DiG-bench: Discovery in Games

Ruairidh M. Battleday[1], Kai Sandbrink[2], Jimi Cullen-Drohan[2], Zihan Yan[2], Timothy Muller[3], Clare Maguire[1], Ales Kubicek[2], Fraser Greenlee-Scott[2], Sukrit Sumant[2], Tri Dao[4], Jürgen Schmidhuber[5, 6], Michal Valko[7], Joshua Tenenbaum[8], Thomas L. Griffiths[4], Zeb Kurth-Nelson[2], James C.R. Whittington[1, 3]

[1]*Thinking About Thinking,* [2]*Independent,* [3]*University of Oxford,* [4]*Princeton University,* [5]*King Abdullah University of Science and Technology,* [6]*Swiss AI Lab,* [7]*Inria,* [8]*Massachusetts Institute of Technology*

Abstract

Discovery—formulating novel generalizations—is a central part of the scientific process. Despite its importance, there is a gap in the current AI benchmark landscape, with few benchmarks directly probing the capacity for discovering new knowledge with experimentation in controlled environments where the objective is unknown. To address this gap, we release a new benchmark: DiG-bench (Discovery in Games). DiG-bench consists of a set of 70 independent games. Each game is encoded as a short string and has unique transformation rules that must be discovered through interaction and experimentation. The levels of the game present a series of challenges to test whether the rules have been discovered, where the win conditions for each level are also unknown. We provide games at seven tiers of difficulty for AI agents. The lowest tier is routinely solvable by multiple models, while the highest tier challenges the best models in agentic harnesses. All 70 games were solved by at least one human on first attempt. A subset of 21 games is released publicly, and the remainder is held private for secure evaluation.

Benchmark website: https://digbench.ai

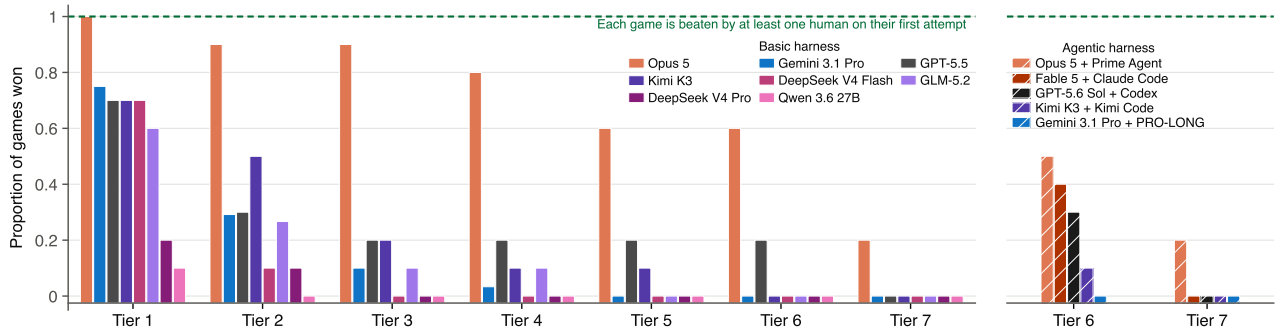

Figure 1: **Model performance on DiG-bench.** Proportion of games won by each model in each tier. Models in the basic harness were evaluated on all seven tiers (left). Models in agentic harnesses were evaluated only on tiers 6 and 7 (right). The horizontal dashed line indicates that humans beat every game.

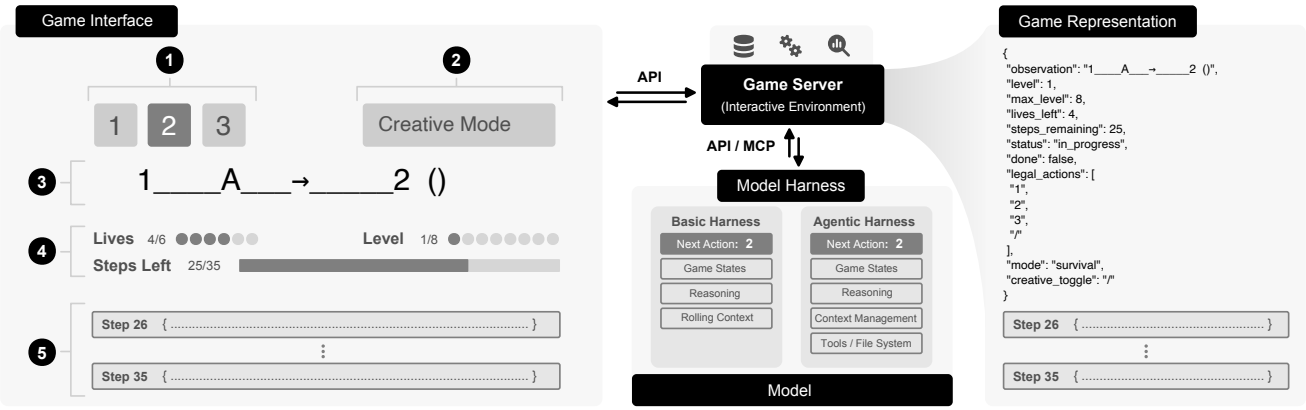

Figure 2: **The DiG-bench platform.** The platform and equivalent human/model interfaces. ❶ the available actions (action '2' is highlighted because the player is selecting it); ❷ creative mode, an option in some games that allows a player to enter a sandbox for experimentation; ❸ the current game observation; ❹ indicators of current level, remaining lives and remaining steps; and ❺ action/state history.

1 Introduction

Discovering new knowledge drives science forward. We define discovery as finding and making sense of previously unexplained regularities in a way that is useful for compressing existing observations [21] and reasoning about new states of a system. There is now enormous interest in building AI systems that can accelerate this process, from AI research to structural biology, chemistry, and mathematics [13, 16].

Measuring progress toward AI that can make genuine discoveries requires a benchmark that isolates the capacity for discovery itself. Some existing benchmarks, like ARC-AGI-3 [1], are framed as tests of discovery and fluid reasoning, but confound this with visual perception. Others, like ARC-AGI-1/2 [6, 8], Bongard-LOGO [19] and IOLBench [14], probe rule induction but not active experimentation. Yet others, such as DiscoveryWorld [15], test experimentation but entangle discovery with prior scientific knowledge or rely on a 'menu' of discoveries. Taken together, this means the current benchmark landscape lacks a targeted, controlled benchmark for active discovery.

We therefore introduce DiG-bench (Discovery in Games), a benchmark of 70 games designed to map the surface of discovery in well-controlled interactive systems. Games have a long history in AI [25]. In DiG-bench, each game is a self-contained miniature world with its own laws, but both the rules and the objective are hidden from the player and must be uncovered through interaction. The games exercise the discovery process end-to-end, with rich possibilities for experimentation.

Six design choices distinguish DiG-bench, and each follows directly from the goal of measuring discovery in isolation.

1. The games operate within the **natural domain of large language models**, creating the best test of the models' true discovery capabilities. The games are purely text-based: observations are short strings, usually on a single line. This is done so that there are no visuospatial confounds [23], leaving discovery as the operative challenge. Games are short enough that most traces fit entirely within the context window of current frontier models, so that we test for discovery "in-context" without any need for weight updates or complicated context management.

2. The games are **handcrafted** by human experts and **novel**, and we keep the majority of them private.

3. We acknowledge that **discovery is effortful**. All games have been solved by at least one human player on that player's first exposure to the game—but players reported finding many games

difficult, and traces show extensive experimentation.

4. Discoveries across games reflect a **rich and diverse set of mechanisms**, meaning that the benchmark is not solved by solving a single challenge, such as vision.

5. **Experimentation** is a central part of the discovery process. For this, many games contain a special creative mode that allows players to test mechanics in a sandbox-like environment with a less restrictive step limit. Within a game, the ability to perform informative experiments often depends on understanding previous experiences.

6. Difficulty is **calibrated to the frontier**. We find tasks that are right at the tipping point of what models can do, allowing us to map the surface of discovery accurately.

The remainder of this report is organized as follows. We first outline the benchmark (Section 2), then report the performance of humans and LLM agents and its breakdown across difficulty tiers (Section 3). We perform a separate analysis in which we give Gemini 3.1 Pro access to the true rules of each game, to evaluate how much easier the games are if the rules are known. We then detail the gameplay setup (Section 4), situate the benchmark against related work (Section 5), and close with a discussion of the skills needed to solve the games and directions for future work (Section 6). Appendices present prompts and further analyses.

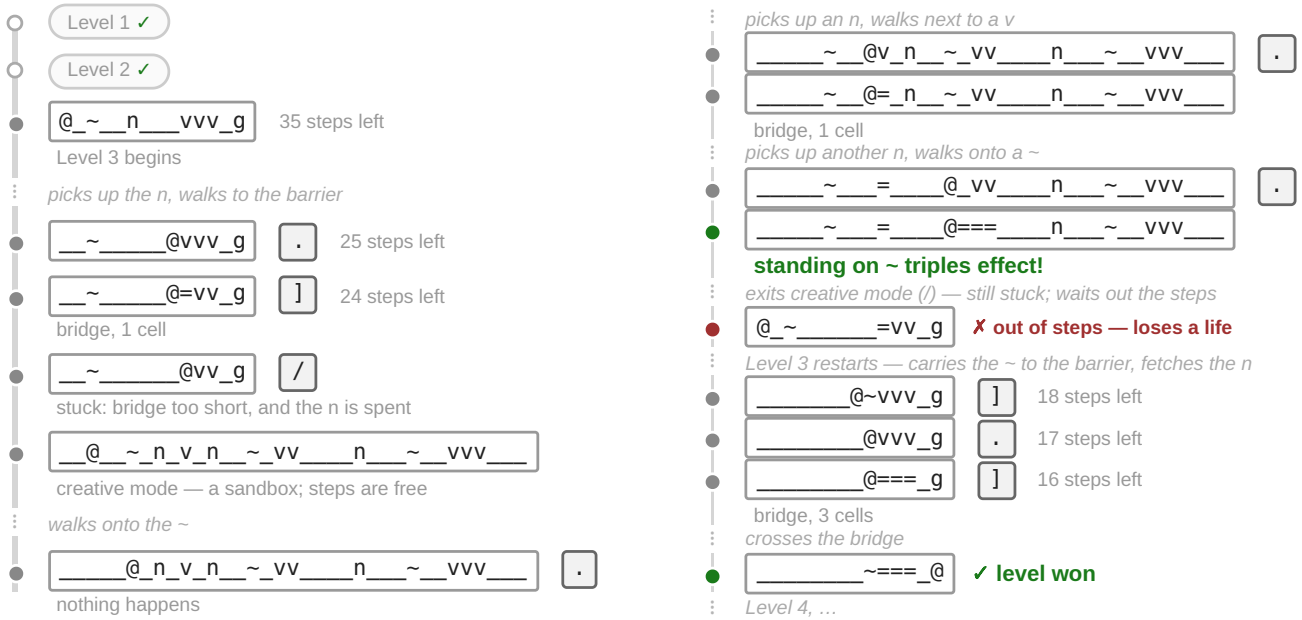

Figure 3: **Gameplay in game P-21.** An abridged example of a gameplay timeline. Having beaten Levels 1 and 2, the player applies the rule learned on Level 2 (not shown): activating a held `n` with '`.`' builds a bridge. But Level 3 contains a triple barrier `vvv`, and the bridge is too short. The player enters creative mode, a sandbox with a less restrictive step limit. Experimenting, the player discovers that standing on `~` has a tripling effect, allowing a longer bridge to be built. Back in survival mode, the level can no longer be won, so the player waits out the remaining steps to restart Level 3. On the second attempt, the player carries the `~` to the barrier, activates the `n` while standing on `~`, and crosses the triple-length bridge to reach the goal `g`, winning the level.

Table 1: **Examples of discoveries in public games.** Each row summarizes one mechanic a player must uncover to solve the game.

Game	Example discovery
P-2	Every level has two parts with analogous structure between the two parts.
P-8	Lilies grow only on the graves of roses.
P-9	Stepping over an underscore reverses the goal sequence order.
P-13	When Knights outnumber Knaves, Knaves tell the truth.
P-14	Voles run away from beetles.
P-15	Reactions have both parallel and rotationally symmetric structure.
P-18	The sliders control rate constants.
P-20	The semantic meaning of each word is informative about how the brackets operate on it.

2 Benchmark

The benchmark contains 70 games. We assigned each game to one of seven tiers according to machine difficulty, with tier 1 being the easiest and tier 7 the hardest (Figure 1). The lowest tier is mostly beaten by one-generation-old models like Gemini 3.1 Pro, while the higher tiers are challenging for state of the art models, including those in agentic harnesses. All 70 games were beaten by at least one human on their first attempt. We release three games publicly for each of the seven tiers, for a total of 21 public games, named P-1 through P-21. We hold the remaining games private for evaluation.

At each step of a game, the game generates an observation based on the current state, the player sees the observation and responds with an action, and the action conditions the next state transition in the game (Figures 2 and 3). The games thus have the form of a partially observable Markov decision process (POMDP). In some games, multiple states alias into the same observation, while other games are fully observable. In our benchmark, observations are text-based and formatted as a short Unicode string. Most games' observations consist of letters, numbers and simple punctuation; a few use special characters like arrows. Many observations fall on a single line; others include a small number of newline characters, for example to show an inventory or to place a cursor beneath a line. Newlines are not used to build large 2D grids. Actions correspond to single characters. Most games have fewer than 10 total possible actions (Figure 4). Which actions are available, and what effect they have, can change dynamically within a game. Each game has between 1 and 16 levels through which players progressively discover the game's structure, and each level has a limited number of steps (Figure 4).

To facilitate experimentation, many games include a creative mode (Figure 3) that can be entered and exited via a special action corresponding to a forward slash "/". Creative mode is a sandbox with rules similar or identical to the main game, but where steps do not count towards the per-level limit, allowing free exploration (there is a large, but finite, separate limit for steps taken in creative mode). The initial state in creative mode is typically different from the main game, so the player cannot simply work out solutions and copy them.

While playing the game, both humans and models are given access to the complete observation-action history in the game (although some model-harness combinations experienced context truncation in a small number of games).

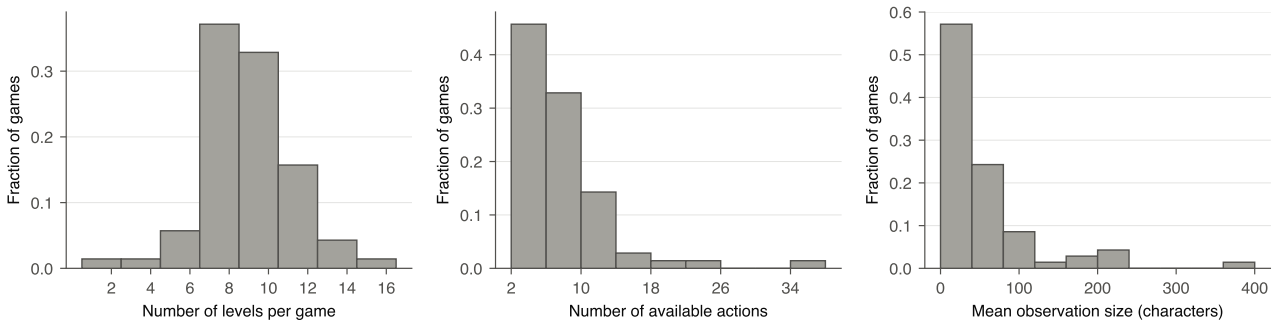

Figure 4: **Descriptive statistics of games.** Distribution over all 70 games of the number of levels per game (left), the number of available actions (center), and the mean observation size (right).

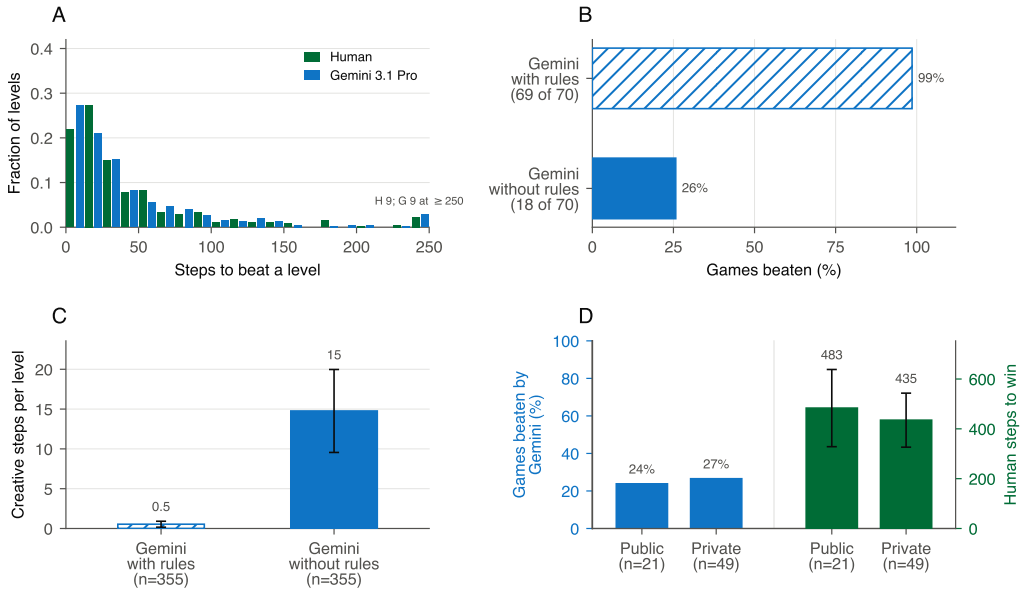

Figure 5: **Gameplay data.** (**A**) Steps to beat a level by humans versus Gemini 3.1 Pro, for all the levels beaten by both; the final bin includes 250 steps or more. (**B**) Giving Gemini access to the ground-truth rules of the game, specified in concise natural language, raises the win rate from 18/70 to 69/70 games. (**C**) Creative-mode steps per level by Gemini with and without access to rules. Error bars are 95% confidence intervals on the mean. (**D**) Left: Gemini win rates are similar between public and private games. Right: human step efficiency is similar between public and private games.

3 Results

We evaluated a range of AI models against our benchmark. DeepSeek V4 Pro Preview, Gemini 3.1 Pro, Opus 5, Kimi K3, DeepSeek V4 Flash 0731, Qwen 3.6 27B, GPT 5.5 and GLM 5.2 were evaluated on all games. Results are shown in Figure 1. The weakest model, Qwen 3.6 27B, beat 1 out of 70 games. The strongest model, Opus 5, beat 50. Taking the best model separately for each game after forming those model×game means, models in the basic harness collectively scored 57 out of 70; within the top two tiers, they scored 9 out of 20.

We wanted to know whether models in an agentic harness would outperform models in the basic harness. We tested Fable 5 in Claude Code, GPT 5.6 Sol in Codex, Kimi K3 in Kimi Code, and Gemini 3.1 Pro in PRO-LONG. The agentic harness conditions were tested only on tiers 6 and 7. In the case of Kimi K3 and Gemini 3.1 Pro, where a direct comparison was available, the model in the agentic harness performed no better than the same model in the basic harness. Finally, we tested Prime Agent (Opus 5 running in the Prime Agent product), which recently achieved 95.5% on ARC-AGI-3 public games, against tiers 6 and 7 of our benchmark. Prime Agent did not improve over basic harness Opus 5.

One aim of our benchmark is to test the player's capacity for deeper experimentation: constructing informative situations that are not directly along the path to a goal. Therefore, unlike ARC-AGI-3, our benchmark does not score step efficiency: exploration that stays within the step limit is not penalized. Of the levels beaten by both, Gemini 3.1 Pro took 46 ± 63 steps per level and humans took 49 ± 78 (mean ± SD; game-level paired Wilcoxon $p = 0.15$; see Figure 5A).

If the games truly challenge discovery rather than other abilities like planning or long-context retrieval, then *telling* models the rules of the game should categorically improve their performance. We therefore reran Gemini 3.1 Pro under the same conditions as before, but supplied with an additional text field which was a compact natural language description of the rules of the game. These rules summarized the dynamics of the game and the win condition; but they excluded any policy-related information like strategies, tactics or move sequences. Gemini beat 69 out of the 70 games with the rules given, against 18 out of 70 without them (Figure 5B). On the one game that Gemini did not beat with the rules, access to the rules nonetheless markedly improved its performance (Appendix B). Furthermore, giving Gemini the rules almost eliminated its use of creative mode (Figure 5C). These data are consistent with the idea that a primary challenge of the benchmark is finding out the rules.

Finally, we had humans play the games to ensure that every game was human beatable. All 70 games were beaten by at least one human on their first attempt at that game. Humans reported finding the games challenging, with some plays lasting for more than an hour of continuous play. Some players reported using a pen and paper. Our favorite quote from the web feedback form we gave players to fill out after they had finished a game: 'during dinner I was trying to figure out how to solve the puzzles'. The number of steps used by humans was no different than Gemini 3.1 Pro in matched levels (Figure 5A).

4 Methods

4.1 Agents

LLM agents interacted with the benchmark via an API that we are making publicly available for the public games. Agents were told that each game has a certain number of levels that they need to complete within a certain number of lives, and that they need to complete each level within a certain number of steps. Agents received the same information as humans as a prompt containing a general task description (see Appendix A), the current observation, level, lives, remaining steps, status, and legal actions, plus a game-mode field and level-transition message where applicable as part of their prompt. They then select an action using a tool call or in their output. We describe the agent harnesses we used in Section 4.1.1. We evaluated a broad set of models, with coverage varying by model and tier. Closed-source models

were run using official provider or AWS Bedrock APIs; some models were run with a cost cap listed in Table 2. Open-source models were run on a hosted server using NVIDIA H200 GPUs (on their high reasoning effort setting where available). Most model×game pairs had only a single run, so the results are effectively single-seed and we do not report variability across seeds. In some instances a model played a game more than once (with fresh context) due to evolving research plans. In Figure 1, we average those runs, because the aim is a fair comparison between models. In Figure 5 and 7, we count a game as beaten by Gemini 3.1 Pro if any Gemini run on that game resulted in a win, because we are interested in whether the games are beatable.

Model identifiers The exact model identifiers recorded for these results are `global.anthropic.claude-opus-5`, `openai.gpt-5.5`, `gemini-3.1-pro-preview`, `kimi-k3`, `glm-5.2`, `deepseek-v4-pro`, `deepseek-v4-flash`, `qwen3.6-27b`, `global.anthropic.claude-fable-5`, and `openai.gpt-5.6-sol`. The Anthropic and OpenAI models were accessed through Amazon Bedrock; Gemini 3.1 Pro was accessed through Google's own API. Because they were served through Bedrock, the OpenAI models were limited to a 272k-token context window rather than the 1M they support natively. For open-weights models the identifier is the checkpoint we served rather than an API identifier. Where a reasoning-effort setting was exposed, it was set to `high`. The runs reported were collected between 13 May and 11 August 2026; human plays were collected between 23 June and 5 August 2026.

Runs stopped at a cap A run that reaches its per-game cost or wall-clock cap is counted as a finished attempt that did not beat the game: a capped run is a loss. Most runs used a $200 cost cap. Kimi K3 used a 262k-token context window—rather than the 1M it supports, for speed and cost—and a 12-hour wall-clock cap per game. GLM-5.2, DeepSeek V4 Pro Preview, DeepSeek V4 Flash 0731, and Qwen3.6 27B likewise used a 12-hour wall-clock cap per game.

4.1.1 Harnesses

We evaluated models in one of two conditions: a minimal *basic harness* or an *agentic harness*, which layers a general-purpose agent product on top of the same game interface.

Basic harness The basic harness tests the ability of base LLMs to make discoveries without additional scaffolding. It preserves each provider's native reasoning-continuity mechanism where available, such as Gemini thought signatures, Anthropic thinking signatures, OpenAI encrypted reasoning items, or visible reasoning in SGLang-served open models. This lets models carry their own reasoning state across turns, giving models the best chance of performing well [3].

At the start of each run, the harness sends the model the task description and initial game state. It then executes a simple loop: the updated game state is appended to the conversation, the model submits exactly one legal action, and the action is applied, repeating until the game ends.

In very long games, the observation-action history is truncated to fit within the context window, excluding the oldest steps. This occurred in 6.1% of runs, mostly Qwen3.6 27B and GPT-5.5. For GPT-5.5, truncation occurred because the model as hosted on Amazon Bedrock had a 272k-token context window rather than the model's 1M-token context window. No Gemini 3.1 Pro or Opus 5 runs had truncated context. However, we did observe anecdotally that Gemini's performance appeared to degrade as the context length grew.

Agentic harness The agentic harness conditions reported in Figure 1 use Claude Code with Fable 5, Codex with GPT-5.6 Sol, Kimi Code with Kimi K3, Prime Agent [17] with Opus 5, and PRO-LONG [10] (a Read-Grep-Bash agent based on OpenCode scaffolding) with Gemini 3.1 Pro. In the harnesses wrapping Claude Code, Codex and Kimi Code, the game was exposed through a Model Context Protocol

(MCP) server. Prime Agent has no MCP support, so it received the same interface as a Python module preloaded into its IPython kernel. The PRO-LONG runs predate our MCP server, so PRO-LONG received the same Python-module interface as Prime Agent. Each run executed in a Docker sandbox whose only access to the game was the interface described above, and it was scoped to a single running session so the agents could not cheat by opening a new session. Within the sandbox, the agents had access to their full native tool set. For agentic harnesses, context was managed by the harness itself.

Harness	**Model**	**Effort**	**Budget Cap**	**Context Size**
basic harness	Claude Opus 5	high	$200	1M
basic harness	Gemini 3.1 Pro Preview	high	$200	1M
basic harness	GPT-5.5	high	$100	272k
basic harness	Kimi K3	max	12-hour	262k
basic harness	GLM-5.2	max	12-hour	1M
basic harness	DeepSeek V4 Pro Preview	-	12-hour	1M
basic harness	DeepSeek V4 Flash 0731	-	12-hour	1M
basic harness	Qwen3.6 27B	-	12-hour	262k
agentic harness (Prime Agent)	Claude Opus 5	high	$200	1M
agentic harness (Claude Code)	Claude Fable 5	high	$200	1M
agentic harness (Codex)	GPT-5.6 Sol	high	$200	272k
agentic harness (Kimi Code)	Kimi K3	max	12-hour	1M
agentic harness (PRO-LONG)	Gemini 3.1 Pro Preview	high	$200	1M

Table 2: Model Configuration

4.2 Humans

We collected human gold standard solutions on the games. Humans played the game via a web interface on a private portal (shown in Figure 2). Players were invited through personal and professional networks, direct outreach to academics, requests for professors to nominate students, and puzzle-solving communities. The players tended to be motivated by solving puzzles. These game plays were used simply to confirm that the games could be beaten by humans rather than gaining generalizable insight into human behavior. All data is reported only in aggregate and anonymized form.

The gameplay experience was standardized between humans and AI agents to the greatest extent possible. Humans were given the same information as agents, except additionally instructed that they should take their time, and that they were encouraged to use a paper and pen to write down their thoughts (see Appendix A). The rules of the game, including the goal, were not provided.

The current observation appeared in a central box. Players could select actions either via keypress or by clicking on one of the buttons appearing above the central box. The interface also showed remaining lives, remaining steps (which reset on every level completion or lost life) and current level. The history was rendered under the interface and was accessible throughout the entire game (abridged in Figure 2). The interface allowed humans to play each game multiple times. After finishing a game, players were asked to describe in free text how they thought it worked. Human performance is reported only from first attempts. We count a play as a first attempt only if that player had zero steps of prior exposure to the game.

5 Related works

Interactive discovery environments A number of benchmarks require an agent to uncover hidden task structure. Prominently, the third installment of the Abstraction and Reasoning Corpus for Artificial General Intelligence benchmark series (ARC-AGI-3) also presents games without instructions that require the agent to explore, plan, and discover the rules through interaction [1]. However, these games are intentionally grounded in human spatial and visual priors for difficulty, representing the environment visually in a two-dimensional grid. This means that part of the difficulty is perceptual, with simpler underlying abstract concepts than those in our benchmark. To compare the ARC-AGI-3 games with our games more accurately, we translated five of the games into our text-based framework, each of which was beaten by Gemini 3.1 Pro in a similar number of steps as humans (Appendix C). Other benchmarks that test the ability of agents to beat games in visuospatial environments include the BALROG (Benchmarking Agentic LLM and VLM Reasoning On Games) agentic-reasoning suite, which features games like Baba Is AI, MiniHack, and NetHack that require deducing environmental mechanics [20], and the AI GameStore [26], which aims at evaluating a model's general intelligence by adapting a set of a hundred human games. Although some of these games contain similar elements of mechanism and discovery, they explain many if not all of the rules and contain additional confounds like visual perception (in many games), long-term planning, reasoning beyond a single context window, spatial reasoning, and navigation. This makes it hard to attribute differences in performance to a single factor. Finally, benchmarks like FALSIFYBENCH [2], in which agents need to discover the semantic properties of a target by proposing test cases that not only confirm but also disconfirm hypotheses, study the ability of LLMs to make inferences in abstract domains. Similarly, Geng et al. [12] ask agents to reverse-engineer black-box programs, formal languages and equations, and find that active intervention helps only patchily.

Scientific discovery environments Another set of interactive environments casts discovery as science, asking the agent to run experiments against a simulated world. ScienceWorld [22] places agents in a text environment in which they must design grounded experiments based on an elementary-school science curriculum. DiscoveryWorld [15] extends this to include what the authors call the complete cycle of novel scientific discovery, including hypothesis generation, experimental design, execution, and analysis. SciGym [9] simulates biological systems, asking agents to iteratively design experiments against them. A second group tests how efficiently an agent can probe for the hidden structure of a parametric law or model. BoxingGym [11] draws generative probabilistic models from domains like psychology and ecology; PhysGym [4] controls the amount of prior knowledge supplied in interactive physics problems; and NewtonBench [29] applies counterfactual shifts to canonical physical laws to prevent recall of memorized information. CausalGame [5] hides a structural causal model behind biased and confounded observations.

Rule induction from fixed demonstrations A final family of benchmarks tests rule inference from a fixed set of demonstrations. ARC-AGI-1 and ARC-AGI-2 present a few input–output grid examples that demonstrate some hidden transformation rule, and the system must infer that rule from the handful of examples and apply it to a new input [6–8]. ConceptARC [18] uses this format to target single spatial or semantic concepts at varying levels of abstraction to evaluate models' understanding of them more completely. ARC-AGI 1 and 2 drew on classic psychometric tests like Raven's Progressive Matrices, which test human ability to infer abstract attribute rules by asking them to complete a matrix of figures based on existing examples. Procedurally Generated Matrices [28] directly adapted this format for machine learning. Bongard-LOGO [19] instead supplies a handful of positive and negative examples, and the Compositional Visual Relations (CVR) benchmark [27] measures sample efficiency and compositional transfer across abstract rules. Other benchmarks remove the visual component to focus on symbolic manipulation, such as rule induction in letter-string analogies [24] or grammatical constructions [14].

This class of benchmarks isolates rule abstraction cleanly, but it cannot measure how well an agent chooses what to try next, which is the ability our benchmark is built around.

6 Discussion

We introduce DiG-bench to measure a central capability for scientific progress: discovering new knowledge through active experimentation. DiG-bench consists of 70 new games, which do not exist anywhere on the internet, across seven tiers of difficulty. The games are designed with unfamiliar, non-obvious mechanisms discovered through interaction and experimentation. The observation of each game consists of a short string.

We tested the games on a variety of different models, either with the basic harness or with agentic harnesses. We found that many games are unbeatable by agents, even though the games are based in text—the natural domain of language models. The agentic harness conditions, including Prime Agent, did not boost performance over the basic harness. These results are suggestive that existing harnesses offer limited gains in discovery ability.

All 70 games were beaten by at least one human on a first attempt, suggesting that all of the games are beatable without excessive computation or access to unknowable information. At the same time, many of the games are substantially effortful for humans—discovery often takes work. Giving Gemini access to the ground truth rules of each game in natural language dramatically improved its performance, consistent with the idea that a primary challenge of the games is finding out the rules.

Games embedded in short strings give the best opportunity to probe the real discovery capabilities of LLMs. We found that the latest generation of models can experiment to discover unknown rules in many kinds of games created de novo. We speculate this is due to the trend in frontier models toward including a vast array of RL environments in training; therefore many short-horizon games effectively lie at points that can be interpolated from training data. Of course, discovering the rules of a game is only one kind of discovery—the most general form includes discovery of all possible kinds of knowledge.

Scientific discoveries made using AI could potentially improve human lives, ecosystems and social systems enormously. Fulfilling this potential to further science requires developing AI's ability to discover new knowledge and make open-ended discoveries in the real world. The same capability is also one of the few remaining bottlenecks to autonomous recursive self-improvement and potentially dangerous misuse; therefore, it needs to be evaluated carefully to ensure safety. Finally, current progress in AI and related fields represents an opportunity to renew a study of human discovery, improving our understanding of human science, and potentially providing ways we can improve our own ability to make discoveries.

References

[1] ARC Prize Foundation. ARC-AGI-3: A New Challenge for Frontier Agentic Intelligence, March 2026. URL http://arxiv.org/abs/2603.24621. arXiv:2603.24621 [cs.AI].

[2] Leonardo Bertolazzi, Katya Tentori, and Raffaella Bernardi. FALSIFYBENCH: Evaluating Inductive Reasoning in LLMs with Rule Discovery Games, June 2026. URL http://arxiv.org/abs/2606.04751. arXiv:2606.04751 [cs.CL].

[3] Ilan Bigio and Ted Sanders. How enabling two settings tripled our scores on the ARC-AGI-3 benchmark, July 2026. URL https://openai.com/index/how-two-settings-tripled-our-arc-agi-3-scores/.

[4] Yimeng Chen, Piotr Piękos, Mateusz Ostaszewski, Firas Laakom, and Jürgen Schmidhuber. PhysGym: Benchmarking LLMs in Interactive Physics Discovery with Controlled Priors, July 2025. URL http://arxiv.org/abs/2507.15550. arXiv:2507.15550 [cs.AI].

[5] Zhenhao Chen, Yongqiang Chen, Chenxi Liu, Junchi Yu, Xiangchen Song, Zijian Li, Jialin Li, Philip Torr, Bo Han, and Kun Zhang. CausalGame: Benchmarking Causal Thinking of LLM Agents in Games, July 2026. URL http://arxiv.org/abs/2607.04293. arXiv:2607.04293 [cs.CL].

[6] Francois Chollet, Mike Knoop, Gregory Kamradt, and Bryan Landers. ARC Prize 2024: Technical Report, January 2025. URL http://arxiv.org/abs/2412.04604. arXiv:2412.04604 [cs.AI].

[7] François Chollet. On the Measure of Intelligence, November 2019. URL http://arxiv.org/abs/1911.01547. arXiv:1911.01547 [cs.AI].

[8] François Chollet, Mike Knoop, Gregory Kamradt, Bryan Landers, and Henry Pinkard. ARC-AGI-2: A New Challenge for Frontier AI Reasoning Systems, May 2025. URL http://arxiv.org/abs/2505.11831. arXiv:2505.11831 [cs.AI].

[9] Haonan Duan, Stephen Zhewen Lu, Caitlin Fiona Harrigan, Nishkrit Desai, Jiarui Lu, Michał Koziarski, Leonardo Cotta, and Chris J. Maddison. Measuring Scientific Capabilities of Language Models with a Systems Biology Dry Lab, July 2025. URL http://arxiv.org/abs/2507.02083. arXiv:2507.02083 [cs.AI].

[10] Alexis Fox, Junlin Wang, Paul Rosu, and Bhuwan Dhingra. PRO-LONG: Programmatic memory enables long-horizon reasoning, July 2026. URL https://arxiv.org/abs/2607.20064. arXiv:2607.20064 [cs.AI].

[11] Kanishk Gandhi, Michael Y. Li, Lyle Goodyear, Agam Bhatia, Louise Li, Aditi Bhaskar, Mohammed Zaman, and Noah D. Goodman. BoxingGym: Benchmarking Progress in Automated Experimental Design and Model Discovery, January 2025. URL http://arxiv.org/abs/2501.01540. arXiv:2501.01540 [cs.AI].

[12] Jiayi Geng, Howard Chen, Dilip Arumugam, and Thomas L. Griffiths. Are Large Language Models Reliable AI Scientists? Assessing Reverse-Engineering of Black-Box Systems, May 2025. URL http://arxiv.org/abs/2505.17968. arXiv:2505.17968 [cs.CL].

[13] Ali Essam Ghareeb, Benjamin Chang, Ludovico Mitchener, Angela Yiu, Caralyn J. Szostkiewicz, Dmytro Shved, Gavin J. Gyimesi, Jon M. Laurent, Samantha M. Wright, Muhammed T. Razzak, Andrew D. White, Silvia C. Finnemann, Michaela M. Hinks, and Samuel G. Rodriques. A multi-agent system for automating scientific discovery. *Nature*, pages 1–3, May 2026. ISSN 1476-4687. doi: 10.1038/s41586-026-10652-y. URL https://www.nature.com/articles/s41586-026-10652-y.

[14] Satyam Goyal and Soham Dan. IOLBENCH: Benchmarking LLMs on Linguistic Reasoning, September 2025. URL http://arxiv.org/abs/2501.04249. arXiv:2501.04249 [cs.CL].

[15] Peter Jansen, Marc-Alexandre Côté, Tushar Khot, Erin Bransom, Bhavana Dalvi Mishra, Bodhisattwa Prasad Majumder, Oyvind Tafjord, and Peter Clark. DISCOVERYWORLD: A Virtual Environment for Developing and Evaluating Automated Scientific Discovery Agents, October 2024. URL http://arxiv.org/abs/2406.06769. arXiv:2406.06769 [cs.AI].

[16] Andrej Karpathy. karpathy/autoresearch, June 2026. URL https://github.com/karpathy/autoresearch. original-date: 2026-03-06T22:00:43Z.

[17] Seth Karten, Alex L. Zhang, Kevin Thomas, Sebastian Müller, and Prime Intellect Team. Prime agent: A self-improving rlm harness. *Prime Intellect Blog*, August 2026. https://www.primeintellect.ai/blog/prime-agent.

[18] Arseny Moskvichev, Victor Vikram Odouard, and Melanie Mitchell. The ConceptARC Benchmark: Evaluating Understanding and Generalization in the ARC Domain, May 2023. URL http://arxiv.org/abs/2305.07141. arXiv:2305.07141 [cs.LG].

[19] Weili Nie, Zhiding Yu, Lei Mao, Ankit B Patel, Yuke Zhu, and Anima Anandkumar. Bongard-LOGO: A New Benchmark for Human-Level Concept Learning and Reasoning. In *Advances in Neural Information Processing Systems*, volume 33, pages 16468–16480. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/hash/bf15e9bbff22c7719020f9df4badc20a-Abstract.html.

[20] Davide Paglieri, Bartłomiej 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. BALROG: Benchmarking Agentic LLM and VLM Reasoning On Games, April 2025. URL http://arxiv.org/abs/2411.13543. arXiv:2411.13543 [cs.AI].

[21] Jürgen Schmidhuber. Driven by compression progress: A simple principle explains essential aspects of subjective beauty, novelty, surprise, interestingness, attention, curiosity, creativity, art, science, music, jokes. In *Workshop on anticipatory behavior in adaptive learning systems*, pages 48–76. Springer, 2008.

[22] Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. ScienceWorld: Is your Agent Smarter than a 5th Grader?, November 2022. URL http://arxiv.org/abs/2203.07540. arXiv:2203.07540 [cs.CL].

[23] Xinhe Wang, Jin Huang, Xingjian Zhang, Tianhao Wang, and Jiaqi W. Ma. Your Reasoning Benchmark May Not Test Reasoning: Revealing Perception Bottleneck in Abstract Reasoning Benchmarks, January 2026. URL http://arxiv.org/abs/2512.21329. arXiv:2512.21329 [cs.CL] version: 2.

[24] Taylor Webb, Keith J. Holyoak, and Hongjing Lu. Emergent Analogical Reasoning in Large Language Models, August 2023. URL http://arxiv.org/abs/2212.09196. arXiv:2212.09196 [cs.AI].

[25] Georgios N Yannakakis and Julian Togelius. *Artificial intelligence and games*, volume 2. Springer.

[26] Lance Ying, Ryan Truong, Prafull Sharma, Kaiya Ivy Zhao, Nathan Cloos, Kelsey R. Allen, Thomas L. Griffiths, Katherine M. Collins, José Hernández-Orallo, Phillip Isola, Samuel J. Gershman, and Joshua B. Tenenbaum. AI Gamestore: Scalable, Open-Ended Evaluation of Machine General Intelligence with Human Games, February 2026. URL http://arxiv.org/abs/2602.17594. arXiv:2602.17594 [cs.AI].

[27] Aimen Zerroug, Mohit Vaishnav, Julien Colin, Sebastian Musslick, and Thomas Serre. A Benchmark for Compositional Visual Reasoning, June 2022. URL http://arxiv.org/abs/2206.05379. arXiv:2206.05379 [cs.CV].

[28] Chi Zhang, Feng Gao, Baoxiong Jia, Yixin Zhu, and Song-Chun Zhu. RAVEN: A Dataset for Relational and Analogical Visual rEasoNing, March 2019. URL http://arxiv.org/abs/1903.02741. arXiv:1903.02741 [cs.CV].

[29] Tianshi Zheng, Kelvin Kiu-Wai Tam, Newt Hue-Nam K. Nguyen, Baixuan Xu, Zhaowei Wang, Jiayang Cheng, Hong Ting Tsang, Weiqi Wang, Jiaxin Bai, Tianqing Fang, Yangqiu Song, Ginny Y. Wong, and Simon See. NewtonBench: Benchmarking Generalizable Scientific Law Discovery in LLM Agents, October 2025. URL http://arxiv.org/abs/2510.07172. arXiv:2510.07172 [cs.CL].

Appendix

A Game instructions

Humans and agents alike were told that the rules must be discovered, and were given the level, life, and step structure, without being told anything about the mechanics of any individual game (except for in the special runs where Gemini agents were given access to the ground truth rules of the game, which are shown in Figure 5). The texts below reproduce the standing prompt and first turn texts given to the basic harness, the prompt template given to the agentic harness, and the instructions for human players.

Fields in braces are filled in per run: the session and game identifiers (`{session_id}`, `{game}`), the names of the tools it is given (`{tools}`), its starting and first move indices (`{step_index}`, `{first_step}`), the requested pace (`{pace}`), and the observation that is currently shown (`{state}`). Angle brackets, by contrast, are part of the prompt text itself, telling the agent what to substitute at call time.

Onboarding text shown to human players

We are not going to tell you the rules of this game—you have to figure them out for yourself.

These games are not easy! It may take quite a bit of thinking and tinkering to figure out what's going on. Initially, it won't make any sense at all. This is normal.

We recommend you use a paper and pen to note down your thoughts and workings.

Levels, lives and steps

The aim is to complete all the levels.

You advance levels by reaching certain states within the game. You will have to figure out what these are.

Within each level, you have a limited number of steps. If you run out of steps, you lose a life. If you lose all your lives, the game is over.

It is also possible to lose a life by reaching certain states within the game.

Important: creative mode

At any time, you can use a button to switch into "creative mode", where you can experiment safely without losing steps or lives.

It may be necessary to use creative mode in order to discover the rules of the game without running out of steps.

System instruction given to the basic harness

We are not going to tell you the rules of this game—you have to figure them out for yourself.

Levels, lives and steps:

- The aim is to reach as high a level as possible. For each level you reach you will be awarded a bonus.

- You advance levels by reaching certain states within the game. You will have to figure out what these are.
- Within each level, you have a limited number of steps. If you run out of steps, you lose a life. If you lose all your lives, the game is over.
- It is also possible to lose a life by reaching certain states within the game.

Creative mode:

- The game may also have a creative mode. If it does, you can switch into it at any time to experiment safely without losing steps or lives, and it may be necessary to use it to discover the rules without running out of steps. You will know creative mode is available when the state carries a `mode` field and a toggle action appears in your `legal_actions`.

How you play:

- A short TASK DESCRIPTION at the start gives the objective and any special actions (not the rules). Each turn you then receive the current state: a text `observation` (the rendered screen), `level`, `max_level`, `lives_left`, `steps_remaining`, the list of `legal_actions`, plus `mode` and a `transition` message when those apply.
- Infer what each action does from how the state changes, and build on what you learn across turns. Reason carefully, then call `make_move` with EXACTLY ONE action from `legal_actions`.

First turn given to the basic harness

You are now playing this game.

TASK DESCRIPTION (objective + any special actions, NOT the rules):

`{description}`

INITIAL STATE:

`{state}`

Reason about it, then call `make_move`.

Prompt template given to the agentic harness

We are not going to tell you the rules of this game—you have to figure them out for yourself.

Levels, lives and steps:

- The aim is to reach as high a level as possible. For each level you reach you will be awarded a bonus.
- You advance levels by reaching certain states within the game. You will have to figure out what these are.
- Within each level, you have a limited number of steps. If you run out of steps, you lose a life. If you lose all your lives, the game is over.
- It is also possible to lose a life by reaching certain states within the game.

Important: creative mode

At nearly any time, you can use a button to switch into "creative mode", where you can experiment safely without losing steps or lives.
It may be necessary to use creative mode in order to discover the rules of the game without running out of steps.

Call the "step" tool with action `"/"` to enter creative mode.
Call the "step" tool with action `"/"` again to return to survival mode.
Only submit `"/"` when it appears in the state's `actions` list.

How you play—use the "Agent Benchmark API" MCP tools to drive the game:

1. Your game session is ALREADY started for you: `session_id="{session_id}"`, `game="{game}"`. You do NOT start or choose a game—you only have the `{tools}` tools, scoped to this one session, and you must pass this `session_id` to every call. Your starting state (`step_index={step_index}`) is: `{state}`
2. Each turn, read the current state and reason from these fields: `observation` (the rendered screen), `level`, `max_level`, `lives_left`, `steps_remaining`, `status`, `done`, the `actions` list (your legal moves), and `mode`/`transition` when present.
3. Make a move with the "step" tool: `session_id="{session_id}"`, `step_index=<the server's last returned step_index + 1>`, `action=<EXACTLY ONE string from the current state's actions list>`. Your first move uses `step_index={first_step}`. A `step_index` mismatch is a 409—always step off the server's last returned `step_index`. Use the "get_session" tool if you ever need to re-read the current state. Infer what each action does from how the state changes, and build on what you learn across turns. Keep playing, `{pace}`, until the state's `done` is true (status `game_over` or `completed`).

When the game is done, STOP making moves and write your debrief: the mechanics you discovered, the objective, useful strategies, and remaining uncertainties.

Prompt template given to the Prime Agent

We are not going to tell you the rules of this game—you have to figure them out for yourself.

Levels, lives and steps:

- The aim is to reach as high a level as possible. For each level you reach you will be awarded a bonus.
- You advance levels by reaching certain states within the game. You will have to figure out what these are.
- Within each level, you have a limited number of steps. If you run out of steps, you lose a life. If you lose all your lives, the game is over.
- It is also possible to lose a life by reaching certain states within the game.

Important: creative mode

At nearly any time, you can use a button to switch into "creative mode", where you can experiment

safely without losing steps or lives.
It may be necessary to use creative mode in order to discover the rules of the game without running out of steps.

Call `game_tool.step` with action `"/"` to enter creative mode.
Call `game_tool.step` with action `"/"` again to return to survival mode.
Only submit `"/"` when it appears in the state's `actions` list.

How you play—use the `game_tool` Python module (already importable in your IPython kernel) to drive the game:

1. Your game session is ALREADY started for you:
   `session_id="{session_id}"`, `game="{game}"`. You do NOT start or choose a game—you only have the `game_tool.step` and `game_tool.get_session` functions, scoped to this one session, and you must pass this `session_id` to every call. Your starting state (`step_index={step_index}`) is:
   `{state}`
2. Each turn, read the current state and reason from these fields: `observation` (the rendered screen), `level`, `max_level`, `lives_left`, `steps_remaining`, `status`, `done`, the `actions` list (your legal moves), and `mode`/`transition` when present.
3. Make a move with `game_tool.step(session_id="{session_id}", step_index=<the server's last returned step_index + 1>, action=<EXACTLY ONE string from the current state's actions list>)`. Your first move uses `step_index={first_step}`. A `step_index` mismatch is a 409—always step off the server's last returned `step_index`. Use `game_tool.get_session(session_id="{session_id}")` if you ever need to re-read the current state.
4. Infer what each action does from how the state changes, and build on what you learn across turns. Keep playing, one move per turn, until the state's `done` is true (status `game_over` or `completed`).

When the game is done, STOP making moves and write your debrief: the mechanics you discovered, the objective, useful strategies, and remaining uncertainties.

B Game unbeaten with rules

One game resisted Gemini 3.1 Pro even when the ground truth rules were supplied (Section 3). Figure 6 compares its progress with and without rules given.

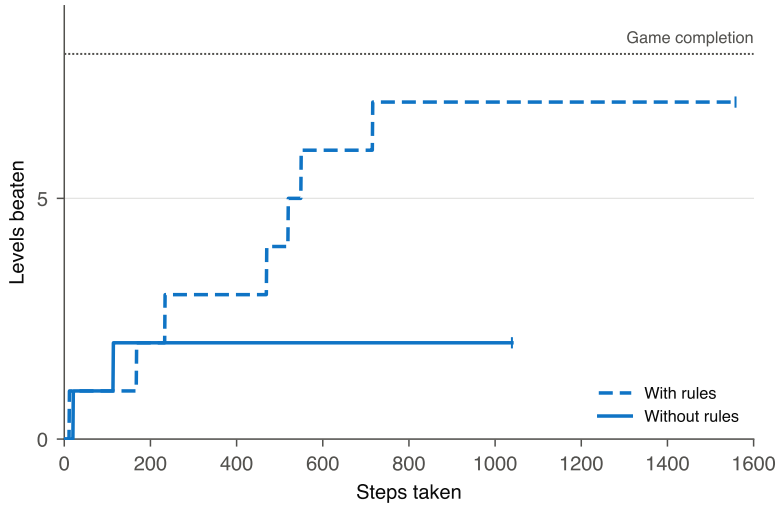

Figure 6: **Progress on the one game Gemini 3.1 Pro did not beat even when given the rules.** Levels beaten against steps taken. Dashed line is the run with rules supplied, solid line is without rules. Both conditions are Gemini 3.1 Pro. The dotted horizontal line marks game completion (8 levels). With the rules, Gemini beat 7/8 levels across 1,558 steps, without them 2/8 levels across 1,039 steps.

C Performance on ARC-like games

ARC-AGI-3 (2026) [1] is the interactive installment of the ARC-AGI (Abstraction and Reasoning Corpus for Artificial General Intelligence) series, created by François Chollet and maintained by the ARC Prize Foundation. It is designed to measure skill-acquisition efficiency and adaptation to novel environments. ARC-AGI-3 presents games without instructions, requiring the player to explore, plan, and discover the rules by acting. However, the games are conceptually simpler and require less experimentation to uncover the winning principles. The ARC benchmark evaluates step efficiency as a key metric, rewarding agents that reach a solution in fewer actions. It also represents its environments visually in a two-dimensional grid. To gauge how much of the difficulty of ARC-AGI-3 comes from the reliance on visual and spatial reasoning, we ported five public ARC-AGI-3 games into our text-based framework. Gemini 3.1 Pro cleared every level of all five, and the human step-efficiency advantage reported on the originals did not reproduce (Figure 7).

We created five other games (not included in the benchmark) intended to mimic as closely as possible the logic of five public ARC-AGI-3 games while being expressed in short strings rather than grids. It was not possible to make the game logic precisely isomorphic to the ARC-AGI-3 games, because some of the puzzle logic in ARC is intrinsically spatial. Nevertheless, these new games provided a crude view on how challenging the ARC puzzles would be if they were not confounded with perception and spatial reasoning.

Gemini 3.1 Pro cleared every level of all five ARC-like games (Figure 7A). This result is consistent with the difficulty of the originals depending partly on other aspects of the ARC-AGI-3 environment, such as visual perception. On these ported games we also did not reproduce the human step-efficiency advantage of the original ARC-AGI-3 set. Gemini's mean efficiency was slightly better at 29 steps per level against 47 for humans (Figure 7B). The medians were nearly identical: 23 steps for humans and 24 for Gemini.

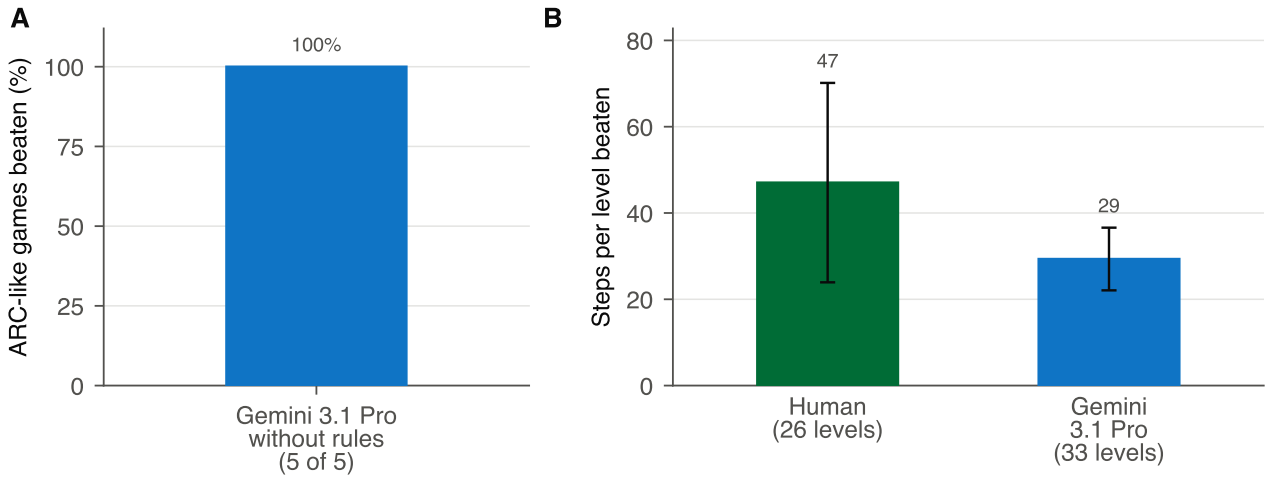

Figure 7: **Performance on the five ARC-like games.** (**A**) Games beaten by Gemini 3.1 Pro without rules. (**B**) Mean steps taken to beat a level by humans versus Gemini 3.1 Pro. Every level beaten by the best human play or the best Gemini 3.1 Pro run on each of the five games contributes one observation, pooled across games. Error bars are 95% confidence intervals. Gemini 3.1 Pro runs are pooled from the basic harness and the PRO-LONG agentic harness.

D Runs stopped at cost or time caps

Table 3 reports the number of runs stopped by each condition's per-game cost or wall-clock cap.

Harness	Model	Cap-stopped runs
basic harness	Claude Opus 5	1
basic harness	Gemini 3.1 Pro Preview	4
basic harness	GPT-5.5	3
basic harness	Kimi K3	6
basic harness	GLM-5.2	2
basic harness	DeepSeek V4 Pro Preview	0
basic harness	DeepSeek V4 Flash 0731	0
basic harness	Qwen3.6 27B	0
agentic harness (Prime Agent)	Claude Opus 5	0
agentic harness (Claude Code)	Claude Fable 5	5
agentic harness (Codex)	GPT-5.6 Sol	1
agentic harness (Kimi Code)	Kimi K3	1
agentic harness (PRO-LONG)	Gemini 3.1 Pro Preview	0

Table 3: Runs stopped by a per-game cost or wall-clock cap.