Mini Amusement Parks (MAPs): A Testbed for Modelling Business Decisions

Stéphane Aroca-Ouellette [*1] **Ian Berlot-Attwell** [*1 2 3] **Panagiotis Lymperopoulos** [1] **Abhiramon Rajasekharan** [1]
Tongqi Zhu [1] **Herin Kang** [1] **Kaheer Suleman** [1] **Sam Pasupalak** [1]

Abstract

Despite rapid progress in artificial intelligence, current systems struggle with the interconnected challenges that define real-world decision making. Practical domains, such as business management, require optimizing an open-ended and multi-faceted objective, actively learning environment dynamics from sparse experience, planning over long horizons in stochastic settings, and reasoning over spatial information. Yet existing human–AI benchmarks isolate subsets of these capabilities, limiting our ability to assess holistic decision-making competence. We introduce Mini Amusement Parks (MAPS), an amusement-park simulator designed to evaluate an agent's ability to model its environment, anticipate long-term consequences under uncertainty, and strategically operate a complex business. We provide human baselines and a comprehensive evaluation of state-of-the-art LLM agents, finding that humans outperform these systems by 6.5× on easy mode and 9.8× on medium mode. Our analysis reveals persistent weaknesses in long-horizon optimization, sample-efficient learning, spatial reasoning, and world modelling. By unifying these challenges within a realistic, scalable environment, MAPS offers a new foundation for developing and benchmarking agents capable of adaptable decision making.

1. Introduction

AI systems have achieved or surpassed human performance on tasks such as board games (Silver et al., 2017), competitive programming (Li et al., 2022), and professional exams including the Bar (Katz et al., 2023) and MCAT (Liu et al., 2024a). Yet these successes are in environments with narrow objectives and minimal uncertainty. This differs from many tasks in human domains. Consider, for example, the task of running a business. Success hinges on efficiently coordinating interdependent decisions—allocating staff, developing infrastructure, investing in R&D, and conducting market research—while simultaneously reasoning over long horizons about noisy, partially observable customer behavior shaped by multiple, often multi-modal, interacting factors. Moreover, business owners must actively and efficiently learn how to navigate these challenges, or risk bankruptcy.

Each of these dimensions, though frequently navigated by humans, presents a major hurdle for state-of-the-art AI systems. (1) When applied directly, large language models (LLMs) perform poorly on long-horizon planning and sequential reasoning tasks (Valmeekam et al., 2023; Aroca-Ouellette et al., 2025), often exhibiting myopic optimization for immediate concerns (Xie et al., 2025; Ma et al., 2025). Methods to mitigate this, such as tree search or iterative reasoning (Yao et al., 2023a), can quickly become prohibitively inefficient and expensive, and are constrained by the LLM's often inadequate evaluation of nodes (Chen et al., 2024). (2) AI systems remain inefficient learners, struggling to acquire skills from a handful of examples (Chollet, 2019). Humans, in contrast, actively construct and test hypotheses to accelerate learning (Gopnik et al., 1999) a capacity largely unexplored in current AI research (3) Existing models lag behind humans in reasoning about non-linguistic concepts, such as spatial or visual relationships (Hao et al., 2025; Bai et al., 2025). (4) Lastly, LLMs are unreliable when estimating causal and probabilistic relationships (Liu et al., 2024b), particularly when disentangling complex conditional distributions from samples (Anonymous, 2025).

Effective performance in real-world domains, such as business management, requires integrating all of these capabilities. To identify fruitful avenues of research and track progress toward robust, real-world decision-making, we introduce Mini Amusement Parks (MAPS), an amusement-park simulator that evaluates an agent's ability to actively model its environment, anticipate long-term outcomes, understand spatial relations, and optimize under uncertainty.

Prior benchmarks with human baselines fall into three categories. First, works such as Crafter (Hafner, 2022) and

[*]Equal contribution [1]Skyfall.ai [2]University of Toronto [3]Vector Institute.

	CH1	CH2	CH3	CH4	CH5	Features		
Environment	Open-Ended Objective	Long-Horizon Planning	Active WM Learning	Spatial Reasoning	Stochastic Transitions	Online Leaderboard	Variable Difficulty	Playable Online
ARC-AGI2	✗	✗	▲	✓	✗	✓	✓	✓
AutumnBench	✗	▲	✓	✓	✓	✗	✗	✓
Crafter	✗	✓	▲	✓	✗	✗	✗	✗
IVRE	✗	✗	▲	✗	✗	✗	✗	✗
DiscoveryWorld	✗	✓	▲	✓	✗	✗	✓	✗
WebShop	✗	✓	▲	✓	✗	✗	✗	✗
WebArena	✗	✓	▲	✓	✗	✓	✗	✗
OSWorld	✗	✓	▲	✓	✗	✓	✗	✗
VendingBench	✓	✓	▲	✗	✓	✓	✗	✗
MAPS	✓	✓	✓	✓	✓	✓	✓	✓

Table 1. Comparison of benchmarks that report human performance. ✗ = absence ▲ = partial presence ✓ = presence.

DiscoveryWorld (Jansen et al., 2024) emphasize long-term planning through hierarchical task structures but simplify entity relationships to static prerequisites, omitting the rich, dynamic interdependencies of complex systems. Second, abstract reasoning benchmarks such as ARC-AGI and AutumnBench (Chollet, 2019; Chollet et al., 2025; Warrier et al., 2025) probe few-shot generalization and fluid intelligence. While they reveal substantial human–AI gaps, their synthetic grid-based puzzles lack real-world semantics. Finally, applied benchmarks expect agents to complete practical tasks. OSWorld evaluates agents on computer tasks (Xie et al., 2024), and VendingBench evaluates an agents ability to operate a vending machine (Backlund & Petersson, 2025). Notably, of all these environments, only VendingBench evaluates open-ended optimization under uncertainty – all others are deterministic or have binary goals.

MAPS bridges these gaps. Like DiscoveryWorld, it requires long-horizon planning but features richer, more interwoven dynamics and incorporates stochastic transitions. Like ARC-AGI2, it demands sample efficiency and spatial reasoning, yet is grounded in a realistic domain. Like VendingBench, it emphasizes an open-ended objective under uncertainty, while additionally requiring actively learning environment dynamics and spatial reasoning. By integrating these challenges, MAPS provides a unique, comprehensive, and scalable AI testbed for robust business-style decision-making.

Our primary contribution is the introduction of MAPS and a comprehensive empirical evaluation of both human participants and state-of-the-art (SOTA) AI systems across multiple settings. These experiments reveal clear challenges facing the community: **CH1.** SOTA models lag far behind humans on the open-ended, multi-faceted task of business-style decision-making in MAPS, achieving only 7.16% of human performance. **CH2.** Their relative performance deteriorates substantially when long-horizon planning is required, underscoring persistent myopia in current models. **CH3.** Even with access to an exploratory sandbox mode, SOTA models fail to effectively learn environment dynamics—a capability we term *active world-model learning*. **CH4.** Agents show significant difficulty with spatial reasoning. **CH5.** SOTA systems struggle to model and act robustly under stochastic transitions, leading to brittle behaviour in uncertain environments. Together, these findings demonstrate that MAPS exposes a broad spectrum of persistent gaps in contemporary AI systems and provides actionable direction for advancing more capable, general-purpose decision-making agents.

2. Related Work

We position MAPS relative to benchmarks that include both an environment and human baselines across the five challenges posed by MAPS. Table 1 shows an overview of related benchmarks. See Section A for further related works.

2.1. Benchmarks without Open-Ended Tasks

Most benchmarks that include human baselines are designed around task completion, framing success as a binary outcome. Yet human performance naturally spans a spectrum, and these gradations carry important signal about capability. In addition, such environments almost always assume deterministic transition dynamics, in sharp contrast to the stochastic settings humans routinely navigate.

A first set of environments focus on abstract tasks far removed from real-world domains. ARC-AGI2 (Chollet et al., 2025) tests fluid intelligence through few-shot generalization in a grid-based colouring environment. Though humans vastly outperform LLMs, the task bears no resemblance to the real world. Similarly, Interactive Visual Reasoning (IVRE) (Xu et al., 2023) tests causal reasoning through an environment that emulates the Blicket test (Gopnik & Sobel, 2000) from child development literature. Concurrent to our work, AutumnBench (Warrier et al., 2025) assesses world model learning from interactions; like ARC-AGI they use satisfaction tasks in coloured grid domains. Due to the lack of grounding, their tasks are divorced from real-world

decision making tasks, and must be simple to make learning tractable with minimal prior tractable: 70% of their planning tasks are solvable by a random agent. Further their environments have at most 8 latent variables — in MAPS, each individual guest is modelled by 14.

Another line of work investigates long-term planning through dependency trees. In Crafter (Hafner, 2022) and HeroBench (Anokhin et al., 2025) players navigate a grid-world, find resources, craft increasingly complex items, and fight monsters. DiscoveryWorld (Jansen et al., 2024) requires players to use items to iteratively perform scientific experiments and build knowledge through discoveries.

Finally there are benchmarks revolving around real computer-based tasks. WebShop (Yao et al., 2022) requires agents to navigate, customize, and purchase items in simulated e-commerce websites; WebArena (Zhou et al., 2023) introduces 812 long-horizon tasks that involve interacting with a websites at-large; and OSWorld (Xie et al., 2024) comprises 369 real-world computer tasks ranging from file system manipulation to formatting spreadsheets.

2.2. Benchmarks with Open-Ended Objectives

We next look at the smaller set of environments that provide a human baseline *and* focus on optimizing an objective. Most related is VendingBench (Backlund & Petersson, 2025), a vending machine simulation spanning hundreds of days with stochastic customers. It reports human performance, but only from a single participant, limiting the strength of comparison. Moreover, it was designed as a zero-shot probe rather than a setting requiring agents to learn and adapt — a core hallmark of human intelligence (Tenenbaum et al., 2011). It also does not require any spatial reasoning, something common in human tasks. Lastly, it is already saturated, with LLMs outperforming humans.

Crafting and survival-based environments evaluate long-horizon, dependency-driven reasoning and require substantial game-specific knowledge to be successful, but typically evaluate task completion (Fan et al., 2022; Anokhin et al., 2025). The Nethack Learning Environment (NLE) (Küttler et al., 2020) is an exception, instead optimizing the game's multi-faceted score. It does not, however, map to real world tasks, nor provide an evaluation framework to support sample-efficient world modelling.

Lastly, Alchemy (Wang et al., 2021) and ScienceWorld (Wang et al., 2022) evaluate an agent's ability to apply scientific principles. Alchemy emulates chemical interactions, requiring spatial reasoning to identify a sequence of potions to maximize the value of stones. ScienceWorld introduces a text-based environment where agents have to design experiments based on a fifth-grade science curriculum. Both resemble real human tasks, but are fully deterministic.

3. Method

In **Mini Amusement Parks (MAPS)**, a player takes the role of an amusement park manager and must maximize the value of the park within a given time horizon. Every morning they perform an action such as building new a ride or shop, hiring staff, or setting a research agenda. Guests then spend the day interacting with the park.

MAPS consists of six main components—terrain, rides, shops, staff, subclasses and research, and guests—and can be played in easy or medium mode. We briefly outline of these components below, but refer the reader to the game's official documentation (cf. Section K) for more detail.

The park is a 20×20 grid with an entrance, an exit, a connecting path, and water tiles that increase the excitement of nearby rides. Rides (one of carousels, Ferris wheels, or roller coasters), drive the park's capacity and park rating. In turn, these determine how many guests visit the park. Shops cater to the needs of guests, providing food and drink. There are also specialty shops that provide a range of unique benefits, such as a souvenir shops, ATMs, and info booths. As in a real business, the inventory of shops must be managed: over-stocking leads to waste and under-stocking causes shops to go out of service. The park can also employ three types of staff: janitors clean dirty tiles, mechanics repair broken rides, and specialist perform a variety of useful tasks such as restocking shops or entertaining guests.

Each ride, shop, and staff subtype has tiered subclasses: yellow, blue, green, and red. Yellow entities are basic and affordable, while red entities provide greater value at higher costs. In easy mode, everything is unlocked from the start. In medium mode, they are unlocked in order via research.

Guests visit the park with a varying initial money, energy, hunger, thirst, and happiness. Throughout the day, guests sample what to do based on their status. While guests act according to general patterns (e.g., going to a food shop when hungry) their specific decisions (e.g., which food shop to go to) are sampled from a distribution.

MAPS design gives rise to complex dynamics based not only on the most recent action, but the overall design of the park.

3.1. Challenges and Research Questions

CH1. Open-Ended Objective At its core, MAPS is designed to highlight gap between current AI and human capabilities. To this end,

RQ1: How do frontier models compare to humans in the integrated setting of MAPS*, which jointly tests long-horizon planning, active world-model learning, spatial reasoning, and reasoning under stochasticity?*

To answer this, we implement a standard ReAct (Yao et al.,

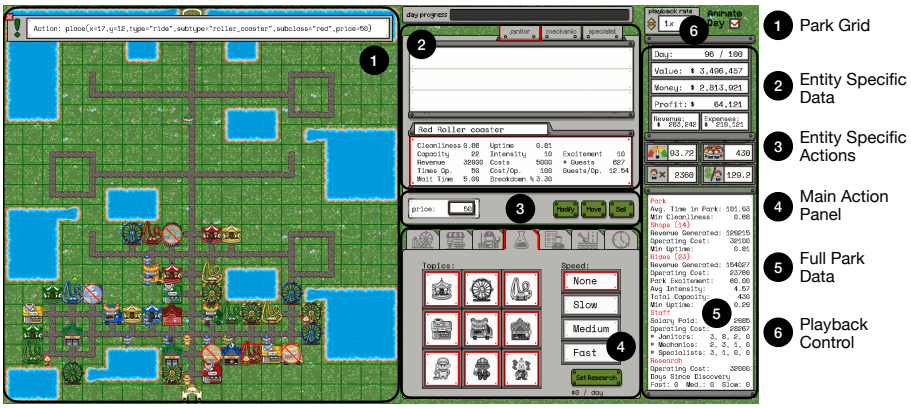


Figure 1. The GUI view of MAPS.

2023b) baseline in which we condition agents on the history of past actions and observations before thinking and generating the next action. We limit the history to the past 5 states and actions to avoid issues with context given the large and growing state space of the game. We include the objective of maximizing park value, the game manual, and the current settings in the system prompt. For additional details, including prompts, see Appendices B and J.1.

CH2. Long-Horizon Planning Long-horizon planning ability is a critical for enterprise AI. To assess this ability, MAPS provides two difficulty modes, easy and medium, where the importance of long-horizon planning is the key differentiator. On medium difficulty the horizon is extended from 50 to 100 and research introduced. The research mechanics places heavy emphasis on planning, as players must anticipate the needs of their long-term strategy. For example: an expensive ride researched too early will be unaffordable, researching attractions that are unnecessary for the strategy at hand wastes time and money, and building a high guest-count park without researching better staff can cause cascading deterioration and significant loss of potential revenue. Success in medium mode therefore depends on forecasting future constraints and opportunities, and sequencing investments so that necessary attractions and staff are available when needed. This design directly tests whether agents can reason across extended horizons, balance short-term gains against long-term outcomes, and plan under uncertainty.

RQ2: How does agent performance change as long-horizon planning demands increase from easy to medium mode?

CH3. Active WM learning Humans efficiently explore and experiment in novel environments, rapidly learning underlying dynamics by adapting prior knowledge (Gopnik et al., 1999; Tenenbaum et al., 2011; Collignon, 2020). This ability to learn from limited experience and to direct exploration towards important areas of uncertainty, is central to success in business domains such as risk management, market analysis, and product development, where data collection opportunities are scarce.

Inspired by this, MAPS tests active world-model learning, also known as world-model induction (Ying et al., 2025). In standard active learning, systems improve task performance by selectively querying an oracle. In comparison, active world-model learning agents must design and execute informative experiments by interacting with the environment. Agents must recognize their own uncertainty, identify the aspects of the environment most relevant to their goal, and strategically explore to reduce that uncertainty under a limited budget. To succeed, systems must construct appropriate experimental conditions to tolerate the noisy, partially observable worlds. Notably, the goal is not to replicate the transition function of the environment, but to create substantiated and impactful theories of the environment capable of supporting long-horizon predictions over key desiderata and guiding an effective policy.

To support such exploration, MAPS includes a novel sandbox mode that expands the action space with auxiliary actions, such as granting funds, unlocking research, switching layouts (between 5 unique non-evaluation layouts), and undoing actions. These enable counterfactual testing and efficiently observing the distribution of possible outcomes for a given action. The sandbox mode is designed to be used a small number of samples, mirroring businesses that cannot afford endless focus groups or trial product launches. The demand for sample efficiency, combined with the actions it provides for active, sample-efficient world-model learning, distinguishes the task from standard RL where millions of interactions are feasible and where exploration is not traditionally done via curated experiments. In realistic domains, agents must achieve useful world models from only tens of trials. To our knowledge, the use of exploration actions is unique in the world-modelling literature; prior works (Allen et al., 2020; Warrier et al., 2025) have operated in simpler environments, only providing a reset action.

RQ3: Can current LLMs effectively leverage a sandbox mode to improve their performance?

To answer this question, we modify our ReAct agent to learn from 100 in-game days in the sandbox mode. First, we adjust the system prompt to explain that the agent is learning in sandbox mode and should discover useful information in preparation for the final evaluation. Second, we add a "Learnings" section before the reason and act steps where the LLM is instructed to make useful notes for its future self; it is informed that it will have access to these notes in both sandbox and evaluation stages. Once the sandbox budget is depleted the notes are extracted and added to the ReAct prompt (see Section J.1). We also repeat this experiment for a subset of models where we withhold the manual.

CH4. Spatial Reasoning. Spatial reasoning plays a central role in business decision-making: storefront location, products' position on shelves, ad placement, logistics – all hinge on understanding spatial interdependencies. Similarly, in MAPS, profitability demands spatial understanding in park design. Guest behaviour is shaped by spatial relationships: they dislike walking too much between attractions, rides near water are more appealing, and specialty shops—such as ATMs or info booths—are not directly targeted by guests so must be placed in high-traffic areas to attract customers.

To probe the spatial reasoning of LLMs, we develop a heuristic that selects the position of attractions based on three rules: (1) to encourage density, sub-select the four empty tiles closest to existing buildings and the entrance, (2) upweight ride and downweight shop locations based on the number of adjacent water tiles (3) for specialty shops, upweight locations based on the number of adjacent path tiles. This simple heuristic captures only first-order spatial dependencies between attractions and guest flow; optimal gameplay requires a richer and more adaptive understanding of multi-entity interactions, crowd dynamics, and evolving park structure.

RQ4: Do LLMs exhibit spatial reasoning abilities that surpass a simple heuristic baseline, effectively leveraging complex spatial relationships to improve park performance?

We test this by comparing a baseline ReAct LLM with one where we replace the all (x, y) placement parameters with the (x, y) position generated from heuristic. We run these evaluations using the easy difficulty to focus the comparison on spatial reasoning rather than long-horizon planning.

CH5. Stochasticity. Real-world decision-making is inherently stochastic: identical actions produce varying outcomes due to epistemic and aleatoric uncertainty. Like in real businesses, action outcomes in MAPS are non-deterministic due to probabilistic guest behaviour. Guest decisions depend on unobservable latent variables (e.g., hunger and thirst), partially controllable factors (e.g., proximity, cleanliness, the availability of unique attractions), and irreducible randomness arising from individual sampling. Understanding this variability is essential: agents must distinguish genuine performance trends from random fluctuations, avoiding overreaction to noise while still learning from the feedback.

This stochasticity is particularly relevant when combined with long-horizon planning. World models (WMs) are a promising approach to planning under uncertainty, yet many recent works are limited to deterministic environments (Tang et al., 2024; Dainese et al., 2024), evaluated under deterministic assumptions (Piriyakulkij et al., 2025), and/or cannot scale to the complexity of MAPS (Curtis et al., 2025; Khan et al., 2025). Among scalable approaches, the most applicable prompt LLMs for state prediction (Hao et al., 2023; Zhao et al., 2023). WALL-E is one such model, augmenting an LLM with learned rules (Zhou et al., 2024).

RQ5: How much variability exists in MAPS*, and can current AI systems model and act effectively under this uncertainty?*

To quantify stochasticity, we measure both day-to-day and trajectory-level variance across repeated runs. We replay identical trajectories ten times using the same sequence of actions and compute the coefficient of variation, $CV = \frac{\sigma}{\mu}$, for three key metrics: revenue, money, and overall park value. Revenue variance captures the unpredictability of guest behaviour. Money variance reflects how stochasticity propagates to resource constraints for future actions. Park-value variance assesses MAPS's consistency in evaluating player performance. For day-to-day variance we repeatedly reset the environment and execute the action. For trajectory-level variance, we repeat entire playthroughs using identical action sequences, discarding runs that diverge due to invalid actions caused by early-stage randomness.

Finally, to evaluate how effectively AI systems can model and act under uncertainty, we extend the ReAct baseline with model-predictive control (MPC) using either (i) an oracle WM or (ii) WALL-E as a learned WM (Zhou et al., 2024). In both we adopt random-shooting MPC—performing five rollouts under the LLM policy and selecting the first action in the best trajectory after four WM-simulated steps.

3.2. Observation & Action Space

To support both human and AI play, MAPS provides two equivalent observation modes: a *graphical interface* and a *textual JSON representation* (via Pydantic). Both contain identical information, including aggregate statistics (e.g., profit, park rating, guest count) and detailed states of employees and attractions. The GUI is shown in Figure 1, and an example JSON observation in Section I. Actions are strings formatted as `Python` function calls. Available actions include *place*, *move*, and *remove* attractions or staff; *modify* attractions; *set research*; *wait* (no-op); and *survey guests*. Full action specifications are in the documentation.

LLM	Score (%)	Runtime (s)	Cost (USD)
EASY DIFFICULTY			
Human	*835,318*	$1109_{\pm 62}$	
GPT-5 Nano	$0.36_{\pm 0.28}$	$1{,}311_{\pm 305}$	$0.13_{\pm 0.02}$
GPT-5	$13.89_{\pm 11.16}$	$3{,}774_{\pm 505}$	$3.75_{\pm 0.32}$
Grok 4	$6.62_{\pm 14.50}$	$2{,}908_{\pm 437}$	$6.60_{\pm 0.63}$
Sonnet 4.5	$4.53_{\pm 5.75}$	$937_{\pm 124}$	$6.06_{\pm 0.60}$
Gemini 2.5 Pro	$3.96_{\pm 8.23}$	$787_{\pm 84}$	$3.06_{\pm 0.30}$
MEDIUM DIFFICULTY			
Human	*2,062,681*	$3559_{\pm 406}$	
GPT-5 Nano	$0.64_{\pm 0.78}$	$2{,}506_{\pm 705}$	$0.27_{\pm 0.04}$
GPT-5	$7.16_{\pm 8.36}$	$7{,}583_{\pm 764}$	$8.36_{\pm 0.48}$
Grok 4	$0.62_{\pm 0.39}$	$5{,}438_{\pm 683}$	$13.00_{\pm 0.86}$
Sonnet 4.5	$1.43_{\pm 1.40}$	$1{,}713_{\pm 86}$	$12.19_{\pm 1.03}$
Gemini 2.5 Pro	$0.64_{\pm 0.29}$	$1{,}423_{\pm 126}$	$5.97_{\pm 0.45}$

Table 2. Results of SotA LLMs on MAPS. Human score is the mean park value across layouts. Score (%) indicates performance relative to humans per layout. 100.00% indicates parity.

3.3. Evaluation Protocols

Beyond difficulty settings, we define four resource settings that vary the amount of prior knowledge and experience provided to an agent: (1) Documentation: only the game documentation. (2) Few-shot: only 100 in-game days in the sandbox mode (excluding sandbox-specific exploration actions). (2) Few-shot + documentation: 100 days in the sandbox environment, plus the game documentation. (3) Unlimited: no restrictions.

Unless otherwise noted, agents are evaluated in the documentation only setting. Each agent is evaluated on three held-out evaluation layouts (i.e., not provided in sandbox training) with three random seeds per layout. The primary metric is the park's final value: the sum of final money on hand, the sell price of rides and shops, and a small contribution from intellectual property accrued through research.

To establish a high-water mark, we collect human performance through an online leaderboard, which is available at: maps.skyfall.ai. As participants can retry layouts or share strategies, these results reflect an unlimited setting and serve as an empirical upper bound rather than as a direct comparison. To facilitate aggregation across layouts, scores are normalized by human performance for each layout and difficulty, with 100% indicating human parity.

4. Results

We organize our quantitative and qualitative findings by challenge. For further results see Section E.

4.1. CH1. Open-Ended Objective

We first benchmark GPT-5 Nano, GPT-5 (OpenAI, 2025), Grok-4 (xAI, 2025), Claude Sonnet 4.5 (Anthropic, 2025), and Gemini 2.5 Pro (Gemini Team, Google DeepMind, 2025) on MAPS. Table 2 shows that all benchmarked systems lag significantly behind humans; even the best model, GPT-5, achieves an average park value that is only 7.16% of that achieved by humans. Further, GPT-5 took over two hours on average to play a full game, compared to the 50 minutes humans took to complete the game.

Reviewing various collected trajectories on medium difficulty, we observed some common trends: (1) Sub-optimal strategies for when and what to research. GPT-5 consistently uses research better than the other models tested, a key reason for its higher scores. We delve deeper in Section 4.2. (2) Poor spatial placements. Constructed parks tend to lack density, fail to place rides near water, and lack local diversity. Again, GPT-5 outperforms other models but remains well below human performance. We discuss further in Section 4.3. (3) Under utilization of shops. This is particularly noticeable for Sonnet 4.5 and Gemini, which frequently place a single food and drink shop. (4) Too much waiting. While top human scores almost never use the wait action (instead modifying stock quantities or hiring affordable staff), models tend to wait until able to reach their myopic goal. Claude and Gemini wait in over 40% of their turns. This is another area where GPT-5 outperforms the other, waiting in only 2.65% of its turns. (5) Lack of diversity. The game incentivizes unique attractions, but agents do not identify the value in this. GPT-5 is the worst offender, placing $>$ 9x times more roller coasters than carousels. (6) Agents do not take into account that yellow drink shops are free to stock, and should therefore be stocked to very high numbers to avoid adjustments or selling out.

Further general qualitative analysis, including model-specific behaviours, can be found in Section H.

4.2. CH2: Long-Horizon Reasoning

LLM	Easy Score (%)	Med. Score (%)
GPT-5 Nano	$0.36_{\pm 0.28}$	$0.64_{\pm 0.78}$
GPT-5	$13.89_{\pm 11.16}$	$7.16_{\pm 8.36}$
Grok 4	$6.62_{\pm 14.50}$	$0.62_{\pm 0.39}$
Sonnet 4.5	$4.53_{\pm 5.75}$	$1.43_{\pm 1.40}$
Gemini 2.5 Pro	$3.96_{\pm 8.23}$	$0.64_{\pm 0.29}$

Table 3. The impact of long-horizon planning is seen in the relative performance drop from easy to medium mode. Medium difficulty doubles the horizon and emphasizes long-term planning by introducing research as a game mechanic. Score (%) indicates the score relative to the human baseline, with 100.00% indicating parity.

To evaluate the impact of long-horizon planning, we compare the relative performance of models under easy and medium difficulty. I.e., amplifying the need for foresight and strategic sequencing by extending the planning horizon and introducing research. As shown in Table 3, relative per-

formance to humans declines significantly. GPT5, the most performant model we tested, drops from 13.89% to 7.16%, indicating that even the strongest systems struggle with increased temporal and structural complexity. Notably, human players achieve higher park values in medium than easy, as the longer-horizon outweighs the additional challenge of research; the inverse holds for AI systems.

Examining the research patterns of models reveals several consistent trends. All models exhibit a strong bias toward rides, with roller coasters being the most frequent. This choice is typically suboptimal, as the blue roller coaster is prohibitively expensive. Excluding GPT-5, in 13 of the 14 trajectories where it is researched, it is never built. While GPT-5 over prioritizes blue roller coasters, it is the only model to consistently use all of its unlocked research, placing at least one instance of every researched entity. Moreover, GPT-5 with sandbox learning is the only agent to reach the green tier, completing a single run with both a green roller coaster and a green Ferris wheel. A possible reason for this success is that GPT-5 continues expansion until profits can sustain research; Grok-4 and Gemini create small parks and wait until they can afford a single research step, leading to inferior performance. Only three agents—GPT-5 (with learning) and Grok-4 (with and without learning)—research shops, and this occurs in fewer than one-third of their trajectories. Staff research virtually absent, with a single observed case of GPT-5 with learning unlocking blue janitors and mechanics. This diverges sharply from expert human play, where every entity type is typically researched to at least the blue level (further progression depends on park layout and strategic intent).

Overall, these findings suggest that current LLM-based agents adopt a myopic approach to success. Rather than planning for future park requirements and constraints, they pursue greedy strategies that fail to account for the long-term dependencies imposed by research and expansion.

4.3. CH3: Active World Modelling

Table 4 shows the results of the tested models with no learning and with the learnings from interacting in the sandbox mode for 100 in-game days. We additionally evaluate GPT-5 Nano and GPT-5 both with and without learning when the game's documentation is not provided; see Table 7.

In most cases we find either no tangible benefit or a quantifiable *decrease* in average performance. From a qualitative analysis of the learnings, we hypothesize this is likely due to the learnings generated: we find they often parrot the same concepts provided in the manual, are overly specific to the current transition, fail to consider stochasticity or necessary context in the state, and fail to capture true causal relationships and instead incorrectly attribute changes to their current action rather than the consequence of their overall park design. The few correct insights are not clearly actionable, or are for late stages of the game that the model never reaches in evaluation. Overall, learnings provide little to no benefit, merely diluting the context.

LLM	Score (%)	Score (%) w. Learning
EASY DIFFICULTY		
GPT-5 Nano	$0.36_{\pm 0.28}$	$0.15_{\pm 0.18}$
GPT-5	$13.89_{\pm 11.16}$	$15.47_{\pm 13.35}$
Grok 4	$6.62_{\pm 14.50}$	$2.66_{\pm 2.86}$
Sonnet 4.5	$4.53_{\pm 5.75}$	$0.83_{\pm 0.84}$
Gemini 2.5 Pro	$3.96_{\pm 8.23}$	$1.21_{\pm 1.47}$
MEDIUM DIFFICULTY		
GPT-5 Nano	$0.64_{\pm 0.78}$	$0.11_{\pm 0.12}$
GPT-5	$7.16_{\pm 8.36}$	$10.21_{\pm 10.07}$
Grok 4	$0.62_{\pm 0.39}$	$1.24_{\pm 1.14}$
Sonnet 4.5	$1.43_{\pm 1.40}$	$0.88_{\pm 0.72}$
Gemini 2.5 Pro	$0.64_{\pm 0.29}$	$0.60_{\pm 0.64}$

Table 4. Scores achieved when models are provided with and without the opportunity to learn in sandbox for 100 in-game steps. Score (%) indicates the score relative to the human baseline, with 100.00% indicating parity.

There are two exceptions. Grok-4 improves from 0.62 to 1.24% in medium mode and GPT-5 improves from 13.89 to 15.47% and from 7.16 to 10.21% in easy and medium modes respectively. They suffers from similar issues, but to a lesser extent; being able to find a few general and more actionable insights. As mentioned in the previous section, a clear outcome of this learning is that GPT-5 becomes more successful at research. Whether this is due to a more directed approach toward it, or simply a more successful early game that enables research faster is unclear. A breakdown of a subset of learnings for each model can be found in Section F.

Without the game documentation, we see an expected drop in GPT-5 performance. Manual inspection reveals that GPT-5 acts more reactively, impatiently selling off attractions over repositioning or adjusting inventory levels. Interestingly, while sandbox learning is still useful in easy mode, performance drops in medium (TODO% → TODO%). The root cause of this drop is unclear, but manual inspection reveals a greater tendency to sell attractions – see Section H

4.4. CH4: Spatial Reasoning

We test the spatial reasoning heuristic on three models of varying capabilities: GPT-5 Nano, Claude Sonnet 4.5, and GPT-5. The results, shown in Section 4.4, show a clear trend that providing this simple heuristic improves performance across all three models. Looking at the generated parks (see Section G), we note that parks with the heuristic are more compact, rides are more frequently by water, and shops are more frequently at intersections. The base models, particularly GPT-5, tend to repeatedly place the same kinds of

rides near each other, something which the heuristic disrupts. This local diversity is important for a successful park, and an unintended positive consequence of the heuristic.

Another issue we identify is that agents struggle with spatial understanding of paths, constructing inaccessible rides on test layouts with disconnected path networks. While models generally recover from this after a few turns by selling or moving the attraction, they waste turns doing so. Relatedly, GPT-5 sometimes clusters rides in ways that are nearby as the crow flies but distant for guests following a winding path – further suggesting a flawed understanding of paths.

Lastly, we note that even with the heuristic, the park design is still far from optimal, highlighting a clear avenue for fruitful future work.

LLM	Score (%)	Score (%) w. Heuristic
GPT-5 Nano	$0.36_{\pm 0.28}$	$0.96_{\pm 1.09}$
GPT-5	$13.89_{\pm 11.16}$	$21.55_{\pm 14.33}$
Sonnet 4.5	$4.53_{\pm 5.75}$	$8.39_{\pm 7.90}$

Table 5. Agent performance with and without a simple placement heuristic. The spatial reasoning of LLMs currently underperforms a simple heuristic. Score (%) indicates the score relative to the human baseline, with 100.00% indicating parity.

4.5. CH5: Stochasticity

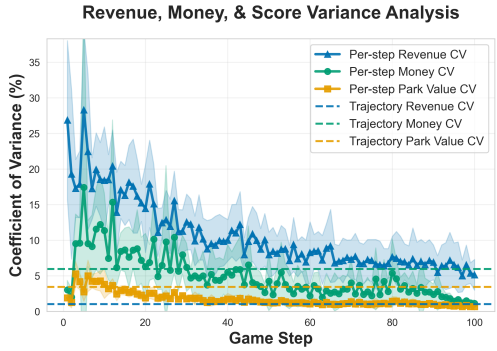


Figure 2. The per-day and full trajectory coefficients of variation for revenue, money, and park value across several full games.

Figure 2 illustrates the variance in MAPS, plotting the coefficient of variation, per-step and per-trajectory. Note that as the game progresses and the number of guests increases, the variance decreases due to the law of large numbers. Comparing variables, revenue has the highest variance indicating that guest behaviour is non-deterministic. Money (i.e. the budget limiting possible actions) also has a relatively high-variance, showing that agents must have a robust policy that adapts to the outcomes that arise. Lastly, park value, the main evaluation metric, has a comparatively low variance, demonstrating that MAPS can reliably detect differences in performance. We note that order of the per-trajectory CVs is largely dictated by the inverse means. This is because the revenue's variance is passed on to money and park value, resulting in all three having a similar variance.

We next look at how current systems handle non-determinism. Section 4.5 highlights that directly using an LLM to predict stochastic outcomes (i.e., the Wall-E WM) is ineffective: MPC with these predictions degrades performance, is ∼8 times slower, and is ∼29 times more expensive than using the policy directly. However, an oracle WM boosts performance with a >4x improvement; under medium mode the oracle MPC even outperforms Sonnet-4.5 on average ($1.43_{\pm 1.40} \rightarrow 2.03_{\pm 0.90}$) at a lower cost ($12.19_{\pm 1.03} \rightarrow 8.09_{\pm 1.31}$), though speed remains an issue ($1{,}713_{\pm 86} \rightarrow 22{,}725_{\pm 3{,}715}$). Notably, these benefits occur with a mere 4-step look-ahead, whereas humans often plan much further ahead. These highlights that world modelling holds promise, but work addressing accurate stochastic modelling, inference speed, and search speed is required.

Agent	Score (%)	Runtime (s)	Cost (USD)
EASY DIFFICULTY			
GPT-5 Nano	$0.36_{\pm 0.28}$	$1{,}311_{\pm 305}$	$0.13_{\pm 0.02}$
GPT-5 Nano$_{\text{WALL-E}}$	$0.07_{\pm 0.08}$	$27{,}238_{\pm 3{,}524}$	$7.22_{\pm 0.61}$
GPT-5 Nano$_{\text{Oracle}}$	$1.53_{\pm 1.05}$	$10{,}787_{\pm 1{,}218}$	$3.78_{\pm 0.30}$
MEDIUM DIFFICULTY			
GPT-5 Nano	$0.64_{\pm 0.78}$	$2{,}506_{\pm 705}$	$0.27_{\pm 0.04}$
GPT-5 Nano$_{\text{Oracle}}$	$2.03_{\pm 0.90}$	$22{,}725_{\pm 3{,}715}$	$8.09_{\pm 1.31}$

Table 6. Wall-E training cost $2.47. World modelling is one proposal for improving long-term planning, but world modelling in stochastic environments such as MAPS is understudied. We find that WALL-E (Zhou et al., 2024) combined with model predictive control (MPC) fails to improve performance, but that an oracle world model yields a 3-5x improvement, even outperforming some frontier models on medium difficulty. Score (%) indicates the score relative to the human baseline, with 100.00% indicating parity.

5. Conclusion.

Real-world decision-making demands agents that can plan over long horizons, learn efficiently from limited interactions, reason about spatial and causal structure, and act robustly under uncertainty. Our results show that even the strongest contemporary AI systems fall far short of human performance when these capabilities must be integrated. By introducing MAPS—a scalable, realistic, and diagnostically rich benchmark—we provide a unified testbed for measuring progress on these challenges. We hope this environment, along with the human and model evaluations presented here, inspires new approaches for building agents that can reason, learn, and act with the flexibility and robustness required in complex domains.

References

Allen, K. R., Smith, K. A., and Tenenbaum, J. B. Rapid trial-and-error learning with simulation supports flexible tool use and physical reasoning. *Proceedings of the National Academy of Sciences of the United States of America*, 117(47):29302—29310, November 2020. ISSN 0027-8424. doi: 10.1073/pnas.1912341117. URL https://europepmc.org/articles/PMC7703630.

Anokhin, P., Khalikov, R., Rebrikov, S., Volkov, V., Sorokin, A., and Bissonnette, V. Herobench: A benchmark for long-horizon planning and structured reasoning in virtual worlds, 2025. URL https://arxiv.org/abs/2508.12782.

Anonymous. Reasoning under uncertainty: Exploring probabilistic reasoning capabilities of LLMs. In *Submitted to The Fourteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=VMLUFna6VI. under review.

Anthropic. Introducing claude sonnet 4.5. https://www.anthropic.com/news/claude-sonnet-4-5, September 2025.

Aroca-Ouellette, S., von der Wense, K., and Roncone, A. Re-Seeding latent states for sequential language understanding. In Christodoulopoulos, C., Chakraborty, T., Rose, C., and Peng, V. (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 25233–25247, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1281. URL https://aclanthology.org/2025.emnlp-main.1281/.

Backlund, A. and Petersson, L. Vending-bench: A benchmark for long-term coherence of autonomous agents, 2025. URL https://arxiv.org/abs/2502.15840.

Bai, M., Cohen, A. K., Koss, E., and Lichtenbaum, C. Stuck in the matrix: Probing spatial reasoning in large language models, 2025. URL https://arxiv.org/abs/2510.20198.

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, jun 2013.

Chen, Z., White, M., Mooney, R., Payani, A., Su, Y., and Sun, H. When is tree search useful for LLM planning? it depends on the discriminator. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13659–13678, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.738. URL https://aclanthology.org/2024.acl-long.738/.

Chevalier-Boisvert, M., Bahdanau, D., Lahlou, S., Willems, L., Saharia, C., Nguyen, T. H., and Bengio, Y. BabyAI: First steps towards grounded language learning with a human in the loop. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJeXCo0cYX.

Chollet, F. On the measure of intelligence, 2019. URL https://arxiv.org/abs/1911.01547.

Chollet, F., Knoop, M., Kamradt, G., Landers, B., and Pinkard, H. Arc-agi-2: A new challenge for frontier ai reasoning systems, 2025. URL https://arxiv.org/abs/2505.11831.

Collignon, N. *Self-directed learning in new and changing environments: understanding human algorithms for exploration*. PhD thesis, University of Edinburgh, Jul 25 2020. URL https://hdl.handle.net/1842/37293. PhD Thesis, Informatics, School of Informatics.

Curtis, A., Tang, H., Veloso, T., Ellis, K., Tenenbaum, J. B., Lozano-Pérez, T., and Kaelbling, L. P. Llm-guided probabilistic program induction for POMDP model estimation. *CoRR*, abs/2505.02216, 2025. doi: 10.48550/ARXIV.2505.02216. URL https://doi.org/10.48550/arXiv.2505.02216.

Dainese, N., Merler, M., Alakuijala, M., and Marttinen, P. Generating code world models with large language models guided by monte carlo tree search. In Globersons, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J. M., and Zhang, C. (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/6f479ea488e0908ac8b1b37b27fd134c-Abstract-Conference.html.

Fan, L., Wang, G., Jiang, Y., Mandlekar, A., Yang, Y., Zhu, H., Tang, A., Huang, D.-A., Zhu, Y., and Anandkumar, A. Minedojo: Building open-ended embodied agents with internet-scale knowledge. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. URL https://openreview.net/forum?id=rc8o_j8I8PX.

Gemini Team, Google DeepMind. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long

context, and next generation agentic capabilities. Technical Report arXiv:2507.06261, Google DeepMind, July 2025.

Gopnik, A. and Sobel, D. M. Detecting blickets: How young children use information about novel causal powers in categorization and induction. *Child Development*, 71(5): 1205–1222, 2000. doi: 10.1111/1467-8624.00224.

Gopnik, A., Meltzoff, A., and Kuhl, P. *The Scientist in the Crib: Minds, Brains, and How Children Learn*. Harper Collins, 1999.

Hafner, D. Benchmarking the spectrum of agent capabilities. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=1W0z96MFEoH.

Hao, S., Gu, Y., Ma, H., Hong, J., Wang, Z., Wang, D., and Hu, Z. Reasoning with language model is planning with world model. In Bouamor, H., Pino, J., and Bali, K. (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 8154–8173, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.507. URL https://aclanthology.org/2023.emnlp-main.507/.

Hao, Y., Gu, J., Wang, H. W., Li, L., Yang, Z., Wang, L., and Cheng, Y. Can MLLMs reason in multimodality? EMMA: An enhanced multimodal reasoning benchmark. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=v26vwjxOEz.

Jansen, P., Côté, M.-A., Khot, T., Bransom, E., Mishra, B. D., Majumder, B. P., Tafjord, O., and Clark, P. Discoveryworld: A virtual environment for developing and evaluating automated scientific discovery agents. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=cDYqckEt6d.

Katz, D. M., Bommarito, M. J., Gao, S., and Arredondo, P. D. Gpt-4 passes the bar exam. *SSRN*, 2023.

Khan, Z., Prasad, A., Stengel-Eskin, E., Cho, J., and Bansal, M. One life to learn: Inferring symbolic world models for stochastic environments from unguided exploration, 2025. URL https://arxiv.org/abs/2510.12088.

Küttler, H., Nardelli, N., Miller, A., Raileanu, R., Selvatici, M., Grefenstette, E., and Rocktäschel, T. The nethack learning environment. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 7671–7684. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/569ff987c643b4bedf504efda8f786c2-Paper.pdf.

Li, Y., Choi, D., Chung, J., et al. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097, 2022.

Liu, M., Okuhara, T., Chang, X., Shirabe, R., Nishiie, Y., Okada, H., and Kiuchi, T. Performance of chatgpt across different versions in medical licensing examinations worldwide: systematic review and meta-analysis. *Journal of medical Internet research*, 26:e60807, 2024a.

Liu, X., Wu, Z., Wu, X., Lu, P., Chang, K.-W., and Feng, Y. Are LLMs capable of data-based statistical and causal reasoning? benchmarking advanced quantitative reasoning with data. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 9215–9235, Bangkok, Thailand, August 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.548. URL https://aclanthology.org/2024.findings-acl.548/.

Ma, C., Zhao, H., Zhang, J., He, J., and Kong, L. Nonmyopic generation of language models for reasoning and planning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=OoNazl6T7D.

OpenAI. Introducing gpt-5. https://openai.com/index/introducing-gpt-5/, August 2025.

Phan, L., Mazeika, M., Zou, A., and Hendrycks, D. Textquests: How good are llms at text-based video games?, 2025. URL https://arxiv.org/abs/2507.23701.

Piriyakulkij, W. T., Liang, Y., Tang, H., Weller, A., Kryven, M., and Ellis, K. Poe-world: Compositional world modeling with products of programmatic experts. *CoRR*, abs/2505.10819, 2025. doi: 10.48550/ARXIV.2505.10819. URL https://doi.org/10.48550/arXiv.2505.10819.

Reiter, R., Hoffmann, J., Reinhardt, D., Messerer, F., Baumgärtner, K., Sawant, S., Boedecker, J., Diehl, M., and Gros, S. Synthesis of model predictive control and reinforcement learning: Survey and classification. *CoRR*, abs/2502.02133, 2025. doi: 10.48550/ARXIV.2502.02133. URL https://doi.org/10.48550/arXiv.2502.02133.

Shridhar, M., Yuan, X., Cote, M.-A., Bisk, Y., Trischler, A., and Hausknecht, M. {ALFW}orld: Aligning text

and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=0IOX0YcCdTn.

Silver, D., Schrittwieser, J., Simonyan, K., et al. Mastering the game of go without human knowledge. *Nature*, 550 (7676):354–359, 2017.

Tang, H., Key, D. Y., and Ellis, K. Worldcoder, a model-based LLM agent: Building world models by writing code and interacting with the environment. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=QGJSXMhVaL.

Tenenbaum, J. B., Kemp, C., Griffiths, T. L., and Goodman, N. D. How to grow a mind: Statistics, structure, and abstraction. *Science*, 331(6022):1279–1285, 2011. doi: 10.1126/science.1192788. URL https://www.science.org/doi/abs/10.1126/science.1192788.

Valmeekam, K., Marquez, M., Sreedharan, S., and Kambhampati, S. On the planning abilities of large language models - a critical investigation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=X6dEqXIsEW.

Wang, J. X., King, M., Porcel, N. P. M., Kurth-Nelson, Z., Zhu, T., Deck, C., Choy, P., Cassin, M., Reynolds, M., Song, H. F., Buttimore, G., Reichert, D. P., Rabinowitz, N. C., Matthey, L., Hassabis, D., Lerchner, A., and Botvinick, M. Alchemy: A benchmark and analysis toolkit for meta-reinforcement learning agents. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=eZu4BZxlRnX.

Wang, R., Jansen, P., Côté, M.-A., and Ammanabrolu, P. ScienceWorld: Is your agent smarter than a 5th grader? In Goldberg, Y., Kozareva, Z., and Zhang, Y. (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11279–11298, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.775. URL https://aclanthology.org/2022.emnlp-main.775/.

Warrier, A., Nguyen, D., Naim, M., Jain, M., Liang, Y., Schroeder, K., Yang, C., Tenenbaum, J. B., Vollmer, S., Ellis, K., and Tavares, Z. Benchmarking world-model learning, 2025. URL https://arxiv.org/abs/2510.19788.

xAI. Grok 4 model card. Technical report, xAI, August 2025.

Xie, J., Zhang, K., Chen, J., Yuan, S., Zhang, K., Zhang, Y., Li, L., and Xiao, Y. Revealing the barriers of language agents in planning. In Chiruzzo, L., Ritter, A., and Wang, L. (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1872–1888, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.93. URL https://aclanthology.org/2025.naacl-long.93/.

Xie, T., Zhang, D., Chen, J., Li, X., Zhao, S., Cao, R., Hua, T. J., Cheng, Z., Shin, D., Lei, F., Liu, Y., Xu, Y., Zhou, S., Savarese, S., Xiong, C., Zhong, V., and Yu, T. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=tN61DTr4Ed.

Xu, M., Jiang, G., Liang, W., Zhang, C., and Zhu, Y. Interactive visual reasoning under uncertainty. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S. (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 42409–42432. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/844f722dbbcb27933ff5baf58a1f00c8-Paper-Datasets_and_Benchmarks.pdf.

Yao, S., Chen, H., Yang, J., and Narasimhan, K. R. Webshop: Towards scalable real-world web interaction with grounded language agents. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=R9KnuFlvnU.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., and Narasimhan, K. R. Tree of thoughts: Deliberate problem solving with large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a. URL https://openreview.net/forum?id=5Xc1ecxO1h.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K. R., and Cao, Y. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=WE_vluYUL-X.

Ying, L., Collins, K. M., Sharma, P., Colas, C., Zhao, K. I., Weller, A., Tavares, Z., Isola, P., Gershman, S. J., An-

dreas, J. D., Griffiths, T. L., Chollet, F., Allen, K. R., and Tenenbaum, J. B. Assessing adaptive world models in machines with novel games, 2025. URL https://arxiv.org/abs/2507.12821.

Zhao, Z., Lee, W. S., and Hsu, D. Large language models as commonsense knowledge for large-scale task planning. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S. (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/65a39213d7d0e1eb5d192aa77e77eeb7-Abstract-Conference.html.

Zhou, S., Xu, F. F., Zhu, H., Zhou, X., Lo, R., Sridhar, A., Cheng, X., Bisk, Y., Fried, D., Alon, U., et al. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*, 2023.

Zhou, S., Zhou, T., Yang, Y., Long, G., Ye, D., Jiang, J., and Zhang, C. Wall-e: World alignment by rule learning improves world model-based llm agents. *arXiv preprint arXiv:2410.07484*, 2024.

Zhou, S., Zhou, T., Yang, Y., Long, G., Ye, D., Jiang, J., and Zhang, C. WALL-E 2.0: World alignment by neurosymbolic learning improves world model-based LLM agents. *CoRR*, abs/2504.15785, 2025. doi: 10.48550/ARXIV.2504.15785. URL https://doi.org/10.48550/arXiv.2504.15785.

A. Extended Related Works

A.1. Task Completion Benchmarks

Another set is crafting and survival-based environments such as MineDojo (Fan et al., 2022), HeroBench (Anokhin et al., 2025). All evaluate long-horizon, dependency-driven reasoning and require substantial game-specific knowledge to be successful. However, MineDojo and HeroBench are both deterministic and satisfaction based tasks. Only MineDojo involves spatial reasoning, and neither maps to real world tasks.

A.2. Optimization Benchmarks

The Atari Learning Environment (ALE) (Bellemare et al., 2013) has long served as a canonical human benchmark in AI. While impactful historically, it is now saturated, deterministic, and far removed from real-world domains.

Instruction-following benchmarks such as BabyAI (Chevalier-Boisvert et al., 2019) and AlfWorld (Shridhar et al., 2021) test navigation and object manipulation via natural language. BabyAI is grid-based, whereas AlfWorld simulates a household with common objects. These environments model tasks closer to the real world and are inherently multi-modal, yet still remain satisfaction-oriented in deterministic settings.

Lastly, a unique benchmark is TextQuests (Phan et al., 2025). TextQuests probes LLMs in a zero-shot manner through interactive text-based adventures. It features open-ended action spaces and long horizons, but does not require learning on the fly, and the tasks are fully text-based and deterministic.

B. Implementation Details

B.1. Evaluation Protocol

Unless otherwise noted, all methods have full access to the games documentation, which can be found in Section K. Note that the section of the documentation on sandbox actions is omitted when the agent is not in sandbox mode.

In addition to the documentation, we also perform experiments where the agent is allowed limited access to a sandbox learning mode prior to evaluation. In sandbox mode, the agents are given a budget of 100 standard Mini Amusement Parks action, and access to a number of sandbox action. These sandbox actions include maximizing money, changing train layout, unlocking research etc... The full list of actions is outlined in the sandbox documentation reproduced in Section K. To prevent infinite loops we impose a limit of 250 sandbox actions – past this point sandbox actions are also deducted from the action budget. None of the agents tested hit this limit.

We evaluated all our agents on 3 trajectories per testing layout. We reported the mean and standard deviation of the final park value (cash on hand, plus the value of physical assets and intellectual property – see the game documentation in Section K for details).

B.2. AI System Evaluation

To build a world model for MAPS we adapt Walle 1.0 (Zhou et al., 2024). We modify all the original prompts to incorporate our new state and action specifications. Both the state and action specifications are represented using JSON objects. We further modify the prompt for predicting the next state given the action to incorporate and provide and examples of state transitions. For few shot examples, we associate transitions from the training data with an embedding of the action string, then at inference time we retrieve the top $k = 5$ transitions based on the current action. For rule generation we follow (Zhou et al., 2024). We adopt two of the simplifications introduced in (Zhou et al., 2025), namely going directly from generated natural language rules to generating code (i.e., eliminating the intermediate revision phase), and removing the ability of rules to invoke an LLM. The symbolic scenegraph introduced by (Zhou et al., 2025) is effectively provided by our state representation.

For game playing experiments, and after conferring with the authors of WallE to best align with their set up, we combined the WallE world model with a ReACT (Yao et al., 2023b) agent using a Model Predicitive Control mechanism (MPC), specifically random shooting (Reiter et al., 2025) . I.e., actions are selected by selecting the best of k rollouts guided by a world model. We select $k = 5$ independent rollouts and roll out for 4 steps, re-sampling the action if the world-model predicts that the action is invalid (e.g., an invalid value for price, or attempting to place a building at coordinates that are

outside of the bounds of the park). The ReAct history of the best action is is then used to initialize the history of the next k rollouts.

C. Learning from the Documentation & the Sandbox.

Agent	LLM	Score (%)	Runtime (s)	Cost (USD)	Train Cost
		EASY DIFFICULTY			
ReAct	GPT-5 Nano	$0.36_{\pm 0.28}$	$1{,}311_{\pm 305}$	$0.13_{\pm 0.02}$	\$0.00
ReAct+SB	GPT-5 Nano	$0.15_{\pm 0.18}$	$1{,}355_{\pm 336}$	$0.14_{\pm 0.02}$	\$0.31
ReAct-ND	GPT-5 Nano	$0.17_{\pm 0.15}$	$1{,}231_{\pm 238}$	$0.12_{\pm 0.01}$	\$0.00
ReAct+SB-ND	GPT-5 Nano	$0.12_{\pm 0.08}$	$1{,}369_{\pm 358}$	$0.13_{\pm 0.04}$	\$0.35
ReAct	GPT-5	$13.89_{\pm 11.16}$	$3{,}774_{\pm 505}$	$3.75_{\pm 0.32}$	\$0.00
ReAct+SB	GPT-5	$15.47_{\pm 13.35}$	$3{,}655_{\pm 346}$	$4.16_{\pm 0.22}$	\$10.26
ReAct-ND	GPT-5	$3.69_{\pm 4.24}$	$3{,}803_{\pm 288}$	$3.44_{\pm 0.28}$	\$0.00
ReAct+SB-ND	GPT-5	$7.78_{\pm 8.87}$	$3{,}225_{\pm 245}$	$3.87_{\pm 0.33}$	\$7.86
ReAct	Grok 4	$6.62_{\pm 14.50}$	$2{,}908_{\pm 437}$	$6.60_{\pm 0.63}$	\$0.00
ReAct+SB	Grok 4	$2.66_{\pm 2.86}$	$2{,}187_{\pm 199}$	$7.38_{\pm 0.51}$	\$18.23
ReAct	Sonnet 4.5	$4.53_{\pm 5.75}$	$937_{\pm 124}$	$6.06_{\pm 0.60}$	\$0.00
ReAct+SB	Sonnet 4.5	$0.83_{\pm 0.84}$	$879_{\pm 51}$	$7.31_{\pm 0.55}$	\$16.06
ReAct	Gemini 2.5 Pro	$3.96_{\pm 8.23}$	$787_{\pm 84}$	$3.06_{\pm 0.30}$	\$0.00
ReAct+SB	Gemini 2.5 Pro	$1.21_{\pm 1.47}$	$697_{\pm 97}$	$3.21_{\pm 0.29}$	\$6.22
		MEDIUM DIFFICULTY			
ReAct	GPT-5 Nano	$0.64_{\pm 0.78}$	$2{,}506_{\pm 705}$	$0.27_{\pm 0.04}$	\$0.00
ReAct+SB	GPT-5 Nano	$0.11_{\pm 0.12}$	$2{,}568_{\pm 483}$	$0.29_{\pm 0.03}$	\$0.35
ReAct-ND	GPT-5 Nano	$0.09_{\pm 0.07}$	$2{,}345_{\pm 803}$	$0.22_{\pm 0.08}$	\$0.00
ReAct+SB-ND	GPT-5 Nano	$0.19_{\pm 0.23}$	$2{,}532_{\pm 440}$	$0.27_{\pm 0.02}$	\$0.30
ReAct	GPT-5	$7.16_{\pm 8.36}$	$7{,}583_{\pm 764}$	$8.36_{\pm 0.48}$	\$0.00
ReAct+SB	GPT-5	$10.21_{\pm 10.07}$	$7{,}897_{\pm 822}$	$9.16_{\pm 0.58}$	\$9.82
ReAct-ND	GPT-5	$5.57_{\pm 7.41}$	$7{,}797_{\pm 674}$	$7.67_{\pm 0.75}$	\$0.00
ReAct+SB-ND	GPT-5	$1.36_{\pm 2.32}$	$5{,}960_{\pm 716}$	$7.15_{\pm 0.73}$	\$7.90
ReAct	Grok 4	$0.62_{\pm 0.39}$	$5{,}438_{\pm 683}$	$13.00_{\pm 0.86}$	\$0.00
ReAct+SB	Grok 4	$1.24_{\pm 1.14}$	$5{,}234_{\pm 526}$	$15.08_{\pm 1.05}$	\$16.92
ReAct	Sonnet 4.5	$1.43_{\pm 1.40}$	$1{,}713_{\pm 86}$	$12.19_{\pm 1.03}$	\$0.00
ReAct+SB	Sonnet 4.5	$0.88_{\pm 0.72}$	$1{,}808_{\pm 221}$	$14.68_{\pm 0.90}$	\$16.82
ReAct	Gemini 2.5 Pro	$0.64_{\pm 0.29}$	$1{,}423_{\pm 126}$	$5.97_{\pm 0.45}$	\$0.00
ReAct+SB	Gemini 2.5 Pro	$0.60_{\pm 0.64}$	$1{,}208_{\pm 124}$	$6.53_{\pm 0.46}$	\$7.36

Table 7. Results comparing models that were and were not provided the sandbox mode to learn in as well as the models that were and were not provided with the game's documentation. Score (%) indicates the score relative to the human baseline, with 100.00% indicating matching human performance. +SB: includes learning in sandbox mode. -ND: model is not provided with the game's documentation

Table 7 shows the results of the models under the standard set up (which includes the documentation, but not the sandbox learning) as well as models without the documentation (-ND), and/or with the sandbox learning experience (+SB). While there is a strong trend that reasoning models are able to effectively leverage the documentation to improve gameplay, the results are more mixed on the benefits of sandbox learning. Notably, GPT5 appears to be the only model that seems capable of generating useful learnings during sandbox mode, and is only able to do so on the easy difficulty. This aligns with our qualitative inspection of the learnings generated. Interestingly, while the learnings are only slighlty beneficial when documentation is provided, they provide substantial benefits when documentation is not provided. We hypothesize that when in sandbox mode without the documentation, it is able to learn some of the valuable information in the documentation from the interactions. From human experience, this sandbox mode is incredibly useful, and we believe exploring how to effectively use these interactions is an exciting avenue for future work.

D. Future Work

We are actively developing MAPS to improve its use as an ongoing testbed for world modelling and human-AI benchmarking. To this end, we are developing a hard difficulty which will boast a horizon of 250 steps and feature at least three additional mechanics including: terraforming, guest preferences, and debt. Lastly, we will be expanding the observation options to include both a grid-vector based representation suitable for more traditional reinforcement learning, as well as an image-text based observation that is suitable for VLMS. Using these observations, we will benchmark an even larger set of existing AI systems.

E. Detailed Model Evaluations

The absolute scores by human and AI systems are outlined in Tables 8 through 10, for each of the 3 test layouts respectively.

Agent	LLM	Score	Runtime (s)	Cost (USD)	Train Cost
		EASY DIFFICULTY			
ReAct	GPT-5 Nano	$2{,}297_{\pm 1{,}887}$	$1{,}276_{\pm 425}$	$0.14_{\pm 0.03}$	$0.00
ReAct-MPC$_{\text{WALL-E}}$	GPT-5 Nano	$397_{\pm 89}$	$27{,}939_{\pm 2{,}186}$	$7.54_{\pm 0.56}$	$2.47
ReAct-MPC$_{\text{Oracle}}$	GPT-5 Nano	$18{,}142_{\pm 11{,}333}$	$9{,}755_{\pm 877}$	$3.77_{\pm 0.20}$	$0.00
ReAct+SB	GPT-5 Nano	$1{,}751_{\pm 2{,}454}$	$1{,}329_{\pm 485}$	$0.15_{\pm 0.03}$	$0.31
ReAct-ND	GPT-5 Nano	$1{,}361_{\pm 1{,}815}$	$1{,}105_{\pm 241}$	$0.12_{\pm 0.01}$	$0.00
ReAct+SB-ND	GPT-5 Nano	$619_{\pm 541}$	$1{,}083_{\pm 536}$	$0.11_{\pm 0.07}$	$0.35
ReAct+SpatialHeur	GPT-5 Nano	$5{,}469_{\pm 7{,}752}$	$1{,}352_{\pm 297}$	$0.15_{\pm 0.02}$	$0.00
ReAct	GPT-5	$154{,}456_{\pm 82{,}848}$	$3{,}634_{\pm 113}$	$4.12_{\pm 0.13}$	$0.00
ReAct+SB	GPT-5	$209{,}816_{\pm 82{,}785}$	$3{,}838_{\pm 531}$	$4.42_{\pm 0.13}$	$10.26
ReAct-ND	GPT-5	$18{,}646_{\pm 9{,}141}$	$4{,}051_{\pm 370}$	$3.78_{\pm 0.21}$	$0.00
ReAct+SB-ND	GPT-5	$44{,}729_{\pm 22{,}838}$	$3{,}484_{\pm 186}$	$4.29_{\pm 0.07}$	$7.86
ReAct+SpatialHeur	GPT-5	$181{,}436_{\pm 62{,}948}$	$5{,}330_{\pm 437}$	$4.21_{\pm 0.15}$	$0.00
ReAct	Grok 4	$122{,}871_{\pm 190{,}128}$	$3{,}111_{\pm 352}$	$7.39_{\pm 0.27}$	$0.00
ReAct+SB	Grok 4	$41{,}353_{\pm 28{,}268}$	$2{,}110_{\pm 231}$	$8.02_{\pm 0.25}$	$18.23
ReAct	Sonnet 4.5	$72{,}763_{\pm 61{,}477}$	$946_{\pm 141}$	$6.77_{\pm 0.41}$	$0.00
ReAct+SB	Sonnet 4.5	$12{,}582_{\pm 6{,}387}$	$886_{\pm 24}$	$8.03_{\pm 0.10}$	$16.06
ReAct+SpatialHeur	Sonnet 4.5	$101{,}229_{\pm 83{,}995}$	$1{,}048_{\pm 207}$	$6.92_{\pm 0.43}$	$0.00
ReAct	Gemini 2.5 Pro	$70{,}543_{\pm 107{,}185}$	$782_{\pm 89}$	$3.35_{\pm 0.23}$	$0.00
ReAct+SB	Gemini 2.5 Pro	$18{,}567_{\pm 15{,}783}$	$751_{\pm 13}$	$3.53_{\pm 0.06}$	$6.22
Human		760,954			
		MEDIUM DIFFICULTY			
ReAct	GPT-5 Nano	$4{,}367_{\pm 6{,}992}$	$2{,}040_{\pm 854}$	$0.26_{\pm 0.06}$	$0.00
ReAct-MPC$_{\text{Oracle}}$	GPT-5 Nano	$93{,}305_{\pm 37{,}018}$	$24{,}186_{\pm 5{,}739}$	$9.20_{\pm 1.45}$	$0.00
ReAct+SB	GPT-5 Nano	$733_{\pm 801}$	$2{,}184_{\pm 626}$	$0.29_{\pm 0.04}$	$0.35
ReAct-ND	GPT-5 Nano	$3{,}895_{\pm 3{,}406}$	$1{,}938_{\pm 1{,}401}$	$0.20_{\pm 0.15}$	$0.00
ReAct+SB-ND	GPT-5 Nano	$3{,}576_{\pm 3{,}499}$	$2{,}445_{\pm 510}$	$0.29_{\pm 0.02}$	$0.30
ReAct	GPT-5	$79{,}194_{\pm 25{,}890}$	$7{,}117_{\pm 278}$	$8.91_{\pm 0.14}$	$0.00
ReAct+SB	GPT-5	$173{,}657_{\pm 123{,}554}$	$8{,}385_{\pm 554}$	$9.82_{\pm 0.32}$	$9.82
ReAct-ND	GPT-5	$57{,}091_{\pm 23{,}314}$	$7{,}834_{\pm 472}$	$8.12_{\pm 0.24}$	$0.00
ReAct+SB-ND	GPT-5	$11{,}114_{\pm 6{,}942}$	$6{,}442_{\pm 610}$	$7.92_{\pm 0.36}$	$7.90
ReAct	Grok 4	$20{,}511_{\pm 5{,}383}$	$5{,}024_{\pm 701}$	$14.01_{\pm 0.36}$	$0.00
ReAct+SB	Grok 4	$15{,}225_{\pm 5{,}083}$	$5{,}533_{\pm 283}$	$16.44_{\pm 0.22}$	$16.92
ReAct	Sonnet 4.5	$31{,}265_{\pm 6{,}050}$	$1{,}759_{\pm 75}$	$13.27_{\pm 0.36}$	$0.00
ReAct+SB	Sonnet 4.5	$25{,}384_{\pm 24{,}292}$	$1{,}710_{\pm 89}$	$15.73_{\pm 0.05}$	$16.82
ReAct	Gemini 2.5 Pro	$23{,}554_{\pm 9{,}057}$	$1{,}328_{\pm 140}$	$6.55_{\pm 0.12}$	$0.00
ReAct+SB	Gemini 2.5 Pro	$8{,}598_{\pm 2{,}834}$	$1{,}258_{\pm 119}$	$6.98_{\pm 0.04}$	$7.36
Human		4,262,963			

Table 8. TODO UPDATE HUMAN Full experiment results with absolute scores for the test layout `the_islands`. Average and standard deviation over 3 trials.

Agent	LLM	Score	Runtime (s)	Cost (USD)	Train Cost
		EASY DIFFICULTY			
ReAct	GPT-5 Nano	$2{,}068_{\pm 1{,}194}$	$1{,}329_{\pm 388}$	$0.13_{\pm 0.02}$	$0.00
ReAct-MPC$_{\text{WALL-E}}$	GPT-5 Nano	$373_{\pm 48}$	$24{,}620_{\pm 1{,}719}$	$6.82_{\pm 0.18}$	$2.47
ReAct-MPC$_{\text{Oracle}}$	GPT-5 Nano	$8{,}910_{\pm 1{,}056}$	$11{,}291_{\pm 826}$	$3.65_{\pm 0.14}$	$0.00
ReAct+SB	GPT-5 Nano	$1{,}106_{\pm 994}$	$1{,}509_{\pm 347}$	$0.15_{\pm 0.02}$	$0.31
ReAct-ND	GPT-5 Nano	$2{,}017_{\pm 2{,}108}$	$1{,}191_{\pm 66}$	$0.11_{\pm 0.01}$	$0.00
ReAct+SB-ND	GPT-5 Nano	$1{,}208_{\pm 469}$	$1{,}615_{\pm 27}$	$0.15_{\pm 0.00}$	$0.35
ReAct+SpatialHeur	GPT-5 Nano	$21{,}582_{\pm 19{,}134}$	$1{,}540_{\pm 179}$	$0.15_{\pm 0.01}$	$0.00
ReAct	GPT-5	$168{,}727_{\pm 117{,}429}$	$4{,}281_{\pm 617}$	$3.63_{\pm 0.09}$	$0.00
ReAct+SB	GPT-5	$169{,}018_{\pm 177{,}938}$	$3{,}567_{\pm 299}$	$4.11_{\pm 0.03}$	$10.26
ReAct-ND	GPT-5	$91{,}328_{\pm 81{,}325}$	$3{,}726_{\pm 178}$	$3.32_{\pm 0.03}$	$0.00
ReAct+SB-ND	GPT-5	$190{,}074_{\pm 167{,}726}$	$3{,}158_{\pm 142}$	$3.73_{\pm 0.13}$	$7.86
ReAct+SpatialHeur	GPT-5	$445{,}397_{\pm 105{,}250}$	$5{,}303_{\pm 903}$	$3.82_{\pm 0.18}$	$0.00
ReAct	Grok 4	$35{,}031_{\pm 38{,}810}$	$2{,}462_{\pm 183}$	$6.30_{\pm 0.11}$	$0.00
ReAct+SB	Grok 4	$12{,}786_{\pm 6{,}894}$	$2{,}328_{\pm 144}$	$7.20_{\pm 0.09}$	$18.23
ReAct	Sonnet 4.5	$20{,}083_{\pm 31{,}576}$	$849_{\pm 20}$	$5.58_{\pm 0.13}$	$0.00
ReAct+SB	Sonnet 4.5	$2{,}304_{\pm 802}$	$833_{\pm 49}$	$6.87_{\pm 0.05}$	$16.06
ReAct+SpatialHeur	Sonnet 4.5	$121{,}672_{\pm 67{,}600}$	$924_{\pm 53}$	$5.99_{\pm 0.32}$	$0.00
ReAct	Gemini 2.5 Pro	$3{,}948_{\pm 1{,}419}$	$818_{\pm 85}$	$2.86_{\pm 0.11}$	$0.00
ReAct+SB	Gemini 2.5 Pro	$6{,}823_{\pm 6{,}700}$	$714_{\pm 56}$	$3.12_{\pm 0.02}$	$6.22
Human		1,263,302			
		MEDIUM DIFFICULTY			
ReAct	GPT-5 Nano	$12{,}966_{\pm 11{,}081}$	$2{,}539_{\pm 778}$	$0.27_{\pm 0.06}$	$0.00
ReAct-MPC$_{\text{Oracle}}$	GPT-5 Nano	$27{,}960_{\pm 8{,}460}$	$20{,}261_{\pm 1{,}631}$	$7.13_{\pm 0.45}$	$0.00
ReAct+SB	GPT-5 Nano	$1{,}733_{\pm 1{,}862}$	$2{,}848_{\pm 386}$	$0.29_{\pm 0.03}$	$0.35
ReAct-ND	GPT-5 Nano	$507_{\pm 200}$	$2{,}504_{\pm 348}$	$0.23_{\pm 0.02}$	$0.00
ReAct+SB-ND	GPT-5 Nano	$992_{\pm 205}$	$2{,}495_{\pm 510}$	$0.26_{\pm 0.02}$	$0.30
ReAct	GPT-5	$177{,}940_{\pm 74{,}558}$	$8{,}420_{\pm 722}$	$8.28_{\pm 0.22}$	$0.00
ReAct+SB	GPT-5	$240{,}658_{\pm 27{,}591}$	$6{,}901_{\pm 50}$	$8.87_{\pm 0.13}$	$9.82
ReAct-ND	GPT-5	$134{,}125_{\pm 101{,}480}$	$8{,}411_{\pm 452}$	$8.06_{\pm 0.65}$	$0.00
ReAct+SB-ND	GPT-5	$38{,}507_{\pm 31{,}171}$	$6{,}317_{\pm 248}$	$7.22_{\pm 0.22}$	$7.90
ReAct	Grok 4	$11{,}523_{\pm 1{,}717}$	$5{,}252_{\pm 425}$	$12.80_{\pm 0.35}$	$0.00
ReAct+SB	Grok 4	$25{,}009_{\pm 13{,}157}$	$5{,}477_{\pm 350}$	$14.57_{\pm 0.26}$	$16.92
ReAct	Sonnet 4.5	$24{,}141_{\pm 23{,}098}$	$1{,}660_{\pm 74}$	$11.56_{\pm 0.93}$	$0.00
ReAct+SB	Sonnet 4.5	$11{,}254_{\pm 12{,}670}$	$1{,}858_{\pm 402}$	$13.88_{\pm 0.71}$	$16.82
ReAct	Gemini 2.5 Pro	$7{,}995_{\pm 4{,}373}$	$1{,}454_{\pm 111}$	$5.58_{\pm 0.10}$	$0.00
ReAct+SB	Gemini 2.5 Pro	$11{,}641_{\pm 6{,}282}$	$1{,}213_{\pm 53}$	$6.41_{\pm 0.03}$	$7.36
Human		1,037,310			

Table 9. TODO UPDATE HUMAN AND ADD WALLE Full experiment results with absolute scores for the test layout `ribs`. Average and standard deviation over 3 trials.

Agent	LLM	Score	Runtime (s)	Cost (USD)	Train Cost
		EASY DIFFICULTY			
ReAct	GPT-5 Nano	$2{,}960_{\pm 1{,}381}$	$1{,}328_{\pm 193}$	$0.13_{\pm 0.01}$	$0.00
ReAct-MPC$_{\text{WALL-E}}$	GPT-5 Nano	$681_{\pm 551}$	$29{,}156_{\pm 5{,}042}$	$7.29_{\pm 0.85}$	$2.47
ReAct-MPC$_{\text{Oracle}}$	GPT-5 Nano	$7{,}295_{\pm 1{,}023}$	$11{,}314_{\pm 1{,}444}$	$3.90_{\pm 0.51}$	$0.00
ReAct+SB	GPT-5 Nano	$676_{\pm 486}$	$1{,}227_{\pm 190}$	$0.13_{\pm 0.01}$	$0.31
ReAct-ND	GPT-5 Nano	$774_{\pm 273}$	$1{,}395_{\pm 313}$	$0.11_{\pm 0.01}$	$0.00
ReAct+SB-ND	GPT-5 Nano	$829_{\pm 570}$	$1{,}410_{\pm 92}$	$0.13_{\pm 0.00}$	$0.35
ReAct+SpatialHeur	GPT-5 Nano	$2{,}147_{\pm 1{,}399}$	$1{,}226_{\pm 159}$	$0.13_{\pm 0.01}$	$0.00
ReAct	GPT-5	$38{,}655_{\pm 64{,}599}$	$3{,}407_{\pm 107}$	$3.50_{\pm 0.25}$	$0.00
ReAct+SB	GPT-5	$26{,}298_{\pm 21{,}567}$	$3{,}558_{\pm 180}$	$3.96_{\pm 0.13}$	$10.26
ReAct-ND	GPT-5	$6{,}672_{\pm 303}$	$3{,}632_{\pm 137}$	$3.23_{\pm 0.01}$	$0.00
ReAct+SB-ND	GPT-5	$11{,}667_{\pm 6{,}144}$	$3{,}032_{\pm 151}$	$3.58_{\pm 0.04}$	$7.86
ReAct+SpatialHeur	GPT-5	$26{,}673_{\pm 14{,}533}$	$4{,}970_{\pm 325}$	$3.62_{\pm 0.16}$	$0.00
ReAct	Grok 4	$4{,}506_{\pm 4{,}142}$	$3{,}150_{\pm 399}$	$6.10_{\pm 0.20}$	$0.00
ReAct+SB	Grok 4	$7{,}325_{\pm 5{,}060}$	$2{,}123_{\pm 199}$	$6.92_{\pm 0.11}$	$18.23
ReAct	Sonnet 4.5	$11{,}738_{\pm 8{,}585}$	$1{,}018_{\pm 141}$	$5.84_{\pm 0.27}$	$0.00
ReAct+SB	Sonnet 4.5	$3{,}089_{\pm 3{,}191}$	$917_{\pm 45}$	$7.02_{\pm 0.11}$	$16.06
ReAct+SpatialHeur	Sonnet 4.5	$10{,}832_{\pm 9{,}517}$	$934_{\pm 61}$	$5.87_{\pm 0.19}$	$0.00
ReAct	Gemini 2.5 Pro	$11{,}057_{\pm 12{,}005}$	$760_{\pm 104}$	$2.97_{\pm 0.29}$	$0.00
ReAct+SB	Gemini 2.5 Pro	$3{,}159_{\pm 3{,}920}$	$625_{\pm 147}$	$2.99_{\pm 0.28}$	$6.22
Human		481,700			
		MEDIUM DIFFICULTY			
ReAct	GPT-5 Nano	$5{,}048_{\pm 4{,}624}$	$2{,}939_{\pm 214}$	$0.28_{\pm 0.02}$	$0.00
ReAct-MPC$_{\text{Oracle}}$	GPT-5 Nano	$10{,}822_{\pm 3{,}233}$	$23{,}727_{\pm 2{,}405}$	$7.92_{\pm 1.14}$	$0.00
ReAct+SB	GPT-5 Nano	$1{,}413_{\pm 340}$	$2{,}672_{\pm 190}$	$0.29_{\pm 0.02}$	$0.35
ReAct-ND	GPT-5 Nano	$1{,}228_{\pm 712}$	$2{,}594_{\pm 338}$	$0.24_{\pm 0.03}$	$0.00
ReAct+SB-ND	GPT-5 Nano	$3{,}345_{\pm 3{,}011}$	$2{,}658_{\pm 467}$	$0.26_{\pm 0.01}$	$0.30
ReAct	GPT-5	$21{,}963_{\pm 15{,}128}$	$7{,}211_{\pm 390}$	$7.88_{\pm 0.22}$	$0.00
ReAct+SB	GPT-5	$29{,}821_{\pm 28{,}585}$	$8{,}406_{\pm 395}$	$8.77_{\pm 0.48}$	$9.82
ReAct-ND	GPT-5	$21{,}685_{\pm 7{,}294}$	$7{,}145_{\pm 432}$	$6.82_{\pm 0.37}$	$0.00
ReAct+SB-ND	GPT-5	$1{,}021_{\pm 225}$	$5{,}122_{\pm 158}$	$6.31_{\pm 0.06}$	$7.90
ReAct	Grok 4	$2{,}379_{\pm 590}$	$6{,}038_{\pm 587}$	$12.17_{\pm 0.24}$	$0.00
ReAct+SB	Grok 4	$8{,}354_{\pm 4{,}044}$	$4{,}692_{\pm 490}$	$14.21_{\pm 0.19}$	$16.92
ReAct	Sonnet 4.5	$10{,}868_{\pm 8{,}456}$	$1{,}720_{\pm 106}$	$11.73_{\pm 0.77}$	$0.00
ReAct+SB	Sonnet 4.5	$8{,}458_{\pm 1{,}840}$	$1{,}857_{\pm 67}$	$14.42_{\pm 0.12}$	$16.82
ReAct	Gemini 2.5 Pro	$5{,}344_{\pm 2{,}266}$	$1{,}486_{\pm 104}$	$5.80_{\pm 0.12}$	$0.00
ReAct+SB	Gemini 2.5 Pro	$4{,}095_{\pm 6{,}773}$	$1{,}154_{\pm 189}$	$6.21_{\pm 0.60}$	$7.36
Human		887,772			

Table 10. TODO: UPDATE HUMAN Full experiment results with absolute scores for the test layout `zig_zag`. Average and standard deviation over 3 trials.

F. Qualitative Analysis of Learnings

Coalescing the analysis in Table 12, we see that GPT-5 is able to identify some simple, but useful trends: janitors and mechanics are important early, park rating can lag a day, and stocking should be considered carefully. However, most of these can be directly inferred from the documentation with some thought. Further, the useful trends are surrounded by very specific information about that specific day and the learnings are also not particularly actionable beyond their most straightforward application. Additionally, when encountering more complicated relationships, it over-generalizes and does a poor job of untangling causal effects. Instead it presumes that its action is the main driver of all change. While it does explore different components of the game, from yellow carousels to red food shops, but spends a lot of time testing high-level

Table 11. GPT-5 Learnings

Step	Learning	Notes
4	A single yellow ferris_wheel priced near its max ($4) attracted guests and generated revenue, and the ride and adjacent path accumulated dirt even without breakdowns—indicating that rides quickly impact cleanliness and likely benefit from early janitor staffing.	A simple, but useful observation
10	Hiring a blue mechanic restored ride uptime to near full and minimized out-of-service time, indicating mechanics quickly recover breakdowns and sustain revenue; rating impacts can lag a day due to start-of-day calculation.	Identifies lag in park rating bump, however everything here is already noted in the game documentation.
11	Park rating improved the day after uptime stabilized (all rides at 1.0) following hiring a mechanic, confirming rating effects can lag and that sustained uptime materially lifts rating and revenue.	Confirms the above, but again no new information garnered.
19	After adding a high-capacity ride, drink demand exceeded 300/day and the yellow drink shop still stocked out, while increasing the yellow food shop's order_quantity to 120 eliminated stockouts but left 23 units unsold—confirming that zero-cost drink shops should be overstocked aggressively, whereas food shops with item costs should be tuned close to observed demand to minimize waste.	Useful to note that the zero-cost drink shops should be overstocked, but is surrounded by detail that is much too specific that specific day.
43	After adding distributed drink coverage and letting a full day pass, drink visits per guest rose (1.36), but higher footfall strained cleanliness (ride/-path cleanliness dipped to 0.82–0.98) and uptime on high-breakdown rides sagged—showing that as capacity and traffic grow, janitorial coverage must scale to maintain rating and guest happiness.	Again filled with overly specific details. Noting scaling can be useful, but could be inferred simply from the documentation.
48	Fresh survey results (with cleanliness and uptime stabilized) still show many exits due to "Too few unique drink/food shops" and "Too few unique attractions", indicating that distributing additional shops across high-traffic corridors and adding more attractions are key levers for raising happiness, rating, and value.	Positive use of guest surveys to get more information. But the conclusions it draws are weak: drink and food directly impact thirst and hunger, which, if too high, can impact happiness; impact on value is even more indirect.
53	Increasing the ATM's order quantity eliminated its stockouts and the park rating rose the next day, confirming specialty shops can bottleneck guest spending and that tuning their inventory supports rating and value.	Poor causal understanding. ATMs impact money spent, not park rating. It likely over emphasizes the impact of its action rather than the state of the park. Park ratings can pendulum: a dirty park will lower park rating leading to fewer guests leading to a cleaner park leading higher park rating again. Over generalizes to specialty shops, but each specialty shop is unique.
77	After a day, the new green food shop (price 18) maintained 1.0 uptime but sold only 50–61 items against an order_quantity of 200 (item cost 4), indicating significant daily waste; tuning nonzero-cost shop inventory to observed demand should lift profit without hurting uptime.	Noting that stocking accurately is useful, but it does not include any actionable learnings.
88	The newly placed red food shop at max price achieved solid same-day demand (50 sales) with 1.0 uptime; because its items have nonzero cost, over-ordering (e.g., 80 with 30 leftover) wastes money, so inventory should be tuned close to observed demand.	Repetitive learning from day 43 (not the only example of this)

Table 12. A subset of learnings generated by GPT-5 during its 100 days in sandbox mode.

subclasses, without understanding that it will not have access to the full range of subclasses. Perhaps most important to note here, is that it does not try to provide any information about which higher subclasses are more valuable, which would be invaluable in guiding research during the evaluation stage.

Table 13. Grok-4 Learnings

Step	Learning	Notes
1	Water tiles adjacent to a ride increase its excitement by 1 per adjacent water tile, as evidenced by a red roller coaster with base excitement 10 achieving 12 when adjacent to two water tiles. Placing a single high-intensity ride attracts a limited number of guests up to its capacity but results in low park rating and some cleanliness decline without staff or diversity.	Useful information, but information that exists in the documentation. Interesting to start learning with the red roller coaster, which is very difficult to attain.
4	Without janitors, ride cleanliness can drop to critically low levels over multiple days, resulting in zero guest usage of the ride despite available capacity, as evidenced by cleanliness reaching 0.0 and no guests entertained in a day with high guest turnover due to "too few rides."	Repeating information that can be found in the manual
19	A yellow specialist (clown) continues to boost guest happiness in lines over multiple days, as evidenced by entertaining 351 guests in the second day (up from 280), correlating with improved park rating from 15.91 to 30.25, increased guests from 79 to 142, and higher average rides visited from 0.93 to 1.35.	Useful insight that is not directly in the manual.
20	Attractions cannot be placed on tiles that already contain a path; attempting to do so results in an invalid action error. They must be on empty tiles adjacent to paths.	Again, this is in the manual.
40	Same constraint as Step 20—attempting to place attractions on path tiles results in invalid actions.	Repeating its own mistakes.
68	Hiring a blue specialist (stocker) at a valid path location successfully places it, but it may not perform restocking or movement if no shops drop below threshold during the day, as evidenced by success_metric_value 0.0 and tiles_traversed 0 in the initial day.	Found in the documentation and generally uninformative.
71	Removing a ride of the same subtype as an existing one reduces excitement, capacity, guests, and rating, but can increase avg rides visited per guest and profit, e.g., excitement 28.16 $\rightarrow$ 22.56, capacity 88 $\rightarrow$ 76, guests 114 $\rightarrow$ 84, rating 24.18 $\rightarrow$ 19.18, avg rides visited 1.07 $\rightarrow$ 1.46, and profit -18 880 $\rightarrow$ +6 966 after selling a duplicate roller coaster.	Poor causal and long-horizon understanding. The large swing in profit likely arises from buying a ride one turn (leading to a high cost), and then selling it the next (high revenue). This short-sided analysis shows the opposite of the long term impact.
75	Guest surveys with one ride reveal "Too few unique attraction" and sometimes "Too few drink shops," with thirst 0.4-0.7 and varying happiness; 22/25 results cite uniqueness issues.	Positive use of guest surveys, but does not leverage the information toward actionable learnings.
80	Hiring a yellow mechanic to repair a broken high-value ride (red roller coaster) performs $\sim$1000 repair steps at cost $\sim$1000 in one day but does not immediately restore uptime (0), showing that one mechanic is insufficient.	Over generalization. One yellow mechanic is insufficient for a red roller coaster.
81	A yellow mechanic continues repairs over multiple days ($\sim$1000 steps/day @ 1000 cost/day) but still 0 uptime after 2 days $\rightarrow$ shows slow repair rate.	Correct, but uniformative.
82	Adding a red mechanic assists repairs (2450 steps + yellow 98) $\rightarrow$ repair completed in 1 day, uptime 0.9, guests return; shows multi-mechanic synergy.	Useful, but three days were spent to learn what is largely intuitive given the documentation.

Table 14. A subset of learnings generated by Grok-4 during its 100 days in sandbox mode.

Coalescing the analysis in Table 14, we see that Grok-4 does a very poor job of learning useful actionable insights. It spends many turns to identify knowledge that readily available in the manual. Like GPT-5, it also struggles with identifying causal relations and over attributes outcomes to the most recent action at hand, without consideration to the existing park layout. Lastly, it appears to spend very little time learning about the early stages of the game, which are critical to reach the later states it does explore.

Table 15. Gemini 2.5-Pro Learnings

Step	Learning	Notes
6	Placing a food shop effectively addresses hunger, increases spend per guest, and generates profit.	Useful, but already in the documentation.
7	Placing a drink shop satisfies thirst and generates revenue, but an order_quantity of 50 was insufficient, causing stockouts. High-excitement rides (e.g., blue roller coaster) break down often.	Useful, but already in the documentation.
8	A high-tier (red) mechanic can repair a broken ride within one day, drastically improving uptime and revenue potential.	Somewhat useful, but fails to note the high price of red mechanics and the research required to get there.
9	Increasing a shop's order_quantity via modify prevents stockouts, ensuring sustained revenue. Research on roller coasters completed, unlocking red subclass.	
48	With park growth, two red janitors may not maintain cleanliness and three red mechanics may struggle → min uptime 0.75, min cleanliness 0.62.	It misses crucial context: with 20 red rides, this may very well be true. With a few yellow rides, two red janitors is excessive.
55	Setting research speed = fast unlocks next tier (blue) in 1 day but at high cost; high traffic increases dirtiness (cleanliness 0.65 even with 3 janitors).	Useful to note the timing of research, but fast research is too expensive to be used frequently. Testing the speeds for slow or medium would likely be more useful.
57	Upgrading a high-demand shop (drink) to higher tier (blue) dramatically boosts rating, guests, and revenue, but guest surge lowers cleanliness.	Upgrade insight can be gleaned from the documentation. Noting guest surge can be useful, if it uses that to preemptively hire janitors (which it won't do).
74	Hiring 7 red mechanics + 6 red janitors → restores uptime 0.97 and strong profit → optimal staffing ratio for large park.	Again, it omits critical context here.
94	Hiring a 12th red mechanic did not solve uptime (0.93) and caused -12,749 loss → more mechanics ≠ more profitability.	Again, it omits critical context here.

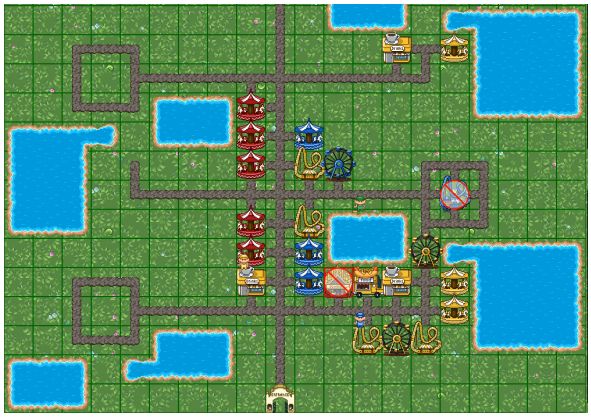
Coalescing the analysis in Table 15, we see that Gemini also does a very poor job of learning useful actionable insights. It largely regurgitates information that is already found in the documentation, and on few instances it tries to identify more specific trends, it omits necessary context.

Analyzing Table 16, we see that Claude Sonnet 4.5 tends to be extremely specific on changes in numbers, but does not properly consider factors outside of its most immediate action. It also does not consider stochasticity or other potential causal links in its insights, leading to several spurious correlations.

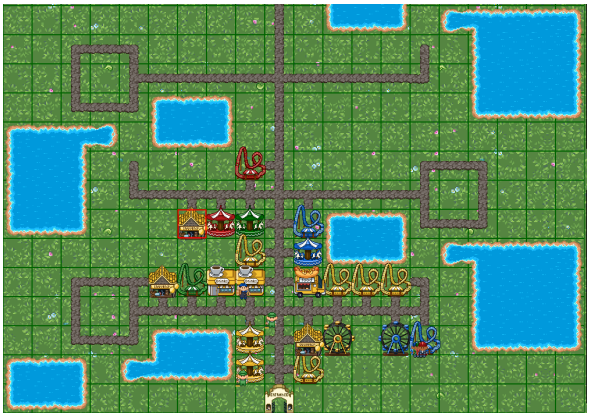
G. Qualitative Analysis of Spatial Placement

Table 16. Claude Sonnet Learnings

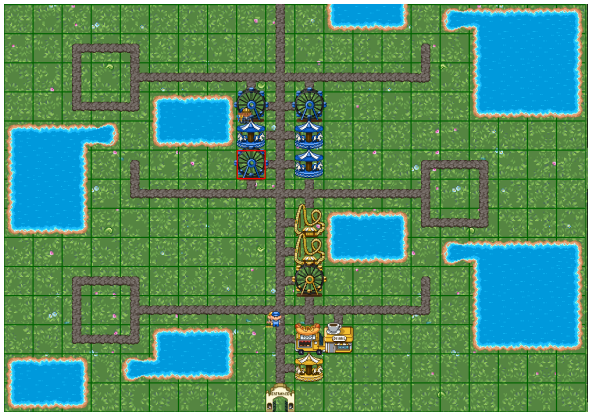
Step	Learning	Notes
7	Adding a food shop near rides improved rating 11.36 → 46.02; guests visited 0.35 food shops on average → meeting hunger needs boosts rating and satisfaction.	While the general trend that food shops can help is true, the rating gap is way too large to be attributed to a single food shop. It is much more likely to be due to the rubber effect of cleanliness.
12	A yellow mechanic (repair rate 2) took a full day (998 steps) to repair a red roller coaster ($100 000) → low-tier mechanics too slow for expensive rides.	True, but an odd experiment to run, and this information is available in the documentation.
20	A green mechanic (repair rate 20) maintained red roller coaster at 61% uptime; revenues $4150, costs $1600 + $3780 → -$165 profit → even high tiers struggle to make it profitable.	Overly specific to a specific context, but also omits the necessary context. Fails to understand that stochasticity can be the cause of the low uptime for this specific turn.
29	Park with 4 diverse rides, 2 shops, 2 staff → consistent $1088 profit, 69 guests, rating 36.53 → ride diversity + inventory management + mechanic = stable profit.	Reasonable and useful assessment.
32	Adding too many same rides hurts returns → 3rd yellow carousel → guests 102→87, profit 1681→888, rating 37.49→31.62 → diversity > capacity for profitability.	Again the claim can be true, but the data here is misleading. Adding a new ride can only increase the park rating in that turn (on a longer horizon, the happiness of guests may be impacted leading to a lower park rating). Drop in park rating is more likely due to the cleanliness.
46	Green mechanic (repair rate 20) maintained blue coaster 91%, fixed others to 99/97%, 980 steps @ $980 cost, profit 5214→3993 → effective but expensive maintenance.	Extremely specific.
91	Once repaired, blue mechanic (repair rate 8) maintains 100% uptime at $100/day salary (no repair costs) → excellent value for mid-tier maintenance.	Potentially useful, the causally incorrect. 100% can only occur because of the lack of a breakdown, which is a stochastic event.



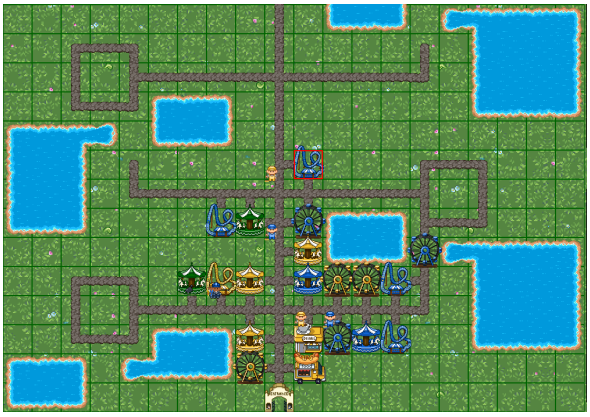
(a) Final park of GPT-5's best playthrough on Ribs Easy.



(b) Final park of GPT-5 with the added spatial heuristic's best playthrough on Ribs Easy.



(c) Final park of Claude Sonnet 4.5's best playthrough on Ribs Easy.



(d) Final park of Claude Sonnet 4.5 with the added spatial heuristic's best playthrough on Ribs Easy.

Figure 3. Comparison of the park design with and without the added spatial heuristic.

H. Qualitative Analysis of Failures and Successes

Grok-4 is strongly biased towards surveying guests. In some cases this allows it to address guest concerns brought to its attention (e.g., by hiring a clown or constructing a drink shop). However, due to the relatively high cost of surveying a guest (500 per guest, twice the price of the cheapest ride), its tendency to survey frequently (e.g. once every 5 turns) rapidly drains its coffers.

A related failure mode ensnares both Grok-4 and Gemini, which is prematurely launching research. This choice to invest in research rather than expanding the base park results in slower growth. Compounding this issue, Grok-4 is sometimes unable to track what it is researching, and so performs a single step of research, accumulates funds over several turns, and then performs a second step of research on a different topic. This prevents it from making progress.

Sonnet (and to a lesser extent Grok-4) tend to be overly reactive and risk averse with building placement. Both not uncommonly construct, and immediately sell an attraction – this results in a loss of 33% of the value which could be prevented by either avoiding the initial construction, or by experimenting with building placement or ticket pricing. In one case, Grok-4 placed a souvenir shop near the entrance (an excellent placement as souvenir shops are high profit margin and rely on foot traffic – which will be high near the entrance), but immediately sold it citing "The specialty shop added at (17,10) is underutilized (only 0.06 average visits, 2 guests served) but incurred high stocking costs ($172 operating, only $30 revenue), leading to negative profit last day. Removing it will recover 66% of build cost ($165) and eliminate ongoing losses". This was overly reactive, as the stock could have been reduced instead.

Closely related, Sonnet (and to a lesser extend Gemini) suffers a failure where instead of performing the free action of moving a ride, they instead sell and newly construct an attraction.

Sonnet tends to play conservatively; in some cases prematurely concluding that expansion is not possible (due to its poor placement combined with a tendency to panic and act reactively), and instead decides to wait until the conclusion of the game.

With sandbox learning, we observed that Sonnet struggled with the longer context. Specifically, partway through six of the nine runs it would repeatedly fail to format its output correctly, and ultimately decide that waiting until the game was completed was its best option. In one trajectory we saw Gemini similarly enter a loop where it repeatedly failed to format its output correctly, but it was able to recover after 4 failed attempts and continued play normally.

In MAPS the yellow specialist (i.e., clown) is an employee that improves guest happiness in lines – only indirectly improving revenue by increasing park rating (leading to more guests in the future), and by making guests stay and spend in the park for longer. We observe that Sonnet never hires clowns, and gemini only rarely. Grok-4 hires clowns extensively (as mentioned, sometimes in response to a guest survey highlighting guest happiness as an issue), though sometimes enters over-hiring (e.g., about 5 clowns in one of the runs – it did correct this by firing some of the clowns). GPT-5 also typically hires clowns, further improving its performance compared to other models.

GPT-5 is a stronger player; its shortcoming are more subtle. While it's generally quite good at building near the entrance and clustered, it still fails with the subtleties of paths and connectivity. For example, on `zig_zag` it will build attractions that are nearby as the crow flies, but quite far apart for guests due to the winding path connecting the attractions. Similarly, on `the_islands` it consistently initially fails to understand that different islands are not connected by paths until after it has constructed attractions in inaccessible locations. It is generally able to recover and move these attractions.

GPT-5 plays most aggressively of the models studied, building new attractions, in some cases as much as 10 duplicate rides (typically yellow or blue roller coasters). This aggression – combined with its competent placement – gives it stronger revenue in the early game making research more practical. However, GPT-5 tends to begin by researching blue roller coasters, the most expensive blue unlock. This causes it to lose time once the research is complete (e.g., in one evaluation there were 16 turns between the unlocking of the blue coaster and its first construction). In most cases, once it has performed its research it only builds blue coasters – preferring continued aggressive expansion with blue coasters over ride diversification. This also leads to problems with park management: due to the high guests counts the yellow staff are unable to keep the park clean or the rides maintained. This results in bimodal park behaviour where guests attendance oscillates – high attendance leads to dirtiness leading to low attendance the next day leading to a maintainable janitorial load leading to high attendance the next day. Instead of performing research into better staff to address this issue, GPT-5 continues aggressive expansion, ignoring the reduction in visitors (as well as the penalties for duplicated rides). Relating to this preference for immediate expansion over all else, in some evaluations GPT-5 never performs research, instead only building yellow attractions. Interestingly, we

observe that with sandbox learning GPT-5 is much more willing to perform research – sometimes researching staff (the only agent studied to do so), and researching blue Ferris wheels in 50% of the evaluations.

H.1. GPT-5 Sandbox Learning without Documentation

As to why sandbox learning decreases performance in medium difficulty when the documentation is removed, the cause is unclear. The notes still explore various tiers of attractions, and appear similar to those learned under different conditions. Studying the medium trajectories with sandbox learning, GPT-5 tends to construct a Ferris wheel very early in the game – a poor choice as when guest counts are low the high-capacity Ferris wheel will have long lines. Combined with the increased impulsiveness when documentation is removed, this results in GPT-5 repeatedly constructing and demolishing attractions for under performing. Why this occurs is unclear. Reviewing the learnings we found one encouraging the construction of yellow Ferris wheels: `Adding even a low-tier unique ride (yellow ferris wheel) near a secondary hub increased park rating without hurting uptime or cleanliness, reinforcing that additional attractions (not just high-tier) can lift value when operations are stable.`. This is largely true, but only if guest counts are sufficiently high. Given that this is the only relevant learning, it is unclear if this is responsible for the behaviour observed. If so, then this would suggest that online learning would benefit future MAPS agents.

I. Observation Example


```
{
  "available_entities": {
    "carousel": ["yellow"], "drink": ["yellow"], "ferris_wheel": ["yellow"], "food":
    ↪  ["yellow"], "janitor": ["yellow"], "mechanic": ["yellow"], "roller_coaster":
    ↪  ["yellow"], "specialist": ["yellow"], "specialty": ["yellow"]},
  "entrance": [0,1],
  "exit": [19,18],
  "expenses": 122,
  "fast_days_since_last_new_entity": 0,
  "guest_survey_results": {"age_of_results": 5, "list_of_results": []},
  "guests": {
    "avg_drink_shops_visited": 0.59,
    "avg_food_shops_visited": 0.64,
    "avg_money_spent": 6.95,
    "avg_rides_visited": 0.5,
    "avg_specialty_shops_visited": 0.0,
    "avg_time_in_park": 113.36,
    "total_guests": 30
  },
  "horizon": 100,
"medium_days_since_last_new_entity": 0,
  "min_cleanliness": 0.85,
  "money": 387,
  "new_entity_available": false,
  "parkId": "263dde2b-95b5-4a7a-b6a6-c7c15d50843b",
  "park_rating": 31.68,
  "paths": [
    {
      "cleanliness": 0.98,
      "x": 0,
      "y": 2
    },
    ...
    {
      "cleanliness": 0.96,
      "x": 19,
      "y": 17
    }
  ],
  "profit": 91,
  "research_operating_cost": 0,
  "research_speed": "none",
```

```
"research_topics": ["carousel", "ferris_wheel", "roller_coaster", "drink", "food",
↪  "specialty", "janitor", "mechanic", "specialist"],
"revenue": 213,
"rides": {
  "avg_intensity": 1.0,
  "min_uptime": 1.0,
  "ride_list": [
    {
      "avg_guests_per_operation": 1.07,
      "avg_wait_time": 12.13,
      "breakdown_rate": 0.001,
      "capacity": 6,
      "cleanliness": 0.87,
      "cost_per_operation": 1,
      "excitement": 1,
      "guests_entertained": 16,
      "intensity": 1,
      "operating_cost": 15,
      "out_of_service": false,
      "revenue_generated": 64,
      "subclass": "yellow",
      "subtype": "carousel",
      "ticket_price": 4,
      "times_operated": 15,
      "uptime": 1.0,
      "x": 0,
      "y": 4
    }
  ],
  "total_capacity": 6,
  "total_excitement": 0.8,
  "total_operating_cost": 15,
  "total_revenue_generated": 64,
  "total_rides": 1
},
"shops": {
  "min_uptime": 1.0,
  "shop_list": [
    {
      "cleanliness": 0.92,
      "guests_served": 18,
      "inventory": 82,
      "item_cost": 0,
      "item_price": 3,
      "number_of_restocks": 0,
      "operating_cost": 0,
      "order_quantity": 100,
      "out_of_service": false,
      "revenue_generated": 54,
      "subclass": "yellow",
      "subtype": "drink",
      "uptime": 1.0,
      "x": 0,
      "y": 5
    },
    {
      "cleanliness": 0.94,
      "guests_served": 19,
      "inventory": 1,
      "item_cost": 1,
      "item_price": 5,
      "number_of_restocks": 0,
      "operating_cost": 20,
      "order_quantity": 20,
      "out_of_service": false,
      "revenue_generated": 95,
```

```
        "subclass": "yellow",
        "subtype": "food",
        "uptime": 1.0,
        "x": 1,
        "y": 2
      }
    ],
    "total_operating_cost": 20,
    "total_revenue_generated": 149,
    "total_shops": 2
  },
  "slow_days_since_last_new_entity": 0,
  "staff": {
    "staff_list": [
      {
        "operating_cost": 2,
        "salary": 25,
        "subclass": "yellow",
        "subtype": "janitor",
        "success_metric": "amount_cleaned",
        "success_metric_value": 0.05,
        "tiles_traversed": 1,
        "x": 0,
        "y": 4
      },
      {
        "operating_cost": 0,
        "salary": 60,
        "subclass": "yellow",
        "subtype": "specialist",
        "success_metric": "guests_entertained",
        "success_metric_value": 97.0,
        "tiles_traversed": 2,
        "x": 0,
        "y": 4
      }
    ],
    "total_janitors": [1, 0, 0, 0],
    "total_mechanics": [0, 0, 0, 0],
    "total_operating_cost": 2,
    "total_salary_paid": 85,
    "total_specialists": [1, 0, 0, 0],
  },
  "step": 5,
  "value": 750,
  "waters": [
    {
      "x": 2,
      "y": 4
    },
    {
      "x": 2,
      "y": 5
    },
    ...
    {
      "x": 16,
      "y": 18
    }
  ]
}
```

J. Prompt Templates

J.1. Base Prompt Templates

For all models we begin with these base templates and evaluate on a test layout for 3 steps to check model behaviour. These base ReAct prompts are modified from LangChain[1]. We budget 30 minutes per model for prompt tuning to correct any observed misbehaviour on the part of the model.

State observations are provided as json strings, with any error messages from the Mini Amusement Parks environment or the world model appended to the state along with the last action taken; e.g., `NOTE: While attempting the action 'place(x=12, y=9, type="shop", subtype="drink", subclass="yellow", price=3, order_quantity=-1)' the error '{'message': 'Inventory order_quantity cannot be negative: -1', 'type': 'invalid_action'}' occurred.`

```
You are playing a game where you are the manager of a theme park.
You can only take a single action each day.
Your objective is to maximize park value over a {horizon} day period.
Note that you are playing in *{difficulty} difficulty*.

Use the following format to structure your output:

Thought: you should always think about what to do
Action: the action to take, should be one of the following: {actions_list}
Action Input: the input to the action, as correctly formatted list of python arguments -- e.g.: arg1="hi", arg2=34
Observation: the user will provide you with result of the action
... (this Thought/Action/Action Input/Observation repeats until the game is complete)

Below is the documentation of the game:

{{GAMEPLAY_RULES}}
```

Listing 1. ReAct System Prompt Template

```
{REACT_HISTORY}

Observation: {state_with_error_msg}
Question: What is the best action I should take now to take to maximize park value?
Thought:
```

Listing 2. ReAct User Prompt Template

In sandbox learning the following prompt was used:

```
Instructions:
You are tasked with playing an amusement park simulator.
Before starting the layouts that you will be evaluated on,
You will have {max_sandbox_steps} in-game days in a sandbox mode to explore and interact with the environment.
Use this period to learn how the game works and experiment with different actions and strategies.
During this sandbox mode, you will have access to several additional commands:
undo_day, max_money, max_research, reset, and switch_layouts.
These are designed to help you learn the games dynamics more efficiently.
These actions do not count toward your learning budget.
However, they will not be available during the evaluation phase.
You are free to learn on any of the provided layouts.
```

[1] https://python.langchain.com/api_reference/_modules/langchain/agents/react/agent.html

```
After completing the sandbox stage, you will begin the evaluation.
This will occur on a layout not available during sandbox mode.
During evaluation, your goal will be to maximize the park's value over a {horizon} day period.

You will be playing on {difficulty} difficulty.

Documentation:
A detailed guide for the game has been provided.
Before playing, please read through the documentation carefully.
The documentation contains useful information, rules, and tips to help you succeed.

Strictly obey the following format to structure your output:

Learnings Summary: a one or two sentence summary of the useful information you have learned from the previous
    interaction (if anything). This information should be general enough to apply even on a game layout with a different
    arrangement of paths. This information should be about normal gameplay -- i.e., not specific to sandbox mode.
    Remember, this is for only information *supported by evidence* (no unbased speculation and no reasoning about
    your next action!) that will be useful when you play again in the future. Your learnings from all turns will be made
    available to you, both during exploration and when you're evaluated.
Thought: you should always think about what to do
Action: the action to take, should be one of the normal game actions ({actions_list}) or the sandbox game actions
    ({sandbox_actions_list})
Action Input: the input to the action, as correctly formatted list of python arguments -- e.g.: arg1="hi", arg2=34
Observation: the user will provide you with result of the action
... (this Thought/Action/Action Input/Observation repeats until the game is complete)


For example:

Learnings Summary: I can increase park value by...
Thought: Based on this, I should try to...
Action: place
Action Input: x=5, y=12, type="ride", subtype="carousel", subclass="yellow", price=3


Below is the documentation for the game:

{{SANDBOX_GAMEPLAY_RULES}}
```

Listing 3. Sandbox Learning ReAct System Prompt

```
# Collated Past Learnings Summaries
{{LEARNINGS_LIST_TO_DATE}}

# Recent Actions

{REACT_HISTORY}

# Current State

Observation: {state_with_error_msg}
Question: What is the best action you can take next to explore the environment with the aim of learning how to maximize
    park value? You have {sandbox_steps_left} non-sandbox actions remaining.
Learnings Summary:
```

Listing 4. Sandbox Learning ReAct User Prompt

```
You are playing a game where you are the manager of a theme park.
You can only take a single action each day.
Your objective is to maximize park value over a {horizon} day period.
Note that you are playing in *{difficulty} difficulty*.

Use the following format to structure your output:

Thought: you should always think about what to do
Action: the action to take, should be one of the following: {actions_list}
Action Input: the input to the action, as correctly formatted list of python arguments -- e.g.: arg1="hi", arg2=34
Observation: the user will provide you with result of the action
... (this Thought/Action/Action Input/Observation repeats until the game is complete)

Below is the documentation of the game:

{{GAMEPLAY_RULES}}

You previously had the chance to explore the game and make notes. These are the notes you made:

Step0: {{STEP_0_LEARNING}}

Step1: {{STEP_1_LEARNING}}

Step2: {{STEP_2_LEARNING}}

...

Step100: {{STEP_100_LEARNING}}
```

Listing 5. ReAct System Prompt Augmented with Learnings

The prompt for sandbox GPT-5 Nano was modified to better separate learnings from thoughts about next steps.

The prompt for Claude Sonnet 4.5 was modified to discourage the model from hallucinating next state observations.

K. Game Documentation

The following pages include the games documentation.

Game Mechanics

The goal of the game is to maximize your amusement park's value within a set number of days. As the manager of the park, you perform one action at the start of each day. The park then opens, and guests interact with the park for a full day.

There are three difficulty modes in the game (*Easy*, *Medium*, and *Hard*):

- **Easy:** All attractions are available from the beginning. Water tiles are disabled. Short time horizon (50 days).
- **Medium:** Only yellow attractions are available from the beginning; other attractions must be researched. Layouts can include water tiles. Medium time horizon (100 days).
- **Hard:** Coming soon.

There are seven primary components to the game. We provide a high-level overview here, followed by detailed descriptions of each:

- **The Park:** Defined by a square grid (defaults to 20x20) and contains all other components. The amusement park has an entrance and an exit connected by a path. Your park also has a *rating* that reflects guest satisfaction, park upkeep, and overall park quality.
- **Terrain:** There are three kinds of terrain -- Paths, Water, and Empty.
- **Rides:** Rides are one of two types of attractions you can place in your amusement park. They are the core of the park and draw guests in. There are three subtypes of rides: Carousels, Ferris Wheels, and Roller Coasters.
- **Shops:** Shops are the second type of attraction. They allow you to cater to your guests' needs. There are three subtypes of shops: Drink, Food, and Specialty. Drink shops quench guests' thirst; Food shops satisfy their hunger; Specialty shops provide unique services.
- **Staff:** Staff can be hired to maintain your park. There are three subtypes of staff: Janitors, Mechanics, and Specialists. Janitors keep the park clean, Mechanics repair rides, and Specialists perform a variety of support tasks.
- **Subclasses & Research:** Each ride, shop, and staff subtype has four subclasses—yellow, blue, green, and red. In Easy mode these are unlocked from the start; in Medium and Hard modes, higher subclasses must be unlocked through research.
- **Guests:** The people your park is built for! Guests interact with your park, spending money to purchase ride tickets, food, drinks, and more.

Terrain

- **Empty tiles:** Blank tiles on which something can be built.
- **Path tiles:** Used by guests to move around your park. All attractions must be placed on an empty tile adjacent to a path tile.
- **Water tiles:** Each water tile adjacent to a ride increases that ride's excitement by 1.

In Hard mode, terrain can be terraformed. Paths can be built ($1000) and removed ($2500). Water tiles can similarly be added ($5000) and removed ($10000).

Rides

Rides are the core of your amusement park. They are the primary driver in determining how many guests visit your park, they improve guest happiness, and contribute to the overall value of the park. Rides have several key attributes:

- **Capacity:** How many guests can fit on the ride at one time. Capacity also affects how frequently the ride operates. The cumulative capacity of your park is also a key factor in how many guests visit the park.
- **Excitement:** How thrilling the ride is. Higher excitement scores increase guest happiness and park rating.
- **Intensity:** How intense the ride experience is. Keeping the average intensity balanced (around 5) ensures your park caters to a wide range of guests and improves your rating.
- **Ticket price:** The amount guests must pay to ride. If a guest cannot afford the ticket, they are rejected, which decreases their happiness.
- **Cost per operation:** The amount it costs each time the ride runs.

> **TIP:** Building multiple rides of the same kind (i.e., identical subtype and subclass) yields diminishing returns for your park rating. Diversifying your attractions will lead to a higher overall rating.

Rides only operate if guests have boarded. After the first guest boards, rides wait a few turns to allow more guests to join. This waiting time is longer for rides with higher capacity but decreases as more guests board. A full ride always operates immediately.

Rides may break down after operating. When broken, they are out of service and will turn away guests, negatively affecting their happiness and your park rating.

> **TIP:** Hire mechanics to fix broken rides promptly.

There are three subtypes of rides:

- **Carousels:** Cheap to build and operate, and they rarely break down. However, they provide limited excitement and capacity.
- **Ferris Wheels:** Intermediate rides with the highest capacities among all subtypes.
- **Roller Coasters:** Expensive but high-value rides that offer the highest excitement and intensity scores, at the cost of frequent breakdowns.

For the exact parameters of each ride, see All Rides.

Shops

Shops are necessary to adequately cater to guest needs, and provide additional value to a park. Shops have several key attributes:

- **Order quantity:** The amount of inventory purchased at the start of each day. Unsold inventory is lost at the end of the day.
- **Item cost:** The cost of purchasing one unit of inventory.
- **Item price:** The price guests pay for one unit of inventory.

At the start of each day, shops are stocked according to their order quantity at the item cost. If there are insufficient funds to fully restock all shops, partial stocking will occur. If a shop runs out of inventory during the day, it will go out of service, turning away guests and lowering their happiness and your park rating.

> **TIP:** Leave enough funds after your action so your shops can be adequately restocked.
> **TIP:** Order quantity can be updated using the *modify* action.

There are three subtypes of shops:

- **Drink:** Sells beverages that quench guests' thirst.
- **Food:** Sells food that satisfies guests' hunger.
- **Specialty:** Provides unique services based on subclass:
  - Yellow (Souvenir Shops): Boosts guest happiness.
  - Blue (Info Booths): Informs guests about attractions and their prices.
  - Green (ATMs): Allows guests to withdraw additional funds.
  - Red (Billboards): Encourages guests to seek food and ATMs.

By default, guests do not actively seek out Specialty Shops; they visit them only if they pass by.

> **TIP:** Place specialty shops in high-traffic areas to increase the likelihood of guest interaction.

For exact parameters of each shop, see All Shops.

Staff

Staff are necessary for the smooth operation of your park, ensuring attractions run properly and guests remain satisfied. Multiple staff can occupy and work on the same tile at any given time. All staff have a daily salary, and some incur additional operating costs when performing tasks.

There are three subtypes of staff:

- **Janitors:** Move through the park toward dirty areas. When on dirty tiles, they clean them. Each cleaning action incurs an operating cost.
- **Mechanics:** Move toward rides that are broken down. When they reach a tile with a broken ride, they repair it. The time and operating cost of repairs depend on the ride's building cost and the mechanic(s) performing the repair.
- **Specialists:** Perform different roles based on subclass:
  - Yellow (Clowns): Increases the happiness of guests waiting in line.
  - Blue (Stockers): Restocks shops with low inventory.
  - Green (Park Criers): Informs guests about out-of-service or dirty attractions, as well as current line wait time for rides.
  - Red (Vendors): Provides food and drink to guests waiting in line.

For exact parameters of all staff members, see All Employees.

Subclasses & Research

Each ride, shop, and staff subtype has four subclasses, ordered by price: yellow (cheapest), blue, green, and red (most expensive). Generally, more expensive subclasses provide greater benefits. However, at times cheaper subclasses may better suit your needs.

In Easy mode, all attractions are available from the start and research is disabled. In Medium and Hard modes, you begin with only yellow subclasses and must perform research to unlock the rest. To do this, you must set your research speed (none/slow/medium/fast) and topics (ride/shop/staff subtypes). Research continues daily until you change your settings, run out of funds, or unlock all available subclasses for the chosen topics. If funds run out or all topics are complete, research speed automatically reverts to *none*. Progress is paused, not lost. Research performed provides a small amount of added value to the park in the form of intellectual property, but this value does not increase with research speed.

Research unlocks subclasses in the following order: blue → green → red. Once a subclass is unlocked, research continues to the next topic in your list.

> **TIP:** If for example, you wanted to unlock the red roller coaster as quickly as possible, set the research speed to *fast* and select only "roller coaster" as your research topic.

Research speed costs:

- **none:** $0/day — research halted.
- **slow:** $2000/day.
- **medium:** $8000/day.
- **fast:** $32000/day.

The default setting is a research speed of *none* and all attraction subtypes selected as topics.

Guests

Guests are the heart of your park. The number of guests who visit depends on your park's rating and capacity.

Capacity determines both how many guests can be in the park at once and how many potential guests consider visiting. Capacity is determined entirely by the cumulative capacity of your rides.

> **TIP:** Since only rides increase capacity, a park with no rides will receive no visitors.

Park Rating determines the likelihood that potential guests decide to enter. New guests cannot enter if the park is already at capacity. Park rating increases with the total excitement of the rides and when guests leave the park happy. Park rating decreases when attractions are frequently out of service, the park is dirty, or the average ride intensity is too high or too low.

> **TIP:** Park ratings are calculated at the start of each day. In some cases, it may take a full day for the action you have taken to effect the park. In these cases, the park rating will only be reflected the subsequent day (two days after the action).

Each guest brings a certain amount of money. When they run out of funds or energy, they leave. Each guest also has hunger, thirst, happiness, and energy levels. Hunger and thirst increase over time, while happiness slowly decreases. Hungry guests seek food shops, thirsty guests seek drink shops, and unhappy guests seek rides. If any of these needs become critical, the guest's happiness drops and they may leave the park.

> **TIP:** A guest's hunger and thirst will continue to increase as they wait in lines. Proximity to food and drink options is especially important for high capacity rides that have longer lines.

Guests favor novelty, preferring kinds of attractions they haven't visited before. They also favor nearby attractions but never visit the same attraction twice in a row. Visiting a ride they can't afford or that's out of service reduces their happiness.

Unhappy guests are more likely to litter, reducing cleanliness. Visiting dirty tiles or attractions decreases happiness further. If a ride is too dirty, guests may turn away.

In Hard mode, guests may have **preferences**, meaning they only enjoy certain attraction types. Visiting a ride that doesn't match their preference reduces their happiness. Providing guests with information through an Info Booth (blue Specialty Shop) allows them to identify attractions that suit them.

> **TIP:** You can learn more about guests by surveying them using the *SurveyGuest* action. This reveals why guests left, their needs at departure, and more. You can survey up to 25 guests at a cost of $500 per guest.

Park Value

The total value of the park is the sum of the total money, the money that would be made by selling all constructed attractions, and a flat amount per day of research (the value of the discovered intellectual property). Each day of slow, medium, and fast research adds $1500, $3000, and $6000 respectively to the park's value.

All Rides

Note: Any existing ride can be sold for 66.0% of its building cost. A ride that breaks down incurs a cost equal to 4.5% of its building cost to repair.

Carousels

Yellow

- Building Cost: 250
- Cost per Operation: 1
- Capacity: 6
- Max Ticket Price: 4
- Excitement: 1
- Intensity: 1
- Breakdown Rate: 0.001

Blue

- Building Cost: 1500
- Cost per Operation: 2
- Capacity: 14
- Max Ticket Price: 6
- Excitement: 4
- Intensity: 3
- Breakdown Rate: 0.002

Green

- Building Cost: 8000
- Cost per Operation: 16
- Capacity: 28
- Max Ticket Price: 14
- Excitement: 3
- Intensity: 4
- Breakdown Rate: 0.003

Red

- Building Cost: 22000
- Cost per Operation: 12
- Capacity: 24
- Max Ticket Price: 24
- Excitement: 8
- Intensity: 5
- Breakdown Rate: 0.005

Ferris Wheels

Yellow

- Building Cost: 600
- Cost per Operation: 10
- Capacity: 10
- Max Ticket Price: 5
- Excitement: 2
- Intensity: 2
- Breakdown Rate: 0.006

Blue

- Building Cost: 10000
- Cost per Operation: 20
- Capacity: 20
- Max Ticket Price: 7
- Excitement: 5
- Intensity: 3
- Breakdown Rate: 0.009

Green

- Building Cost: 50000
- Cost per Operation: 55
- Capacity: 40
- Max Ticket Price: 15
- Excitement: 4
- Intensity: 6
- Breakdown Rate: 0.023

Red

- Building Cost: 75000
- Cost per Operation: 75
- Capacity: 30
- Max Ticket Price: 28
- Excitement: 9
- Intensity: 8
- Breakdown Rate: 0.018

Roller Coasters

Yellow

- Building Cost: 1000
- Cost per Operation: 9
- Capacity: 4
- Max Ticket Price: 10
- Excitement: 3
- Intensity: 4
- Breakdown Rate: 0.01

Blue

- Building Cost: 18000
- Cost per Operation: 25
- Capacity: 12
- Max Ticket Price: 20
- Excitement: 7
- Intensity: 7
- Breakdown Rate: 0.02

Green

- Building Cost: 60000
- Cost per Operation: 45
- Capacity: 28
- Max Ticket Price: 34
- Excitement: 6
- Intensity: 9
- Breakdown Rate: 0.025

Red

- Building Cost: 100000
- Cost per Operation: 100
- Capacity: 22
- Max Ticket Price: 50
- Excitement: 10
- Intensity: 10
- Breakdown Rate: 0.033

All Shops

Note: Any existing shop can be sold for 66.0% of its building cost.

Drink Shops

Yellow

- Building Cost: 100
- Item Cost: 0
- Max Item Price: 3
- Thirst Reduction: 0.35

Blue

- Building Cost: 2250
- Item Cost: 1
- Max Item Price: 6
- Thirst Reduction: 0.7

Green

- Building Cost: 17500
- Item Cost: 3
- Max Item Price: 17
- Thirst Reduction: 0.9

- Happiness Boost: 0.4

In addition to quenching thirst, green drink shops additionally provide a boost to guest happiness.

Red

- Building Cost: 48000
- Item Cost: 5
- Max Item Price: 25
- Thirst Reduction: 0.7
- Energy Boost: 20
- Caffeinated Steps: 50

Red drinks caffeinate guests, which boosts a guest's energy and allows them to move twice as fast

Food Shops

Yellow

- Building Cost: 200
- Item Cost: 1
- Max Item Price: 5
- Hunger Reduction: 0.25

Blue

- Building Cost: 3600
- Item Cost: 2
- Max Item Price: 9
- Hunger Reduction: 0.7

Green

- Building Cost: 28000
- Item Cost: 4
- Max Item Price: 18
- Hunger Reduction: 0.8
- Thirst Reduction: 0.4

Green food shops both satiate hunger and quench thirst.

Red

- Building Cost: 60000
- Item Cost: 8
- Max Item Price: 34
- Hunger Reduction: 0.9
- Happiness Boost: 0.6

Red food shops sell luxury food. This greatly satiates hunger and increases happiness

Specialty Shops

TIP: Guests will not target specialty shops, and will only visit specialty shops if the walk adjacent to one.

Yellow (Souvenir Shop)

- Building Cost: 250
- Item Cost: 4
- Max Item Price: 14
- Happiness Boost: 0.5

These provide a happiness boost to guests that buy them the first time, but this happiness boost diminishes with each subsequent souvenir purchased.

Blue (Information Booth)

- Building Cost: 10000
- Item Cost: 2
- Max Item Price: 5

These provide information to guests about the attractions in the park and ensures that guests only visit rides that fall within their budget and preferences.

Green (ATM)

- Building Cost: 50000
- Item Cost: 3
- Max Item Price: 5
- Money Withdrawal: 64

ATMs allow guests to withdraw more money. The amount of money withdrawn decreases exponentially with every subsequent withdrawal, to a minimum of 16

Red (Billboard)

- Building Cost: 10000
- Item Cost: 0
- Max Item Price: 0
- Thirst Boost: 0.2
- Hunger Boost: 0.2

Billboards make guests more hungry, thirsty, and happy. They will additionally reset the visit count of attractions (meaning that guests are more likely to revisit

attractions), and, if the guest has less than $25, will direct the guest to an ATM.

All Employees

Employees perform up to 500 actions per day. For example, moving to an adjacent tile costs a move action.

Janitors

Janitors clean a tile up to their cleaning threshold before moving to the next tile to clean. In addition to their salary, janitors incur $1 per cleaning action in cleaning supplies.

Yellow

- Daily Salary: 25
- Clean Rate: 0.025
- Cleaning Threshold: 0.85

Blue

- Daily Salary: 100
- Clean Rate: 0.075
- Cleaning Threshold: 0.95

Blue, green, and red janitors move at double speed.

Green

- Daily Salary: 500
- Clean Rate: 0.2
- Cleaning Threshold: 1.0

Red

- Daily Salary: 2000
- Clean Rate: 0.35
- Cleaning Threshold: 1.2

Red janitors perform preventative cleaning, allowing them to clean a tile to 1.2; normally tiles have a maximum cleanliness of 1.0

Mechanics

Yellow

- Daily Salary: 15
- Repair Rate: 2

Blue

- Daily Salary: 100
- Repair Rate: 8

Blue, green, and red mechanics move at double speed.

Green

- Daily Salary: 250
- Repair Rate: 20

Red

- Daily Salary: 1000
- Repair Rate: 50

Red mechanics also perform preventative maintenance -- this allows them to partially repair a ride before it breaks down, further reducing the time required for repairs.

Specialists

Yellow (Clown)

- Daily Salary: 60
- Happiness Boost: 0.25

Clowns move between different rides, increasing the happiness of guests waiting in line.

Blue (Stocker)

- Daily Salary: 350
- Stocking Rate: 0.1
- Max Inventory: 100
- Idle Ticks: 30
- Restock Threshold: 0.25

Stockers restock the inventory of shops when the proportion of the shop's remaining inventory drops below the Restock Threshold. When this happens a stocker will move to an entrance or exit, purchase 10.0% of the daily order quantity, and take it to the shop. Stockers can carry at most 100 units of inventory and will not order more than this quantity. Stockers will stop making new purchases in the last 30 actions of the day to prevent them from restocking a shop immediately before closing.

Green (Crier)

- Daily Salary: 400

Criers move between attractions, providing guests about information related to the cleanliness and operation (i.e., if an attraction is out of service) of attractions, as well as current line wait times. This prevents guests from visiting out of service attractions, and makes them more likely to visit cleaner attractions and those with shorter wait times.

Red (Vendor)

- Daily Salary: 1000
- Hunger Reduction: 0.3
- Thirst Reduction: 0.5

Vendors move between rides, providing food and drink to guests waiting in line. Vendors do not incur additional costs or make additional profits from their activities.

Action Space

Actions must be written as python function calls with keyword arguments. These action functions must be in the following format:

```
action_name(param_1=<param1_value>, param_2=<param1_value>, ... )
```

The full list of available actions, including the action names, parameters, and a description of what they do is below:

place
Description: Place an entity (ride or shop or staff) in the theme park
Parameters:
- x: The x position of the entity
- y: The y position of the entity
- type: The type of entity. Must be one of ride or shop or staff
- subtype: The subtype of the entity to be placed. For rides, must be one of carousel, ferris wheel, or roller coaster. For shops, must be one of drink, food, or specialty. For staff, must be one of janitor, mechanic, or specialist
- subclass: The specific instance of the entity to be placed. Must be one of yellow, blue, green, or red
- price: The price of the ticket or item. Only for rides and shops.
- order_quantity: The maximum order_quantity of inventory. Only for shops.

---

move
Description: Move an entity (ride or shop or staff) in the theme park
Parameters:
- type: The type of entity. Must be one of ride or shop or staff
- subtype: The subtype of the entity to be moved. For rides, must be one of carousel, ferris wheel, or roller coaster. For shops, must be one of drink, food, or specialty. For staff, must be one of janitor, mechanic, or specialist
- subclass: The specific instance of the entity to be moved. Must be one of yellow, blue, green, or red
- x: The current x position of the entity
- y: The current y position of the entity
- new_x: The new x position
- new_y: The new y position

---

remove
Description: Remove an entity (ride or shop or staff). If the entity is a ride or shop, it will be sold for a price.
Parameters:
- type: The type of entity. Must be one of ride or shop or staff

- subtype: The subtype of the entity to be removed. For rides, must be one of carousel, ferris wheel, or roller coaster. For shops, must be one of drink, food, or specialty. For staff, must be one of janitor, mechanic, or specialist
- subclass: The specific instance of the entity to be removed. Must be one of yellow, blue, green, or red
- x: The x position of the entity
- y: The y position of the entity

---

modify
Description: Change the price and (as applicable) order_quantity of inventory for an attraction
Parameters:
- type: The type of attraction. Must be one of ride or shop
- x: The x position of the attraction
- y: The y position of the attraction
- price: The new price
- order_quantity: The new maximum order_quantity of inventory. Only for shops.

---

set_research
Description: Set the research speed and topic(s) for the theme park. Only available in medium or hard difficulty.
Parameters:
- research_speed: The research speed. Must be one of none, slow, medium, or fast
- research_topics: The topics(s) of research. Must be a subset of ['carousel', 'ferris_wheel', 'roller_coaster', 'drink', 'food', 'specialty']

---

survey_guests
Description: Retrieve a sample of information from guests
Parameters:
- num_guests: The number of guests to survey

---

add_path
Description: Add a path tile to the theme park. Only available in hard difficulty.
Parameters:
- x: The x position of the path tile
- y: The y position of the path tile

---

remove_path
Description: Remove a path tile from the theme park. Only available in hard difficulty.
Parameters:
- x: The x position of the path tile
- y: The y position of the path tile

---

add_water
Description: Add a water tile to the theme park. Only available in hard difficulty.
Parameters:
- x: The x position of the water tile
- y: The y position of the water tile

---

remove_water
Description: Remove a water tile from the theme park. Only available in hard difficulty.

Parameters:
- x: The x position of the water tile
- y: The y position of the water tile

---

wait
Description: runs the day without any new action
Parameters:
No parameters

Sandbox Mode

The game can be placed in sandbox mode to help with learning or exploring the environment. In sandbox mode, players have access to new actions that are useful for experimenting with the game. These actions are likewise written as python function calls (see Action Space).

When in sandbox mode, the game permits the player to perform 100 normal actions and 250 of sandbox actions.

Sandbox Action Space

The full list of available sandbox actions, including the action names, parameters, and a description of what they do is below:

---

undo_day
Description: Undo the last day, reverting to the previous state.
Parameters:
No parameters

---

max_money
Description: Set the park's money to the maximum value ($99,999,999).
Parameters:
No parameters

---

max_research
Description: Unlock all attractions and staff.
Parameters:
No parameters

---

reset
Description: Reset the park to initial state
Parameters:
No parameters

---

change_settings
Description: Change the park's difficulty and layout settings, and reset the park using the new settings.
Parameters:
- difficulty: The difficulty level. Must be one of easy, medium, or hard
- layout: The park layout. Must be one of the available layouts

---