan, Robert Kirk, Vitaly Kurin, Jack Parker-Holder, Minqi Jiang, Eric Hambro, Fabio Petroni, Heinrich Küttler, Edward Grefenstette, and Tim Rocktäschel. 2021. MiniHack the Planet: A Sandbox for Open-Ended Reinforcement Learning Research. In NeurIPS 2021 Datasets and Benchmarks Track.
- [33] Claude E. Shannon. 1950. Programming a Computer for Playing Chess. Philos. Mag. 41, 314 (1950), 256–275.
- [34] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language Agents with Verbal Reinforcement Learning. arXiv preprint arXiv:2303.11366 (2023).
- [35] Penny Sweetser. 2024. Large Language Models and Video Games: A Preliminary Scoping Review. arXiv preprint arXiv:2403.02613 (2024).
- [36] Al Sweigart. 2023. PyAutoGUI. https://pypi.org/project/PyAutoGUI/. Version 0.9.54.
- [37] Weihao Tan, Wentao Zhang, Xinrun Xu, Haochong Xia, Ziluo Ding, Boyu Li, Bohan Zhou, Junpeng Yue, Jiechuan Jiang, Yewen Li, Ruyi An, Molei Qin, Chuqiao Zong, Longtao Zheng, Yujie Wu, Xiaoqiang Chai, Yifei Bi, Tianbao Xie, Pengjie Gu, Xiyun Li, Ceyao Zhang, Long Tian, Chaojie Wang, Xinrun Wang, Börje F. Karlsson, Bo An, Shuicheng Yan, and Zongqing Lu. 2024. Cradle: Empowering Foundation Agents Towards General Computer Control. arXiv preprint arXiv:2403.03186 (2024). https://doi.org/10.48550/arXiv.2403.03186
- Gemini Team, Google, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy Lillicrap, Angeliki Lazaridou, Orhan Firat, James Mollov, Michael Isard, Paul R, Barham, Tom Hennigan, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, Ryan Doherty, Eli Collins, Clemens Meyer, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, Jack Krawczyk, Cosmo Du, Ed Chi, Heng-Tze Cheng, Eric Ni, Purvi Shah, Patrick Kane, Betty Chan, Manaal Faruqui, Aliaksei Severyn, Hanzhao Lin, Yaguang Li, Yong Cheng, Abe Ittycheriah, Mahdis Mahdieh, Mia Chen, Pei Sun, Dustin Tran, Sumit Bagri, Balaji Lakshminarayanan, Jeremiah Liu, Andras Orban, Fabian Güra, Hao Zhou, Xinying Song, Aurelien Boffy, Harish Ganapathy, Steven Zheng, HyunJeong Choe, Ágoston Weisz, Tao Zhu, Yifeng Lu, Siddharth Gopal, Jarrod Kahn, Maciej Kula, Jeff Pitman, Rushin Shah, Emanuel Taropa, Majd Al Merey, et al. 2024. Gemini: A Family of Highly Capable Multimodal Models. arXiv preprint arXiv:2312.11805 (2024). https://doi.org/10.48550/arXiv.2312.11805 Version 4, last revised 17 Jun 2024.
- [39] Juan Terven, Diana-Margarita Córdova-Esparza, and Julio-Alejandro Romero-González. 2023. A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS. Machine Learning and Knowledge Extraction 5, 4 (Nov. 2023), 1680–1716. https://doi.org/10.3390/make5040083
- [40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, A. N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. arXiv preprint arXiv:1706.03762 (2017).
- 41] Oriol Vinyals, Andrei Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and Tell: A Neural Image Caption Generator. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 3156–3164.
- [42] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023. Voyager: An Open-Ended Embodied Agent with Large Language Models. arXiv:2305.16291 [cs.AI] https://arxiv.org/ ph/2305.16201.
- [43] Jiaqi Wang, Zihao Wu, Yiwei Li, Hanqi Jiang, Peng Shu, Enze Shi, Huawen Hu, Chong Ma, Yiheng Liu, Xuhui Wang, et al. 2024. Large Language Models for Robotics: Opportunities, Challenges, and Perspectives. arXiv preprint arXiv:2401.04334 (2024).
- [44] Xinyu Wang, Bohan Zhuang, and Qi Wu. 2025. Are Large Vision Language Models Good Game Players? arXiv preprint arXiv:2503.02358 (2025).
- [45] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. 2015. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In Proceedings of the International Conference on Machine Learning (ICML). 2048–2057.

- [46] Y. Xu, S. Wang, P. Li, et al. 2023. Exploring Large Language Models for Communication Games: An Empirical Study on Werewolf. arXiv preprint arXiv:2309.04658 (2023).
- (2023).
 [47] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. ReAct: Synergizing Reasoning and Acting in Language Models. arXiv:2210.03629 [cs.CL] https://arxiv.org/abs/2210.03629