TowerMind: A Tower Defence Game Learning Environment and Benchmark for LLM as Agents

Dawei Wang[1], **Chengming Zhou**[1], **Di Zhao**[2], **Xinyuan Liu**[1], **Marci Chi Ma**[1], **Gary Ushaw**[1], **Richard Davison**[1]

[1]Newcastle University, United Kingdom
[2]University of Auckland, New Zealand
{d.wang28, c.zhou10, x.liu89, c.ma20, gary.ushaw, richard-gordon.davison}@newcastle.ac.uk, dzha866@aucklanduni.ac.nz

Abstract

Recent breakthroughs in Large Language Models (LLMs) have positioned them as a promising paradigm for agents, with long-term planning and decision-making emerging as core general-purpose capabilities for adapting to diverse scenarios and tasks. Real-time strategy (RTS) games serve as an ideal testbed for evaluating these two capabilities, as their inherent gameplay requires both macro-level strategic planning and micro-level tactical adaptation and action execution. Existing RTS game-based environments either suffer from relatively high computational demands or lack support for textual observations, which has constrained the use of RTS games for LLM evaluation. Motivated by this, we present TowerMind, a novel environment grounded in the tower defense (TD) subgenre of RTS games. TowerMind preserves the key evaluation strengths of RTS games for assessing LLMs, while featuring low computational demands and a multimodal observation space, including pixel-based, textual, and structured game-state representations. In addition, TowerMind supports the evaluation of model hallucination and provides a high degree of customizability. We design five benchmark levels to evaluate several widely used LLMs under different multimodal input settings. The results reveal a clear performance gap between LLMs and human experts across both capability and hallucination dimensions. The experiments further highlight key limitations in LLM behavior, such as inadequate planning validation, a lack of multifinality in decision-making, and inefficient action use. We also evaluate two classic reinforcement learning algorithms: Ape-X DQN and PPO. By offering a lightweight and multimodal design, TowerMind complements the existing RTS game-based environment landscape and introduces a new benchmark for the AI agent field.

1 Introduction

One of the fundamental challenges in artificial intelligence (AI) is equipping agents with the ability to solve tasks across a broader range of scenarios (Russell and Norvig 2016). Recent breakthroughs in large language models (LLMs) (Devlin et al. 2019; Achiam et al. 2023) have made them a promising approach to addressing this challenge. Benefiting from their extensive cross-domain knowledge and diverse abilities, including reasoning (Wei et al. 2022b; Wang, Deng, and Sun 2022) and problem-solving (Lingo, Arroyo, and Chhajer 2024; Renze and Guven 2024), LLM-based agents have shown potential in various domains, such as healthcare (Li et al. 2024), office automation (Zhang et al. 2024), and design (Çelen et al. 2024). Despite differences in context and specifics, these tasks consistently require two foundational capabilities from LLMs: **long-term planning** and **decision-making**, which are essential for accomplishing tasks: (1) LLMs leverage long-term planning to decompose a high-level task into a sequence of subgoals that guide progress toward the final objective; (2) LLMs perform decision-making to translate this sequence of subgoals into executable actions, conditioned on the evolving task state.

Real-time strategy (RTS) games are an ideal platform for evaluating long-term planning and decision-making abilities, as they require players to engage simultaneously in both macromanagement and micromanagement (Barros e Sá and Madeira 2025). Specifically, RTS games provide a battlefield setting where macromanagement tends toward long-term strategic planning, in which players formulate high-level strategies such as overall unit deployment and resource allocation; whereas micromanagement focuses on real-time decision-making, where players flexibly control units in response to dynamic changes on the battlefield to execute their combat plans. Currently, several RTS game-based benchmarks have recently been proposed for evaluating LLMs, including TextStarCraft II (Ma et al. 2025b), LLM-PySC2 (Li et al. 2024), and VLMs Play StarCraft II (Ma et al. 2025a), all of which are based on the StarCraft II Learning Environment (SC2LE) (Vinyals et al. 2017), known for its relatively high computational demands. These challenging RTS game-based benchmarks are effective for assessing the long-term planning and decision-making capabilities of LLMs; nevertheless, the need for low-cost evaluation environments still persists in the field (Dubois et al. 2024). For example, in fast-paced continuous development pipelines (Koc 2025) and in the usage of reward models for instruction tuning (Yuan et al. 2024), lightweight benchmarks offer clear advantages. While several lightweight RTS game-based environments (e.g., ELF (Tian et al. 2017), DeepRTS (Andersen, Goodwin, and Granmo 2018), Gym-μrts (Huang et al. 2021)) have been proposed to alleviate the computational demands of SC2LE-based platforms, they fundamentally lack support for textual observations and action interfaces, which makes them incompatible with LLMs.

Figure 1: These are screenshots from four different TowerMind levels. The icons in the four corners of each image display key gameplay information, including the number of player's current gold coins, player's base health, and remaining enemy waves. The maps feature irregular, intersecting roads along which enemies advance toward the player's base in successive waves. Players must strategically build different types of towers at designated locations along these roads to repel the incoming enemies. The cloud-shaped white areas represent fog of war, introducing partial observability to the environment.

To address the lack of lightweight RTS game-based environments with textual observation capabilities, we propose **TowerMind**, a newly developed game environment built upon the tower defense (TD) subgenre of RTS games (Liu et al. 2019), its screenshot is shown in Figure 1. TD games share the same core game mechanics as classic RTS games (Tian et al. 2017), providing a battlefield scenario where players must build towers and deploy units to defend against waves of invading enemies, requiring them to demonstrate long-term planning and decision-making. Unlike the player-versus-player mechanics of classic RTS games, TD games focus solely on defending against predefined waves of enemies. This allows for a more isolated evaluation of LLMs' ability to finish complex tasks using long-term planning and decision-making, without interference from opponent unpredictability. Furthermore, the fixed tower placement options and predefined enemy roads in TD games facilitate clearer analysis of the strategies employed by LLMs. In this work, TowerMind significantly reduces the computational demands compared to existing RTS game-based LLM benchmarks. Specifically, existing RTS game-based benchmarks for LLMs rely on the SC2LE environment, which requires approximately 30 GB of disk space, 2 GB of RAM, and a dedicated GPU. In contrast, TowerMind requires only 0.15 GB of disk space and RAM, runs efficiently on CPUs without the need for a dedicated GPU, and additionally offers advantages in ease of deployment and integration. This makes it well-suited for rapid research iteration, large-scale parallel training or fine-tuning, and similar scenarios in the LLM domain (Peng et al. 2023). Meanwhile, TowerMind supports pixel-based, textual, and structured game-state observations, enabling evaluation of multimodal LLMs. A comprehensive comparison of TowerMind and other lightweight RTS game-based environments in terms of supported features is provided in Table 1.

In addition to addressing the limitations of existing RTS game-based environments, the design of TowerMind incorporates two new features: **(1) Hallucination Evaluation**: In evaluating LLMs, our metrics consider not only in-game score as a measure of performance, but also the executability of actions as an indicator of *hallucination*. Hallucination refers to LLM outputs that conflict with factual or contextual information (Bang et al. 2023); in our setting, this specifically denotes actions that are invalid or inconsistent with the game state or rules. Such a metric design allows for simultaneous evaluation of LLM capabilities and reliability; **(2) Customizability**: As both a TD environment and engine, TowerMind includes a graphical level editor that enables researchers to conveniently create custom levels. These levels can range from trivially easy to extremely difficult or structurally unique, supporting diverse research needs and reducing the risk of data contamination.

The contributions of our work are four-fold: (1) We present TowerMind, a lightweight and multimodal TD environment for evaluating long-term planning and decision-making in LLMs, while also supporting hallucination analysis and offering strong customizability. (2) The evaluation results show that while commercial LLMs outperform open-source ones, there remains a significant gap between LLMs and human experts. Furthermore, we observe several behavioural shortcomings during evaluation, including inadequate planning validation, a lack of multifinality in decision-making, and inefficient action use. (3) Based on the experimental results, we discuss three key aspects: how visual input enhances LLM capabilities, how correctness relates to effectiveness, and how LLMs handle misleading information. These findings provide insights for future research and highlight TowerMind's potential as a versatile benchmark. (4) We evaluate two popular RL algorithms, Ape-X DQN (Horgan et al. 2018) and PPO (Schulman et al. 2017), and the results demonstrate that TowerMind is a challenging environment that broadens the diversity of RL benchmarks.

2 Related Work

Long-Term Planning and Decision-Making with LLMs. Recent advances in LLMs have sparked growing interest in their capabilities beyond text generation, particularly in long-term planning and decision-making tasks (Brown et al.

Environment	Pixel Observation	Textual Observation	Stochastic Environments	Partial Observability	Level Editor	Gym Interface
ELF (Tian et al. 2017)	✓	✗	✓	✓	✗	✗
DeepRTS (Andersen, Goodwin, and Granmo 2018)	✓	✗	✗	✓	✗	✗
Gym-μrts (Huang et al. 2021)	✗	✗	✓	✓	✗	✓
Mini HoK (Liu et al. 2024)	✗	✗	✓	✗	✓	✗
TowerMind (Ours)	✓	✓	✓	✓	✓	✓

Table 1: Comparison between TowerMind and other lightweight RTS game-based environments.

2020). ReAct (Yao et al. 2023) and Toolformer (Schick et al. 2023) demonstrate that LLMs can be guided to plan and act through reasoning traces and tool use, respectively. More recent approaches such as AutoGPT (Yang, Yue, and He 2023) and BabyAGI (Calegario et al. 2023) attempt to leverage LLMs in open-ended task planning by forming feedback loops that allow the models to iteratively set subgoals and update plans. Whether originating from LLMs themselves or enabled through prompt engineering or agentic systems, long-term planning and decision-making capabilities need to be systematically evaluated to support further improvement. Different benchmarks adopt various perspectives when evaluating long-term planning and decision-making. PlanGen-LLMs (Wei et al. 2025), AGENTBENCH (Liu et al. 2023), and PLANET (Li et al. 2025) focus on simulated operating systems, web tasks, and other interactive environments. TowerMind, by leveraging the TD genre as a subclass of RTS games, preserves the strengths of RTS-style evaluation while providing a more computationally efficient alternative.

RTS Game-Based Environments and Benchmarks. RTS games have long been one of the game genres most deeply intertwined with AI research (Buro 2003). Currently, available RTS game-based environments include SC2LE (Vinyals et al. 2017), StarCraft Multi-Agent Challenge (SMAC) (Samvelyan et al. 2019), Honor of Kings Arena (HoK) and Honor of Kings 3v3 Arena (HoK3v3) (Wei et al. 2022a), along with several lightweight alternatives designed for lower computational demands, such as ELF (Tian et al. 2017), DeepRTS (Andersen, Goodwin, and Granmo 2018), Gym-μrts (Huang et al. 2021) and Mini Honor of Kings (Mini HoK) (Liu et al. 2024). To facilitate LLM-related research, several RTS game benchmarks featuring text-based observations have been introduced, such as TextStarCraft II (Ma et al. 2025b), LLM-PySC2 (Li et al. 2024), and VLMs Play StarCraft II (Ma et al. 2025a), all of which face relatively high computational demands because they are based on SC2LE. TowerMind serves as a computationally efficient alternative to existing RTS game-based benchmarks for LLM evaluation. Moreover, it extends beyond current lightweight RTS game-based environments by supporting multimodal observations and functionalities that reflect the evolving needs of AI agent research.

Tower Defense Games and AI Research. TD games represent a rule-convergent subgenre of RTS games, characterized by a combination of simple mechanics and substantial strategic depth (Avery et al. 2011). With the advancement of AI, TD games have increasingly attracted attention in AI research. Early work focused on applying AI algorithms to address TD-specific challenges (Rummell 2011; Tan et al. 2013; Wong and Kang 2015), while more recent studies have adopted TD games as testbeds for reinforcement learning (RL) research (Dias, Foleiss, and Lopes 2020; Bergdahl, Sestini, and Gisslén 2024) and human-AI collaboration (Haduong et al. 2024). However, the only TD game available to the research community is a small module in the ELF (Tian et al. 2017) environment, which is highly limited in tower and enemy variety and lacks units control. To date, the research community still lacks a dedicated and sufficiently challenging TD game environment. TowerMind fills this gap and enriches the diversity of benchmarks in the AI agent research domain.

3 The TowerMind Environment

The TowerMind environment is built on top of the Unity game engine and extended into an AI environment using the Unity ML-Agents Toolkit (Juliani et al. 2018).

3.1 Game Mechanics

TowerMind consists of a series of independent levels, each with a distinct map and enemy configuration, where enemies spawn in sequential waves and advance toward the player's base. Players must strategically build towers and control units to prevent enemies from reaching the base and depleting the player's base health.

Maps. In TowerMind, the map is defined as a square area with a side length of 6, centered at $(0.0, 0.0)$, where both the horizontal and vertical coordinates lie within the interval $[-3.0, 3.0]$. It consists of two fundamental elements: roads and tower points. Roads (the red and blue directional curves in Figure 2) are fixed paths that guide enemy movement through the map, represented as sequences of 2D coordinate waypoints along which enemies move in straight lines, often starting from different locations and converging at the player's base. Tower points (the label "F" in Figure 2) are predefined locations on the map where players can construct defensive towers, typically positioned along both sides of the roads. Some tower points, however, are placed far from the roads and cannot engage any enemies, thus serving as misleading tower points. Together, the varying shapes of roads and the diverse locations of tower points form rich and dynamic map features, serving as a critical factor influencing players' long-term planning and decision-making.

Towers, Knights and Hero. Players can control three types of game entities to defend against enemy attacks: towers, knight units, and a hero unit, with all player actions related

to these three entities. There are three types of buildable towers: Archer Tower, Magician Tower, and Knight Tower, each with distinct construction costs, attack styles, and optimal use cases. The Archer Tower deals strong single-target damage, the Magician Tower performs area-of-effect (AoE) attacks, and the Knight Tower summons knight units that can be manually controlled by the player. Knight units (the label "H" in Figure 2) are melee fighters designed for individual combat with enemy units. They can either be summoned from knight towers or directly deployed anywhere on the map via the knight reinforcements action. And in each level, players can control a hero unit with greater health, attack damage, and other attributes than knight units, and can manipulate its movement and skill usage with finer granularity. Overall, towers are relatively static and reflect high-level strategic planning, whereas knight units and the hero unit are more flexible and emphasize low-level tactical decision-making and action execution.

Enemies. Enemies (the label "I" in Figure 2) appear in waves, with each wave consisting of a predefined number of enemies and a fixed time interval between waves. There are 15 distinct enemy types, each exhibiting different attributes in terms of health, movement speed, and attack damage. Some also possess special abilities; for example, the Orc Sorcerer can disable nearby towers. By varying the types and quantities of enemies within each wave, the game generates diverse enemy compositions, making it impossible for players to rely on a single fixed strategy to clear all levels.

Resources. Gold coins (the label "A" in Figure 2) serve as the sole in-game resource and are required for constructing towers, upgrading existing towers, and enhancing the hero's maximum health. Gold coins periodically appear at random locations on the map (the label "G" in Figure 2) and must be actively collected by dispatching knight units or the hero to the corresponding positions. Moreover, when the hero's AoE skill inadvertently eliminates friendly knight units, a Friendly Fire Compensation mechanism is triggered, awarding the player an amount of gold coins as compensation.

Fog of War. In each level, a white, cloud-shaped fog of war region is present and moves randomly across the map (the label "E" in Figure 2), introducing partial observability to the environment. All towers, knight units, the hero unit, and enemies located within the fog of war are excluded from the observation space, and friendly units in these areas become inactive and do not attack enemies. Fog of war increases the difficulty and uncertainty of the environment.

3.2 Environment Interface

The TowerMind environment conforms to the OpenAI Gym standard (Brockman et al. 2016) for easy integration with existing frameworks.

Observations. TowerMind provides three distinct types of observations: pixel-based, textual, and structured game-state observations. The pixel-based observation is a $512 \times 512 \times 3$ colour image representing the raw game screen. The textual and structured game-state observations represent two different formats derived from the same underlying game-state information, which includes both level-specific and real-time status data. For instance, `Level_Initial_Gold_Coins` indicates the initial number of gold coins at the beginning of a level, while `Level_Enemies_Realtime_Status` provides real-time data about each enemy, such as their health points and coordinates. The textual observation explicitly encodes this information in JSON format, clearly associating each piece of data with its corresponding semantic field name, facilitating comprehension by LLMs. In contrast, the structured game-state observation presents the same information flattened into a one-dimensional array.

Actions. The action space in this environment is designed as a *hybrid action space*, where each action a is represented by a three-dimensional vector: $a = (x, y, c)$. Here, the first two dimensions (x, y) are continuous variables specifying the spatial location of the action, expressed as Cartesian coordinates within a two-dimensional plane centered at the map's midpoint, with both horizontal and vertical coordinates constrained by: $x, y \in [-3.0, 3.0]$, consistent with the spatial boundaries of the game map. The third dimension c denotes the discrete action type (e.g., upgrade a tower, sell a tower, dispatch knight reinforcements), represented by an integer index: $c \in \{0, 1, 2, \ldots, 11\}$. This action space design integrates continuous spatial coordinates with discrete action selections, enabling diverse interactions within the environment. Figure 2 provides a more intuitive illustration of the action space. Additionally, only actions that comply with the game rules and current state are considered executable, and we refer to these as *valid actions*. In contrast, actions that violate these constraints, such as issuing a tower-building command at coordinates where no tower point exists or when insufficient gold is available, are classified as invalid and are not executed.

Reward. TowerMind provides a sparse reward signal, assigning a reward of -1.0 for each enemy that reaches the player's base, which aligns with the game mechanic where the player's base health (the label "D" in Figure 2) is reduced by one in the same situation.

Episode Dynamics. Each level in TowerMind is treated as a single finite episode, terminating either when the player's base health reaches zero or when all enemy waves have been eliminated. In all our experiments, actions are executed by default every 16 game steps, corresponding to 187 actions per minute, similar to the action frequency in SC2LE.

3.3 Levels and Difficulty

TowerMind includes five built-in benchmark levels with increasing difficulty from Level 1 to Level 5. To characterize each level, we propose a quantitative metric system grounded in the game mechanics described in Section 3.1, enabling the modeling and comparison of level difficulty. We define the difficulty of a level l in TowerMind as a scalar value $D(l)$, composed of four components: $D(l) = d_r(l) + d_t(l) + d_e(l) + d_{re}(l)$.

Road: $d_r(l) = \frac{R_l}{R_{\max}}$, where R_l is the number of roads in level l, and R_{max} is the design-time maximum number of roads in the TowerMind environment.

Tower: $d_t(l) = \frac{T_l}{T_{\max}}$, where T_l is the number of tower points in level l, and T_{max} is the design-time maximum number of tower points in the TowerMind environment.

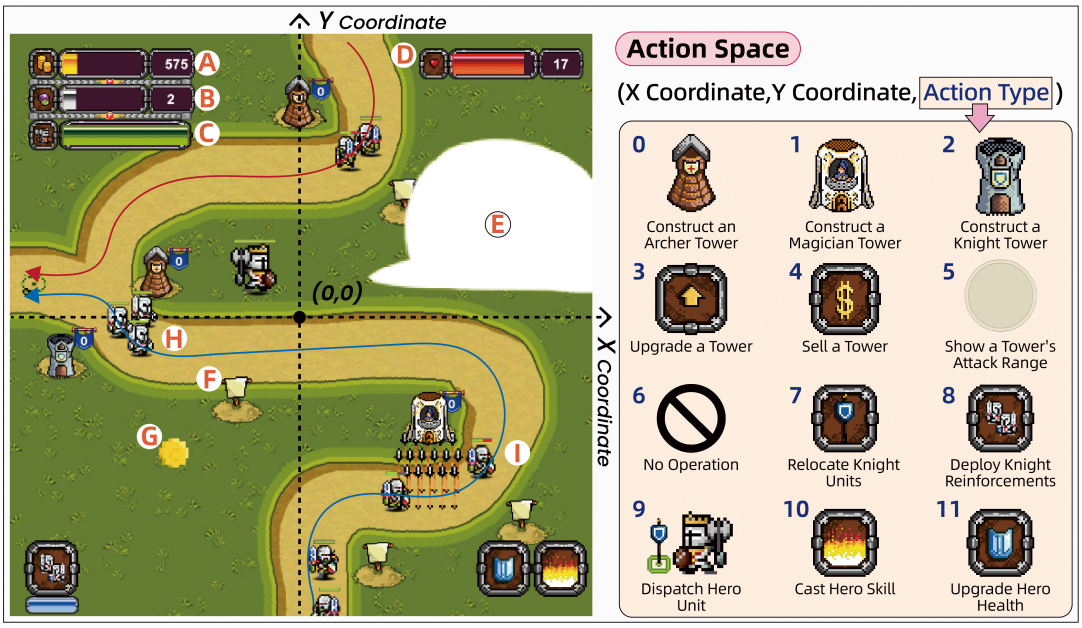

Figure 2: Left: A screenshot of the TowerMind environment with key game elements annotated. The coordinate axes illustrate the alignment between the game map and the 2D coordinate system. The red and blue arrowed curves represent the two roads used by enemies to attack. Labels A–G indicate: (A) player's current gold coins; (B) remaining enemy waves; (C) hero unit's current health; (D) player's base health; (E) fog of war; (F) an unbuilt tower point; and (G) gold coins dropped on the map awaiting collection; (H) knights units; (I) an enemy. Right: Illustrations and brief descriptions of the 12 action types.

Enemy: $d_e(l) = \frac{E_l}{E_{\text{total}}} + \frac{\bar{N}l}{N\max}$, where E_l is the number of enemy types in level l, and E_{total} is the total number of enemy types, $\bar{N}_l$ is the average number of enemies per wave in level l, and N_{max} is the design-time maximum number of enemies per wave in the TowerMind environment.

Resource: $d_{re}(l) = \frac{1}{3}\left(\frac{I_{\min}}{I_l} + \frac{G_{\min}}{G_l} + (1 - r_{\text{sellback}}(l))\right)$, where I_{min} is the design-time minimum initial gold coins in this environment, I_l is the initial gold coins in level l, G_{min} is the design-time minimum gold coins drop in the TowerMind environment, G_l is the gold coins drop amount in level l, and $r_{\text{sellback}}(l)$ is the tower sell-back ratio in level l.

3.4 Environment Customizability

The customizability of TowerMind spans three aspects: (1) **Level Customization**, new levels can be easily created using the graphical level editor; (2) **Parameter Customization**, most game parameters are modifiable, allowing flexible adjustment of towers, enemies, heroes, and other elements; (3) **Feature Customization**, researchers can enable or disable various modes and tools, such as debugging or human trajectory recording.

4 Evaluation for LLMs

4.1 Evaluation Setting

We define two evaluation metrics: **score** and **valid action rate**. The score metric is identical to the raw reward signal provided by the TowerMind environment interface. As all benchmark levels assign the player base a fixed health of 20, the score metric is a real number ranging from –20 to 0. The valid action rate metric is calculated as the ratio of valid actions to total actions within a given level: $\frac{\#\text{Valid Actions}}{\#\text{Total Actions}}$, ranges from 0 to 1.

Our evaluation covers a range of popular commercial and open-source models, including GPT-4.1 (Achiam et al. 2023), Gemini-2.5-Pro (Comanici et al. 2025), Claude 3.7 Sonnet (Anthropic 2025), Llama 3.2 (90B/11B) (MetaAI 2024), and Qwen2.5-VL (72B/7B) (Bai et al. 2025). We employ a zero-shot prompting strategy, using identical prompts across all models to ensure fairness and consistency. The prompt comprises four parts: (1) a natural-language description of the game's objective and rules; (2) a natural-language description of the action space; (3) JSON tables detailing numerical attributes of towers, knights, the hero unit, knight reinforcements, and enemies; and (4) a JSON-formatted observation–action history (default length 3) capturing recent agent–environment interactions. The language-only modality receives only the prompt, whereas the vision-language modality includes both the prompt and a $512 \times 512 \times 3$ pixel-based observation. To support a comprehensive benchmark, we evaluated the performance of five human experts across five benchmark levels to establish a human experts baseline.

4.2 Results

In our experiments, each model was evaluated using five random seeds across each benchmark level under both language-only and vision-language modalities. Tables 2 and 3 respectively present the score and valid action rate performances of different LLMs on each benchmark level. All

values in these tables are normalized relative to the human experts baseline. **Bold** values indicate the best-performing results on each benchmark level under the language-only modality, underlined values denote the best-performing results on each benchmark level under the vision-language modality, and values highlighted in *italic* indicate performance worse than the random baseline.

Model	Lv 1	Lv 2	Lv 3	Lv 4	Lv 5	Avg.
Language-Only						
GPT-4.1	0.59	0.49	0.32	0.19	0.07	0.33
Gemini-2.5-Pro	0.52	0.42	0.31	0.11	0.01	0.27
Claude 3.7 Sonnet	**0.62**	**0.51**	**0.40**	**0.24**	**0.15**	**0.38**
Llama 3.2 90B	0.42	0.32	0.19	0.12	0.00	0.21
Llama 3.2 11B	0.17	0.09	0.00	0.00	0.00	0.05
Qwen 2.5-VL 72B	0.47	0.36	0.21	0.00	0.00	0.21
Qwen 2.5-VL 7B	0.00	0.00	0.00	0.00	0.00	0.00
Vision-Language						
GPT-4.1	0.63	0.56	0.44	0.32	0.15	0.42
Gemini-2.5-Pro	0.57	0.44	0.33	0.16	0.01	0.30
Claude 3.7 Sonnet	0.67	0.58	0.45	0.20	0.16	0.41
Llama 3.2 90B	0.30	0.05	0.00	0.00	0.00	0.07
Llama 3.2 11B	0.04	0.00	0.00	0.00	0.00	0.01
Qwen 2.5-VL 72B	0.54	0.39	0.20	0.12	0.05	0.26
Qwen 2.5-VL 7B	0.05	0.00	0.00	0.00	0.00	0.01
Random	0.00	0.00	0.00	0.00	0.00	0.00

Table 2: The score performance of different LLMs on the benchmark levels.

Model	Lv 1	Lv 2	Lv 3	Lv 4	Lv 5	Avg.
Language-Only						
GPT-4.1	**0.92**	0.89	0.88	0.84	0.75	0.86
Gemini-2.5-Pro	0.91	**0.90**	**0.89**	0.83	**0.82**	**0.87**
Claude 3.7 Sonnet	0.90	0.87	0.85	**0.85**	0.79	0.85
Llama 3.2 90B	0.48	0.39	0.30	*0.21*	*0.20*	0.32
Llama 3.2 11B	0.28	*0.24*	*0.23*	*0.23*	*0.22*	0.24
Qwen 2.5-VL 72B	0.87	0.78	0.76	0.58	0.51	0.70
Qwen 2.5-VL 7B	*0.11*	*0.05*	*0.03*	*0.01*	*0.01*	*0.04*
Vision-Language						
GPT-4.1	0.86	0.81	0.75	0.68	0.66	0.75
Gemini-2.5-Pro	0.85	0.81	0.80	0.73	0.67	0.77
Claude 3.7 Sonnet	0.85	0.85	0.83	0.80	0.79	0.82
Llama 3.2 90B	0.44	0.38	0.33	0.31	0.30	0.35
Llama 3.2 11B	0.31	*0.19*	*0.18*	*0.13*	*0.11*	*0.18*
Qwen 2.5-VL 72B	0.79	0.72	0.66	0.54	0.43	0.63
Qwen 2.5-VL 7B	*0.21*	*0.15*	*0.05*	*0.04*	*0.02*	*0.09*
Random	0.25	0.25	0.24	0.24	0.22	0.24

Table 3: The valid action rate performance of different LLMs on the benchmark levels.

4.3 Quantitative Analysis

Based on the data in Tables 2 and 3, including both individual benchmark level results and overall averages, we identify the following key findings:

Limited Performance of LLMs. In terms of score, Claude 3.7 Sonnet achieved the best performance in the language-only setting, and GPT-4.1 in the vision-language setting. However, they still lag behind human experts by 62% and 58%, with all other models showing even larger gaps. Notably, on the most difficult level, Level 5, all models underperform human experts by at least 84%.

Vision Input Improves Performance. All evaluated models except Llama 3.2 (90B/11B) show improved score performance under the vision-language modality compared to the language-only modality. This suggests that multimodal cues enhance these models' environmental understanding, whereas Llama 3.2 (90B/11B) seems to struggle with such complex and dynamic visual inputs. This degradation may stem from the model being primarily optimized for static image understanding rather than continuous, temporally evolving scenes, which limits its ability to interpret dynamic environments effectively.

Hallucination Issues in Open-Source LLMs. From the perspective of valid action rate, the three commercial LLMs performed relatively well, each exhibiting a gap of less than 20% compared to human experts. Among the open-source models, only Qwen 2.5-VL 72B showed acceptable results, while the other three models underperformed significantly. In particular, the smaller models Qwen 2.5-VL 7B and Llama 3.2 11B exhibited performance on several benchmark levels that was even lower than the random baseline. The high level of hallucination constrains their long-term planning and decision-making capabilities.

Effect of Level Difficulty on Hallucination. As level difficulty increases, the degree of hallucination also rises across models. This suggests that harder levels tend to include more game elements, resulting in longer prompts that challenge the models' generation stability and consistency.

4.4 Qualitative Analysis

We analyzed model trajectories and identified common challenges and limitations:

Insufficient Validation of Long-Term Planning. In Levels 1 and 2, we placed one or more misleading tower points located far from the enemy attack roads. Building towers on these tower points does not threaten any enemies, thus serving only to waste resources. However, the LLMs consistently chose to build towers on these misleading tower points. Despite having access to all necessary information in the prompt to compute that these towers would not engage any enemies, the models failed to perform such basic spatial or numerical reasoning during tower placement planning.

Decision-Making Without Multifinality Thinking. Multifinality refers to the ability to achieve multiple goals with a single action (Kruglanski et al. 2015), and it is a key decision-making skill for optimizing task efficiency. This type of behavior is frequently observed in human expert gameplay. For example, human experts may direct the hero unit to collect gold coins while simultaneously attacking nearby enemies. However, we have never observed such behavior in the gameplay trajectories of any LLMs.

Limited Use and Understanding of Actions. We frequently observe that LLMs fail to fully utilize or understand the available action space. Typical behaviors include neglecting to upgrade towers despite sufficient gold, sending knight reinforcements to empty areas, or using the hero's

AoE skill in the absence of enemies. This suggests that LLMs tend to interpret the available actions only at a surface level, lacking a deeper understanding of their strategic use and appropriate contexts.

4.5 Insights and Future Directions

Based on the above quantitative and qualitative analyses, we attempt to further discuss the findings in this section, with the hope of providing insights that may inspire future research in the LLM domain.

Effect of Visual Input on Model Performance. The majority of LLMs evaluated in our experiments exhibited improved performance under the vision-language modality compared to the language-only setting. This indicates that LLMs can leverage visual inputs to access information beyond what is conveyed through text, thereby enhancing their reasoning, planning, and decision-making capabilities. Consequently, future research may benefit from further exploring the role of visual information, such as through vision-informed prompt engineering or preprocessing techniques for visual feature extraction.

From Correctness to Effectiveness. In our experiments, we found that the gap between LLMs and human experts in terms of valid action rate is smaller than the gap in score. This suggests that LLMs are capable of understanding the game rules and current game state, and can generate actions that are consistent with both. However, these actions are often limited in their effectiveness toward achieving the intended goals. This is analogous to a common issue observed in LLM-based question answering, where models often produce 'technically correct but ultimately unhelpful' responses. Accordingly, evaluating LLMs in the future should go beyond static knowledge benchmarks like SuperGLUE (Wang et al. 2019) and MMLU (Hendrycks et al. 2020), which test correctness, and increasingly incorporate interactive benchmarks such as AGENTBENCH (Liu et al. 2023) and TowerMind, which assess the effectiveness of model-generated responses in dynamic, decision-making settings.

Identifying Misleading Content. In our experiments, even the most advanced LLMs were observed to waste resources by building towers at misleading tower points that would never engage any enemies. The ability to identify misleading information is critical not only for improving the performance of LLMs, but also for ensuring their safety. LLMs need mechanisms to prevent being misled into producing toxic content (Bianchi and Zou 2024). This highlights the necessity of incorporating validation mechanisms into LLM systems. Importantly, these mechanisms should be outcome-driven, meaning that validation should extend beyond surface-level textual content to include assessment of the effects or implications of the generated content.

5 RL Benchmark

To validate TowerMind's feasibility and challenge in RL, we established a preliminary RL benchmark by adopting two widely-used algorithms as baselines: Ape-X DQN (Horgan et al. 2018) and PPO (Schulman et al. 2017). We trained two algorithms on the five benchmark levels using pixel-based and structured game-state observations. Each algorithm was run three times with different random seeds, using 100 million environment steps per run, as shown in Figure 4. We evaluated the trained models on the benchmark levels using five different random seeds, following the same score metric used in the LLM-based evaluation, as shown in Figure 3.

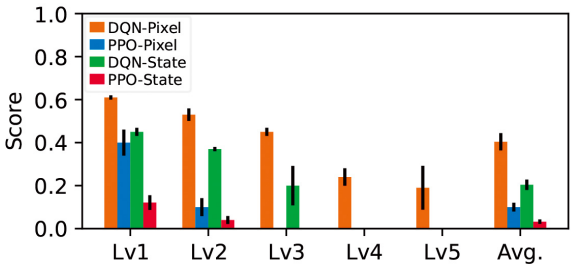

Figure 3: The evaluation results on the benchmark levels, with scores normalized relative to the human expert. Error bars represent the standard error.

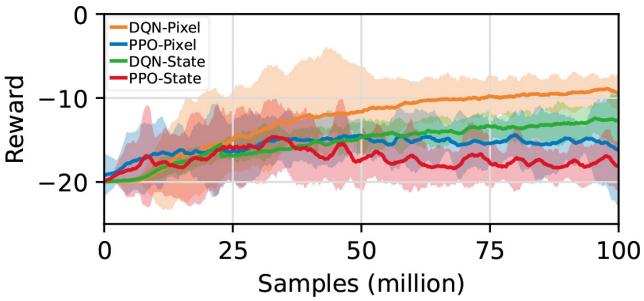

Figure 4: Training curves, the horizontal axis shows the number of training samples, measured in millions. Error bars represent 95% confidence intervals.

The evaluation results indicate that, after 100 million environment steps, both RL algorithms were able to solve simpler levels to some extent, but their performance remained substantially inferior to that of human experts, which suggests that TowerMind is a challenging environment for RL.

6 Conclusion

In this work, we propose TowerMind, a lightweight TD game environment with multimodal observation capabilities designed for evaluating LLMs. Through our evaluation, TowerMind reveals a substantial performance gap between current LLMs and human experts in terms of long-term planning and decision-making. It also clearly demonstrates the roles of visual input, hallucination, and misleading information in contributing to this gap, highlighting TowerMind's practical value for LLM research. Additionally, TowerMind can also be used for RL research. In future work, we plan to incorporate audio into the observation space to further enhance the multimodal richness of TowerMind. We believe that TowerMind can serve as a practical platform to facilitate the development of more capable AI agents.

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Andersen, P.-A.; Goodwin, M.; and Granmo, O.-C. 2018. Deep RTS: a game environment for deep reinforcement learning in real-time strategy games. In *2018 IEEE conference on computational intelligence and games (CIG)*, 1–8. IEEE.

Anthropic. 2025. Claude 3.7 Sonnet and Claude Code. https://www.anthropic.com/news/claude-3-7-sonnet. Accessed: 2025-04-21.

Avery, P.; Togelius, J.; Alistar, E.; and van Leeuwen, R. P. 2011. Computational intelligence and tower defence games. In *2011 IEEE Congress of Evolutionary Computation (CEC)*, 1084–1091.

Bai, S.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; Song, S.; Dang, K.; Wang, P.; Wang, S.; Tang, J.; et al. 2025. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Bang, Y.; Cahyawijaya, S.; Lee, N.; Dai, W.; Su, D.; Wilie, B.; Lovenia, H.; Ji, Z.; Yu, T.; Chung, W.; et al. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023*.

Barros e Sá, G. C.; and Madeira, C. A. G. 2025. Deep reinforcement learning in real-time strategy games: a systematic literature review. *Applied Intelligence*, 55(3): 243.

Bergdahl, J.; Sestini, A.; and Gisslén, L. 2024. Reinforcement Learning for High-Level Strategic Control in Tower Defense Games. In *2024 IEEE Conference on Games (CoG)*, 1–8. IEEE.

Bianchi, F.; and Zou, J. 2024. Large language models are vulnerable to bait-and-switch attacks for generating harmful content. *arXiv preprint arXiv:2402.13926*.

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. OpenAI Gym, June 2016. *arXiv preprint arXiv:1606.01540*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Buro, M. 2003. Real-time strategy games: A new AI research challenge. In *IJCAI*, volume 2003, 1534–1535.

Calegario, F.; Burégio, V.; Erivaldo, F.; Andrade, D. M. C.; Felix, K.; Barbosa, N.; Lucena, P. L. d. S.; and França, C. 2023. Exploring the intersection of Generative AI and Software Development. *arXiv preprint arXiv:2312.14262*.

Çelen, A.; Han, G.; Schindler, K.; Van Gool, L.; Armeni, I.; Obukhov, A.; and Wang, X. 2024. I-design: Personalized llm interior designer. *arXiv preprint arXiv:2404.02838*.

Comanici, G.; Bieber, E.; Schaekermann, M.; Pasupat, I.; Sachdeva, N.; Dhillon, I.; Blistein, M.; Ram, O.; Zhang, D.; Rosen, E.; et al. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 4171–4186.

Dias, A.; Foleiss, J.; and Lopes, R. P. 2020. Reinforcement learning in tower defense. In *International Conference on Videogame Sciences and Arts*, 127–139. Springer.

Dubois, Y.; Galambosi, B.; Liang, P.; and Hashimoto, T. B. 2024. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*.

Haduong, N.; Wang, I.; Lu, B.-R.; Ammanabrolu, P.; and Smith, N. A. 2024. CPS-TaskForge: Generating Collaborative Problem Solving Environments for Diverse Communication Tasks. In *Proceedings of the 1st Workshop on Customizable NLP: Progress and Challenges in Customizing NLP for a Domain, Application, Group, or Individual (CustomNLP4U)*, 86–112.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Horgan, D.; Quan, J.; Budden, D.; Barth-Maron, G.; Hessel, M.; van Hasselt, H.; and Silver, D. 2018. Distributed prioritized experience replay. CoRR abs/1803.00933 (2018). *arXiv preprint arXiv:1803.00933*.

Huang, S.; Ontañón, S.; Bamford, C.; and Grela, L. 2021. Gym-μrts: Toward affordable full game real-time strategy games research with deep reinforcement learning. In *2021 IEEE Conference on Games (CoG)*, 1–8. IEEE.

Juliani, A.; Berges, V.-P.; Teng, E.; Cohen, A.; Harper, J.; Elion, C.; Goy, C.; Gao, Y.; Henry, H.; Mattar, M.; et al. 2018. Unity: A general platform for intelligent agents. *arXiv preprint arXiv:1809.02627*.

Koc, V. 2025. Tiny QA Benchmark++: Ultra-Lightweight, Synthetic Multilingual Dataset Generation & Smoke-Tests for Continuous LLM Evaluation. *arXiv preprint arXiv:2505.12058*.

Kruglanski, A. W.; Chernikova, M.; Babush, M.; Dugas, M.; and Schumpe, B. M. 2015. The architecture of goal systems: Multifinality, equifinality, and counterfinality in means—end relations. In *Advances in motivation science*, volume 2, 69–98. Elsevier.

Li, H.; Chen, Z.; Zhang, J.; and Liu, F. 2025. PLANET: A Collection of Benchmarks for Evaluating LLMs' Planning Capabilities. *arXiv preprint arXiv:2504.14773*.

Li, J.; Lai, Y.; Li, W.; Ren, J.; Zhang, M.; Kang, X.; Wang, S.; Li, P.; Zhang, Y.-Q.; Ma, W.; et al. 2024. Agent hospital: A simulacrum of hospital with evolvable medical agents. *arXiv preprint arXiv:2405.02957*.

Li, Z.; Ni, Y.; Qi, R.; Jiang, L.; Lu, C.; Xu, X.; Liu, X.; Li, P.; Guo, Y.; Ma, Z.; Guo, X.; Huang, K.; and Zhang, X. 2024. LLM-PySC2: Starcraft II learning environment for Large Language Models. *arXiv e-prints*, arXiv:2411.05348.

Lingo, R.; Arroyo, M.; and Chhajer, R. 2024. Enhancing llm problem solving with reap: Reflection, explicit problem deconstruction, and advanced prompting. *arXiv preprint arXiv:2409.09415*.

Liu, L.; Zhao, J.; Hu, C.; Cao, Z.; Zhao, Y.; Ye, Z.; Meng, M.; Wang, W.; He, Z.; Li, H.; et al. 2024. Mini Honor of Kings: A Lightweight Environment for Multi-Agent Reinforcement Learning. *arXiv preprint arXiv:2406.03978*.

Liu, S.; Chaoran, L.; Yue, L.; Heng, M.; Xiao, H.; Yiming, S.; Licong, W.; Ze, C.; Xianghao, G.; Hengtong, L.; et al. 2019. Automatic generation of tower defense levels using PCG. In *Proceedings of the 14th International Conference on the Foundations of Digital Games*, 1–9.

Liu, X.; Yu, H.; Zhang, H.; Xu, Y.; Lei, X.; Lai, H.; Gu, Y.; Ding, H.; Men, K.; Yang, K.; et al. 2023. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*.

Ma, W.; Fu, Y.; Zhang, Z.; and Li, G. 2025a. VLMs Play StarCraft II: A Benchmark and Multimodal Decision Method. *arXiv preprint arXiv:2503.05383*.

Ma, W.; Mi, Q.; Zeng, Y.; Yan, X.; Lin, R.; Wu, Y.; Wang, J.; and Zhang, H. 2025b. Large language models play starcraft ii: Benchmarks and a chain of summarization approach. *Advances in Neural Information Processing Systems*, 37: 133386–133442.

MetaAI. 2024. Llama 3.2: Revolutionizing edge AI and vision with open, customizable models. https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/. Accessed: 2025-04-07.

Peng, B.; Li, C.; He, P.; Galley, M.; and Gao, J. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Renze, M.; and Guven, E. 2024. Self-reflection in llm agents: Effects on problem-solving performance. *arXiv preprint arXiv:2405.06682*.

Rummell, P. A. 2011. Adaptive ai to play tower defense game. In *2011 16th International Conference on Computer Games (CGAMES)*, 38–40. IEEE.

Russell, S. J.; and Norvig, P. 2016. *Artificial intelligence: a modern approach*. pearson.

Samvelyan, M.; Rashid, T.; De Witt, C. S.; Farquhar, G.; Nardelli, N.; Rudner, T. G.; Hung, C.-M.; Torr, P. H.; Foerster, J.; and Whiteson, S. 2019. The starcraft multi-agent challenge. *arXiv preprint arXiv:1902.04043*.

Schick, T.; Dwivedi-Yu, J.; Dessì, R.; Raileanu, R.; Lomeli, M.; Hambro, E.; Zettlemoyer, L.; Cancedda, N.; and Scialom, T. 2023. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36: 68539–68551.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Tan, T. G.; Yong, Y. N.; Chin, K. O.; Teo, J.; and Alfred, R. 2013. Automated evaluation for AI controllers in tower defense game using genetic algorithm. In *International Multi-Conference on Artificial Intelligence Technology*, 135–146. Springer.

Tian, Y.; Gong, Q.; Shang, W.; Wu, Y.; and Zitnick, C. L. 2017. Elf: An extensive, lightweight and flexible research platform for real-time strategy games. *Advances in Neural Information Processing Systems*, 30.

Vinyals, O.; Ewalds, T.; Bartunov, S.; Georgiev, P.; Vezhnevets, A. S.; Yeo, M.; Makhzani, A.; Küttler, H.; Agapiou, J.; Schrittwieser, J.; et al. 2017. Starcraft ii: A new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782*.

Wang, A.; Pruksachatkun, Y.; Nangia, N.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Wang, B.; Deng, X.; and Sun, H. 2022. Iteratively prompt pre-trained language models for chain of thought. *arXiv preprint arXiv:2203.08383*.

Wei, H.; Chen, J.; Ji, X.; Qin, H.; Deng, M.; Li, S.; Wang, L.; Zhang, W.; Yu, Y.; Linc, L.; et al. 2022a. Honor of kings arena: an environment for generalization in competitive reinforcement learning. *Advances in Neural Information Processing Systems*, 35: 11881–11892.

Wei, H.; Zhang, Z.; He, S.; Xia, T.; Pan, S.; and Liu, F. 2025. Plangenllms: A modern survey of llm planning capabilities. *arXiv preprint arXiv:2502.11221*.

Wei, J.; Tay, Y.; Bommasani, R.; Raffel, C.; Zoph, B.; Borgeaud, S.; Yogatama, D.; Bosma, M.; Zhou, D.; Metzler, D.; et al. 2022b. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Wong, A. M. H.; and Kang, D.-K. 2015. Game layout and artificial intelligence implementation in mobile 3D tower defence game. *International Journal of Security and Networks*, 10(1): 42–47.

Yang, H.; Yue, S.; and He, Y. 2023. Auto-gpt for online decision making: Benchmarks and additional opinions. *arXiv preprint arXiv:2306.02224*.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Yuan, W.; Pang, R. Y.; Cho, K.; Sukhbaatar, S.; Xu, J.; and Weston, J. 2024. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 3.

Zhang, X.; Luo, S.; Zhang, B.; Ma, Z.; Zhang, J.; Li, Y.; Li, G.; Yao, Z.; Xu, K.; Zhou, J.; et al. 2024. Tablellm: Enabling tabular data manipulation by llms in real office usage scenarios. *arXiv preprint arXiv:2403.19318*.