MINE ODYSSEY: BENCHMARKING SPATIAL AGENTIC INTELLIGENCE IN THE WILD

Anonymous authors
Paper under double-blind review

ABSTRACT

Advances in foundation models are driving efforts to introduce agents to assist people in the physical world. Such agents require agentic spatial intelligence: exploring unfamiliar environments, updating spatial understanding through interaction, and adapting actions based on feedback to sustain progress toward a sequence of goals. Evaluating these capabilities requires environments that reflect the diversity and structural complexity of real-world places, yet existing benchmarks typically cover only a limited range of spatial layouts, scales, and traversal requirements. To address this gap, we introduce **MINE ODYSSEY**, a benchmark for evaluating agentic spatial intelligence using Minecraft reconstructions of real-world locations. It comprises 180 tasks covering 30 such locations across 20 countries and regions on five continents, including 20 outdoor and 10 indoor settings. These settings are selected to provide broad coverage across spatial scales, layouts, terrains, and connectivity patterns, ranging from the urban streets of Midtown Manhattan to the rural Entrup, and from the steep terrain of Santa Lucía Hill to Buckingham Palace. Within each reconstruction, we select meaningful locations as waypoints (*e.g.*, historic landmarks, buildings, and rooms) and manually verify their accessibility. Each task provides a natural-language instruction specifying which waypoints to visit and in what order. Completing these tasks requires agents to find accessible routes and entrances, open doors, and move between levels using stairs and ladders, while monitoring their progress and recovering from navigation errors. Across eight evaluated state-of-the-art models, GPT-6 Astra achieves the highest success rate of 85.6%. However, the second-best model, Claude Opus 5, completes only 50.6% of tasks, while the strongest evaluated open-weight model, DeepSeek-V4.1-Flash, reaches 23.9%, highlighting substantial room for improvement in the agentic spatial intelligence of current models. Comprehensive analyses and ablation studies on MINE ODYSSEY reveal current models' limitations and provide insights for advancing agentic spatial intelligence.

1 INTRODUCTION

Advances in foundation models are opening new possibilities for agents to move beyond digital workflows and interact with the physical world (Gemini Robotics Team et al., 2025). Recent evaluations of frontier models such as GPT-6 Astra (OpenAI, 2026b) using robot-control tools have begun to explore this possibility in embodied settings (RoboDojo Team, 2026). Potential applications for such agents range from robotic guide dogs that assist people with visual impairments (Hayamizu et al., 2026) to autonomous delivery robots that transport goods (Wang et al., 2025). In these settings, agents must adapt their movements and interactions based on the spatial relationships in their surroundings, including how spaces connect and which objects and locations are reachable. We refer to this capability as *agentic spatial intelligence*: the ability to proactively acquire and continually update spatial information through interaction, use it to guide observation and action, and adapt those actions as new evidence emerges (Pei et al., 2026; Deng et al., 2026).

Existing benchmarks often evaluate agents in relatively narrow settings, such as household environments (Yang et al., 2025b; Qiao et al., 2026) or urban navigation (Wang et al., 2025; Dalal et al., 2026; Wang et al., 2026b; Ju et al., 2026b), leaving their ability to operate across a broader range of spatial structures and traversal requirements insufficiently tested. However, real-world tasks often arise in environments with varied layouts and traversal requirements. An agent may need to follow

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

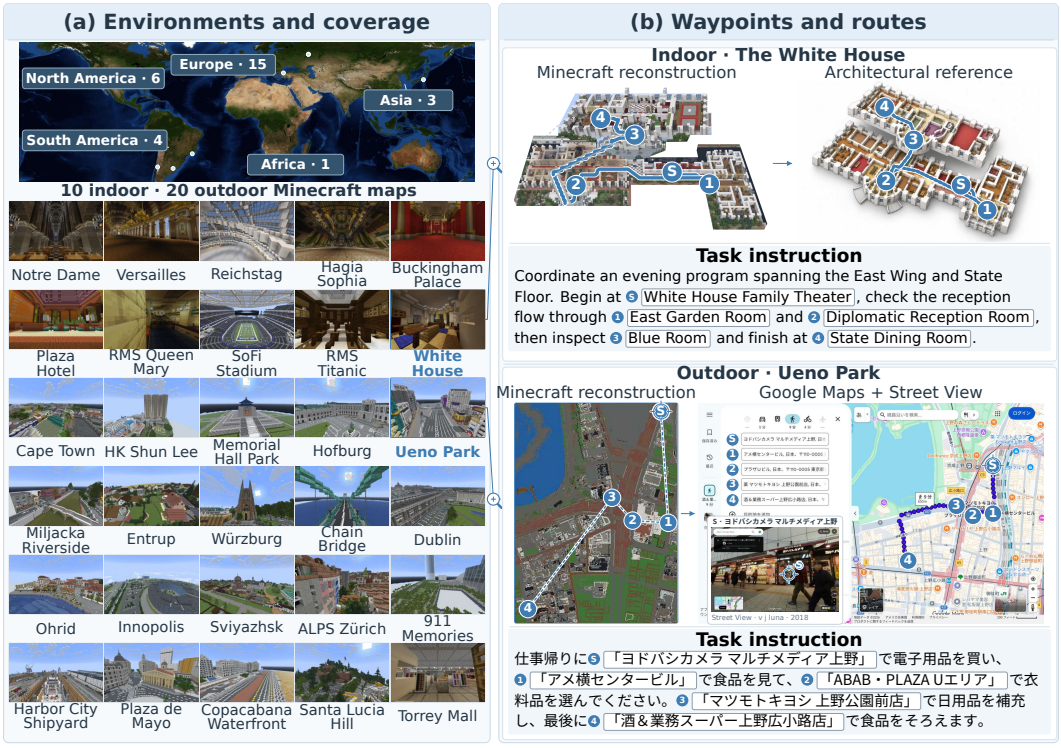

Figure 1: Overview of MINE ODYSSEY environments and task construction. (a) 30 Minecraft reconstructions and geographic coverage. (b) Task construction with White House and Ueno Park examples: reference locations are mapped to waypoints and ordered into navigation tasks. S denotes the start; numbered markers and dashed lines indicate visit order.

irregular paths through a rural settlement, find a traversable route across a steep hillside, or navigate entrances, corridors, and multiple levels within a palace or stadium. These environments differ not only in appearance, but also in scale, connectivity, access constraints, and elevation. This diversity, together with the need to maintain spatial information and revise plans throughout execution, makes reliable performance across environments particularly challenging.

To address these challenges, we introduce MINE ODYSSEY, a benchmark for evaluating agentic spatial intelligence through diverse, realistic, and long-horizon navigation tasks in Minecraft reconstructions of real-world locations. Here we choose Minecraft as our evaluation platform for its rich ecosystem of community-created mods and world saves. Detailed reconstructions of cities, landscapes, and buildings from around the world provide diverse spatial environments. The platform also allows episodes to be reset to specified initial conditions and task completion to be verified automatically from the agent's in-game position. MINE ODYSSEY spans 30 real-world locations recreated in Minecraft, with 180 tasks across 20 countries and regions on five continents, including 20 outdoor and 10 indoor environments. To capture the diversity and structural complexity of real-world spatial environments, we curate spots spanning dense urban districts, rural settlements, mountainous terrain, palace complexes, stadiums, and historic passenger ships. Representative examples include Midtown Manhattan, Cape Town, Entrup, Santa Lucía Hill, Buckingham Palace, SoFi Stadium, and RMS Queen Mary. These places provide variation in street connectivity, terrain, room arrangement, entrances, and vertical circulation. Minecraft makes these structures interactive through first-person movement, collision, and actions such as opening doors and climbing ladders and stairs.

Since these maps don't come with annotated destinations for evaluation, we mark named locations as waypoints that serve as destinations or intermediate stops in navigation tasks. To preserve realism and semantic relevance, we select recognizable places such as historic landmarks, palace rooms, stadium facilities, and building entrances rather than arbitrary points on the map. Each task is then defined as an ordered sequence of these waypoints, requiring agents to find feasible paths and execute the movements and interactions needed to reach each destination in order.

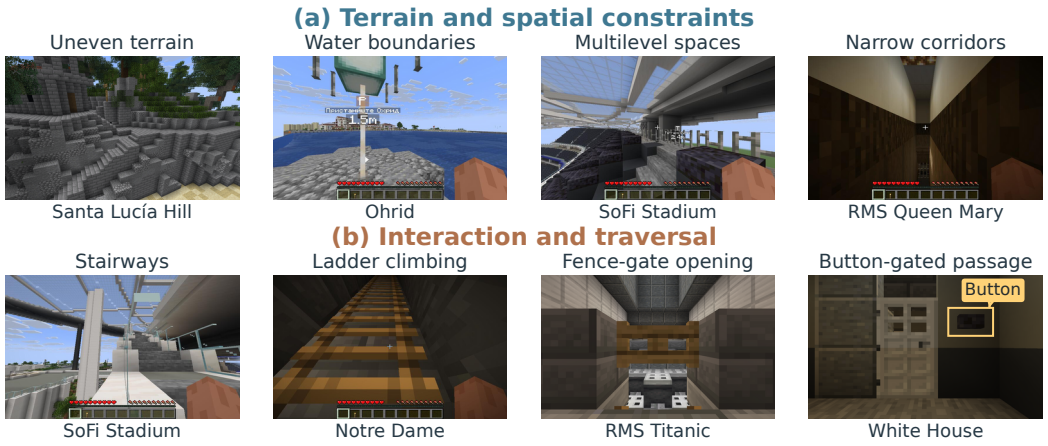

Figure 2: Representative terrain, spatial constraints, and interactions across the diverse collection of maps in MINE ODYSSEY.

We also provide a programmable agent framework that connects visual observations with game APIs and keyboard–mouse control, allowing agents to compose and execute their own action sequences. Agents can inspect intermediate feedback and interrupt ongoing commands, supporting closed-loop interaction as the environment evolves. Isolated execution prevents access to hidden task data and interference between runs, while independent arrival checks support automated, parallel evaluation.

Our evaluation reveals substantial differences in task success and decision efficiency. GPT-6 Astra (OpenAI, 2026b) achieves an 85.6% task success rate, compared with 50.6% for Claude Opus 5 (Anthropic, 2026) and 23.9% for the strongest open-weight model, DeepSeek-V4.1-Flash (DeepSeek, 2026). Performance differences also depend on the environment, with a larger success-rate gap between Astra and Opus indoors than outdoors. In more than half of failed attempts, agents do not reach even the first required destination. We further conduct ablation analysis and case studies to better understand the strengths and limitations of current agents. Our task suite and evaluation framework will also be publicly released to support future research on agentic spatial intellgence in diverse real-world environments.

2 Tasks Environment and Evaluation Framework

2.1 Environments

Game Details Minecraft has been widely adopted as a research platform with its interactive 3D environments and consistent physical mechanics (Guss et al., 2019; Fan et al., 2022; Guo et al., 2025; Ju et al., 2026a). Its open community ecosystem also allows us to integrate diverse, community-built worlds into a unified testbed. Diverse mods provide fine-grained control over agent observation and interaction and enable automatic verification of task outcomes using in-game agent positions. Our experiments adopt Minecraft Java Edition 1.21.11 with NeoForge 21.11.44. We also install these mods: *Xaero's World Map* 1.44.2 for full-screen map, *Xaero's Minimap* 26.4.2 for waypoint markers, *BoccHUD* 0.4.1 with *MaFgLib* 0.4 for HUD, and *LambDynamicLights* 4.9.1 for dynamic lighting. By default, agent in each task begins at a specified position and orientation in first-person view, with a nominal 100° field of view, frozen noon lighting, clear weather, Adventure mode, Peaceful difficulty, and only the *Xaero's World Map* and *LambDynamicLights* are enabled. Additional configurations and restrictions are detailed in Appendix C.1.

Maps We source our maps from BuildTheEarth,[1] a global community project dedicated to reconstructing real-world locations in Minecraft. We aim to evaluate agents in environments that reflect the diversity and spatial realism of real-world places while preserving experimental control. As shown in Figure 1, we use 30 existing Minecraft builds modeled on locations across five continents,

[1] https://buildtheearth.net

spanning 20 different countries and regions. More detailed statistics and can be found in Appendix B. The collection includes 20 outdoor and 10 indoor environments. Outdoor maps cover residential and commercial districts, parks, hillsides, and waterfronts, varying in street connectivity, landmark density, and relief. Indoor maps include palaces, religious buildings, hotels, stadiums, and passenger ships, with variation in room arrangement, corridor structure, entrances, and vertical circulation. These environments present diverse terrain and spatial constraints, including uneven ground, water boundaries, narrow corridors, and multilevel spaces. Navigating them can involve ascending stairs, climbing ladders, opening fence gates, or activating buttons to pass through doors, as illustrated in Figure 2. Versioned world saves allow each task to begin from a consistent map state.

Waypoints We use waypoints to represent the destinations and intermediate stops that an agent may be asked to visit. Each waypoint pairs a human-readable place name with a three-dimensional arrival position in a specific Minecraft world. Our waypoints include landmarks, shops, public facilities, transport entrances, and indoor rooms, covering destinations with different functions and access conditions. We construct each waypoint in three stages. First, an annotator identifies a named place and its approximate real-world location using geographic or venue-specific references, including Google Maps, OpenStreetMap, and site maps. Second, the annotator finds the corresponding place in the Minecraft reconstruction by matching the surrounding street or room layout and visible cues such as building facades, distinctive structures, and signs. Third, the annotator enters the world, visits the matched place, and creates the waypoint at an observed, reachable standing position on the correct floor. During task execution, the agent receives the waypoint name and can access its visual map marker via map, whereas the precise three-dimensional arrival position remains evaluator-only. A waypoint marker will appears in the first-person view when the agent is within 32 blocks of the waypoint, providing local visual guidance toward the destination. The catalog contains 1,059 waypoints, of which 715 distinct identifiers occur in the task pool.

Tasks As shown in Figure 1(b), each task specifies a map, a starting waypoint and orientation, and an ordered sequence of destinations described in a natural-language instruction. The agent must identify and visit these destinations in order, choosing its own route using the permitted controls and ending at the final waypoint. To construct tasks, we provide large language models with waypoint names, place types, and map descriptions to generate candidate task instructions and stop orders with a specific role and purpose. We then manually select and refine these candidates based on three criteria: *semantic coherence*, so that the stops and their order fit the stated purpose; *route plausibility*, avoiding redundant visits, unnecessary backtracking, and implausibly short segments; and *diversity*, covering varied roles, scenarios, and navigation requirements. To ensure task diversity, we vary starting points, distances between stops, elevation changes, and interactions such as opening doors, pressing buttons, or climbing ladders, while limiting repeated ordered waypoint pairs within each map. yx To guarantee task feasibility, we first use Baritone (a Minecraft pathfinding and navigation mod) for a preliminary reachability check, then manually traverse the task routes in the game to verify the reachability. The resulting collection contains 180 tasks with 2–12 stops each.

2.2 VERIFIER.

To complete a task, the agent must reach the specified destinations in order. An independent evaluator is used to verify arrivals by comparing the player's position with the annotated waypoint coordinates. It samples the client player position at 1 Hz throughout execution, including during model inference. The agent is considered to have reached the next required waypoint g_i when its sampled position $p \in \mathbb{R}^3$ satisfies

$$\|p - g_i\|_2 < 3.5 \quad \text{and} \quad |p^y - g_i^y| \leq 1.5$$

All distances are measured in Minecraft blocks. The vertical constraint reduces ambiguity between nearby floors. Only the next required waypoint is eligible for credit; out-of-order visits do not advance task progress. An episode can terminate early with failure under three conditions: (1) *Death*: the player's health reaches zero after taking damage, for example from a fall; (2) *Leaving the game*: the player exits or disconnects from the game world; (3) *Out of bounds*: the player falls beyond the map boundary and cannot return to continue the task. On maps with a configured height boundary, this is detected when the player remains at or below that height for 15 consecutive seconds. After visiting all required destinations, the agent must submit an accepted `claim_done` to complete the task successfully.

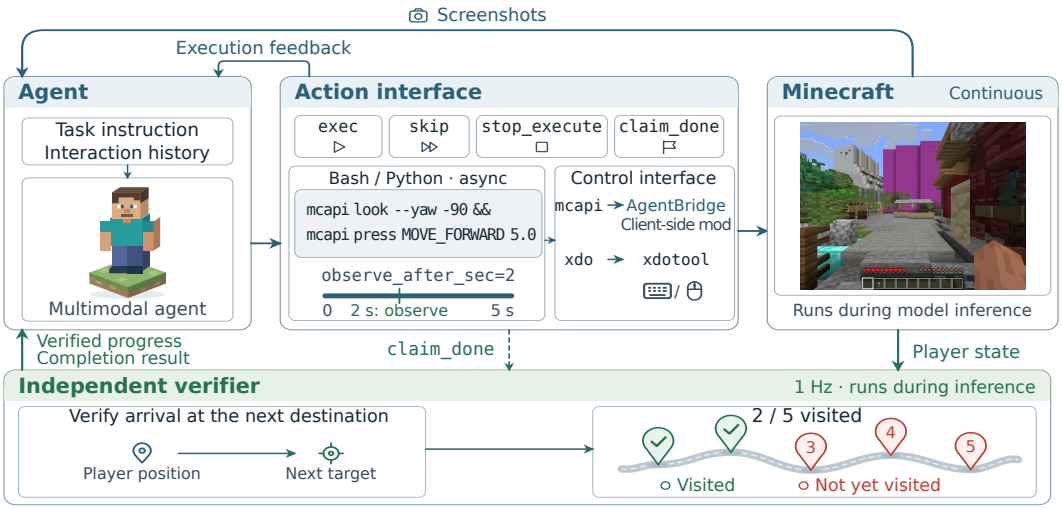

Figure 3: Agent interaction and independent verification in MINE ODYSSEY.

2.3 Agent Framework

Figure 3 shows how the agent interacts with the Minecraft client through the action interface and receives feedback from the independent verifier.

Action Space To enable the agent to control a player and interact with the Minecraft client, we develop a programmable control interface that combines API-based player control with keyboard and mouse input. Specifically, we implement *AgentBridge*, a client-side mod that uses native Minecraft APIs to support player actions such as movement, camera rotation, and item interaction. A command-line client, *mcapi*, sends commands to AgentBridge and returns execution results to the agent. To complement these actions with GUI interactions, we provide *xdo*, a command-line wrapper around `xdotool` that enables the agent to operate the client's graphical interface through keyboard and mouse. The agent uses these tools through Bash, either executing individual operations or composing multiple operations within a single Bash tool call with shell commands or Python scripts. For example, the agent may run `mcapi look --yaw -90 && mcapi press MOVE_FORWARD 2.0` to turn the camera and then move forward for two seconds. Scripts can include calculations, loops, and conditional logic, allowing the agent to execute complex action sequences.

Observation Space The agent receives screenshots of the Minecraft client as, showing either the player's first-person view or an open interface, such as the map or inventory. The in-game map provides a top-down view of the explored environment and supports panning, zooming or searching to inspect different areas. In addition to screenshots, the agent receives textual feedback from command execution, including returned data, execution status, and error messages.

Agent Loop Given these observation and control interfaces, the agent interacts with the environment through an observation–action loop. The game simulation runs asynchronously with this loop: the environment evolves continuously without waiting for the agent to finish reasoning or issue its next action. At each decision step, the agent selects one of four action types based on the latest observations, the task instruction, and its interaction history: (1)*exec*: execute a Bash command, interrupting any command still running. It can run game-control tools such as `mcapi` and `xdo`, execute Python scripts, or perform computations and file operations. Invalid Bash commands return textual errors, while `mcapi state` queries return textual results; both are returned immediately without waiting for the observation interval. (2)*skip*: wait for further observations while allowing the current command to continue. (3)*stop_execute*: terminate the current command and immediately return a screenshot. (4)*claim_done*: indicate that the agent believes it has visited all required waypoints.

Since the environment evolves independently of the agent's decision loop, conditions may change during command execution, especially when a command takes a long time to complete. The agent may therefore need to observe these changes and intervene before the command finishes. So

we provide an optional `observe_after_sec` parameter for `exec` and `skip`, which specifies the requested delay in seconds before the next observation. It allows the agent to assess intermediate progress and decide whether to let the current command continue, stop it, or replace it. For example, the agent may issue `exec(content="mcapi press MOVE_FORWARD 5.0", observe_after_sec=2.0)`, requesting five seconds of forward movement and another observation after two seconds. If the path remains clear, it may select `skip` to wait for another observation while the movement continues. If an obstacle appears, it may select `stop_execute` to terminate the command or issue a new `exec` action to interrupt it and adjust its movement.

Preventing Hacking and Sandbox To preserve the integrity of the evaluation, we restrict access to information and game features that could allow the agent to bypass the intended task requirements. Specifically, we add safeguards to AgentBridge to prevent coordinate disclosure through waypoint editors, settings pages, and chat sharing. The visual world map remains available, while waypoint teleportation and Baritone-based automatic navigation are disabled during evaluation.

In addition, to prevent direct access to hidden task data through Bash, we execute agent-issued commands in a Bubblewrap sandbox within a containerized evaluation environment managed by Podman or Docker. The sandbox uses Linux namespaces and restricted filesystem mounts to expose the required tools and a writable agent workspace while excluding world saves, waypoint files, project files, and evaluator records. A separate network namespace disables external network access, while a Unix-domain socket relay permits access to approved AgentBridge APIs. Such sandbox isolation supports parallel evaluation. Each task runs in its own sandbox with a separate copy, client directory, and agent workspace, preventing interference from others. To improve evaluation efficiency, we also support GPU-accelerated rendering within the containers to offload graphics processing from CPU.

3 Experiments

3.1 Setup

Models and configuration. We evaluate leading proprietary models, including GPT-6 Astra (OpenAI, 2026b), GPT-5.6 Sol (OpenAI, 2026a), Claude Opus 5 (Anthropic, 2026), Gemini 3.8 Flash (Google DeepMind, 2026) and Grok 4.6 (SpaceXAI, 2026), alongside open-weight models DeepSeek-V4.1-Flash (DeepSeek, 2026) and GLM-5.3-Flash (Z.ai, 2026). All models use native tool calls through the same action interface. Each task is limited to 500 agent steps and six hours of wall-clock time, with at most three `claim_done` calls. To manage context growth, we automatically trigger summarization when any of the following three conditions is met: the current context window reaches 100 assistant turns, the latest model call reports more than 200K total tokens, or the active context has more than 100 images. The agent is prompted to summarize its goals, progress, relevant environment state, failed attempts, and next steps. The summary replaces the previous interaction history, while the system instructions and original task are retained. The agent then resumes the same episode from the current game state using the summary and newly received observations.

Tasks and metrics. We evaluate each model on 180 tasks and report **Success Rate** (SR), the fraction of tasks where the agent visits all required stops in order and submits an accepted `claim_done`. We also report **Checkpoint Coverage** (CC), the average fraction of ordered intermediate stops to measure partial progress. To measure movement and route efficiency, we report **Path Length** (PL) and **Success weighted by Path Length** (SPL). PL averages the full-episode 3D distance, computed from consecutive position samples at approximately 1 Hz, over all 180 tasks including failures. SPL measures efficiency by rewarding successful completion along shorter routes. It averages $S_i L_i / \max(L_i, P_i)$ over the same tasks, where S_i indicates success, P_i is the recorded path length, and L_i is the reference path length through all required stops in order. Details of reference path computation are provided in Appendix C.3. Average turns is the average number of agent turns following the agent loop in Section 2.3 and exclude automatic history-summary responses.

3.2 Main results

Table 1 shows that GPT-6 Astra leads by a wide margin, achieving 85.6% SR and outperforming Claude Opus 5 by 35.0 percentage points GPT-5.6 Sol follows at 43.9%, Gemini 3.8 Flash reaches

Table 1: Main results on 180 tasks, with success rates broken down by 125 outdoor and 55 indoor tasks. SR, CC, and SPL are percentages; PL is the mean sampled 3D distance over all tasks.

Model	SR (All)↑	SR (Outdoor)↑	SR (Indoor)↑	CC ↑	SPL ↑	Turns	PL
Proprietary models							
GPT-6 Astra (medium)	**85.56**	**87.20**	**81.82**	**89.04**	**53.70**	105.8	1033.3
Claude Opus 5 (max)	50.56	58.40	32.73	67.53	24.29	242.8	2020.2
Claude Opus 5 (high)	44.44	51.20	29.09	59.99	17.09	231.4	2921.5
Claude Opus 5 (medium)	39.44	47.20	21.82	57.48	12.95	233.8	3311.3
Claude Opus 5 (low)	35.00	39.20	25.45	48.00	7.77	218.8	4454.2
GPT-5.6 Sol (max)	43.89	51.20	27.27	61.36	22.87	306.0	1252.4
Gemini 3.8 Flash (high)	26.67	28.00	23.64	56.32	16.91	410.0	620.1
Grok 4.6 (xhigh)	17.22	20.80	9.09	40.36	7.84	402.1	1088.4
Open-weight models							
DeepSeek-V4.1-Flash (max)	23.89	30.40	9.09	44.37	11.62	362.5	1327.2
GLM-5.3-Flash (max)	15.56	19.20	7.27	36.09	5.72	357.8	1435.6

26.7%, and DeepSeek-V4.1-Flash is the strongest open-weight model at 23.9%. Most evaluated models complete fewer than half of the tasks, highlighting the benchmark's difficulty and the large differences in performance. All configurations have lower indoor SR, but Astra's decline is only 5.4 points, compared with 25.7 for Opus max and 23.9 for Sol. The two environments also differ in layout and route composition. Models with widely separated SR can nevertheless achieve similar intermediate-stop coverage: Sol and Gemini differ by only about 5 points in CC (61.4% versus 56.3%), compared with their 17.2-point gap in SR. The efficiency metrics provide a complementary view. Astra achieves 53.70% SPL, more than twice that of Opus max, with fewer than half as many turns. Gemini nearly matches Opus high's 17.09% SPL at 16.91% despite substantially lower SR. However, its shortest PL of 620.1 blocks comes with the most turns at 410.0, showing that short travel, including failed episodes, need not mean efficient execution. Increasing Opus effort from low to max raises SR from 35.0% to 50.6% and SPL from 7.77 to 24.29, while mean PL falls from 4454.2 to 2020.2 blocks and turns rise only from 218.8 to 242.8. Higher effort thus accompanies more completed tasks and less travel, with a modest increase in agent decisions.

4 Analysis

4.1 Ablations

We evaluate how spatial information and scene appearance affect DeepSeek-V4.1-Flash and GLM-5.3-Flash under six conditions. As illustrated in Figure 4, HUD enable BoccHUD (player position and orientation in the upper right) and Xaero's Minimap (local map in the upper left) and remove the default 32-block display-distance limit on first-person waypoint markers. Multi-view observations combine six camera directions into one image, while third-person view fixes the camera behind the player. Route guidance display a Baritone-planned path as a green trail with yellow directional arrows on the ground, as shown in Figure 4 (d), while leaving movement under agent control. Midnight replaces the default clear-noon lighting and supplies a torch that illuminates the surroundings when held. Rain changes the weather only for outdoor tasks; indoor outcomes remain unchanged. All conditions preserve the required destinations. From Table 2, we can find that visual aids improve SR and CC for both models, with

Table 2: Ablation across six conditions: SR, CC, SPL.

Setting	SR	CC	SPL
DeepSeek-V4.1-Flash			
HUD	30.0 (+6.1)	52.1 (+7.8)	12.8 (+1.2)
Multi-view (6)	26.1 (+2.2)	48.7 (+4.3)	11.3 (−0.3)
Third-person view	28.5 (+4.5)	50.0 (+5.3)	13.5 (+1.8)
Route guidance	21.1 (−2.8)	46.3 (+1.9)	9.7 (−1.9)
Midnight	22.2 (−1.7)	46.3 (+2.0)	10.1 (−1.5)
Rain	20.0 (−3.9)	42.1 (−2.3)	9.9 (−1.7)
GLM-5.3-Flash			
HUD	23.0 (+4.0)	43.8 (+3.2)	9.6 (+2.5)
Multi-view (6)	20.6 (+2.2)	43.3 (+3.1)	8.6 (+1.8)
Third-person view	23.0 (+4.0)	42.8 (+3.6)	9.9 (+2.8)
Route guidance	18.3 (+0.0)	41.8 (+1.6)	8.0 (+1.2)
Midnight	16.7 (−1.7)	39.0 (−1.2)	7.8 (+1.0)
Rain	16.1 (−2.2)	40.3 (+0.1)	5.7 (−1.1)

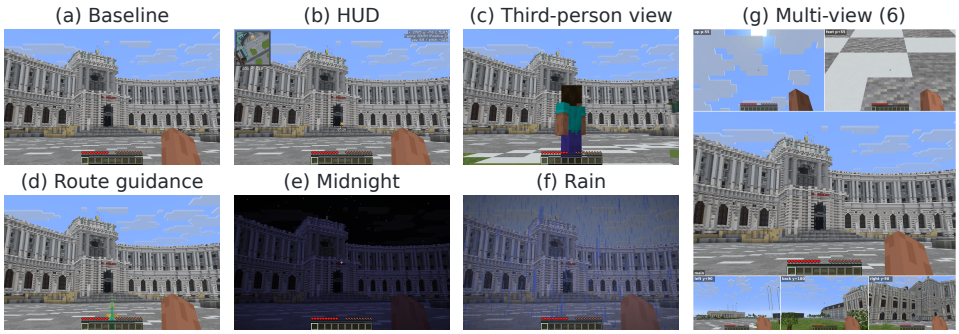

Figure 4: The default view and six ablation conditions are shown outside Hofburg Palace.

HUD raising SR by 6.1 points for DeepSeek and 4.0 for GLM. DeepSeek's multi-view SR gain of 2.2 points accompanies a 0.3 point SPL decrease. Route guidance slightly raises CC but lowers DeepSeek's SR by 2.8 points and leaves GLM's unchanged, indicating a gap between recognizing guidance and effective action. Finally, midnight reduces SR by 1.7 points for both models, with torches available to provide local light. Rain lowers SR by 3.9 points for DeepSeek and 2.2 for GLM.

4.2 Prolonged local stays.

To examine agents' ability to sustain progress, we count local stays of at least ten consecutive turns within one block of a fixed 3D anchor. From Figure 5, we find that Astra rarely remains in the same local area for many turns with only 0.17 episodes and 2.7% of turn, whereas such stays are much more frequent among all other models. Notably, Gemini spends over half of its turns in local stays at 51.7%, more than twice the share of any other model, and averages 7.09 episodes per task. During these stays, Gemini often engages in repeated map inspection, menu interactions, and screenshot processing without resuming movement.

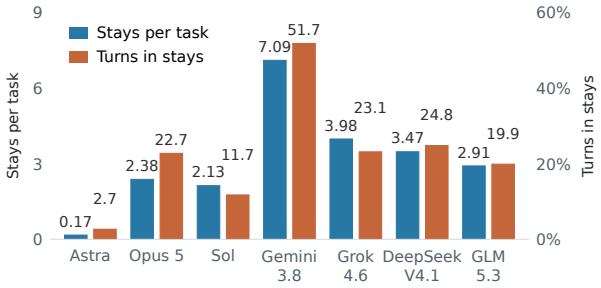

Figure 5: Local stays over at least ten turns within one block. Blue: episodes per task (left axis). Orange: task-mean share of aligned turns (right axis).

4.3 Case Studies

Case Study 1 Figure 6 shows two pairs of agents that pass through a shared location but differ in their subsequent progress. As shown in Figure 6(a), both DeepSeek and Astra enter through the same western doorway. DeepSeek recognizes the blocked passage but spent 142.7 minutes repeatedly revisiting the nearby rooms until the step limit. Astra instead inspects its surroundings, returns toward a southern doorway, and follows the hallway around the wall to Queen's Bedroom, completing the displayed detour in 2.4 minutes. In Figure 6(b), Astra and Grok both leave the restaurant floor at S and explore the upper patio at M. Grok remains upstairs for 135.7 minutes before reaching the step limit. Astra instead retraces its route to the stairs, descends to the restaurant floor, and reaches the target in 12.6 minutes. In both cases, Astra redirects its search and follows an alternative route to the target, while the unsuccessful models continues exploring within the same local region.

Case Study 2 As illustrated in Figure 7, Astra and Opus receive nearly matching views of the same passage entrance. After entering the passage, they see the same door ahead (outlined in blue). Astra turns right in place, revealing the stairs and identifying an upward route toward the target. While Opus instead approaches the door and returns outside without exploring the ascending branch, even though the stair opening is visible on its right (outlined in gold). This contrast illustrates how incomplete local exploration can leave an available connection undiscovered, even when its entrance is visible.

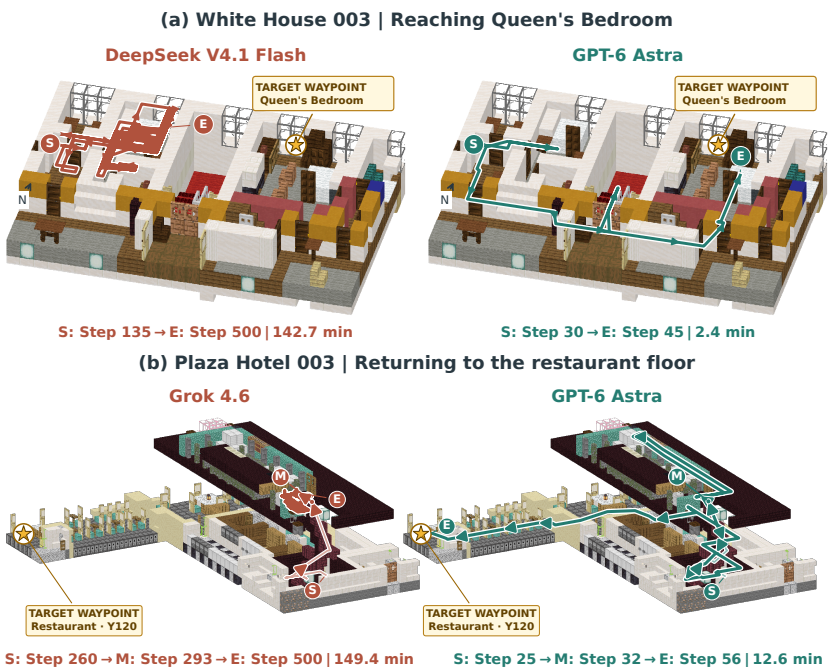

Figure 6: (a) White House: Astra takes a detour; DeepSeek revisits rooms. (b) Plaza Hotel: Astra returns downstairs; Grok remains upstairs. S/E: start/end; M: shared upper landing; stars: targets. Times cover the S–E segments.

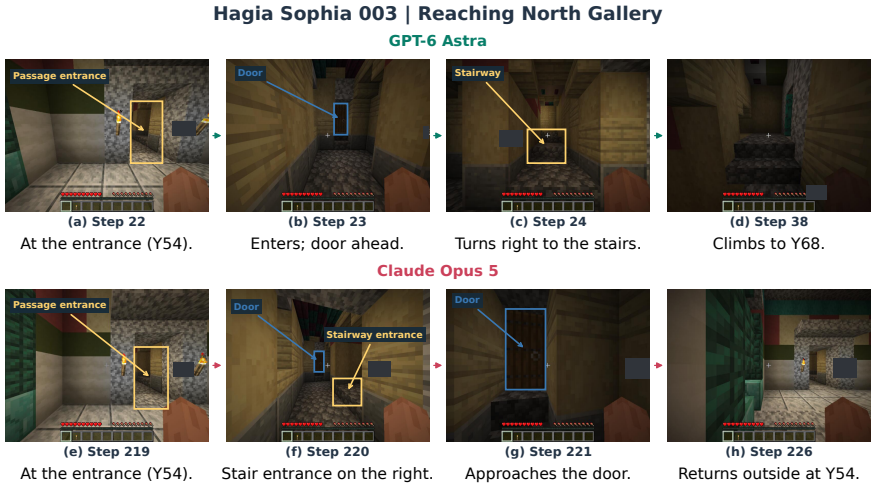

Figure 7: Discovering an upstairs route in Hagia Sophia 003. Astra explores the side branch and ascends, while Opus approaches the end door and leaves without exploring the branch.

5 Conclusion

We introduce Mine Odyssey, a benchmark for evaluating agentic spatial intelligence on 180 spatial tasks spanning 30 Minecraft reconstructions of diverse real-world locations. Our evaluation reveals substantial performance gaps: GPT-6 Astra achieves 85.6% success, while the second-best model, Claude Opus 5, reaches 50.6% and the strongest evaluated open-weight model, DeepSeek-V4.1-Flash, reaches 23.9%, leaving considerable room for improvement. Through detailed analyses, we identify key challenges in grounding spatial information, discovering traversable connections, and recovering from unsuccessful routes. We believe Mine Odyssey will support the development of more capable and reliable agents for sustained interaction in complex spatial environments.

AI USE STATEMENT

In this work, We used generative AI tools to assist with finding related work, research ideation, implementing analysis tools, generating synthetic data, and drafting and polishing the manuscript. All AI-assisted content, code, and data were carefully reviewed and revised by the authors. The authors take full responsibility for the accuracy and integrity of the final work.

REFERENCES

Andon Labs. Blueprint-Bench 2. https://andonlabs.com/evals/blueprint-bench-2, 2026.

Anthropic. Introducing Claude Opus 5, 2026. URL https://www.anthropic.com/news/claude-opus-5.

Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. Objectnav revisited: On evaluation of embodied agents navigating to objects. *arXiv preprint arXiv:2006.13171*, 2020. URL https://arxiv.org/abs/2006.13171.

Sharat Bhat, Harshita Khandelwal, Tushar Kataria, and Vivek Gupta. Mapverse: A benchmark for geospatial question answering on diverse real-world maps. In *2026 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 8168–8178, 2026. URL https://openaccess.thecvf.com/content/WACV2026/papers/Bhat_MapVerse_A_Benchmark_for_Geospatial_Question_Answering_on_Diverse_Real-World_WACV_2026_paper.pdf.

Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14455–14465, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Chen_SpatialVLM_Endowing_Vision-Language_Models_with_Spatial_Reasoning_Capabilities_CVPR_2024_paper.html.

Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 12538–12547, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Chen_TOUCHDOWN_Natural_Language_Navigation_and_Spatial_Reasoning_in_Visual_Street_CVPR_2019_paper.html.

Dwip Dalal, Utkarsh Mishra, Narendra Ahuja, and Nebojsa Jojic. Can mllms find their way in a city? exploring emergent navigation from web-scale knowledge. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8279–8303, 2026. URL https://aclanthology.org/2026.eacl-long.387/.

DeepSeek. Introducing DeepSeek-V4.1-Flash: Smarter, faster, more efficient, September 2026. URL https://www.deepseek.com/en/news/deepseek-v4-1-flash/.

Yucheng Deng, Pingrui Lai, Xinhai Li, Chenjia Bai, Xiaoheng Deng, Chengnuo Sun, Xuelong Li, and Hua Yang. Spacevln: A zero-shot vision-and-language navigation agent with online spatial cognitive memory and reasoning. *arXiv preprint arXiv:2606.08992*, 2026. URL https://arxiv.org/abs/2606.08992.

Mengfei Du, Binhao Wu, Zejun Li, Xuan-Jing Huang, and Zhongyu Wei. Embspatial-bench: Benchmarking spatial understanding for embodied tasks with large vision-language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 346–355, 2024. URL https://aclanthology.org/2024.acl-short.33/.

Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. In *Advances in Neural Information Processing Systems*, volume 35, pp. 18343–18362, 2022. URL https://papers.nips.cc/paper_files/paper/2022/hash/74a67268c5cc5910f64938cac4526a90-Abstract-Datasets_and_Benchmarks.html.

Hongcheng Gao, Hailong Qu, Jingyi Tang, Jiahao Wang, Zihao Huang, Hengkang Qiao, Shihong Huang, Junming Yang, Yi Li, Hongyixuan Yuan, et al. SpatialWorld: Benchmarking interactive spatial reasoning of multimodal agents in real-world tasks. *arXiv preprint arXiv:2606.09669*, 2026. URL https://arxiv.org/abs/2606.09669.

Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, et al. Gemini Robotics: Bringing AI into the physical world. *arXiv preprint arXiv:2503.20020*, 2025. URL https://arxiv.org/abs/2503.20020.

Google DeepMind. Gemini 3.8 Flash: Model card, 2026. URL https://deepmind.google/models/model-cards/gemini-3-8-flash/.

Junliang Guo, Yang Ye, Tianyu He, Haoyu Wu, Yushu Jiang, Tim Pearce, and Jiang Bian. MineWorld: A real-time and open-source interactive world model on minecraft. *arXiv preprint arXiv:2504.08388*, 2025. URL https://arxiv.org/abs/2504.08388.

William H. Guss, Brandon Houghton, Nicholay Topin, Phillip Wang, Cayden Codel, Manuela Veloso, and Ruslan Salakhutdinov. MineRL: A large-scale dataset of minecraft demonstrations. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pp. 2442–2448, 2019. URL https://dl.acm.org/doi/abs/10.5555/3367243.3367379.

Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4:100–107, 1968. URL https://ieeexplore.ieee.org/abstract/document/4082128?casa_token=rkP6LfyouWkAAAAA:HHTH9is5wy1WCguFF4A9rYMIATfiH7MSme6tgDnJdY7mkHgs8GoiduVzZQ6sBCH9vDL7MW5X.

Yohei Hayamizu, David DeFazio, Hrudayangam Mehta, Zainab Altaweel, Jacqueline Choe, Chao Lin, Jake Juettner, Furui Xiao, Jeremy Blackburn, and Shiqi Zhang. From woofs to words: Towards intelligent robotic guide dogs with verbal communication. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 38560–38569, 2026. URL https://ojs.aaai.org/index.php/AAAI/article/view/41198.

Yining Hong, Jiageng Liu, Han Yin, Manling Li, Leonidas Guibas, Li Fei-Fei, Jiajun Wu, and Yejin Choi. ESI-Bench: Towards embodied spatial intelligence that closes the perception-action loop. *arXiv preprint arXiv:2605.18746*, 2026. URL https://arxiv.org/abs/2605.18746.

Xinmiao Huang, Qisong He, Zhenglin Huang, Boxuan Wang, Zhuoyun Li, Guangliang Cheng, Yi Dong, and Xiaowei Huang. Spatial-dise: A unified benchmark for evaluating spatial reasoning in vision-language models. In *International Conference on Learning Representations*, volume 2026, pp. 135833–135865, 2026. URL https://iclr.cc/virtual/2026/poster/10008617.

Tianjie Ju, Yueqing Sun, Zheng Wu, Wei Zhang, Yaqi Huo, Xi Su, Qi Gu, Xunliang Cai, Gongshen Liu, and Zhuosheng Zhang. Mineexplorer: Evaluating open-world exploration of mllm agents in minecraft. *arXiv preprint arXiv:2605.30931*, 2026a. URL https://arxiv.org/abs/2605.30931.

Tianjie Ju, Zheng Wu, Yueqing Sun, Yuhan Cui, Bobo Li, Shengqiong Wu, Pengzhou Cheng, Haodong Zhao, Zongru Wu, Xinbei Ma, Doris Zhang, Kunling Li, Mong-Li Lee, Wynne Hsu, Hao Fei, Qi Gu, Gongshen Liu, and Zhuosheng Zhang. UrbanGround: From local perception to spatial agency in a real-scale city. *arXiv preprint arXiv:2608.27456*, 2026b. URL https://arxiv.org/abs/2608.27456.

Mukul Khanna, Ram Ramrakhya, Gunjan Chhablani, Sriram Yenamandra, Theophile Gervet, Matthew Chang, Zsolt Kira, Devendra Singh Chaplot, Dhruv Batra, and Roozbeh Mottaghi. Goat-bench: A benchmark for multi-modal lifelong navigation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16373–16383. IEEE, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Khanna_GOAT-Bench_A_Benchmark_for_Multi-Modal_Lifelong_Navigation_CVPR_2024_paper.html.

Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision*, pp. 104–120. Springer, 2020. URL https://link.springer.com/chapter/10.1007/978-3-030-58604-1_7.

Junlong Li, Wenshuo Zhao, Jian Zhao, Weihao Zeng, Haoze Wu, Xiaochen Wang, Rui Ge, Yuxuan Cao, Yuzhen Huang, Wei Liu, Junteng Liu, Zhaochen Su, Yiyang Guo, Fan Zhou, Lueyang Zhang, Juan Michelini, Xingyao Wang, Xiang Yue, Shuyan Zhou, Graham Neubig, and Junxian He. The tool decathlon: Benchmarking language agents for diverse, realistic, and long-horizon task execution. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=z53s5p0qhf.

Shunyu Liu, Yaoru Li, Kongcheng Zhang, Zhenyu Cui, Wenkai Fang, Yuxuan Zheng, Tongya Zheng, and Mingli Song. Odyssey: Empowering minecraft agents with open-world skills. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, pp. 187–195, 2025. URL https://doi.org/10.24963/ijcai.2025/22.

Tianhui Liu, Jie Feng, Zhiheng Zheng, Shengyuan Wang, Yiming Guo, Yanxin Xi, Hangyu Fan, Yong Li, and Pan Hui. Spatialact: Probing spatial reasoning-to-action capabilities of vlm agents in 3d scenes. *arXiv preprint arXiv:2605.31148*, 2026. URL https://arxiv.org/abs/2605.31148.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating llms as agents. In *International Conference on Learning Representations*, volume 2024, pp. 52989–53046, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/e9df36b21ff4ee211a8b71ee8b7e9f57-Abstract-Conference.html.

Piotr Mirowski, Andras Banki-Horvath, Keith Anderson, Denis Teplyashin, Karl Moritz Hermann, Mateusz Malinowski, Matthew Koichi Grimes, Karen Simonyan, Koray Kavukcuoglu, Andrew Zisserman, et al. The streetlearn environment and dataset. *arXiv preprint arXiv:1903.01292*, 2019. URL https://arxiv.org/abs/1903.01292.

OpenAI. GPT-5.6 Sol model, 2026a. URL https://openai.com/index/gpt-5-6/.

OpenAI. GPT-6 Astra model, 2026b. URL https://openai.com/index/gpt-6-astra/.

Mingyu Ouyang, Siyuan Hu, Kevin Qinghong Lin, Hwee Tou Ng, and Mike Zheng Shou. Gameworld: Towards standardized and verifiable evaluation of multimodal game agents. In *European Conference on Computer Vision*, pp. 41–59, 2026. URL https://arxiv.org/abs/2604.07429.

Davide Paglieri, Bartłomiej Cupiał, Samuel Coward, Ulyana Piterbarg, Maciej Wołczyk, Akbir Khan, Eduardo Pignatelli, Łukasz Kuciński, Lerrel Pinto, Rob Fergus, et al. Balrog: Benchmarking agentic llm and vlm reasoning on games. In *International Conference on Learning Representations*, volume 2025, pp. 96666–96702, 2025. URL https://openreview.net/forum?id=fp6t3F669F.

Xuetong Pei, Jian Liu, Vidura Munasinghe, Bo Miao, U-Xuan Tan, Wenrui Ding, and Na Zhao. SAP-Nav: Spatial semantic representation meets active perception for hierarchical open-vocabulary object navigation. *arXiv preprint arXiv:2608.12707*, 2026. URL https://arxiv.org/abs/2608.12707.

Lukas Petersson, Axel Backlund, Axel Wennström, Hanna Petersson, Callum Sharrock, and Arash Dabiri. Blueprint-bench: Comparing spatial intelligence of llms, agents and image models. *arXiv preprint arXiv:2509.25229*, 2025. URL https://arxiv.org/abs/2509.25229.

Yanyuan Qiao, Haodong Hong, Wenqi Lyu, Dong An, Siqi Zhang, Yutong Xie, Xinyu Wang, and Qi Wu. Navbench: Probing multimodal large language models for embodied navigation. In *Advances in Neural Information Processing Systems*, volume 38, pp. 95365–95385, 2026. URL https://papers.nips.cc/paper_files/paper/2025/hash/8932bbe603280f51d425e2781cd6ea6e-Abstract-Conference.html.

RoboDojo Team. An unexpected robot policy: Early evaluations of GPT-6 Astra on RoboDojo and beyond. Online technical report, 2026. URL https://robodojo-benchmark.com/report/gpt-6-astra-eval.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied AI research. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9338–9346, 2019. URL https://openaccess.thecvf.com/content_ICCV_2019/html/Savva_Habitat_A_Platform_for_Embodied_AI_Research_ICCV_2019_paper.html.

Ankur Sikarwar, Debangan Mishra, Sudarshan Nikhil, Ponnurangam Kumaraguru, and Aishwarya Agrawal. Communicating about space: Language-mediated spatial integration across partial views. *arXiv preprint arXiv:2603.27183*, 2026. URL https://arxiv.org/abs/2603.27183.

Xinshuai Song, Weixing Chen, Yang Liu, Weikai Chen, Guanbin Li, and Liang Lin. Towards long-horizon vision-language navigation: Platform, benchmark and method. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12078–12088. IEEE, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Song_Towards_Long-Horizon_Vision-Language_Navigation_Platform_Benchmark_and_Method_CVPR_2025_paper.html.

SpaceXAI. Introducing Grok 4.6, August 2026. URL https://x.ai/news/grok-4-6.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024a. URL https://openreview.net/forum?id=ehfRiF0R3a.

Junhui Wang, Dongjie Huo, Zehui Xu, Yongliang Shi, Yimin Yan, Yuanxin Wang, Chao Gao, Yan Qiao, and Guyue Zhou. Openbench: A new benchmark and baseline for semantic navigation in smart logistics. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 16202–16208, 2025. URL https://ieeexplore.ieee.org/abstract/document/11127476/?casa_token=LtRoOvc2_ZYAAAAA:8kdO9m__dDRTjs8AscJP9f75SYsOMxa12ig7ep_efmLpy7s-uRhyYBxBvBMEYYtEvc-bz5cU.

Sen Wang, Bangwei Liu, Zhenkun Gao, Lizhuang Ma, Xuhong Wang, Yuan Xie, and Xin Tan. Explore with long-term memory: A benchmark and multimodal llm-based reinforcement learning framework for embodied exploration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 37098–37108, 2026a. URL https://openaccess.thecvf.com/content/CVPR2026/html/Wang_Explore_with_Long-term_Memory_A_Benchmark_and_Multimodal_LLM-based_Reinforcement_CVPR_2026_paper.html.

Siqi Wang, Chao Liang, Yunfan Gao, Erxin Yu, Sen Li, Jing Li, and Haofen Wang. Cityseeker: How do vlms explore embodied urban navigation with implicit human needs? In *International Conference on Learning Representations*, volume 2026, pp. 32729–32766, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/3767842e2ebee900236f534855aa2c36-Abstract-Conference.html.

Zihao Wang, Shaofei Cai, Anji Liu, Yonggang Jin, Jinbing Hou, Bowei Zhang, Haowei Lin, Zhaofeng He, Zilong Zheng, Yaodong Yang, et al. Jarvis-1: Open-world multi-task agents with memory-augmented multimodal language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(3):1894–1907, 2024b. URL https://ieeexplore.ieee.org/document/10778628/.

Saim Wani, Shivansh Patel, Unnat Jain, Angel Chang, and Manolis Savva. Multion: Benchmarking semantic map memory using multi-object navigation. In *Advances in Neural Information Processing Systems*, volume 33, pp. 9700–9712, 2020. URL https://papers.nips.cc/paper/2020/hash/6e01383fd96a17ae51cc3e15447e7533-Abstract.html.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh J Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems*, volume 37, pp. 52040–52094, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/5d413e48f84dc61244b6be550f1cd8f5-Abstract-Datasets_and_Benchmarks_Track.html.

Jihan Yang, Runyu Ding, Ellis Brown, Xiaojuan Qi, and Saining Xie. V-IRL: Grounding virtual intelligence in real life. In *European Conference on Computer Vision*, pp. 36–55, 2024. URL https://www.ecva.net/papers/eccv_2024/papers_ECCV/html/6117_ECCV_2024_paper.php.

Jihan Yang, Shusheng Yang, Anjali W Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10632–10643, 2025a. URL https://openaccess.thecvf.com/content/CVPR2025/html/Yang_Thinking_in_Space_How_Multimodal_Large_Language_Models_See_Remember_CVPR_2025_paper.html.

Rui Yang, Hanyang Chen, Junyu Zhang, Mark Zhao, Cheng Qian, Kangrui Wang, Qineng Wang, Teja Venkat Koripella, Marziyeh Movahedi, Manling Li, Heng Ji, Huan Zhang, and Tong Zhang. EmbodiedBench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 70576–70631. PMLR, 2025b. URL https://proceedings.mlr.press/v267/yang25f.html.

Sihan Yang, Runsen Xu, Yiman Xie, Sizhe Yang, Mo Li, Jingli Lin, Chenming Zhu, Xiaochen Chen, Haodong Duan, Xiangyu Yue, et al. Mmsi-bench: A benchmark for multi-image spatial intelligence. In *International Conference on Learning Representations*, volume 2026, pp. 157051–157088, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/fe5d4bd3e2af823701b5d8f70a3c7602-Abstract-Conference.html.

Mengqi Yuan, Zilong Zhou, Xinzhuang Xiong, Weiming Wu, Jiayang Sun, Jiamin Song, Kaiqian Cui, Bowen Wang, Haoyuan Wu, Yitong Li, Dunjie Lu, Haikong Lu, Qi Zhen, Xinyuan Wang, Jiaqi Deng, Yuhao Yang, Cheng Chen, Boyuan Zheng, Alex Su, Xiao Yu, Hao Zou, Saaket Agashe, Xing Han Lu, Manpreet Kaur, Zhengyang Qi, Vincent Sunn Chen, Frederic Sala, Dayiheng Liu, Junyang Lin, Zhou Yu, Yu Su, Siva Reddy, Xin Eric Wang, Peng Qi, Tianbao Xie, and Tao Yu. OSWorld 2.0: Benchmarking computer use agents on long-horizon real-world tasks. *arXiv preprint arXiv:2606.29537*, 2026. URL https://arxiv.org/abs/2606.29537.

Z.ai. GLM-5.3-Flash, 2026. URL https://z.ai/blog/glm-5.3-flash.

Alex L. Zhang, Thomas L. Griffiths, Karthik R. Narasimhan, and Ofir Press. VideoGameBench: Can vision-language models complete popular video games? *arXiv preprint arXiv:2505.18134*, 2025. URL https://arxiv.org/abs/2505.18134.

Yuyou Zhang, Radu Corcodel, Chiori Hori, Anoop Cherian, and Ding Zhao. Spin-bench: Perspective and rotation as a lens on spatial reasoning in vlms. In *International Conference on Learning Representations*, volume 2026, pp. 70072–70141, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/724be4472168f31ba1c9ac630f15dec8-Abstract-Conference.html.

Xinyue Zheng, Haowei Lin, Kaichen He, Zihao Wang, Qiang Fu, Haobo Fu, Zilong Zheng, and Yitao Liang. MCU: An evaluation framework for open-ended game agents. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 78221–78259. PMLR, 2025. URL https://proceedings.mlr.press/v267/zheng25j.html.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, volume 2024, pp. 15585–15606, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/4410c0711e9154a7a2d26f9b3816d1ef-Abstract-Conference.html.

Yan Zhuang, Jiawei Ren, Xiaokang Ye, Jianzhi Shen, Tianai Yue, Muhammad Faayez, Xuhong He, Xiyan Zhang, Ziqiao Ma, Lianhui Qin, et al. Simworld-robotics: Synthesizing photorealistic and dynamic urban environments for multimodal robot navigation and collaboration. In *Advances in Neural Information Processing Systems*, volume 38, pp. 51854–51895, 2025. URL https://papers.neurips.cc/paper_files/paper/2025/hash/4a81c1aad3064fb8243b8938a49ecca0-Abstract-Conference.html.

A Related Work

A.1 Spatial Intelligence Evaluation

Spatial intelligence evaluation examines how models infer relations, integrate observations, and use spatial representations. Research has expanded from spatial relations in individual images (Du et al., 2024; Chen et al., 2024) to integration across views and videos (Yang et al., 2025a; 2026; Sikarwar et al., 2026), with finer diagnoses of perspective-taking and rotation (Huang et al., 2026; Zhang et al., 2026). Map understanding and reconstruction further test explicit spatial representations (Bhat et al., 2026; Petersson et al., 2025; Andon Labs, 2026). Recent interactive benchmarks extend this line to scene refinement (Liu et al., 2026), active evidence gathering (Hong et al., 2026), and spatial task completion across simulators (Gao et al., 2026). Within this line, we study agentic spatial intelligence in goal-directed navigation, examining how agents align allocentric maps with egocentric observations, select feasible actions under physical constraints, and revise decisions as their actions reveal new evidence.

A.2 Embodied Navigation Evaluation

Embodied navigation evaluates whether agents can reach specified goals in 3D environments through perception and action. Instruction-following and object-goal tasks (Batra et al., 2020) evaluate navigation to individual destinations. Multi-goal and long-horizon benchmarks (Wani et al., 2020; Khanna et al., 2024; Song et al., 2025; Wang et al., 2026a) further evaluate whether agents can complete successive navigation stages, adjusting their plans as new observations reveal obstacles, wrong turns, or failed movements. These navigation tasks are studied in both indoor and outdoor environments. Indoor benchmarks evaluate navigation through reconstructed rooms and buildings, including low-level movement in continuous environments (Savva et al., 2019; Krantz et al., 2020). For outdoor navigation, street-view environments provide access to real streets but restrict traversal to sampled viewpoints (Chen et al., 2019; Mirowski et al., 2019; Yang et al., 2024). City simulators enable continuous navigation through generated urban scenes (Zhuang et al., 2025) and real-city reconstructions such as UrbanGround's Hong Kong environment, where agents can use maps to navigate (Ju et al., 2026b). Mine Odyssey broadens environmental coverage to diverse outdoor and indoor reconstructions of real-world places, while opening navigation execution to agent-written programs. A persistent workspace allows agents to compose navigation commands into control routines, process state feedback, and retain task information across decisions.

A.3 Interactive Evaluation of Generalist Agents

Interactive benchmarks evaluate generalist agents through task completion in stateful environments (Liu et al., 2024). WebArena and OSWorld study realistic web and desktop workflows (Zhou et al., 2024; Xie et al., 2024), while OSWorld 2.0 and Toolathlon extend evaluation to sustained execution across longer workflows (Yuan et al., 2026; Li et al., 2026). Game benchmarks examine planning, exploration, and control over extended play (Paglieri et al., 2025; Zhang et al., 2025). Beyond task design, GameWorld studies how action interfaces affect agent performance using state-verifiable task outcomes (Ouyang et al., 2026).

Within Minecraft, agent methods investigate reusable skills and memory for extended execution (Wang et al., 2024a;b; Liu et al., 2025). Benchmarks assess complementary aspects of this execution: MCU evaluates composable tasks (Zheng et al., 2025), while MineExplorer tests whether agents can discover and satisfy unstated prerequisites through exploration, measuring progress with rule-based milestones (Ju et al., 2026a). In Mine Odyssey, agents operate through a programmable workspace, where they can process spatial evidence, compose control routines, and retain intermediate information. Together, interaction logs, movement trajectories, and evaluator-recorded visits support analysis of how agents use map information, organize control, and attempt recovery during the task.

B Map Collection and Environment Diversity

Table 3 lists task counts for all 30 maps in the selected 180-task benchmark.

Table 3: Formal task counts in the selected benchmark.

Map	Tasks	Map	Tasks
Buckingham Palace	5	Cape Town / Bo-Kaap	9
Chain Bridge	1	Copacabana Waterfront	3
Dublin Departments District	7	Entrup	6
Hagia Sophia	3	Hofburg	7
Innopolis	9	Memorial Hall Park	1
Miljacka Riverside	4	Harbor City Shipyard	11
Notre-Dame	4	NYC 9/11 Memorials	6
Ohrid	5	Plaza de Mayo	8
Plaza Hotel	6	Reichstag	5
RMS Queen Mary	10	RMS Titanic	6
Santa Lucía Hill	5	Shun Lee	4
SoFi Stadium	6	Sviyazhsk	6
Torrey Mall	5	Ueno Park	12
Versailles	3	White House	7
Würzburg	10	Zürich	6
		Total	180

The collection represents locations in countries and regions including South Africa, Brazil, Ireland, Japan, Germany, France, Türkiye, the United Kingdom, the United States, and Hong Kong. Examples range from city districts in Cape Town, Rio de Janeiro, Dublin, Tokyo, and New York City to Buckingham Palace, Versailles, Hagia Sophia, the Reichstag, SoFi Stadium, and historic passenger ships. Outdoor maps cover residential and commercial districts, parks, hillsides, and waterfronts, with variation in street connectivity, landmark density, and relief. Indoor maps include palaces, religious buildings, hotels, stadiums, and passenger ships, varying in room arrangement, corridor structure, entrances, and vertical circulation. Together, these settings support tasks from route planning across city districts to finding building entrances and moving between floors. Stairs, ladders, doors, and gates provide additional local traversal and interaction contexts, as illustrated in Figure 2. In Figure C, we present in-game world-map views of all 30 benchmark environments. Each map is labeled with its name and independently scaled to show its full layout while preserving its aspect ratio.

C Default Settings and Game Configuration

C.1 Default settings.

Table 4 summarizes the settings shared by the main-comparison runs. Controlled variants modify the designated settings as described in Section 4.1.

Table 4: Default settings for the main comparison. Controlled variants override the relevant entries.

Factor	Default
Time / weather	Noon (frozen) / clear (frozen)
Perspective	First person
Observation	Event-based, single view
Player scale	1.0
Optional HUD / minimap panel	Disabled
Full-screen world map	Available
Navigation hints	Disabled
Target-coordinate access	Locked in client and chat-sharing paths
Action-sandbox network	Public IP access disabled
Game mode / difficulty	Adventure / peaceful
Mobs / initial inventory	Disabled / cleared
Time / decision limits	21,600 seconds / 500 decisions

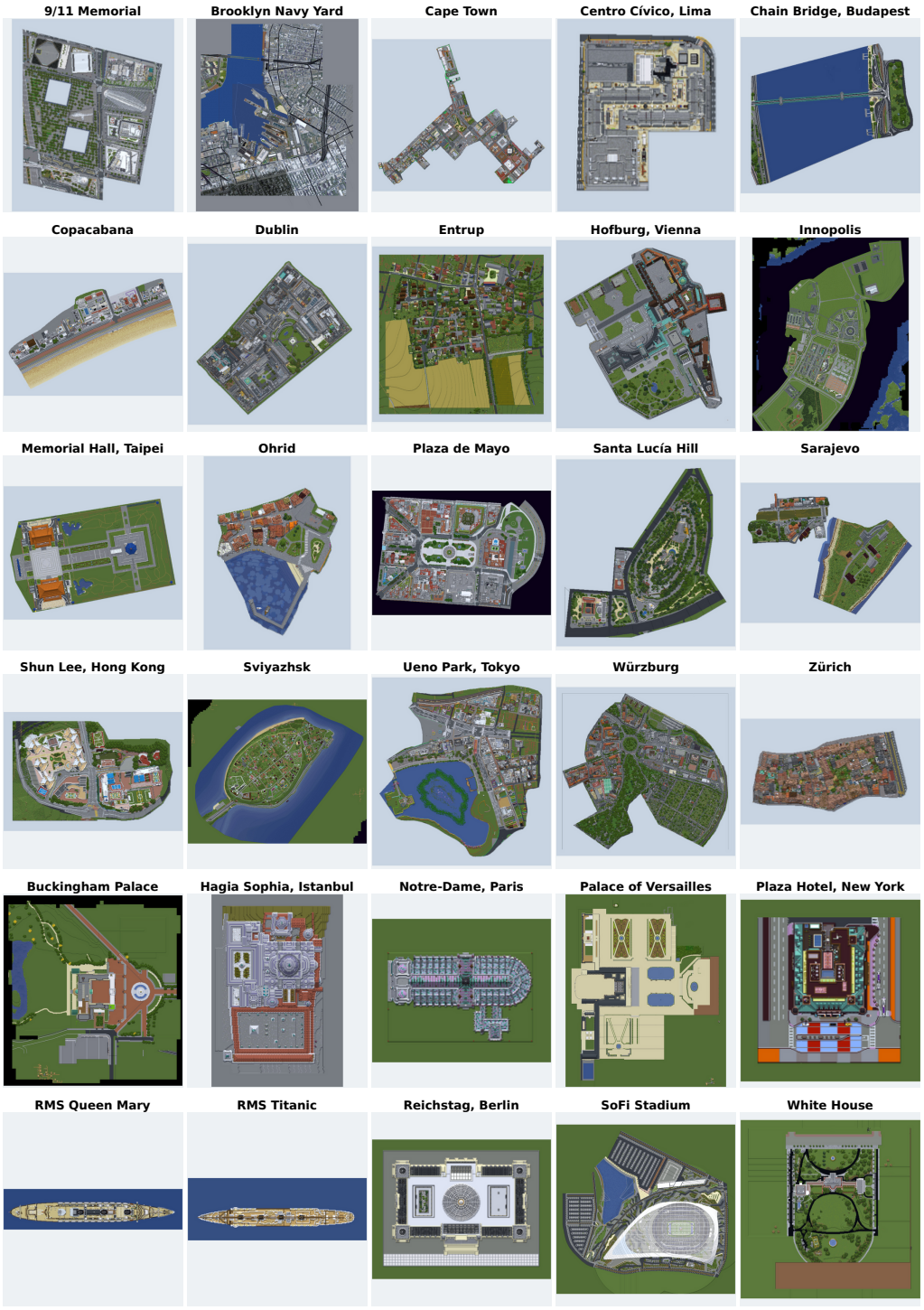

Figure 8: In-game world-map views of all 30 benchmark environments.

Control and condition-specific configurations. Our *AgentBridge* mod (1.0.0) provides player control. In the midnight condition, *LambDynamicLights* provides illumination from held torches. The route-guidance condition additionally loads our *Ground Navigation* mod (0.1.0) and *Baritone* (1.17.0) to display a planned route while leaving movement under agent control (Section 4.1).

C.2 Observation and interaction.

The default observation is a single first-person view. Observations and execution feedback are returned through the agent's interaction loop; the game and verifier continue to run during model inference. Agents can request an observation delay through `observe_after_sec`, allowing inspection before a command finishes. State queries and execution errors can also return textual feedback. Thus, event-based observation does not mean that the world pauses between decisions. Adventure mode permits ordinary locomotion and interactions such as opening doors and climbing ladders, while restricting block modification. Flight is disabled, and Peaceful difficulty removes hostile combat. Agents may consult the full-screen world map and nearby waypoint markers, but cannot access exact target coordinates or evaluator state. The minimap panel, optional HUD, navigation hints, automated navigation, and waypoint teleportation are disabled by default. Agent-issued commands run in an isolated workspace without access to world saves, waypoint files, or evaluator records; a restricted local relay provides access to the approved game-control APIs.

C.3 SPL and reference path construction

Success weighted by Path Length (SPL) measures navigation efficiency by combining task success with path length relative to a reference:

$$\mathrm{SPL} = \frac{1}{N}\sum_{i=1}^{N} S_i \frac{L_i}{\max(L_i, P_i)}, \tag{1}$$

where $N = 180$ is the number of tasks, S_i indicates task success, P_i is the agent's full-episode path length, and L_i is the reference path length through all required stops in order. Failed tasks contribute zero; successful tasks receive higher scores when their paths are shorter relative to the reference. Each task's contribution is capped at one.

We use A*, a shortest-path search algorithm (Hart et al., 1968), to compute a reference path for each task, shared across all models. We represent Minecraft movement as a graph, where nodes denote candidate player positions and edges denote movements allowed by our approximate movement model. Each edge is weighted by the 3D distance between its endpoints. Starting from the recorded initial position, A* finds the minimum-length route through all required arrival regions in order. The graph uses a 0.5-block horizontal lattice and approximate collision geometry. It supports walking, steps, ladders, and bounded falls with a crouched player body; it accounts for cumulative fall damage on maps where damage is enabled. Ordinary doors and registered button-operated gates are modeled as operable. Gap jumps, swimming, dynamic stance changes, and button-operated transport are outside the current graph. Both reference and agent path lengths are approximate. Reference lengths depend on the graph's discretization and simplified movement model, while agent path lengths are estimated from position samples collected at approximately 1 Hz, which may miss movement between samples.

D Performance and Inference Cost

D.1 Resource Accounting

Figures 9 and 10 report estimated monetary cost and output-token usage. Means include all 180 tasks, including failed runs. Turns count model decisions and exclude automatic history-summary responses.

D.2 Performance and Resource Use

The cost comparison distinguishes models with similar spending but different completion rates (Figure 9). At around \$1 per task, DeepSeek-V4.1-Flash achieves 23.9% SR for \$1.11, compared with 15.6% for GLM-5.3-Flash at \$1.20. DeepSeek therefore achieves higher completion at slightly lower estimated cost in this comparison. Moving to Gemini 3.8 Flash raises estimated spending to \$5.13 per task, about $4.6\times$ as much, while improving SR by 2.8 percentage points.

GPT-6 Astra achieves the highest SR and the lowest reported output-token usage, reaching 85.6% SR with 10.3K output tokens per task (Figure 10, left). Among the other models, GPT-5.6 Sol achieves

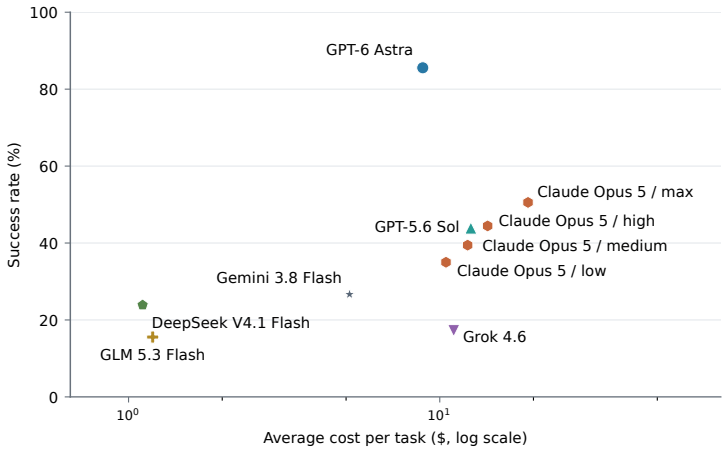

Figure 9: SR versus estimated cost per task, with cost on a log scale. Connected points show Claude Opus 5 effort settings. Costs average over all 180 tasks, including failures, using the accounting in Appendix D.

43.9% SR with 68.5K output tokens, while DeepSeek V4.1 Flash generates 618.0K tokens but reaches 23.9% SR. Output volume alone therefore does not explain the completion ranking. This comparison concerns reported output usage, rather than a direct measurement of inference compute.

D.3 Reasoning Effort

Increasing Claude Opus 5's reasoning effort from low to max raises SR from 35.0% to 50.6%, while mean output usage grows from 73.2K to 249.9K tokens per task, approximately $3.4\times$ (Figure 10, right). Estimated monetary cost increases from \$10.47 to \$19.21 per task, yielding a 15.6-point SR gain for an additional \$8.74. Mean turns increase by only 11.0%, so output usage grows much faster than the number of interaction turns.

The additional output required also differs across effort transitions: low to medium adds 28.3K output tokens per task for a 4.4-point SR gain, whereas high to max adds 105.1K for a 6.1-point gain (Figure 10, right). Higher effort improves completion in this model, but the latter transition requires more additional output per percentage point gained. These comparisons average over all tasks; they do not hold successful-task membership fixed.

E Failure Stages

Figure 11 igure 11 distinguishes failure before the first required stop, failure after reaching it, and successful completion. The first stop is the first intermediate stop, or the destination for tasks without inter- mediate stops. Reaching a stop uses the evaluator's ordered-arrival record; full success additionally requires an accepted completion claim. Failure after the first stop can therefore include runs that visit all stops but do not complete successfully. DeepSeek, Grok, and GLM fail before the first stop on 48.3–55.6% of all tasks. Gemini reaches it more often than Sol (70.0% versus 67.8%), but fails afterward on 43.3% of all tasks, compared with 23.9% for Sol. These outcomes show why initial progress need not translate into full completion.

F Interaction Statistics

Counting and aggregation. We classify 364,358 exec requests from the seven main configurations, each evaluated on the same 180 tasks. Each recognized category is counted at most once per request, even when a request contains repeated calls. For each category, we divide its count by the task's exec-request count and then average these percentages equally over the 180 tasks. We do not pool

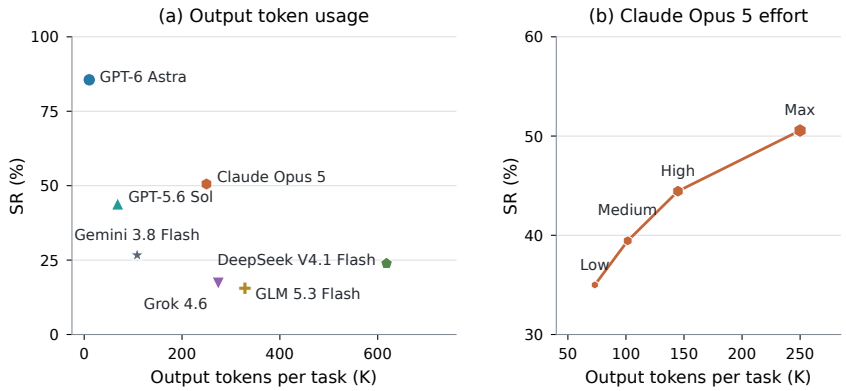

Figure 10: SR versus output-token usage. Left: the seven main model configurations. Right: Claude Opus 5 at low, medium, high, and max reasoning effort; connected points show increasing effort. Values use all 180 tasks, including failures; K denotes thousands. Both panels use linear axes, with different SR ranges.

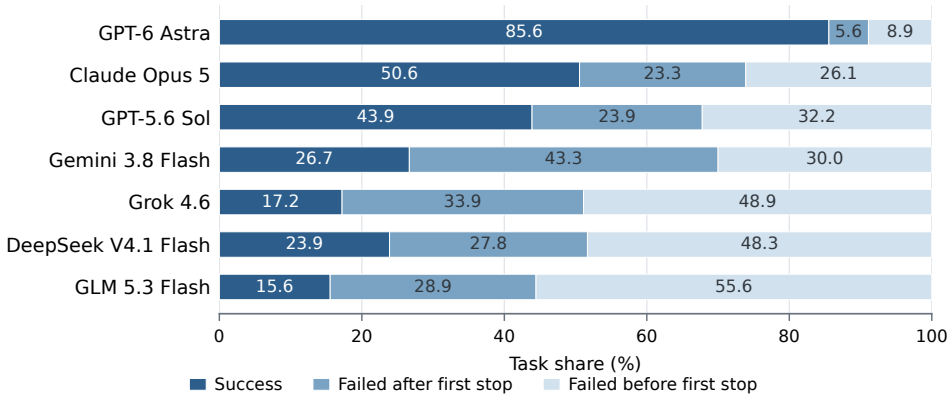

Figure 11: Outcomes over 180 tasks per model. First stop means the first intermediate stop, or the destination if none. Deaths count as failures even after all stops. All shares use the full 180-task denominator. Model configurations match the main comparison.

requests across tasks. A request may contribute to multiple categories, and percentages need not sum to 100%.

Observed interaction patterns As revealed in Figure 12, Astra, Opus, DeepSeek, and GLM combine commands in over 90% of requests, compared with 25.6% for Gemini. Gemini has the highest share of explicit image processing. It also includes state queries less frequently. Section 4.2 also reveals that Gemini often remains in one place for extended periods, repeatedly checking maps, operating menus, and processing screenshots without resuming movement. Opus and Gemini use loops more often than Astra, whose requests predominantly use command sequences. Astra and Grok both adjust the viewing direction in about 59% of requests, while Astra toggles the map more frequently. These patterns distinguish how models request information and organize commands, without directly ranking navigation ability.

G CASE STUDIES

Interaction styles across models

	Information access				Command structure	
	Map key	Look	State query	Image proc.	Shell sequence	Loops
GPT-6 Astra	11.7	59.2	51.3	0.0	90.1	0.2
Claude Opus 5	8.9	39.8	44.5	0.2	92.9	8.3
GPT-5.6 Sol	10.0	32.6	53.8	<0.1	66.5	0.9
Gemini 3.8 Flash	6.9	26.6	16.6	7.9	25.6	8.3
Grok 4.6	4.1	59.4	40.5	<0.1	53.3	0.9
DeepSeek V4.1 Flash	9.5	38.6	55.3	0.3	91.3	2.4
GLM 5.3 Flash	7.7	42.6	57.4	0.3	93.5	8.0

Figure 12: Information access and command structure across seven models. Values are percentages of execution requests, computed per task and averaged over 180 tasks per model; categories can overlap. Map key counts map toggles, Look counts camera-orientation commands, State query includes queries combined with controls, and Image proc. counts explicit image-processing operations Shell sequence counts command sequences or pipelines within a request, and Loops counts requests containing shell or Python loop syntax.



Figure 13: **From map pixels to a movement target.** Step 176 captures the map; Step 177 computes the estimated U20 coordinates; Steps 179 and 181 use the estimate for orientation and movement. S marks the player position and G the estimated goal; dashed lines indicate pixel offsets, not an executed route. The displayed map provides context because the measured PNG is unavailable. The adopted scale is uncalibrated, and execution does not establish arrival.

Image processing can construct an actionable spatial target. Gemini turns map observations into an actionable spatial target through a short Python program. It converts player-to-marker pixel offsets into estimated world coordinates, then uses the result to orient and initiate movement toward U20 (Figure 13). This case illustrates how programming connects visual map grounding to action.

Stair programming. As shown in Figure 14, after testing individual stair movements, Astra combines four 1.3-second forward movements and turns into a single shell sequence. At Step 35, this

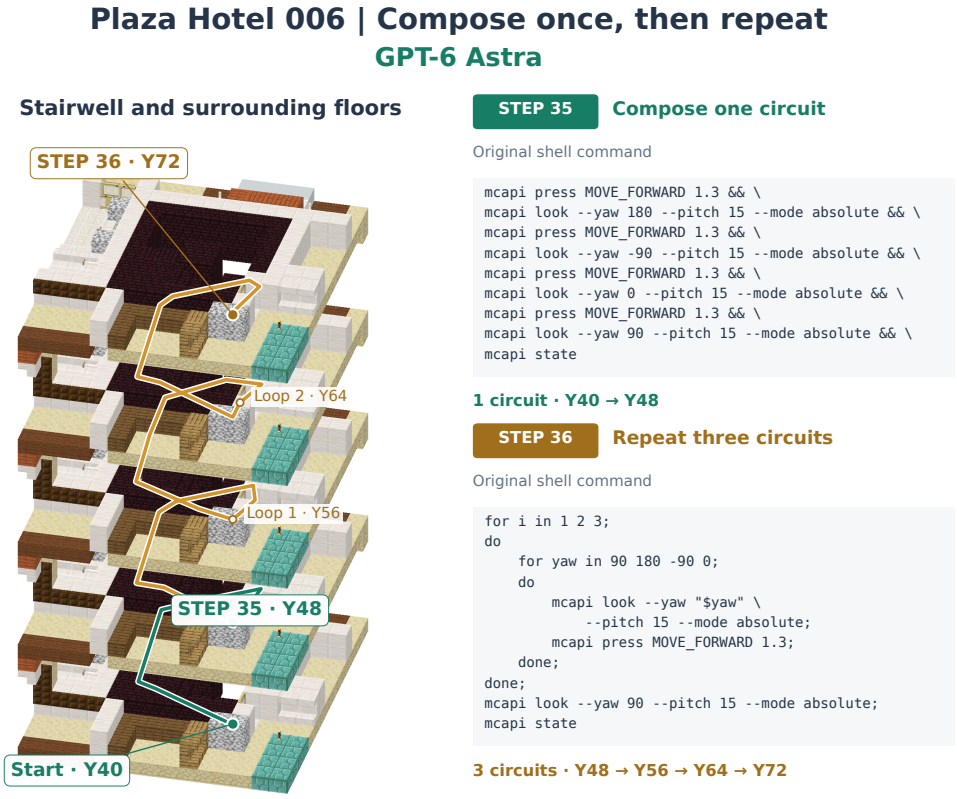

Figure 14: From composition to reuse in Plaza Hotel, Steps 35–36. Step 35 executes one circuit from Y40 to Y48; Step 36 executes all three circuits from Y48 through Y56 and Y64 to Y72. The cutaway uses the retained world's native textures, with selected walls and room volumes removed. Command-window position samples and matched state endpoints are projected at their original heights over the scene for visibility. Matching colors link each step in the enlarged scene to its height transition and complete shell command. The original observation screenshots are omitted. Only whitespace is reformatted in the code. Later steps are outside the displayed window; the cutaway is a rendering, not an agent screenshot.

sequence completes one circuit around the stairwell, raising the agent from Y40 to Y48. At Step 36, Astra uses a loop to repeat the four-heading movement pattern three times within one tool request, ascending through Y56 and Y64 to Y72. The loop reuses the established stair-climbing pattern across successive levels without requiring a new agent turn for each movement.

Together, these cases show how agents use programming to turn visual map information into explicit coordinates and individual movement commands into reusable action sequences.