Current Agents Fail to Leverage World Model as Tool for Foresight

Cheng Qian[1], **Emre Can Acikgoz**[1], **Bingxuan Li**[1], **Xiusi Chen**[1], **Yuji Zhang**[1], **Bingxiang He**[2], **Qinyu Luo**[3], **Dilek Hakkani-Tür**[1], **Gokhan Tur**[1], **Yunzhu Li**[4], **Heng Ji**[1]
[1]UIUC, [2]THU, [3]JHU, [4]Columbia
{chengq9, hengji}@illinois.edu

Abstract

Agents built on vision-language models increasingly face tasks that demand anticipating future states rather than relying on short-horizon reasoning. Generative world models offer a promising remedy: agents could use them as external simulators to foresee outcomes before acting. This paper empirically examines whether current agents can leverage such world models as tools to enhance their cognition. Across diverse agentic and visual question answering tasks, we observe that some agents rarely invoke simulation (fewer than 1%), frequently misuse predicted rollouts (approximately 15%), and often exhibit inconsistent or even degraded performance (up to 5%) when simulation is available or enforced. Attribution analysis further indicates that the primary bottleneck lies in the agents' capacity to decide when to simulate, how to interpret predicted outcomes, and how to integrate foresight into downstream reasoning. These findings underscore the need for mechanisms that foster calibrated, strategic interaction with world models, paving the way toward more reliable anticipatory cognition in future agent systems.

1 Introduction

Modern AI agents are increasingly expected to operate in settings where tasks unfold over long horizons, involve intricate chains of interdependent decisions, and may yield irreversible consequences. Such scenarios, ranging from multi-stage robotics manipulation (Huang et al., 2023, 2024; Feng et al., 2025; Fan et al., 2025) to complex software automation (Deng et al., 2025b; Wang et al., 2025a; Agashe et al., 2025) and real-world planning (Garg et al., 2025; Erdogan et al., 2025; Geng and Chang, 2025), demand that agents not only reason locally but also anticipate how the environment might evolve. Human decision-making in similar contexts is often anchored in our ability to project possible future states, assess risks, and adjust actions accordingly (Gilbert and Wilson, 2007). Classical theories of cognition, such as mental simulation and prospective reasoning in psychology and cognitive science, emphasize that people routinely rely on internal models of how the world behaves to forecast outcomes before acting (Premack and Woodruff, 1978; Johnson-Laird, 1983; Schacter et al., 2007). These enable us to navigate uncertainty, manage delayed rewards, and avoid catastrophic errors.

Large language model (LLM) agents, despite impressive proficiency in planning, decomposition, and interacting with diverse tools or environments, still exhibit a major limitation: they often lack a robust mechanism for foresight (Chae et al., 2024; Jin et al., 2024). Even when an LLM can articulate a goal-achieving plan, this does not necessarily imply that it can anticipate how future states of the environment might unfold.[1] For instance, a household robot may plan to open a window to vent smoke, but fail to anticipate that incoming wind will blow loose papers into a spill, spreading contamination and creating new hazards. Such an error illustrates that effective action depends not only on a plausible plan, but on foresight about how the environment will evolve under interventions, especially when those dynamics introduce delayed, non-local constraints.

Recent research has begun to explore whether foresight can be explicitly supported by equipping agents with mechanisms for simulating the future. One line of work takes a training-free approach, augmenting agents with dedicated foresight modules through prompts (Qian et al., 2024) or encouraging deliberate world-modeling via structured instructions (Wang et al., 2025b). Another line of research focuses on intrinsic solutions, training or embedding learned world models directly into the

[1]*Foresight* in our scope refers to an agent's ability to anticipate how the environment will evolve, rather than to generate plans to achieve goals from its own perspective.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 13686–13723
July 2-7, 2026

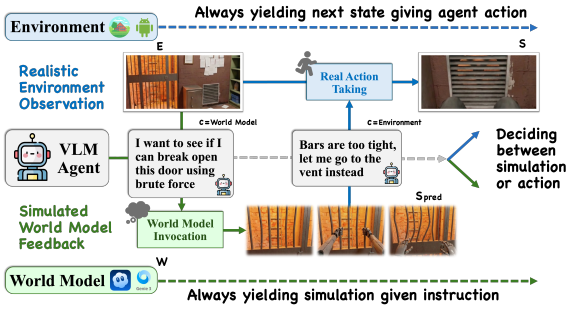

Figure 1: A world model as tool framework illustrated through a high-stakes escape decision. The agent evaluates a reinforced door break versus a ventilation-shaft crawl in simulation before committing to the feasible real world action.

agent itself, as seen in efforts such as code world model or multimodal generative simulators (Carbonneaux et al., 2025). However, simple prompting approaches often behave rigidly, failing to capture the visually diverse world states, while intrinsic methods typically require extensive retraining, substantial computational resources, and are difficult to integrate reliably across model families.

At the same time, world models—especially large-scale video and environment simulators[2] such as Sora, WAN, and other generative dynamics models (Brooks et al., 2024a; Wan et al., 2025)—have advanced to produce reasonably coherent and temporally consistent predictions in open-ended settings. This raises a natural question: instead of building foresight inside the agent or enforcing highly structured prompting, can we **treat world models as off-the-shelf tools that an agent may optionally call upon when needed**? Doing so reframes foresight not as a fixed architectural component but as a resource that an agent might strategically leverage. This perspective opens an important direction: *To what extent can current LLM agents decide when to use world models for environmental foresight, and does this foresight substantially improve their cognition?*

Our preliminary investigation reveals two concrete limitations. First, many LLM agents are reluctant to invoke world-model tools even when it improves environmental foresight, and this reluctance increases with model scale or capability. The behavior is also family-consistent: LLaMA variants show comparatively higher willingness, while GPT and Qwen variants frequently decline tool use across sizes. Second, even when world models are invoked, cognitive benefits are inconsistent because agents often misuse the tool, for example by producing only one deterministic future, overriding simulations with over-confident internal reasoning, or failing to evaluate counterfactual branches.

Taken together, these findings underscore that progress in foresight-augmented agents is more about calibrating its behavior: models should more strategically consult external simulations when benefits are clear, less prone to over-trust inaccurate internal reasoning, and better able to incorporate simulated evidence without either dismissing it or rigidly overfitting to it. Our analysis also reveals that these calibration issues follow systematic patterns. We identify three recurrent governance breakdowns, including misguided input formulation, ambiguous interpretation of simulated outcomes, and unstable action integration, which collectively explain the majority of observed regressions. To summarize:

- We introduce the conceptual framing of *world models as tools*, offering an analysis-focused perspective that examines how existing LLM agents interact with external simulators.
- We propose an evaluation framework that enables systematic study of agents' willingness, correctness, and consistency when employing world models, providing a basis for future training.
- Through extensive analysis, we uncover behavioral patterns, family-specific tendencies, error taxonomy, and structural weaknesses that inform future research on agentic foresight.

2 Related Work

Cognitive science argues that humans plan by mentally simulating actions and outcomes using internal world models, like *theory of mind* (anticipating others' reactions by simulating their inner states), and more generally, predicting how an environment will evolve. Analogously, recent work equips LLM agents with world-modeling machinery for foresight, either by treating an LLM as a dynamics model and searching over imagined trajectories, or by embedding a learned simulator in the decision loop to mitigate short-sighted action selection (Deng et al., 2025a; Hao et al., 2023). This idea extends beyond text: in code, agents can anticipate program outcomes internally before committing to an answer (Carbonneaux et al., 2025); in vision, agents benefit from explicitly sep-

[2]The terms *simulator* and *world model* are used interchangeably in this paper. They both refer to the prediction about future environment state given an agent action.

arating state estimation from action-conditioned transition prediction, often by importing dynamics priors from video world models (Wang et al., 2025b; Zhang et al., 2025b) or future frame prediction (Gladstone et al., 2025). In parallel, scaling generative video models yields increasingly realistic long-horizon simulators (Brooks et al., 2024b), and lightweight action conditioning turns them into controllable rollouts for planning and reinforcement learning (He et al., 2025); recent work also studies how to evaluate and deploy such world models in closed-loop decision-making settings (Zhang et al., 2025a). Complementary evidence shows that language itself can function as a simulator when agents are prompted to hypothesize and reflect (Qian et al., 2024; Wang et al., 2025c), and that closing the loop with external physics engines can enforce feasibility constraints in embodied settings (Meng et al., 2025). In contrast to work that primarily proposes new simulators or integration architectures, our work focuses on the agent-side question of *when* to invoke world models and *how* simulated futures are interpreted and used during downstream reasoning.

3 Preliminaries

In this section, we introduce the evaluation framework used to study how an agent leverages a world model as a decision-making tool. Let A denote the tested agent, E the real environment in which it operates, and W a world model that simulates the environment dynamics. The agent's objective is to reach a task-specific goal state G in E. The concrete form of G depends on the task family (for example, reaching a target location, solving a puzzle, or maximizing accumulated reward). At any time, the real environment is in state S, and the agent maintains a trajectory (or context) T that collects its past actions and observations.

3.1 Interaction Process

The overall interaction procedure is summarized in Algorithm 1. At each decision step, the agent maps its current trajectory T to a pair (c, a), where c specifies *where* the action is to be executed and a is the action itself:

$$(c, a) = A(T). \tag{1}$$

The choice variable c can direct the agent either to the real environment E or to the world model W, and this decision is part of the agent's policy and reasoning process.

Algorithm 1 Agent Interaction Process

Require: Agent A, Environment E, World Model W, Initial State S, Maximum Steps L, Goal Condition G

$T \leftarrow \emptyset$ ▷ Initialize trajectory
for $t = 1, 2, \ldots, L$ **do**
  $(c, a) \leftarrow A(T)$ ▷ Agent chooses target
  if c is **World Model then**
    $S_{\text{pred}} \leftarrow W(S, a)$ ▷ Simulation
    $T \leftarrow T \cup \{(a, S_{\text{pred}})\}$
  else if c is **Environment then**
    $S \leftarrow E(S, a)$ ▷ Execution
    $T \leftarrow T \cup \{(a, S)\}$
  end if
  if S satisfies goal condition G **then**
    break
  end if
end for
return T

When c selects the world model, the agent uses W as a simulator. The action a is applied to the current environment state S inside the world model, which produces a predicted next state

$$S_{\text{pred}} = W(S, a). \tag{2}$$

The agent does not alter the real environment in this branch. Instead, it treats S_{pred} as an informative but hypothetical observation, and the pair (a, S_{pred}) is appended to the trajectory T. In this way, the agent can explore possible futures, anticipate consequences, and refine its plan without incurring real-world cost or risk.

When c selects the real environment, the agent commits to executing the action a in E. The true environment state is updated according to the transition dynamics,

$$S \leftarrow E(S, a), \tag{3}$$

and the resulting state S is returned as an observation to the agent. The pair (a, S) is then appended to the trajectory T. In this branch, the agent makes irreversible progress in the real task and potentially moves closer to, or further away from, the goal G.

The rollout continues by repeatedly querying the agent with the updated trajectory T and state S. The process terminates when the environment reaches a goal state G or when a predefined maximum number of decision steps L is exceeded. In this way, the same protocol can accommodate diverse tasks while keeping the role of the world model explicit and comparable across settings.

3.2 Interaction Modes

A key aspect of the framework is how the agent is informed about the world model and allowed to

interact with it. We control this through the system prompt, which leads to three distinct modes:

- **Normal Mode.** The agent is informed that a world model W exists and may choose to query it or directly act in E. This mode captures realistic decision-making in which simulation is an optional tool whose value must be inferred and used appropriately.
- **WM Invisible Mode.** The agent is agnostic of W, and therefore never queries it. All actions are executed directly in E, serving as the standard baseline consistent with prior evaluations.
- **WM Force Mode.** The agent is instructed to simulate actions in W before executing them in E, making world model use mandatory. This mode reveals whether compulsory planning helps or hinders performance when simulation is no longer a strategic choice.

In our experiments, we examine all three, with particular focus on comparing the first two modes, which isolates the effect of optional access to a world model while keeping all other factors fixed.

4 Experiments

4.1 Task Choices

To understand when and how world models benefit VLM agents, we evaluate two complementary types of tasks that require VLM-based reasoning.

The first type is **agentic decision-making tasks**, which require visual grounding, perception-to-action mapping, and multi-step reasoning in interactive environments. Following VAGEN (Wang et al., 2025b), we select four diverse tasks: *FrozenLake*, *Navigation*, *PrimitiveSkill*, and *Sokoban*. These tasks jointly cover: (i) long-horizon interaction, (ii) reasoning in both 2D symbolic layouts and photorealistic 3D scenes, and (iii) action planning across different embodiments, including household agents, robot manipulation, and game environments. These properties make them ideal for assessing whether simulated rollouts can meaningfully support the agent's cognition.

Although FrozenLake and Sokoban are visually simpler than photorealistic environments, they remain meaningful foresight benchmarks because both contain irreversible failure modes. In FrozenLake, a single mistaken move can end the episode by dropping the agent into a hole; in Sokoban, pushing a box into a wall-adjacent dead-end can make the puzzle unrecoverable. Since one central value of simulation is to anticipate such costly irreversible outcomes before execution, these controlled environments let us isolate when foresight is genuinely helpful and when it is misused.

The second type is pure **VLM reasoning tasks**, where no embodied control is required. Our aim is to test the simulation paradigm's impact when the agent interacts only through perceptual or spatial reasoning without action execution. We include four representative datasets: *3DSRBench* (Ma et al., 2025), *MMSI Bench* (Yang et al., 2025), *SAT* (Ray et al., 2024), and *Spatial-MM Object* (Shiri et al., 2024). These benchmarks are chosen because: (i) they demand rich spatial reasoning and visual grounding, and (ii) their questions contain latent dynamics or hypothetical transformations that can theoretically benefit from simulated outcomes. Without this property, evaluating world model augmentation would be inherently meaningless, since simulation would not provide additional utility beyond static perception.

Together, these two categories allow us to examine world model usage both in interactive control and in non-embodied visual reasoning, revealing whether simulation is effective. Table 1 summarizes the task families and how world model queries fit into the agent–environment loop.

4.2 Experiment Settings

Test Models. We evaluate nine vision-language models across both open- and closed-source families, including GPT (OpenAI, 2025), Llama (AI, 2025), and Qwen (Bai et al., 2025). For each family, we select multiple model sizes to investigate how world model augmentation affects models with different capacities.

World Model. For agentic tasks, the world model is an oracle simulator implemented by cloning the current environment state and rolling out hypothetical actions under the exact same dynamics as the real environment. This design isolates agent-side use of foresight from simulator-quality confounds. For VQA reasoning tasks, no analogous oracle rollout exists because the input is a static image rather than an executable environment state. We therefore use Wan2.1 (Wan et al., 2025) to generate predicted visual outcomes from the agent's textual simulation instructions, and later test a stronger Wan2.2 simulator as a robustness check. Full configurations are provided in Section A and Section E.

Tasks	Goal Condition G	Environment State S	Action a in Environment	Action a in World Model
FrozenLake	Reach the target grid without falling	2D grid observation of agent position	Discrete movement (up, down, left, right)	
Navigation	Arrive at a target location	First-person RGB scene	Embodied moves (forward, rotate, look, etc.)	
PrimitiveSkill	Manipulate objects to a target pose	Arm pose + object pose	Manipulation command (pick, place, move with coordinates)	
Sokoban	Push boxes to goal cells	2D game state with box locations	Discrete movement (up, down, left, right)	
All VQA Tasks	Output the correct answer	Provided problem images	Direct textual answer	Simulation-oriented query on an image

Table 1: Task families and how world model queries integrate with the agent–environment loop.

Model	Frozen Lake	Navigate	Primitive Skill	Sokoban	Avg.
Without World Model Access					
GPT-4o-mini	0.36	0.26	0.32	0.00	0.27
GPT-4o	0.58	0.35	0.51	0.02	0.40
GPT-5-mini	0.86	0.66	0.11	0.03	0.41
GPT-5	0.77	0.74	0.19	0.06	0.47
Llama-4-Maverick	0.70	0.31	0.40	0.00	0.35
Llama-4-Scout	0.59	0.55	0.32	0.02	0.42
Qwen2.5-VL-7B	0.36	0.26	0.13	0.02	0.20
Qwen2.5-VL-32B	0.59	0.36	0.49	0.00	0.40
Qwen2.5-VL-72B	0.61	0.38	0.41	0.00	0.37
With World Model Access					
GPT-4o-mini	$0.39_{\uparrow 0.03}$	$0.24_{\downarrow 0.02}$	$0.22_{\downarrow 0.10}$	0.00	$0.22_{\downarrow 0.05}$
GPT-4o	0.58	$0.31_{\downarrow 0.04}$	$0.46_{\downarrow 0.05}$	0.02	$0.36_{\downarrow 0.04}$
GPT-5-mini	$0.89_{\uparrow 0.03}$	$0.63_{\downarrow 0.03}$	$0.19_{\uparrow 0.08}$	$0.00_{\downarrow 0.03}$	$0.43_{\uparrow 0.02}$
GPT-5	$0.89_{\uparrow 0.12}$	$0.71_{\downarrow 0.03}$	$0.20_{\uparrow 0.01}$	$0.14_{\uparrow 0.08}$	$0.48_{\uparrow 0.01}$
Llama-4-Maverick	$0.66_{\downarrow 0.04}$	$0.20_{\downarrow 0.11}$	$0.32_{\downarrow 0.08}$	$0.03_{\uparrow 0.03}$	$0.27_{\downarrow 0.08}$
Llama-4-Scout	$0.55_{\downarrow 0.04}$	$0.54_{\downarrow 0.01}$	$0.24_{\downarrow 0.08}$	$0.03_{\uparrow 0.01}$	$0.38_{\downarrow 0.04}$
Qwen2.5-VL-7B	$0.45_{\uparrow 0.09}$	0.26	$0.12_{\downarrow 0.01}$	$0.00_{\downarrow 0.02}$	0.20
Qwen2.5-VL-32B	$0.58_{\downarrow 0.01}$	$0.32_{\downarrow 0.04}$	$0.39_{\downarrow 0.10}$	$0.02_{\uparrow 0.02}$	$0.34_{\downarrow 0.06}$
Qwen2.5-VL-72B	$0.45_{\downarrow 0.16}$	$0.37_{\downarrow 0.01}$	$0.32_{\downarrow 0.09}$	0.00	$0.33_{\downarrow 0.04}$

Table 2: Agent Task Success Rate: Comparison between w/wo WM access. Almost all the models in all the agent tasks fail to reach higher performance with WM access.

Model	3DSRBench	MMSI	SAT	Spatial	Avg.
Without World Model Access					
GPT-4o-mini	0.58	0.28	0.52	0.65	0.56
GPT-4o	0.66	0.31	0.71	0.72	0.63
GPT-5-mini	0.67	0.35	0.85	0.78	0.66
GPT-5	0.69	0.38	0.86	0.80	0.68
Llama-4-Maverick	0.61	0.27	0.52	0.74	0.60
Llama-4-Scout	0.59	0.27	0.41	0.74	0.58
Qwen2.5-VL-7B	0.53	0.24	0.59	0.62	0.52
Qwen2.5-VL-32B	0.59	0.30	0.47	0.67	0.57
Qwen2.5-VL-72B	0.61	0.29	0.47	0.71	0.59
With World Model Access					
GPT-4o-mini	$0.59_{\uparrow 0.01}$	$0.27_{\downarrow 0.01}$	$0.57_{\uparrow 0.05}$	$0.66_{\uparrow 0.01}$	0.56
GPT-4o	0.66	$0.30_{\downarrow 0.01}$	$0.73_{\uparrow 0.02}$	0.72	0.63
GPT-5-mini	$0.68_{\uparrow 0.01}$	$0.36_{\uparrow 0.01}$	$0.83_{\downarrow 0.02}$	$0.79_{\uparrow 0.01}$	$0.67_{\uparrow 0.01}$
GPT-5	$0.70_{\uparrow 0.01}$	$0.37_{\downarrow 0.01}$	$0.85_{\downarrow 0.01}$	$0.79_{\downarrow 0.01}$	0.68
Llama-4-Maverick	$0.62_{\uparrow 0.01}$	$0.28_{\uparrow 0.01}$	$0.47_{\downarrow 0.05}$	$0.75_{\uparrow 0.01}$	0.60
Llama-4-Scout	0.59	$0.28_{\uparrow 0.01}$	$0.36_{\downarrow 0.05}$	$0.73_{\downarrow 0.01}$	0.58
Qwen2.5-VL-7B	$0.54_{\uparrow 0.01}$	0.24	$0.66_{\uparrow 0.07}$	$0.63_{\uparrow 0.01}$	0.52
Qwen2.5-VL-32B	$0.58_{\downarrow 0.01}$	$0.28_{\downarrow 0.02}$	0.47	0.67	$0.56_{\downarrow 0.01}$
Qwen2.5-VL-72B	0.61	0.29	$0.48_{\uparrow 0.01}$	$0.73_{\uparrow 0.02}$	0.59

Table 3: VQA Task Accuracy: Comparison between w/wo WM access across VQA benchmarks. All model performances are nearly the same for benchmarks.

Modes and Metrics. Our main evaluations compare *Normal Mode* (world model optional) with *World Model Invisible Mode* (world model unavailable), providing a clean measure of the value of optional simulation. *World Model Force Mode* is additionally applied to agentic tasks to explore whether mandatory simulation helps or harms planning. Across all agentic tasks, the performance is measured by the final task's success rate, consistent with prior work. In VQA tasks, we evaluate answer correctness on multiple-choice questions, as simulation here theoretically aids perceptual reasoning rather than sequential control.

4.3 Experiment Results

Finding 1: World Models Do Not Reliably Improve Performance.. Table 2 and Table 3 compare base models with their world model–augmented variants. Across both agent and VQA settings, the anticipated advantages of explicit foresight fail to materialize. In agent tasks, additional world model signals often introduce noise rather than guidance, leading most models (except the GPT-5 family) to perform *worse* on average. In VQA, the effect is milder but still underwhelming: gains are marginal, and performance with or without world model access is nearly indistinguishable. These findings challenge the assumption that off-policy world model rollouts inherently strengthen downstream reasoning and instead suggest that current models struggle to leverage such information in a meaningful or stable way.

Finding 2: Models Rarely Choose to Invoke the World Model. Table 4 reports each model's world model usage rate, defined as the fraction of cases in which at least one world model tool invocation is made. Usage is consistently low, particularly for VQA, where rates remain below 0.1 for all but the Llama family. This reluctance indicates that models generally do not recognize the world model as a valuable computational tool, even when demonstrations are provided. The hesitation also reflects a deeper issue: models lack a clear internal strategy for when and why world-model rollouts could meaningfully improve their predictions.

Model	Frozen Lake	Navigate	Primitive Skill	Sokoban	Avg.
Agent Task World Model Usage Rate					
GPT-4o-mini	0.3438	0.1367	0.3906	0.3906	0.2749
GPT-4o	0.4531	0.1967	0.1289	0.0469	0.1813
GPT-5-mini	0.7188	0.6100	0.5195	0.8750	0.6111
GPT-5	0.2031	0.3367	0.0742	0.1406	0.2076
Llama-4-Maverick	0.9844	0.9933	1.0000	1.0000	0.9956
Llama-4-Scout	0.8125	0.3300	0.9414	0.9375	0.6608
Qwen2.5-VL-7B	0.1406	0.0167	0.1211	0.0000	0.0658
Qwen2.5-VL-32B	0.6875	0.5433	0.5781	0.1406	0.5322
Qwen2.5-VL-72B	0.6562	0.0033	0.4180	0.0156	0.2208
Model	**3DSRBench**	**MMSI**	**SAT**	**Spatial**	**Avg.**
VQA Task World Model Usage Rate					
GPT-4o-mini	0.0818	0.3012	0.2533	0.0122	0.0972
GPT-4o	0.0014	0.0087	0.0133	0.0000	0.0022
GPT-5-mini	0.0103	0.0087	0.0133	0.0032	0.0087
GPT-5	0.0000	0.0000	0.0000	0.0000	0.0000
Llama-4-Maverick	0.3863	0.7465	0.4533	0.0733	0.3678
Llama-4-Scout	0.3923	0.5320	0.3733	0.1690	0.3639
Qwen2.5-VL-7B	0.0047	0.0054	0.0467	0.0039	0.0054
Qwen2.5-VL-32B	0.0080	0.0033	0.0067	0.0013	0.0060
Qwen2.5-VL-72B	0.0105	0.0358	0.0467	0.0000	0.0121

Table 4: World Model Usage Rate Statistics: Generally models are not that willing to use the world model as tool, especially for VQA tasks.

Finding 3: Different Model Families Exhibit Distinct Calling Behaviors. Despite the overall low usage rates, the patterns vary substantially across families. Llama models are the most proactive in querying the world model, but they show little measurable benefit. Within the GPT series, smaller models query more frequently, seemingly compensating for weaker internal reasoning, while larger models display high self-confidence and thus bypass external help. Qwen models follow a similar trend for agent tasks, except for Qwen2.5-VL-7B, which is unusually unwilling to call the world model despite being the smallest and least capable model among all that are tested. This suggests a form of potential misplaced confidence. Collectively, these patterns reveal that invocation habits are shaped more by a model's perception than by the actual utility of the world model.

Finding 4: Fine-Grained Analysis Shows a Net Neutral Impact. To understand not just whether world models are invoked but whether they *help*, we analyze case-level differences in Figure 2. The distribution reveals that: for VQA tasks, "WM Helps" and "WM Hurts" occur at nearly equal rates, indicating that models cannot robustly leverage the foresight provided, and such guidance might be treated as noise. For agent tasks, harmful rollouts occur even more frequently than helpful ones.

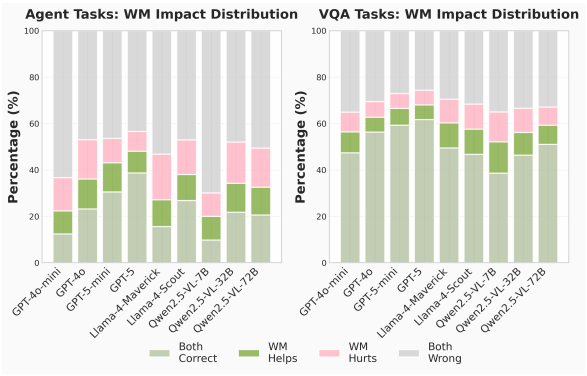

Figure 2: Percentage breakdown of the world model's impact. "WM Helps" is generally less frequent than "WM Hurts" for agent tasks, while VQA shows a more balanced distribution.

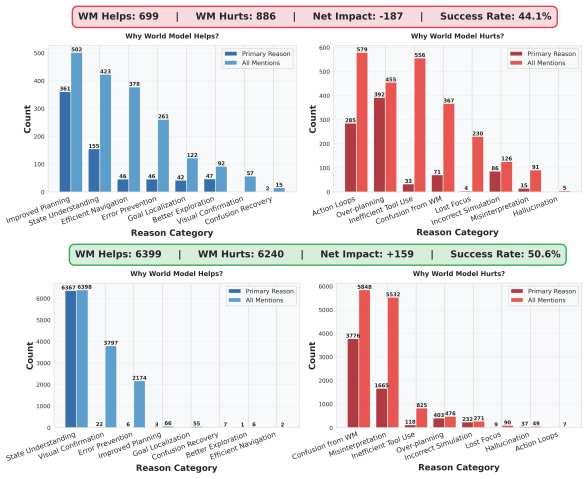

Figure 3: Attributions for when world model use helps or hurts across Agent (top) and VQA (bottom) tasks.

Overall, the detailed statistics reinforce a consistent conclusion: under current interfaces and model behaviors, world-model access offers limited practical advantage and can even be counterproductive. Understanding *why* this happens requires a deeper examination of success and failure modes, which we analyze in the next section.

5 Analysis

5.1 Attribution Analysis

We categorize eight major ways in which world-model simulations influence outcomes and annotate both primary and contributing reasons for each case with the help of GPT-4o. The aggregated statistics in Figure 3 uncover a consistent pattern: models can extract value from foresight, but their use of it is poorly calibrated, leading to fragile gains and frequent regressions. These patterns motivate a structured taxonomy of world model governance suc-

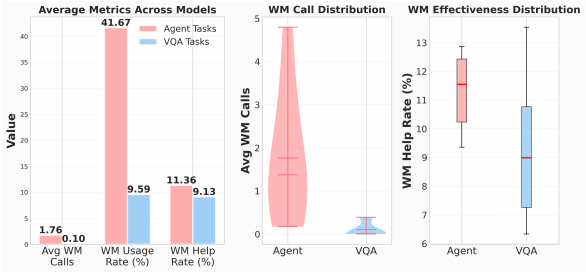

Figure 4: Comparison of Agent vs. VQA tasks. Former shows higher world model usage and higher help rate.

cesses (Figure 12) and failures (Figure 13) across the cognitive pipeline.

Finding 5: Agent Tasks Provide Useful Planning Signals, but Instability in Execution. In agent tasks, rollouts help mainly by improving planning and state understanding, confirming that simulated futures *can* guide multi-step decisions. However, failures are more frequent and more varied. The prevalence of action loops and over-planning shows that models tend to overreact to simulated information, repeatedly re-planning without making meaningful progress. Additional failure modes, including inefficient tool use, misinterpreting simulations, and loss of focus, all point to a deeper issue: models lack a stable policy for *how* to integrate world-model feedback. Instead of grounding simulations in a coherent trajectory, they oscillate between cautious over-analysis and unfocused repetition. This makes foresight beneficial in isolated cases but detrimental over long horizons.

Finding 6: VQA Tasks Can Provide Targeted Benefits but Amplified Ambiguity. For VQA, the influence of world models is more concentrated. Rollouts help primarily through improved state understanding and visual confirmation, which acts as a safeguard against committing an incorrect answer. Yet the dominant failure modes, confusion and misinterpretation, stem from a feedback loop: the model formulates unclear simulation requests, the world model returns ambiguous scenarios, and these further mislead the VLM. Thus, instead of correcting uncertainty, simulations often magnify it when the initial instruction is under-specified.

Implications: The Core Bottleneck Is Foresight Governance. Instead of struggling with future simulation, current agents often fail to govern foresight–deciding what to simulate (input governance), how to interpret it (meaning governance), and when to act upon it (action governance). In

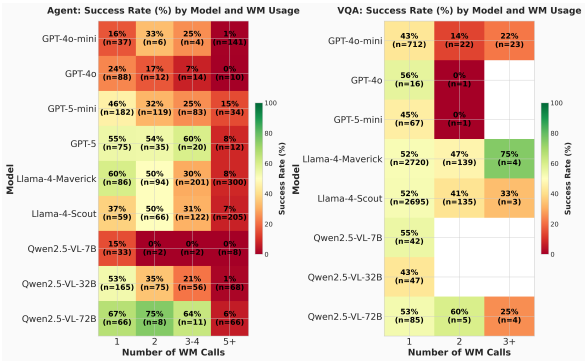

Figure 5: Success rate as a function of world model call count. More calls often correlate with worse outcomes.

agent tasks this leads to unstable execution, while in VQA it amplifies ambiguity. Thus, improving the world model alone is insufficient: what is missing is an internal mechanism for principled, calibrated use of foresight. Designing such mechanisms is crucial for realizing the potential of world model–augmented systems. Figure 12 and Figure 13 further summarize these patterns as complementary taxonomies of world model governance failures and successes, revealing that stable foresight hinges on governance rather than simulation.

5.2 Further Studies

Beyond attribution-level patterns, we further analyze how world model usage behaves across tasks, call frequencies, and beyond.

Task Perspective: Agent Tasks Leverage Foresight More Effectively. As shown in Figure 4, agent tasks consistently exhibit higher world-model usage and a higher probability that rollouts provide actual benefit. This aligns with the structure of agentic problems, where state transitions and sequential decisions naturally create opportunities for foresight to guide behavior. In contrast, VQA demands highly targeted disambiguation; without precise simulation requests, world model information is more likely to distract than assist. The gap between the two tasks suggests that current invocation patterns are better suited for dynamic, stateful environments than for static perception queries, though the latter can also benefit from the world model as tool paradigm.

Call Perspective: More Calls Reflect Uncertainty, Not Better Reasoning. Figure 5 shows a clear negative correlation between the number of world model calls and task success. Rather than accumulating useful evidence, repeated calls often signal unresolved confusion instead of scaling ef-

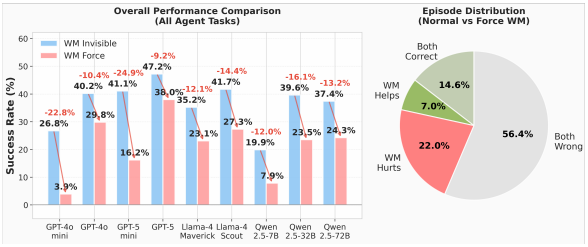

Figure 6: Comparison of agent task performance between world model invisible and forced world model use conditions.

fect: models re-query instead of integrating prior rollouts into a stable plan. This is especially visible in agent tasks, where excessive calls frequently coincide with action loops and degraded execution. The pattern reinforces a central conclusion of our empirical evidence: the challenge is not generating simulations, but knowing when to stop, how to interpret them, and how to integrate them.

Usage Perspective: Forcing World Model Invocation Degrades Performance. Across all agent tasks, we also evaluate a setting where the model is *forced* to consult its world model before producing an action. As shown in Figure 6, this intervention leads to an even steeper drop in performance than the normal mode with world model access. This outcome aligns with earlier findings: when optional world model access already introduces systematic failure modes, mandating its use amplifies those weaknesses. Both the frequency and severity of cases in which the world model hurts increase, and average performance across models declines for every task. Taken together, these results indicate that, under current designs, forcing world model usage is not effective.

5.3 Results Robustness

To test whether our conclusions are driven by simulator quality, prompt wording, or multimodal context load, we run several additional studies. Simulator-fidelity confounds mainly affect VQA, since agentic tasks already use an oracle simulator. We therefore compare the Wan2.1 against the stronger Wan2.2 on VQA, and also test: (i) a prompt variant biased toward world-model use, (ii) text-only simulator feedback, (iii) text foresight without external simulator, and (iv) a small SFT study on world-model interaction.

The stronger-simulator and prompt-variation results rule out two common alternatives. Improving the VQA simulator does not remove the main failures, so simulator quality is unlikely to be the main driver. Likewise, near-zero VQA invocation persists even when the prompt is biased toward tool use, suggesting that reluctance to simulate is not primarily a wording artifact. The text-based ablations refine this picture. Converting rollouts to text slightly helps FrozenLake, indicating that visual context load matters, but it hurts 3DSRBench, and text-only foresight still underperforms optional visual foresight. Together, these results suggest that the limitation is broader: current agents struggle to decide when to query, how to query, and how to update beliefs from simulated evidence.

The SFT result further indicates that this missing behavior is at least partly learnable. Training on only 531 correct interaction trajectories increases world-model usage and gives modest in-domain gains, but transfer out of domain remains weak. We therefore view foresight governance mainly as a training problem rather than a prompt-only problem: future work should jointly optimize invocation, interpretation, and rollout integration, rather than assuming that a stronger simulator alone will solve the issue.

6 Discussion

World Model for Agent Confirmation. Attribution analysis suggests world models mostly help as **visual confirmation** in VQA: the tested model forms a hypothesis and asks the world model to verify it, which can filter out implausible actions. However, when the initial hypothesis is wrong, a confirmation-style query can lock in the error by producing a plausible simulation aligned with that wrong belief, empirically matching the cases where world models hurt. This reflects a broader limitation: current agents treat the world model as a verifier of a single guess rather than a tool to surface alternatives the agent has not yet considered. A more corrective protocol is to promote **discrimination** rather than affirmation. Specifically, the agent may propose several plausible hypotheses, simulate each with the world model, and pick the one whose predicted cues best match the observation. This turns simulation into structured hypothesis testing, thus increasing the chance of correcting an initial error rather than reinforcing it.

Dedicated Modules for Integration of Foresight. Foresight is currently integrated by simply expanding the context over multiple rounds, with demonstrations in the system prompt. However in prac-

Ablation	Setup	Key observation
Employ a stronger VQA simulator	Replace Wan2.1 with Wan2.2 for GPT-5 and GPT-5-mini on 3DSRBench and SAT.	Accuracy changes are negligible (≤ 0.01 in all four comparisons), and the overlap of correctly solved examples is above 95%.
Prompt variation	Use a simpler prompt that favors world-model use and retains only WM-using demonstrations; evaluate GPT-4o and GPT-5 on the PrimitiveSkill and 3DSRBench tasks.	Usage changes only slightly (PrimitiveSkill: GPT-4o $0.1289 \rightarrow 0.1372$, GPT-5 $0.0742 \rightarrow 0.0719$; 3DSRBench: $0.0014 \rightarrow 0.0023$ and $0.0000 \rightarrow 0.0000$).
Apply text-only simulator feedback	Replace returned frames with textual descriptions from Qwen2.5-VL-72B; evaluate GPT-5 and Qwen2.5-VL-7B on FrozenLake and 3DSRBench.	Text feedback slightly helps FrozenLake ($0.89 \rightarrow 0.90, 0.45 \rightarrow 0.47$) but hurts 3DSRBench ($0.70 \rightarrow 0.62, 0.54 \rightarrow 0.51$).
Use text foresight without external simulator	Remove the simulator and require a textual prediction field before the action or answer.	This improves over WM-Invisible in some cases (e.g., GPT-5 on FrozenLake: $0.77 \rightarrow 0.82$) but still trails optional WM (0.89).
Preliminary interaction training	SFT Qwen2.5-VL-7B on 531 correct interaction trajectories from FrozenLake, Navigate, SAT, and Spatial tasks.	Gains are small in-domain (FrozenLake: $0.45 \rightarrow 0.48$; SAT: $0.65 \rightarrow 0.69$), OOD transfer is weak (3DSRBench: $0.54 \rightarrow 0.53$), and WM usage increases by about 13%.

Table 5: Robustness and preliminary-solution ablations. Across stronger simulators, prompt variants, text-based alternatives, and limited interaction training, the main result is unchanged: the key bottleneck is not access to foresight alone, but the ability to invoke, interpret, and integrate it reliably.

tice, agents often discount simulator observations and instead become more confident in their initial (incorrect) beliefs. A more reliable approach is to introduce dedicated modules that enforce a structured interaction loop between reasoning and foresight: (i) **Decider**: determines whether to invoke the world model or execute a real action, and boldly proposes candidate actions to test; (ii) **Reflector**: evaluates observations from real execution or simulation, checks whether outcomes match the expectations, and feeds back what to adjust next; (iii) **Memory**: maintains long and short-term task objectives by storing observations and selectively releasing relevant context to the Decider and Reflector. Overall, a dedicated mechanism can make foresight usage explicit and organized, reducing sprawling, brittle chains of multimodal reasoning where the agent may get lost in the middle.

Agent Training for Better World Model Interactions. Still, without fine-tuning it is difficult to reshape an agent's default behavior: prompting rarely teaches *when* to invoke a world model or *how* to query it effectively. To intrinsically improve this paradigm, training is necessary. One practical route is RL with online, multi-turn rollouts where the world model is an explicit tool, using rewards that go beyond final answer correctness to encourage: (i) **appropriate** world model invocation, and (ii) **diverse, distinctive** queries (e.g., measured semantically). The challenge is credit assignment: we want to reward useful interaction without inducing indiscriminate calling. In addition to penalties for excessive or repeated invocations, two complementary remedies are: (i) constructing high-quality supervised fine-tuning data to set strong cold start interaction habits; and (ii) using indirect objectives such as rewarding information gain (e.g., reductions in hypothesis entropy). These incentives directly target the attribution-identified failure modes and should yield more strategic and exploratory world model use.

7 Conclusion and Future Work

Our investigation reveals that giving agents access to a world model reshapes their behavior in unexpected ways. Rather than serving as a straightforward enhancement, simulation introduces new cognitive pressures: agents must manage hypothetical branches and maintain coherent reasoning across mixed real and imagined experience. The difficulties we observe, including hesitation, over-analysis, and misaligned interpretation, suggest that effective foresight requires more fine-grained governance. Looking ahead, promising directions include learning architectures that maintain stable internal state across simulated and real trajectories, exploration-driven simulation policies, and interfaces that encourage agents to articulate and refine their hypotheses before querying a world model. Beyond performance, studying how agents build and anchor beliefs across possible futures may deepen anticipatory cognition and integrate world model simulation directly into decision-making.

Limitations

Our study covers a limited set of model families; we currently exclude Gemini and Claude due to cost constraints. While this narrows breadth, the models we evaluate are widely used and suffice to probe general trends in the willingness to employ world models and the effectiveness of doing so. Our evaluation also uses different simulators

across task types: agentic tasks rely on a ground-truth simulator, whereas VQA uses WAN2.1 as the simulator. This design choice keeps the focus on agent-side behavior rather than simulator quality, but it may introduce mild cross-task incomparability. In practice, we observe occasional quality artifacts in WAN2.1-generated frames that can confuse agents. Notably, many failure cases trace back to underspecified or ambiguous agent instructions, which degrade simulation quality and in turn affect downstream performance. We view these as actionable future directions. Future work will expand coverage to additional model families and employ stronger, more consistent simulators to better isolate agent effectiveness.

Ethical Statement

This paper investigates "world models as tools" for VLM agents in controlled experiments (simulating in a cloned environment state or via generative rollouts) rather than real-world deployment. A key ethical risk is that simulated futures can be over-trusted or misread, leading to unsafe decisions if transferred to high-stakes domains; in fact, our results show that forcing simulation can degrade performance and create new failure modes, underscoring that naive use is not reliably beneficial. To reduce potential harm, we recommend treating simulations as uncertain hypotheses instead of ground truth, encouraging multi-hypothesis checks rather than "confirmation" rollouts, and avoiding safety-critical deployment without robust oversight and privacy protections.

Acknowledgment

This research is supported by DARPA ITM Program No. FA8650-23-C-7316 and Capital One-Illinois Center for Generative AI Safety, Knowledge Systems, and Cybersecurity (ASKS). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

Saaket Agashe, Kyle Wong, Vincent Tu, Jiachen Yang, Ang Li, and Xin Eric Wang. 2025. Agent s2: A compositional generalist-specialist framework for computer use agents. *arXiv preprint arXiv:2504.00906*.

Meta AI. 2025. The llama 4 herd: The beginning of a new era of natively multimodal ai. https://ai.meta.com/blog/llama-4-multimodal-intelligence/.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. 2025. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. 2024a. Video generation models as world simulators.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, et al. 2024b. Video generation models as world simulators. *OpenAI Blog*, 1(8):1.

Quentin Carbonneaux, Gal Cohen, Jonas Gehring, Jacob Kahn, Jannik Kossen, Felix Kreuk, Emily McMilin, Michel Meyer, Yuxiang Wei, David Zhang, et al. 2025. Cwm: An open-weights llm for research on code generation with world models. *arXiv preprint arXiv:2510.02387*.

Hyungjoo Chae, Namyoung Kim, Kai Tzu iunn Ong, Minju Gwak, Gwanwoo Song, Jihoon Kim, Sunghwan Kim, Dongha Lee, and Jinyoung Yeo. 2024. Web agents with world models: Learning and leveraging environment dynamics in web navigation. *Preprint*, arXiv:2410.13232. ICLR 2025 submission (arXiv preprint).

Mingkai Deng, Jinyu Hou, Zhiting Hu, and Eric Xing. 2025a. Simura: A world-model-driven simulative reasoning architecture for general goal-oriented agents. *arXiv preprint arXiv:2507.23773*.

Xiang Deng, Jeff Da, Edwin Pan, Yannis Yiming He, Charles Ide, Kanak Garg, Niklas Lauffer, Andrew Park, Nitin Pasari, Chetan Rane, Karmini Sampath, Maya Krishnan, Srivatsa Kundurthy, Sean Hendryx, Zifan Wang, Vijay Bharadwaj, Jeff Holm, Raja Aluri, Chen Bo Calvin Zhang, Noah Jacobson, Bing Liu, and Brad Kenstler. 2025b. Swe-bench pro: Can ai agents solve long-horizon software engineering tasks? *arXiv preprint arXiv:2509.16941*.

Lutfi Eren Erdogan, Nicholas Lee, Sehoon Kim, Suhong Moon, Hiroki Furuta, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. 2025. Plan-and-act: Improving planning of agents for long-horizon tasks. *arXiv preprint arXiv:2503.09572*. ICML 2025.

Yiguo Fan, Pengxiang Ding, Shuanghao Bai, Xinyang Tong, Yuyang Zhu, Hongchao Lu, Fengqi Dai, Wei Zhao, Yang Liu, Siteng Huang, Zhaoxin Fan, Badong

Chen, and Donglin Wang. 2025. Long-vla: Unleashing long-horizon capability of vision language action model for robot manipulation. *arXiv preprint arXiv:2508.19958*. Accepted to CoRL 2025.

Yunhai Feng, Jiaming Han, Zhuoran Yang, Xiangyu Yue, Sergey Levine, and Jianlan Luo. 2025. Reflective planning: Vision-language models for multi-stage long-horizon robotic manipulation. *arXiv preprint arXiv:2502.16707*.

Divyansh Garg, Shaun VanWeelden, Diego Caples, Andis Draguns, Nikil Ravi, Pranav Putta, Naman Garg, Tomas Abraham, Michael Lara, Federico Lopez, James Liu, Atharva Gundawar, Prannay Hebbar, Youngchul Joo, Jindong Gu, Charles London, Christian Schroeder de Witt, and Sumeet Motwani. 2025. Real: Benchmarking autonomous agents on deterministic simulations of real websites. *arXiv preprint arXiv:2504.11543*.

Longling Geng and Edward Y. Chang. 2025. Realm-bench: A benchmark for evaluating multi-agent systems on real-world, dynamic planning and scheduling tasks. *arXiv preprint arXiv:2502.18836*.

Daniel T. Gilbert and Timothy D. Wilson. 2007. Prospection: Experiencing the future. *Science*, 317(5843):1351–1354.

Alexi Gladstone, Ganesh Nanduru, Md Mofijul Islam, Peixuan Han, Hyeonjeong Ha, Aman Chadha, Yilun Du, Heng Ji, Jundong Li, and Tariq Iqbal. 2025. Energy-based transformers are scalable learners and thinkers. In *arxiv*.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8154–8173.

Haoran He, Yang Zhang, Liang Lin, Zhongwen Xu, and Ling Pan. 2025. Pre-trained video generative models as world simulators. In *ICLR 2025 Workshop on World Models: Understanding, Modelling and Scaling*.

Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. 2024. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. *arXiv preprint arXiv:2409.01652*.

Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. 2023. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*.

Emily Jin, Zhuoyi Huang, Jan-Philipp Fränken, Weiyu Liu, Hannah Cha, Erik Brockbank, Sarah Wu, Ruohan Zhang, Jiajun Wu, and Tobias Gerstenberg. 2024. Marple: A benchmark for long-horizon inference. *Preprint*, arXiv:2410.01926. Preprint. Under review.

PN Johnson-Laird. 1983. *Mental models: Towards a cognitive science of language, inference, and consciousness*. Harvard University Press.

Wufei Ma, Haoyu Chen, Guofeng Zhang, Yu-Cheng Chou, Jieneng Chen, Celso de Melo, and Alan Yuille. 2025. 3dsrbench: A comprehensive 3d spatial reasoning benchmark. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6924–6934.

Siwei Meng, Yawei Luo, and Ping Liu. 2025. Magic: Motion-aware generative inference via confidence-guided llm. *arXiv preprint arXiv:2505.16456*.

OpenAI. 2025. Gpt-5 system card. `https://cdn.openai.com/gpt-5-system-card.pdf`.

David Premack and Guy Woodruff. 1978. Does the chimpanzee have a theory of mind? *Behavioral and Brain Sciences*, 1(4):515–526.

Cheng Qian, Peixuan Han, Qinyu Luo, Bingxiang He, Xiusi Chen, Yuji Zhang, Hongyi Du, Jiarui Yao, Xiaocheng Yang, Denghui Zhang, et al. 2024. Escapebench: Pushing language models to think outside the box. *arXiv preprint arXiv:2412.13549*.

Arijit Ray, Jiafei Duan, Ellis Brown, Reuben Tan, Dina Bashkirova, Rose Hendrix, Kiana Ehsani, Aniruddha Kembhavi, Bryan A Plummer, Ranjay Krishna, et al. 2024. Sat: Dynamic spatial aptitude training for multimodal language models. *arXiv preprint arXiv:2412.07755*.

Daniel L. Schacter, Donna Rose Addis, and Randy L. Buckner. 2007. Remembering the past to imagine the future: the prospective brain. *Nature Reviews Neuroscience*, 8:657–661.

Fatemeh Shiri, Xiao-Yu Guo, Mona Golestan Far, Xin Yu, Reza Haf, and Yuan-Fang Li. 2024. An empirical analysis on spatial reasoning capabilities of large multimodal models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21440–21455.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. 2025. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*.

Huacan Wang, Ziyi Ni, Shuo Zhang, Shuo Lu, Sen Hu, Ziyang He, Chen Hu, Jiaye Lin, Yifu Guo, Ronghao Chen, Xin Li, Daxin Jiang, Yuntao Du, and Pin Lyu. 2025a. Repomaster: Autonomous exploration and understanding of github repositories for complex task solving. *arXiv preprint arXiv:2505.21577*.

Kangrui Wang, Pingyue Zhang, Zihan Wang, Yaning Gao, Linjie Li, Qineng Wang, Hanyang Chen, Chi Wan, Yiping Lu, Zhengyuan Yang, et al. 2025b. Vagen: Reinforcing world model reasoning for multi-turn vlm agents. *arXiv preprint arXiv:2510.16907*.

Zhenhailong Wang, Haiyang Xu, Junyang Wang, Xi Zhang, Ming Yang, Ji Zhang, Fei Huang, and Heng Ji. 2025c. Mobile-agent-e: Self-evolving mobile assistant for complex real-world tasks. In *Proc. NeurIPS2025 Workshop on Scaling Environments for Agents*.

Sihan Yang, Runsen Xu, Yiman Xie, Sizhe Yang, Mo Li, Jingli Lin, Chenming Zhu, Xiaochen Chen, Haodong Duan, Xiangyu Yue, et al. 2025. Mmsi-bench: A benchmark for multi-image spatial intelligence. *arXiv preprint arXiv:2505.23764*.

Jiahan Zhang, Muqing Jiang, Nanru Dai, Taiming Lu, Arda Uzunoglu, Shunchi Zhang, Yana Wei, Jiahao Wang, Vishal M Patel, Paul Pu Liang, et al. 2025a. World-in-world: World models in a closed-loop world. *arXiv preprint arXiv:2510.18135*.

Kevin Zhang, Kuangzhi Ge, Xiaowei Chi, Renrui Zhang, Shaojun Shi, Zhen Dong, Sirui Han, and Shanghang Zhang. 2025b. Can world models benefit vlms for world dynamics? *arXiv preprint arXiv:2510.00855*.

Appendix

A Evaluation Settings Details

In this section, we describe the evaluation settings and hyperparameters used throughout our experiments. Unless otherwise specified, all tested models are evaluated with an inference temperature of 0.0 and a maximum output length of 2048 tokens per turn. Each episode is constrained to a maximum of 15 turns across all tasks.

For agentic tasks, we directly leverage the real environment simulation as the tested model's foresight mechanism. Specifically, we use all frames produced by the environment when executing the model's predicted simulation actions, and we cap the number of incorporated frames to the reasoning trajectory at three for each step.

For VQA tasks, we adopt WAN2.1 as the world model and use Qwen2.5-3B-VL for prompt augmentation, following official recommendations. The hyperparameters for generating world model frames are provided in the following:

Hyperparameter	Value
Frame Num	21
Steps	15
Shift	16.0
Guide Scale	5.0
Sample Solver	unipc
Seed	-1
Timeout	1200

Table 6: Hyperparameters used in WAN2.1 as world model.

In both categories of tasks, we limit the number of frames incorporated into the model's reasoning trajectory to three. To reduce redundancy, we first filter frames so that the similarity between any two consecutive frames remains below 0.95. From the filtered set, we then uniformly interpolate the sequence to obtain three representative frames. The final frame generated by the world model is always included in the returned set.

For each data point, we perform a single evaluation. Task accuracy is then computed by averaging the accuracy across all data points, where accuracy corresponds to the success rate for agent tasks and the multiple choice accuracy for VQA tasks.

B Additional Experiment Results

Additional Main Experiment Results. In Table 7 and Table 8, we extend our original results by reporting success rates and accuracies under the forced world model usage setting. The results indicate that mandating the use of the world model does not improve, but often degrades, the effectiveness of world model utilization by the tested models. In fact, forcing world model usage generally yields worse performance than allowing the model to access the world model as an optional tool. These findings provide a more direct comparison and further support our main claim that compulsory world model usage is not a scalable or effective strategy for current VLM-based agents.

Model	Frozen Lake	Navigate	Primitive Skill	Sokoban	Avg.
Without World Model Access					
GPT-4o-mini	0.36	0.26	0.32	0.00	0.27
GPT-4o	0.58	0.35	0.51	0.02	0.40
GPT-5-mini	0.86	0.66	0.11	0.03	0.41
GPT-5	0.77	0.74	0.19	0.06	0.47
Llama-4-Maverick	0.70	0.31	0.40	0.00	0.35
Llama-4-Scout	0.59	0.55	0.32	0.02	0.42
Qwen2.5-VL-7B	0.36	0.26	0.13	0.02	0.20
Qwen2.5-VL-32B	0.59	0.36	0.49	0.00	0.40
Qwen2.5-VL-72B	0.61	0.38	0.41	0.00	0.37
With World Model Access					
GPT-4o-mini	0.39	0.24	0.22	0.00	0.22
GPT-4o	0.58	0.31	0.46	0.02	0.36
GPT-5-mini	0.89	0.63	0.19	0.00	0.43
GPT-5	0.89	0.71	0.20	0.14	0.48
Llama-4-Maverick	0.66	0.20	0.32	0.03	0.27
Llama-4-Scout	0.55	0.54	0.24	0.03	0.38
Qwen2.5-VL-7B	0.45	0.26	0.12	0.00	0.20
Qwen2.5-VL-32B	0.58	0.32	0.39	0.02	0.34
Qwen2.5-VL-72B	0.45	0.37	0.32	0.00	0.33
Force WM Use					
GPT-4o-mini	0.06	0.05	0.03	0.00	0.04
GPT-4o	0.58	0.21	0.40	0.00	0.30
GPT-5-mini	0.67	0.21	0.02	0.02	0.16
GPT-5	0.91	0.58	0.09	0.06	0.38
Llama-4-Maverick	0.64	0.12	0.31	0.00	0.23
Llama-4-Scout	0.50	0.36	0.18	0.00	0.27
Qwen2.5-VL-7B	0.25	0.12	0.01	0.00	0.08
Qwen2.5-VL-32B	0.39	0.25	0.24	0.00	0.24
Qwen2.5-VL-72B	0.53	0.27	0.20	0.00	0.24

Table 7: Agent Task Success Rate: Comprehensive comparison between without WM access, with WM access, and force WM use.

Additional Main Experiment Statistics. In Table 9 and Table 10, we report the frequency and rate of world model usage for each agent task across all evaluated models, under both the normal set-

Model	3DSRBench	MMSI	SAT	Spatial	Avg.
Without World Model Access					
GPT-4o-mini	0.58	0.28	0.52	0.65	0.56
GPT-4o	0.66	0.31	0.71	0.72	0.63
GPT-5-mini	0.67	0.35	0.85	0.78	0.66
GPT-5	0.69	0.38	0.86	0.80	0.68
Llama-4-Maverick	0.61	0.27	0.52	0.74	0.60
Llama-4-Scout	0.59	0.27	0.41	0.74	0.58
Qwen2.5-VL-7B	0.53	0.24	0.59	0.62	0.52
Qwen2.5-VL-32B	0.59	0.30	0.47	0.67	0.57
Qwen2.5-VL-72B	0.61	0.29	0.47	0.71	0.59
With World Model Access					
GPT-4o-mini	0.59	0.27	0.57	0.66	0.56
GPT-4o	0.66	0.30	0.73	0.72	0.63
GPT-5-mini	0.68	0.36	0.83	0.79	0.67
GPT-5	0.70	0.37	0.85	0.79	0.68
Llama-4-Maverick	0.62	0.28	0.47	0.75	0.60
Llama-4-Scout	0.59	0.28	0.36	0.73	0.58
Qwen2.5-VL-7B	0.54	0.24	0.66	0.63	0.52
Qwen2.5-VL-32B	0.58	0.28	0.47	0.67	0.56
Qwen2.5-VL-72B	0.61	0.29	0.48	0.73	0.59
Force WM Use					
GPT-4o-mini	0.59	0.28	0.61	0.64	0.56
GPT-4o	0.66	0.29	0.67	0.72	0.63
GPT-5-mini	0.67	0.34	0.81	0.80	0.66
GPT-5	0.69	0.33	0.85	0.79	0.67
Llama-4-Maverick	0.61	0.30	0.50	0.73	0.59
Llama-4-Scout	0.59	0.28	0.41	0.73	0.58
Qwen2.5-VL-7B	0.51	0.22	0.37	0.55	0.48
Qwen2.5-VL-32B	0.57	0.27	0.38	0.66	0.55
Qwen2.5-VL-72B	0.59	0.28	0.49	0.71	0.58

Table 8: VQA Task Accuracy: Comprehensive comparison between without WM access, with WM access, and force WM use.

ting and the forced world model usage setting. We observe that explicitly forcing world model usage in the system prompt leads to a higher absolute number of world model invocations per episode. However, even under this enforced setting, some models do not consistently comply. Notably, for GPT-5, the world model usage rate still fails to reach 1.0, indicating that the model does not use the world model at least once in every episode, despite explicit instructions to do so before each action.

We present analogous statistics for VQA tasks in Table 13 and Table 14, where we observe similar overall trends. An exception arises in SAT tasks, for which models appear particularly reluctant to use the world model as a tool, even under the forced usage setting.

Additionally, in Table 11 and Table 12, we

Model	FrozenLake	Navigation	PrimitiveSkill	Sokoban	Overall
Average WM Calls					
GPT-4o-mini	1.3281	1.2467	4.1484	3.3906	2.5409
GPT-4o	0.6250	0.2100	0.5000	0.1250	0.3494
GPT-5-mini	1.5625	1.2833	0.9375	2.9375	1.3348
GPT-5	0.2188	0.8067	0.0742	0.1875	0.4196
Llama-4-Maverick	3.0000	5.6533	4.0352	5.6250	4.7968
Llama-4-Scout	4.1719	1.0933	4.5352	7.7812	3.2953
Qwen2.5-VL-7B	0.1406	0.0900	0.3164	0.0000	0.1711
Qwen2.5-VL-32B	1.0312	1.7167	1.7969	0.4688	1.5658
Qwen2.5-VL-72B	1.6406	0.0267	3.2305	0.0156	1.3757
WM Usage Rate					
GPT-4o-mini	0.3438	0.1367	0.3906	0.3906	0.2749
GPT-4o	0.4531	0.1967	0.1289	0.0469	0.1813
GPT-5-mini	0.7188	0.6100	0.5195	0.8750	0.6111
GPT-5	0.2031	0.3367	0.0742	0.1406	0.2076
Llama-4-Maverick	0.9844	0.9933	1.0000	1.0000	0.9956
Llama-4-Scout	0.8125	0.3300	0.9414	0.9375	0.6608
Qwen2.5-VL-7B	0.1406	0.0167	0.1211	0.0000	0.0658
Qwen2.5-VL-32B	0.6875	0.5433	0.5781	0.1406	0.5322
Qwen2.5-VL-72B	0.6562	0.0033	0.4180	0.0156	0.2208

Table 9: Agent Task WM Statistics (With WM Access): Upper section shows average WM calls per episode. Lower section shows WM usage rate (percentage of episodes that used WM at least once).

report statistics on the average number of steps per episode and the action validity rate for agent tasks. Given that the maximum number of steps per episode is capped at 15, we find that many models frequently reach this limit across all data points for certain tasks. This behavior may lead to repetitive actions, a phenomenon we analyze in detail in the attribution analysis presented in the main text. Further, we observe no meaningful differences in either step count or action validity rate across different evaluation modes. This suggests that these factors are not primary contributors to the observed performance differences, but instead serve to further corroborate the conclusions drawn in our attribution analysis.

Additional Comparative Analysis. In Table 20, Table 21, Table 22, and Table 23, we present a detailed breakdown of the percentage of cases in which the world model either improves or degrades performance relative to the world model invisible setting. These comparisons are reported separately for agentic and VQA tasks, under both the normal world model access and forced world model usage configurations.

Across all evaluated tasks and models, we observe that the proportion of cases in which the world model improves performance does not consistently exceed the proportion of cases in which it degrades performance. This indicates that the ef-

Model	FrozenLake	Navigation	PrimitiveSkill	Sokoban	Overall
Average WM Calls					
GPT-4o-mini	5.9062	13.9467	12.2969	10.5781	12.2617
GPT-4o	2.5938	7.9900	6.9102	3.2656	6.6389
GPT-5-mini	5.9062	12.1667	8.0664	8.7500	9.7266
GPT-5	1.6719	6.7300	3.0273	1.8594	4.4152
Llama-4-Maverick	2.6094	9.7300	6.9805	7.7500	7.8494
Llama-4-Scout	7.7500	9.1033	8.1367	10.6562	8.7602
Qwen2.5-VL-7B	4.5156	7.1867	6.8164	8.0000	6.8743
Qwen2.5-VL-32B	2.9531	8.6767	6.8594	8.0938	7.4064
Qwen2.5-VL-72B	3.6562	9.1633	8.7891	9.7188	8.5599
WM Usage Rate					
GPT-4o-mini	1.0000	1.0000	1.0000	1.0000	1.0000
GPT-4o	1.0000	1.0000	1.0000	1.0000	1.0000
GPT-5-mini	1.0000	1.0000	0.9492	1.0000	0.9810
GPT-5	0.9375	0.9967	0.7930	0.7812	0.8947
Llama-4-Maverick	1.0000	1.0000	1.0000	1.0000	1.0000
Llama-4-Scout	1.0000	1.0000	1.0000	1.0000	1.0000
Qwen2.5-VL-7B	1.0000	0.9000	1.0000	1.0000	0.9561
Qwen2.5-VL-32B	1.0000	1.0000	1.0000	1.0000	1.0000
Qwen2.5-VL-72B	1.0000	1.0000	1.0000	1.0000	1.0000

Table 10: Agent Task WM Statistics (Force WM Use): Upper section shows average WM calls per episode when forced to use WM. Lower section shows WM usage rate (expected to be 100% in forced mode).

fectiveness of world models is not reliably demonstrated across models or task settings. These statistics further support our main conclusions regarding the limited effectiveness of world models for current VLM-based agents and motivate our subsequent investigations into the specific conditions under which the world model helps or hurts performance, which we analyze through detailed attribution studies.

C Attribution Analysis Details

We begin by manually analyzing 60 cases in which the world model either improves or degrades performance. Through detailed inspection, we identify and summarize the core categories that explain when and why the world model is effective or fails to assist the tested model. Based on this manual analysis, we curate a set of eight attribution categories respectively that capture the primary failure and success modes.

Following this initial human analysis, we employ GPT-4o to assist with large-scale attribution. For each case, GPT-4o is tasked with identifying one primary attribution reason and any number of additional contributing reasons. To perform this analysis, we provide GPT-4o with the interaction trajectory of the tested model when the world model is invisible, along with the corresponding trajectory for the same query when the world model is available (or enforced to use). GPT-4o is then instructed

Model	FrozenLake	Navigation	PrimitiveSkill	Sokoban	Overall
Without WM Access					
GPT-4o-mini	7.81	12.91	4.12	15.00	9.34
GPT-4o	3.75	11.27	4.36	14.78	8.31
GPT-5-mini	1.55	7.95	5.91	14.12	7.17
GPT-5	4.75	7.52	6.92	14.02	7.65
Llama-4-Maverick	2.27	12.73	6.52	15.00	9.64
Llama-4-Scout	3.12	9.86	6.88	14.78	8.58
Qwen2.5-VL-7B	6.09	12.16	8.12	14.80	10.33
Qwen2.5-VL-32B	2.03	11.48	4.98	15.00	8.49
Qwen2.5-VL-72B	2.44	10.96	4.58	15.00	8.15
With WM Access					
GPT-4o-mini	7.89	12.94	8.58	15.00	11.03
GPT-4o	5.08	11.75	5.43	14.78	9.05
GPT-5-mini	3.19	8.68	7.12	14.95	8.17
GPT-5	3.58	8.02	8.24	14.17	8.26
Llama-4-Maverick	6.88	13.80	9.99	14.75	11.81
Llama-4-Scout	7.56	10.01	11.13	14.69	10.64
Qwen2.5-VL-7B	2.05	12.30	8.19	15.00	10.05
Qwen2.5-VL-32B	5.12	11.66	6.75	14.86	9.51
Qwen2.5-VL-72B	4.47	11.01	8.42	15.00	9.80
Force WM Use					
GPT-4o-mini	12.52	14.65	14.52	15.00	14.43
GPT-4o	6.73	13.45	10.17	15.00	11.74
GPT-5-mini	7.44	13.47	14.23	14.77	13.31
GPT-5	5.62	10.76	12.60	14.77	11.34
Llama-4-Maverick	4.83	14.52	11.95	15.00	12.70
Llama-4-Scout	10.52	12.81	12.97	15.00	12.86
Qwen2.5-VL-7B	8.97	14.08	13.73	15.00	13.56
Qwen2.5-VL-32B	6.08	12.86	11.05	15.00	11.75
Qwen2.5-VL-72B	6.00	12.85	12.34	15.00	12.22

Table 11: Agent Task Average Steps: Comparison of average trajectory length between without WM access, with WM access, and forced WM use.

to compare these trajectories and assign attribution labels accordingly.

To assess the reliability of the automated annotations, we manually review all 60 cases that were previously examined and annotated by humans, comparing GPT-4o's primary attribution labels with human judgments. The agreement rate for the primary attribution reaches 81.67%, indicating strong alignment and suggesting that the model's case-by-case annotations are reliable. In addition, we randomly sample 20 further cases for manual verification, finding that all attribution reasons produced by GPT-4o are reasonable. Based on these validation results, we report attribution statistics for all cases using the automated annotations.

D Additional Analysis Results

In this section, we present more analysis results in addition to the analysis in the main text.

In Figure 7, we examine how world model usage and its effectiveness vary jointly with model family and scale, across both agent tasks and VQA settings. A clear pattern emerges that increased

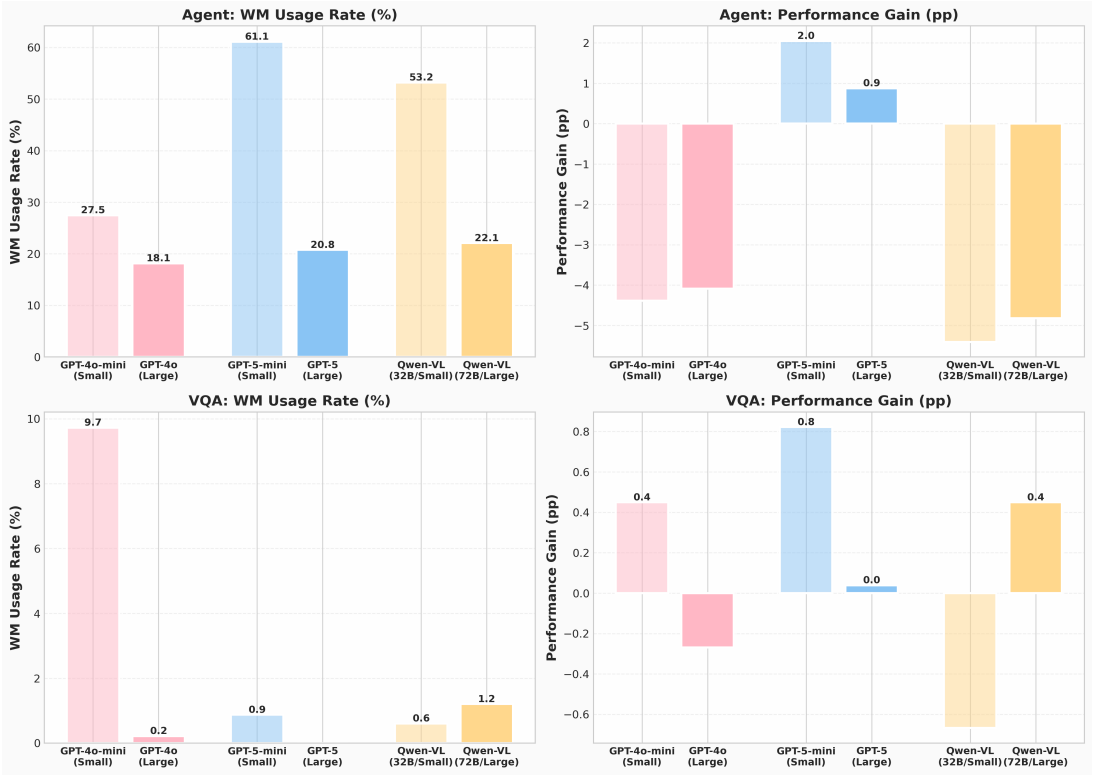

Figure 7: Comparison of world model usage rate and corresponding performance gain across model families and scales, for both agentic tasks (top row) and VQA tasks (bottom row). Results are aggregated by model family (GPT, Qwen) and size, highlighting systematic differences in invocation behavior and the limited, sometimes negative, returns of increased world model usage.

world model invocation does not consistently yield positive returns. For agent tasks, smaller models tend to rely more heavily on the world model and occasionally obtain modest gains, whereas larger models reduce their usage and often experience neutral or negative effects. Qwen models further illustrate that high usage alone is insufficient, as frequent but poorly calibrated world model calls correlate with pronounced performance degradation. In VQA tasks, usage rates remain uniformly low regardless of scale, and performance changes are marginal, reinforcing the conclusion that current models lack a principled mechanism for deciding when foresight is beneficial and how to integrate it effectively.

In Figure 8, we compare how often models invoke the world model when solving agent tasks versus VQA tasks. Across all backbones, agentic settings induce substantially higher call frequencies, often by an order of magnitude, reflecting the natural role of foresight in sequential decision-making. In contrast, VQA tasks trigger extremely sparse usage, with most models making fewer than one call per question on average. This disparity aligns with the main text's analysis that simulation is more naturally integrated into dynamic, stateful environments, whereas in VQA it requires highly targeted and well-specified queries that current models rarely generate. Importantly, the elevated call frequency in agent tasks does not imply better outcomes, reinforcing our broader finding that repeated world model invocation often signals uncertainty rather than effective integration of foresight.

In Figure 10 and Figure 9, we jointly analyze how the frequency of world model invocation correlates with performance in agent and VQA tasks. In both settings, higher world model call counts are associated with lower success rates, as evidenced by negative linear trends and consistent degradation in the binned statistics. The effect is more pronounced in agent tasks, where excessive calls coincide with sharp performance drops, reflecting action loops and over-planning behaviors discussed in the main text. In VQA, although overall call counts are much lower, repeated invocations still correspond to reduced accuracy, suggesting that additional simulations often amplify ambiguity rather than resolve it. Together, these results reinforce the

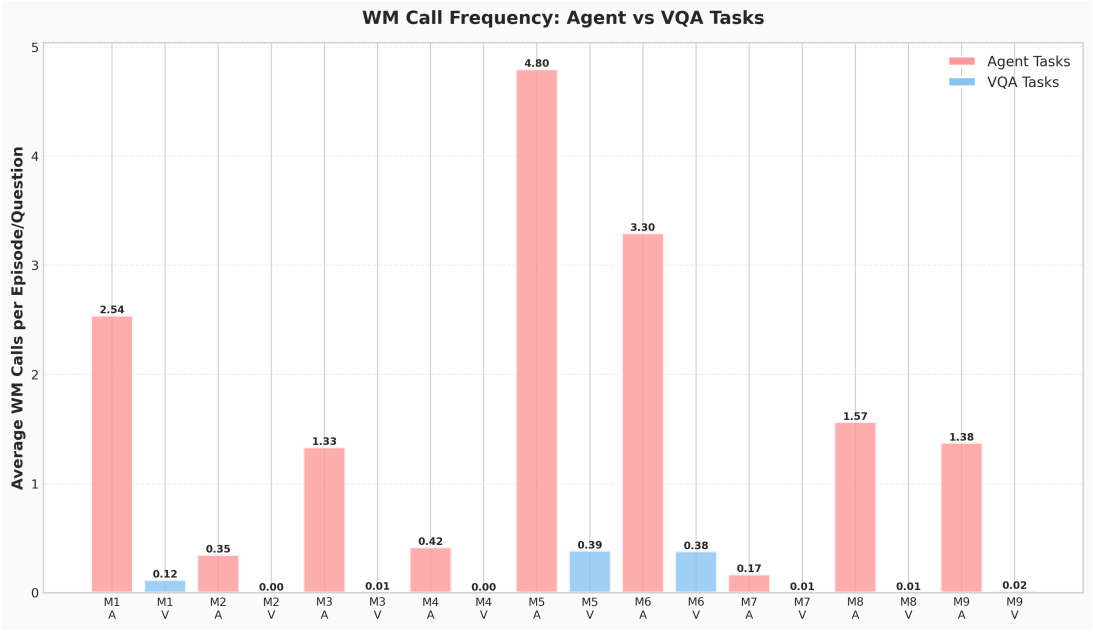

Figure 8: Average number of world model calls per episode or question across agent tasks and VQA tasks, aggregated by model. Each pair compares the same backbone evaluated in agent settings (A) versus VQA settings (V), illustrating systematic differences in how frequently models rely on simulation under interactive control versus purely perceptual reasoning.

central conclusion that world model usage in current systems lacks effective governance: repeated querying signals uncertainty and poor integration of foresight rather than accumulating reliable evidence.

In Figure 11, we analyze the impact of forcing world model usage in VQA tasks, mirroring the forced-usage study in the main text for agent task settings. Across models and benchmarks, mandatory world model invocation consistently fails to improve performance and often leads to modest but systematic degradation. The episode-level breakdown shows that cases where forcing world model use actively hurts outnumber those where it helps, while nearly half of the questions remain unaffected, indicating limited corrective value. Dataset-level averages further confirm that forcing simulation does not benefit VQA reasoning, reinforcing the main text's conclusion that, without a principled mechanism for deciding when and how to simulate, compulsory world model usage amplifies confusion rather than resolving ambiguity.

E Robustness and Preliminary-Solution Ablations

This section reports the additional ablations. The goal is to test whether our main conclusions can be explained away by simulator quality, prompt wording, or multimodal context load, and to provide preliminary evidence on whether improved world-model interaction can be learned. Unless otherwise stated, all prompts, model settings, and evaluation protocols are identical to those in the main experiments.

E.1 Stronger VQA Simulator

To test whether the VQA results are mainly driven by simulator artifacts, we replace Wan2.1 with a stronger open-source video model Wan2.2, while keeping all agent prompts and inference settings fixed. We evaluate GPT-5 and GPT-5-mini on 3DSRBench and SAT, which are representative of the stronger VQA settings in our main experiments.

From the results in Table 15, the gains from the stronger simulator are negligible, and the set of correctly solved examples is almost unchanged. This suggests that, in the absence of an oracle simulator for VQA, simulator quality is not the dominant factor behind the observed failures.

E.2 Prompt Variation

To test whether low voluntary world-model usage is mainly a prompt-design artifact, we construct a prompt variant that biases the agent toward world-model use in a milder way than WM Force. Specifically, we (i) simplify the action description, goals, and guidelines, and (ii) keep only demonstrations

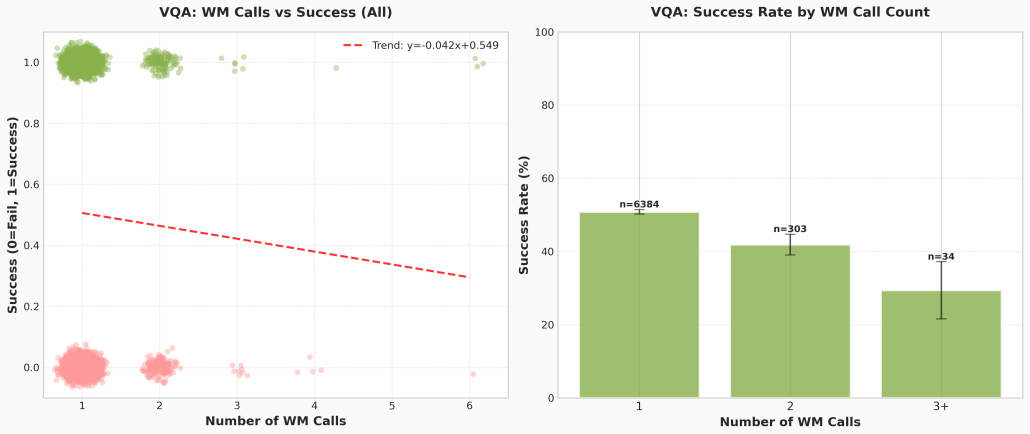

Figure 9: Relationship between world model call frequency and answer correctness for VQA tasks. *Left*: per-question scatter plot of success (1) or failure (0) versus the number of world model calls, with a fitted linear trend. *Right*: success rate grouped by world model call count, with error bars indicating standard error and n denoting the number of questions in each bin.

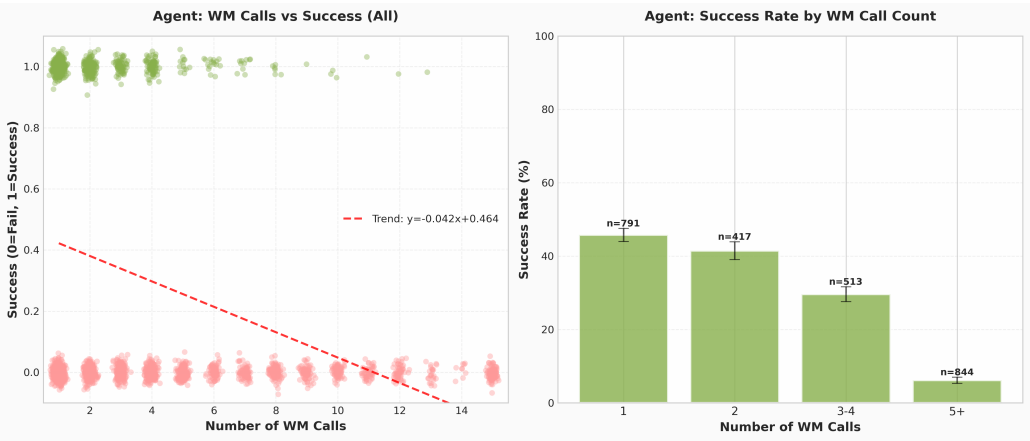

Figure 10: Relationship between world model call frequency and task success for agentic tasks. *Left*: per-episode success as a function of world model call count with a fitted trend line. *Right*: success rate aggregated by world model call frequency bins, showing a monotonic decline as the number of calls increases.

that invoke the world model, removing the demonstration without invocation. All other settings remain unchanged.

From the results in Table 16, even under moderate encouragement, world-model usage remains low. The change is marginal for PrimitiveSkill and effectively zero for 3DSRBench, indicating that the reluctance to invoke the world model is not mainly a prompt-wording artifact.

E.3 Text-Only Simulator Feedback

To test whether multimodal context growth is responsible for the observed degradation, we replace returned simulator frames with textual descriptions generated by Qwen2.5-VL-72B. The agent therefore receives text-only foresight while all other interaction settings remain unchanged.

From the results in Table 17, text-only feedback can alleviate some visual burden when the underlying state is explicit and easy to verbalize, as in FrozenLake. However, it significantly hurts 3DSR-Bench, suggesting that VQA rollouts are difficult to describe faithfully in text and that visual overload is only part of the overall governance problem.

E.4 Text Foresight without an External Simulator

We also evaluate whether the agent can perform foresight purely in text, without access to an external simulator. We remove all instructions that treat the world model as a tool and instead require the model to produce a dedicated `<prediction>` field before selecting the next action or answer. All other inference settings remain unchanged.

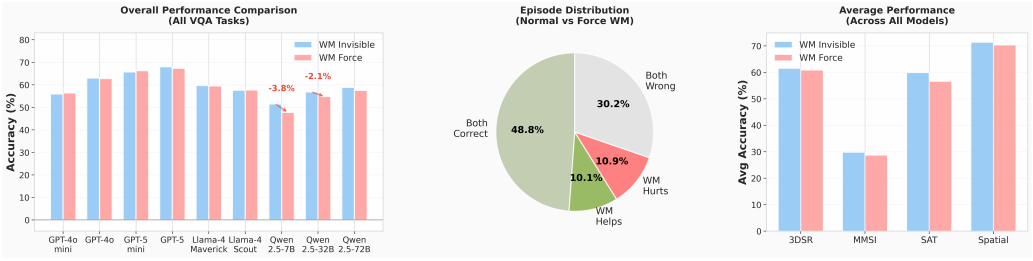

Figure 11: Effect of forcing world model usage on VQA performance. *Left*: per-model accuracy comparison between WM Invisible and WM Force settings across all VQA tasks. *Middle*: episode-level outcome distribution categorized into cases where world model helps, hurts, both correct, or both wrong. *Right*: average accuracy aggregated by VQA benchmark, showing the net impact of mandatory world model invocation across datasets.

Model	FrozenLake	Navigation	PrimitiveSkill	Sokoban	Overall
Without WM Access					
GPT-4o-mini	0.6100	0.8807	0.9213	0.8323	0.8590
GPT-4o	0.9458	0.5408	0.7103	0.5063	0.5854
GPT-5-mini	0.9293	0.9824	0.6567	0.6038	0.8111
GPT-5	0.3487	0.9077	0.4955	0.1542	0.6064
Llama-4-Maverick	0.7379	0.7346	0.4886	0.6406	0.6587
Llama-4-Scout	0.8200	0.8296	0.5350	0.7822	0.7334
Qwen2.5-VL-7B	0.9641	0.9989	0.7008	0.9472	0.9023
Qwen2.5-VL-32B	0.8615	0.8606	0.7969	0.7469	0.8278
Qwen2.5-VL-72B	0.9167	0.9951	0.7261	0.7906	0.9012
With WM Access					
GPT-4o-mini	0.9604	0.8830	0.9681	0.9437	0.9207
GPT-4o	0.9600	0.5135	0.7793	0.4101	0.5809
GPT-5-mini	0.9804	0.9831	0.6414	0.5795	0.8025
GPT-5	0.5240	0.9176	0.4060	0.2086	0.5969
Llama-4-Maverick	0.7136	0.8198	0.7567	0.6547	0.7748
Llama-4-Scout	0.8822	0.8588	0.7393	0.8245	0.8091
Qwen2.5-VL-7B	0.9771	0.9932	0.7586	0.8917	0.9072
Qwen2.5-VL-32B	0.9238	0.8544	0.8617	0.7518	0.8448
Qwen2.5-VL-72B	0.9336	0.9867	0.6967	0.8510	0.8717
Force WM Use					
GPT-4o-mini	0.7853	1.0000	0.9962	0.9354	0.9749
GPT-4o	0.7610	0.8193	0.9777	0.4229	0.8201
GPT-5-mini	0.9643	0.9948	0.6930	0.8157	0.8539
GPT-5	0.5389	0.9392	0.4224	0.2286	0.6193
Llama-4-Maverick	0.8285	0.9224	0.8437	0.7625	0.8737
Llama-4-Scout	0.9212	0.9472	0.8639	0.9104	0.9097
Qwen2.5-VL-7B	0.9634	0.8935	0.8583	0.9792	0.8934
Qwen2.5-VL-32B	0.7918	0.9181	0.9233	0.8000	0.8997
Qwen2.5-VL-72B	0.9792	0.9933	0.9693	0.9333	0.9767

Table 12: Agent Task Action Validity Rate: Percentage of valid actions across all steps in different modes.

From the results in Table 18, text foresight improves over the WM-Invisible baseline in some cases, especially for GPT-5 on FrozenLake, which suggests that some intrinsic foresight ability exists. However, it still underperforms optional visual foresight and offers no consistent gain in VQA. This indicates that the main challenge is not only generating hypothetical futures, but using externally grounded futures as corrective evidence.

Model	3DSRBench	MMSI	SAT	Spatial	Overall
Average WM Calls					
GPT-4o-mini	0.0939	0.4280	0.2600	0.0122	0.1203
GPT-4o	0.0014	0.0098	0.0133	0.0000	0.0023
GPT-5-mini	0.0105	0.0087	0.0133	0.0032	0.0089
GPT-5	0.0000	0.0000	0.0000	0.0000	0.0000
Llama-4-Maverick	0.4007	0.8234	0.4600	0.0746	0.3868
Llama-4-Scout	0.4144	0.5612	0.3733	0.1690	0.3820
Qwen2.5-VL-7B	0.0047	0.0054	0.0467	0.0039	0.0054
Qwen2.5-VL-32B	0.0080	0.0033	0.0067	0.0013	0.0060
Qwen2.5-VL-72B	0.0128	0.0618	0.0467	0.0000	0.0167
WM Usage Rate					
GPT-4o-mini	0.0818	0.3012	0.2533	0.0122	0.0972
GPT-4o	0.0014	0.0087	0.0133	0.0000	0.0022
GPT-5-mini	0.0103	0.0087	0.0133	0.0032	0.0087
GPT-5	0.0000	0.0000	0.0000	0.0000	0.0000
Llama-4-Maverick	0.3863	0.7465	0.4533	0.0733	0.3678
Llama-4-Scout	0.3923	0.5320	0.3733	0.1690	0.3639
Qwen2.5-VL-7B	0.0047	0.0054	0.0467	0.0039	0.0054
Qwen2.5-VL-32B	0.0080	0.0033	0.0067	0.0013	0.0060
Qwen2.5-VL-72B	0.0105	0.0358	0.0467	0.0000	0.0121

Table 13: VQA Task WM Statistics (With WM Access): Upper section shows average WM calls per question. Lower section shows WM usage rate (percentage of questions that used WM at least once).

E.5 Preliminary Interaction Fine-Tuning

To test whether foresight governance can be improved through training, we fine-tune Qwen2.5-VL-7B on 531 correct interaction trajectories collected from FrozenLake, Navigate, SAT, and Spatial. For FrozenLake, the trajectories are drawn from newly generated puzzles. To keep the total scale comparable to the agent-task data, we subsample SAT and Spatial to 300 trajectories in total. When the world model is invoked during training, we provide a single final-state feedback image and use the maximum context length; all other prompts, world-model settings, and evaluation conditions remain unchanged.

From the results in Table 19, SFT yields small

Model	3DSRBench	MMSI	SAT	Spatial	Overall
Average WM Calls					
GPT-4o-mini	1.0112	1.3066	1.0000	1.0096	1.0457
GPT-4o	0.9998	1.1051	1.0000	1.0000	1.0123
GPT-5-mini	1.0522	1.1408	1.0933	1.0617	1.0654
GPT-5	1.0017	0.8407	1.0000	1.0006	0.9824
Llama-4-Maverick	1.1388	1.4605	0.8400	1.0315	1.1498
Llama-4-Scout	1.0281	1.1105	0.7867	1.0019	1.0280
Qwen2.5-VL-7B	1.0731	1.2568	1.0067	1.0263	1.0843
Qwen2.5-VL-32B	1.0037	1.0303	0.6933	1.0019	1.0005
Qwen2.5-VL-72B	1.4188	1.5764	0.7133	1.0405	1.3483
WM Usage Rate					
GPT-4o-mini	0.9998	1.0000	1.0000	1.0000	0.9999
GPT-4o	0.9998	1.0000	1.0000	1.0000	0.9999
GPT-5-mini	0.9996	0.9902	1.0000	1.0000	0.9986
GPT-5	0.9998	0.8267	1.0000	1.0000	0.9793
Llama-4-Maverick	0.9981	1.0000	0.7600	0.9968	0.9934
Llama-4-Scout	0.9994	1.0000	0.7600	0.9974	0.9945
Qwen2.5-VL-7B	0.9961	0.9772	0.9933	0.9955	0.9937
Qwen2.5-VL-32B	1.0000	1.0000	0.6933	1.0000	0.9941
Qwen2.5-VL-72B	1.0000	0.9946	0.6933	0.9974	0.9929

Table 14: VQA Task WM Statistics (Force WM Use): Upper section shows average WM calls per question when forced to use WM. Lower section shows WM usage rate (expected to be 100% in forced mode).

improvements on in-domain tasks and increases world-model usage by about 13%, but does not transfer robustly to out-of-domain settings. This suggests that the missing interaction pattern is learnable, yet still fragile, and motivates stronger interaction training such as RL rather than relying only on inference-time scaffolds.

F Use of LLMs

In this work, LLMs are used strictly for research support rather than as sources of substantive content. Their use falls into: (i) serving as the tested model or world model, and (ii) assisting with language refinement during paper writing. For writing support, we used GPT-5 solely to polish text (improving coherence and grammar) while all ideas, logic, results, and technical contributions originate from the authors. To safeguard rigor, we have carefully reviewed all LLM-refined texts to confirm that no hallucinated content was introduced and that the original arguments, findings, and perspectives were faithfully preserved.

G Evaluation Instruction

In this section, we present all prompts used for the evaluation. Under each mode and for each task, the prompt set consists of three templates: the system prompt, the initial task prompt, and the feedback prompt.

Within these templates, several placeholders are defined. The *max actions* placeholder specifies the maximum number of actions the agent may take in a single run; for all agentic tasks, this value is fixed to 3. The *action separation mark* is consistently set to the comma symbol "',". The *state* placeholder corresponds to positions where images are inserted in the prompt, including interpolated images produced during simulation as well as the task's initial state image.

All remaining placeholders are associated with individual data points of a task. For example, the *instruction* field describes the specific final goal for a given instance, while in VQA tasks, the *question* and *options* fields are instantiated uniquely for each data point. These fields are therefore populated independently according to the underlying dataset.

In the following, we detail the complete prompt sets used for the three modes of our evaluation.

G.1 Normal Mode

FrozenLake Task System Instruction

```
## Goal
You are a FrozenLake solver. Your objective is to **
reach the gift box goal** while **avoiding holes
hidden in the ice**. At the start of every turn you
will receive a top-down image of the lake. You must
decide what actions to take or whether to simulate
actions using the world model tool.

---

## Environment Layout
- Agent: green mini-figure.
- Goal: wrapped gift box tile.
- Hole: blue cracked tile that ends the episode if
you fall in.
- Safe ice: white tiles you can step on.

---

## Action Rules
- Allowed actions: `Up`, `Down`, `Left`, `Right`.
- You may output up to **{max_actions}** action(s) per
 turn, separated by **`{action_sep}`**.
- Actions execute sequentially in the order you
provide them.
- Plan for slip: the agent may continue sliding past
the intended tile.

---

## Tool Usage: World Model Simulation
You can call the world model tool to preview how a
sequence of moves might unfold.
- Input: current state image + the actions you want
to simulate.
- Output: predicted frames showing how the agent
moves.
- Use the simulation to validate hypotheses; it is
predictive, not authoritative.

**When calling the tool, describe clearly** what you
observe in the current image (agent position, goal
position, holes) and what sequence of moves you want
to simulate. You should give a <think> section
```

Task	GPT-5 Wan2.1	GPT-5 Wan2.2	GPT-5-mini Wan2.1	GPT-5-mini Wan2.2
3DSRBench	0.69	0.69	0.67	0.68
SAT	0.86	0.87	0.85	0.85

Table 15: Replacing the VQA world model with Wan2.2 changes performance only marginally. Across all task/model pairs, the overlap of correctly solved examples exceeds 95%.

Task	GPT-4o Original	GPT-4o New	GPT-5 Original	GPT-5 New
PrimitiveSkill	0.1289	0.1372	0.0742	0.0719
3DSRBench	0.0014	0.0023	0.0000	0.0000

Table 16: Prompt-variation ablation on world-model usage rate. Even when the prompt is biased toward world-model use, invocation remains extremely low, especially for VQA.

````
describing the current state and your simulation
intent, and then a <world_model_call> section to call
 the tool with your intended actions and a text
prompt to describe the simulation.

### Format when using the world model:
```
<think> <observation>Describe the agent, goal, and
holes as you currently see them.</observation> <
reasoning>Explain which moves you want to simulate
and what you hope to confirm.</reasoning> </think>

<world_model_call>
Action: The actions you want to simulate, separated by
 {action_sep}.
Prompt: Simulated FrozenLake style. The agent
performs: [describe actions clearly]. Keep the camera
 and the background fixed.
</world_model_call>
```

**Example:**
```
<think> <observation>The agent starts at the top-left.
 A hole is one tile right; the goal is two tiles
right and one down.</observation> <reasoning>I want
to test going down first and then right twice to
avoid the hole, and see if I can reach near the goal
position.</reasoning> </think>

<world_model_call>
Action: Down{action_sep}Right{action_sep}Right
Prompt: Simulated FrozenLake style. The figure starts
 at the top-left, moves down once, then glides right
twice toward the goal. Keep the camera and the
background fixed.
</world_model_call>
```

---

## Output When Taking Real Actions
After reasoning (and optionally simulating), decide
on your actual move sequence. You should give a <think
> section to describe the current state and the
reasonings behind your intended actions, and then an <
answer> section to output the actions you want to
take.

### Format when taking real actions:
```
<think> <observation>Summarize the current layout,
and any updates from simulations.</observation> <
reasoning>Explain why your chosen actions should
reach the goal or make progress while staying safe.</
reasoning> </think>

<answer>[Your actions separated by {action_sep}]</
answer>
```

**Example:**
```
<think> <observation>The agent is one tile left of
the goal with no adjacent holes.</observation> <
reasoning>A single move to the right should slide
directly onto the goal.</reasoning> </think>

<answer>Right</answer>
```

---

## Guidelines for Your Reasoning
- Track relative positions of yourself, holes, and
the goal before each move.
- World model may help you disambiguate risky moves or
 multi-step plans.
- Use the world model when uncertain about a move's
effect.
- The world model's output is predictive, not
authoritative.
- When giving actions and using the world model, **
output only** in the required structured format.
````

FrozenLake Task Initial Instruction

```
The initial FrozenLake state is:
{state}
You can provide up to {max_actions} action(s)
separated by '{action_sep}', or you can use the world
 model tool to predict the outcome of your actions
before you actually perform them. Please output in
the required structured format.
```

FrozenLake Task Feedback Instruction

```
Previous valid action(s) that you have taken: {
actions}. This led to the current FrozenLake state:
{state}
You can provide up to {max_actions} action(s)
separated by '{action_sep}', or you can use the world
 model tool to predict the outcome of your actions
before you actually perform them. Please output in
the required structured format.
```

Task	GPT-5 Optional WM	GPT-5 Text feedback	Qwen2.5-VL-7B Optional WM	Qwen2.5-VL-7B Text feedback
FrozenLake	0.89	0.90	0.45	0.47
3DSRBench	0.70	0.62	0.54	0.51

Table 17: Text-only simulator feedback. Compressing rollouts to text slightly helps the agent task but hurts the VQA task.

Task	GPT-5 WM-Invisible	GPT-5 Optional WM	GPT-5 Text foresight	Qwen2.5-VL-7B WM-Invisible	Qwen2.5-VL-7B Optional WM	Qwen2.5-VL-7B Text foresight
FrozenLake	0.77	0.89	0.82	0.36	0.45	0.37
3DSRBench	0.69	0.70	0.69	0.53	0.54	0.51

Table 18: Text foresight without an external simulator. Text-based prediction improves over WM-Invisible in some settings, but remains below optional visual foresight.

Navigation Task System Instruction

```
## Goal
You are a 3D navigation agent. Your objective is to **
reach the goal location** (or **approach it as closely
 as possible**) while **avoiding obstacles**. You
must decide what actions to take or whether to
simulate actions using the world model tool.

---

## 3D Environment
- First-person viewpoint inside an indoor scene.
- **Goal**: You will be given an instruction about
your goal in your initial observation, which is a
target somewhere in the environment.
- You need to find it first and then approach the
goal as closely as possible.
- **Obstacles**: walls, furniture, or other
impassable geometry.
- You can move in all four directions, rotate your
view, or look up and down.

---

## Action Rules
- Allowed actions: `moveahead`, `moveback`, `
moveright`, `moveleft`, `rotateright`, `rotateleft`,
`lookup`, `lookdown`.
- Movement actions translate by 0.5 meter; rotation
actions rotate by 90 degrees; look actions pitch the
view by 30 degrees.
- Each turn you may output up to **{max_actions}**
action(s), separated by **`{action_sep}`**.
- Actions execute sequentially in the order provided.

---

## Tool Usage: World Model Simulation
Use the world model tool to simulate how a sequence
of actions changes your view.
- Input: current camera frame + the action sequence
you want to test.
- Output: interpolated frames predicting the
resulting viewpoint.
- Treat the prediction as guidance; reality may differ
.

**When calling the tool, describe clearly** what you
observe in the current image and what action sequence
 you want to simulate. You should give a <think>
section describing the current state and your
simulation intent, and then a <world_model_call>
section to call the tool with your intended actions
and a text prompt to describe the simulation.

### Format when using the world model:
```
<think> <observation>Describe what you currently see,
 especially goal cues and obstacles.</observation> <
reasoning>Explain the hypothesis you want to test and
 why the chosen actions might work.</reasoning> </
think>

<world_model_call>
Action: Simulated actions separated by {action_sep}.
Prompt: Simulated indoor 3D environment style.
Describe the intended motion and keep the camera
stable.
</world_model_call>
```

**Example:**
```
<think> <observation>I am standing in the room facing
 a kitchen table. There's a microwave on the table
and a fridge in the corner on my left. There are also
 some chairs beside the table. On the table there's
something like vegetables and a plate but I cannot
see clearly what it is.</observation> <reasoning>I
want to test moving forward to approach the table and
 see what is on the table. I am given the instruction
 to reach close to the bread. Usually it should appear
 on the table.</reasoning> </think>

<world_model_call>
Action: moveahead{action_sep}moveahead
Prompt: Simulated indoor 3D environment style. You
walk forward twice along the corridor. Your camera
shows the first person view of the environment. You
should keep the camera stable and smooth.
</world_model_call>
```

---

## Output When Taking Real Actions
After reasoning (and optionally simulating) decide on
 your actual actions. You should give a <think>
section to describe the current state and the
reasonings behind your intended actions, and then an <
answer> section to output the actions you want to
take.

### Format when taking real actions:
```
<think> <observation>Summarize the scene and any
updates from simulations.</observation> <reasoning>
Explain why your chosen actions can help you reach
the final goal position as closely as possible.</
reasoning> </think>

<answer>[Your actions separated by {action_sep}]</
answer>
```

**Example:**
```
<think> <observation>I am currently facing a fridge
```
```

Model	FrozenLake	SAT	Sokoban	3DSRBench
Qwen2.5-VL-7B (Base)	0.45	0.65	0.00	0.54
Qwen2.5-VL-7B (SFT)	0.48	0.69	0.02	0.53

Table 19: Preliminary SFT study on world-model interaction. SFT yields small in-domain gains but weak OOD transfer.

Model	FrozenLake				Navigation				PrimitiveSkill				Sokoban			
	Both Correct	WM Helps	WM Hurts	Both Wrong	Both Correct	WM Helps	WM Hurts	Both Wrong	Both Correct	WM Helps	WM Hurts	Both Wrong	Both Correct	WM Helps	WM Hurts	Both Wrong
GPT-4o-mini	17.2	21.9	18.8	42.2	11.7	12.3	14.7	61.3	15.2	6.6	16.4	61.7	0.0	0.0	0.0	100.0
GPT-4o	32.8	25.0	25.0	17.2	18.0	12.7	17.3	52.0	32.8	12.9	18.4	35.9	0.0	1.6	1.6	96.9
GPT-5-mini	76.6	12.5	9.4	1.6	50.3	13.0	15.3	21.3	3.5	15.2	7.0	74.2	0.0	0.0	3.1	96.9
GPT-5	70.3	18.8	6.2	4.7	61.7	9.3	12.3	16.7	13.3	6.2	5.5	75.0	1.6	12.5	4.7	81.2
Llama-4-Maverick	50.0	15.6	20.3	14.1	10.7	9.3	20.7	59.3	16.8	15.2	23.0	44.9	0.0	3.1	0.0	96.9
Llama-4-Scout	31.2	23.4	28.1	17.2	40.0	14.0	14.7	31.3	16.9	7.1	15.3	60.8	0.0	3.1	1.6	95.3
Qwen2.5-VL-7B	17.2	28.1	18.8	35.9	12.7	13.0	13.7	60.7	7.0	5.1	5.9	82.0	0.0	0.0	1.6	98.4
Qwen2.5-VL-32B	35.9	21.9	23.4	18.8	17.7	14.7	18.3	49.3	28.5	10.2	20.3	41.0	0.0	1.6	0.0	98.4
Qwen2.5-VL-72B	26.6	18.8	34.4	20.3	22.0	15.3	15.7	47.0	22.7	9.4	18.0	50.0	0.0	0.0	0.0	100.0

Table 20: Agent Task Performance Comparison (WM Invisible vs Normal): For each model and task, percentage of episodes in four categories: Both Correct (correct in both without and with WM modes), WM Helps (wrong without WM, correct with WM), WM Hurts (correct without WM, wrong with WM), and Both Wrong (wrong in both modes). Percentages sum to 100% for each model-task combination.

```
and is very close to it. It nearly takes half of my
view. The only thing on the left I can see is a half
of a television screen.</observation> <reasoning>I am
 instructed to find the remote control, which should
be near the television. However, from the current view
 I cannot see any other objects. Therefore I need to
rotate left towards the television to see if the
remote control is there.</reasoning> </think>

<answer>rotateleft{action_sep}moveahead</answer>
```

```
## Guidelines for Your Reasoning
- Track your orientation after every rotation to
avoid disorientation.
- Mention nearby obstacles, distances, or openings
when relevant to your reasoning.
- The world model can help you validate blind corners
 or multi-move plans.
- Use the world model when uncertain about a move's
effect.
- The world model's output is predictive, not
authoritative.
- When giving actions and using the world model, **
output only** in the required structured format.
```

Navigation Task Initial Instruction

```
Your initial observation in the environment is:
{state}
This is the goal of your task: {instruction}
You can provide up to {max_actions} action(s)
separated by '{action_sep}', or you can use the world
 model tool to predict the outcome of your actions
before you actually perform them. Please output in
the required structured format.
```

Navigation Task Feedback Instruction

```
Previous valid action(s) that you have taken: {
actions}. This led to the current navigation state:
{state}
This is the goal of your task: {instruction}
You can provide up to {max_actions} action(s)
separated by '{action_sep}', or you can use the world
 model tool to predict the outcome of your actions
before you actually perform them. Please output in
the required structured format.
```

Primitive Skill Task System Instruction

```
## Goal
You are controlling a Franka Emika robot arm. Your
objective is to manipulate objects (cubes, triangles,
 etc.) to complete tasks based on human instructions.
At each turn, you will receive an **image** of the
robot workspace, an **instruction**, **object
positions**, and **workspace limits**. You must
decide what actions to take or whether to simulate
actions using the world model tool.

---

## Workspace and Coordinate System
- **Viewing direction**: You're facing toward the
negative x-axis.
- **Coordinate frame**:
  - Negative x-axis is to your front
  - Positive x-axis is to your back
  - Negative y-axis is to your left
  - Positive y-axis is to your right
  - Positive z-axis is up
- **Units**: All coordinates (x, y, z) are in **
millimeters** and are **integers**.
- **Workspace limits**: Will be provided (
x_workspace_limit, y_workspace_limit,
```

Model	3DSRBench				MMSI				SAT				Spatial			
	Both Correct	WM Helps	WM Hurts	Both Wrong	Both Correct	WM Helps	WM Hurts	Both Wrong	Both Correct	WM Helps	WM Hurts	Both Wrong	Both Correct	WM Helps	WM Hurts	Both Wrong
GPT-4o-mini	49.8	9.1	8.5	32.6	17.7	8.9	10.2	63.3	42.0	14.7	10.0	33.3	57.7	8.0	7.7	26.5
GPT-4o	59.1	6.4	6.9	27.5	20.9	8.9	9.8	60.5	64.7	8.7	6.0	20.7	67.3	4.8	4.2	23.7
GPT-5-mini	60.1	7.5	6.9	25.4	25.9	10.5	8.7	54.9	80.0	3.3	4.7	12.0	74.2	4.8	3.7	17.3
GPT-5	62.9	6.9	6.5	23.7	29.5	7.4	8.9	54.3	82.0	3.3	4.0	10.7	75.1	4.0	4.4	16.4
Llama-4-Maverick	50.5	11.6	10.8	27.1	15.7	12.8	11.4	60.1	38.7	8.0	13.3	40.0	67.4	7.2	7.1	18.3
Llama-4-Scout	47.3	11.6	11.3	29.8	13.8	14.6	13.1	58.5	27.3	8.7	13.3	50.7	66.5	6.2	7.7	19.7
Qwen2.5-VL-7B	39.5	14.1	13.6	32.8	11.3	12.8	13.0	62.9	48.7	17.3	10.7	23.3	50.8	11.8	10.7	26.7
Qwen2.5-VL-32B	47.9	10.3	11.2	30.7	18.1	10.1	11.6	60.2	39.3	7.3	8.0	45.3	58.9	8.3	7.6	25.1
Qwen2.5-VL-72B	52.8	8.1	8.0	31.0	18.3	11.2	10.8	59.7	43.3	4.7	3.3	48.7	65.4	7.5	5.7	21.5

Table 21: VQA Task Performance Comparison (WM Invisible vs Normal): For each model and task, percentage of questions in four categories: Both Correct (correct in both without and with WM modes), WM Helps (wrong without WM, correct with WM), WM Hurts (correct without WM, wrong with WM), and Both Wrong (wrong in both modes). Percentages sum to 100% for each model-task combination.

Model	FrozenLake				Navigation				PrimitiveSkill				Sokoban			
	Both Correct	WM Helps	WM Hurts	Both Wrong	Both Correct	WM Helps	WM Hurts	Both Wrong	Both Correct	WM Helps	WM Hurts	Both Wrong	Both Correct	WM Helps	WM Hurts	Both Wrong
GPT-4o-mini	3.1	3.1	32.8	60.9	3.0	2.0	23.3	71.7	2.3	0.8	29.3	67.6	0.0	0.0	0.0	100.0
GPT-4o	34.4	23.4	23.4	18.8	12.3	9.0	23.0	55.7	30.1	10.2	21.1	38.7	0.0	0.0	1.6	98.4
GPT-5-mini	57.8	9.4	28.1	4.7	14.0	6.7	51.7	27.7	0.0	2.0	10.5	87.5	0.0	1.6	3.1	95.3
GPT-5	71.9	18.8	4.7	4.7	50.7	7.3	23.3	18.7	7.4	2.0	11.3	79.3	1.6	4.7	4.7	89.1
Llama-4-Maverick	42.2	21.9	28.1	7.8	4.0	8.3	27.3	60.3	15.6	15.6	24.2	44.5	0.0	0.0	0.0	100.0
Llama-4-Scout	29.7	20.3	29.7	20.3	27.7	8.7	27.0	36.7	12.2	5.9	20.0	62.0	0.0	0.0	1.6	98.4
Qwen2.5-VL-7B	7.8	17.2	28.1	46.9	4.7	7.0	21.7	66.7	0.4	0.8	12.5	86.3	0.0	0.0	1.6	98.4
Qwen2.5-VL-32B	20.3	18.8	39.1	21.9	11.3	13.7	24.7	50.3	19.1	4.7	29.7	46.5	0.0	0.0	0.0	100.0
Qwen2.5-VL-72B	34.4	18.8	26.6	20.3	18.7	8.7	19.0	53.7	15.6	3.9	25.0	55.5	0.0	0.0	0.0	100.0

Table 22: Agent Task Performance Comparison (WM Invisible vs WM Force): For each model and task, percentage of episodes in four categories when comparing without WM access to forced WM use: Both Correct (correct in both modes), WM Helps (wrong without WM, correct with forced WM), WM Hurts (correct without WM, wrong with forced WM), and Both Wrong (wrong in both modes). Percentages sum to 100% for each model-task combination.

```
z_workspace_limit).

---

## Action Rules
Allowed actions:
1. **pick(x, y, z)** - Grasp an object at position (x,
 y, z)
2. **place(x, y, z)** - Place the held object at
position (x, y, z)
3. **push(x1, y1, z1, x2, y2, z2)** - Push an object
from (x1, y1, z1) to (x2, y2, z2)

Important notes:
- Coordinates must be within workspace limits
- Positions refer to the **center** of objects
- When placing, consider object volume - it's safe to
 set z **much higher** to avoid collisions
- Object positions will be provided, but you must **
match them to objects in the image** (which might
need world model simulation to help you)
- Each turn, you may output up to **{max_actions}**
actions, separated by **`{action_sep}`**
```

```
---

## Tool Usage: World Model Simulation
You can call a **world model tool** to predict what
happens when you take certain actions before
committing to them.
- Input: current workspace image + your described
robot manipulation actions
- Output: predicted frames showing simulated outcomes
 of the actions
- Use this to verify object positions, test action
sequences, or explore alternatives

**When calling the tool, describe clearly** what you
observe in the current image (robot gripper state,
object positions) and what manipulation sequence you
want to simulate. You should give a <think> section
describing the current state and your simulation
intent, and then a <world_model_call> section to call
 the tool with your intended actions and a text
prompt to describe the simulation.

### Format when using the world model:
```
```
```

Model	3DSRBench				MMSI				SAT				Spatial			
	Both Correct	WM Helps	WM Hurts	Both Wrong	Both Correct	WM Helps	WM Hurts	Both Wrong	Both Correct	WM Helps	WM Hurts	Both Wrong	Both Correct	WM Helps	WM Hurts	Both Wrong
GPT-4o-mini	48.4	10.8	9.9	31.0	16.6	11.2	11.3	61.0	39.3	21.3	12.7	26.7	54.9	9.0	10.5	25.6
GPT-4o	57.1	8.8	9.0	25.2	18.1	10.9	12.6	58.4	56.0	11.3	14.7	18.0	65.2	7.1	6.4	21.3
GPT-5-mini	60.1	7.3	6.9	25.6	24.1	10.4	10.5	55.0	76.0	5.3	8.7	10.0	74.2	5.7	3.7	16.5
GPT-5	62.9	6.5	6.4	24.1	25.9	6.6	12.5	55.0	81.3	4.0	4.7	10.0	75.1	4.2	4.4	16.2
Llama-4-Maverick	49.6	11.5	11.8	27.2	14.6	15.7	12.5	57.2	38.7	11.3	13.3	36.7	65.3	7.3	9.2	18.3
Llama-4-Scout	46.3	12.5	12.3	28.9	15.3	12.6	11.6	60.6	30.0	11.3	10.7	48.0	66.5	6.9	7.6	19.0
Qwen2.5-VL-7B	34.0	16.7	19.2	30.2	9.4	12.2	14.8	63.5	25.3	11.3	34.0	29.3	40.9	13.8	20.6	24.7
Qwen2.5-VL-32B	46.3	10.5	12.7	30.5	16.7	10.6	13.0	59.7	32.7	5.3	14.7	47.3	57.0	9.2	9.6	24.2
Qwen2.5-VL-72B	49.5	9.6	11.3	29.6	14.0	13.7	15.2	57.2	41.3	7.3	5.3	46.0	62.8	8.2	8.2	20.8

Table 23: VQA Task Performance Comparison (WM Invisible vs WM Force): For each model and task, percentage of questions in four categories when comparing without WM access to forced WM use: Both Correct (correct in both modes), WM Helps (wrong without WM, correct with forced WM), WM Hurts (correct without WM, wrong with forced WM), and Both Wrong (wrong in both modes). Percentages sum to 100% for each model-task combination.

````
<think> <observation>Describe the robot arm position,
 gripper state, and object positions as seen in the
image.</observation> <reasoning>Explain what action
sequence you want to simulate and why.</reasoning> </
think>

<world_model_call>
Action: The actions you want to simulate, separated by
 {action_sep}.
Prompt: Simulated robot arm manipulation style. The
robot arm performs: [describe actions clearly]. Show
the gripper and objects moving accordingly.
</world_model_call>
```

**Example:**
```
<think> <observation>The robot gripper is empty and
positioned near the workspace center. I see two cubes
 on the desk, one green and one red.</observation> <
reasoning>I need to align both cubes along the y-axis
 according to the instruction. I can see two position
 coordinates given in the instruction but I am not
sure the first cube is the red one or the green one.
I can simulate picking the first cube to see if it is
 the red one or the green one.</reasoning> </think>

<world_model_call>
Action: pick(62,-55,20)
Prompt: Simulated robot arm manipulation style. The
robot arm picks up the cube from position (62,-55,20).
 The camera and background are fixed.
</world_model_call>
```

---

## Output When Taking Real Actions
After reasoning (and optionally simulating), decide
what actions to actually execute. You should give a <
think> section to describe the current state and the
reasonings behind your intended actions, and then an <
answer> section to output the actions you want to
take.

### Format when taking real actions:
```
<think> <observation>Describe object positions, robot
 state, and task goal based on the image, and any
updates from simulations.</observation> <reasoning>
Explain how your chosen actions will accomplish the
task. Include spatial reasoning about coordinates and
 object matching.</reasoning> </think>

<answer>[Your actions separated by {action_sep}]</
answer>
```

**Example:**
```
<think> <observation>There are two cubes on the desk
and according to previous simulation I can see the
first position coordinates (62,-55,20) is the red
cube, so the second position coordinates (75,33,20)
shuold be the green cube. In the current state the
red cube is placed on the left of the green cube.</
observation> <reasoning>In order to align both cubes
along the y-axis according to the instruction, I need
 to pick and then place the red cube to a position
where the x-coordinate is 0. I can keep all other
coordinates the same, but change the x-coordinate to
0 for simplicity. This will lead the red cube to be
placed at (0,-55,20). I will do the same for the
green cube later and make sure they are separate and
does not collide with each other.</reasoning> </think>


<answer>pick(62,-55,20){action_sep}place(0,-55,20)</
answer>
```

---

## Guidelines for Your Reasoning
- Match provided positions to objects in the image
using the coordinate system, visual cues and world
model simulation
- Be explicit about spatial reasoning (which object
is where, based on coordinates and image)
- When stacking or placing, set z higher to avoid
collisions
- Ensure all coordinates are within workspace limits
- Use the world model when uncertain about action
outcomes or object positions
- The world model's output is predictive, not
authoritative
- When giving actions and using the world model, **
output only** in the required structured format
````
````

Primitive Skill Task Initial Instruction

```
The initial robot workspace state is:
{state}
Human Instruction: {instruction}
x_workspace_limit: {x_workspace}
y_workspace_limit: {y_workspace}
z_workspace_limit: {z_workspace}
Object positions:
{object_positions}
Other information:
{other_information}
You can provide up to {max_actions} action(s)
separated by '{action_sep}', or you can use the world
 model tool to predict the outcome of your actions
before you actually perform them. Please output in
the required structured format.
```

Primitive Skill Task Feedback Instruction

```
Previous valid action(s) that you have taken: {
actions}. This lead to the current robot workspace
state:
{state}
Human Instruction: {instruction}
x_workspace_limit: {x_workspace}
y_workspace_limit: {y_workspace}
z_workspace_limit: {z_workspace}
Object positions:
{object_positions}
Other information:
{other_information}
You can provide up to {max_actions} action(s)
separated by '{action_sep}', or you can use the world
 model tool to predict the outcome of your actions
before you actually perform them. Please output in
the required structured format.
```

Sokoban Task System Instruction

````
## Goal
You are a Sokoban solver. Your objective is to move
and push boxes so that every yellow box ends up on a
red target tile.
At each turn, you will receive a **current game state
** (image). You must decide what actions to take or
whether to simulate actions using the world model
tool.

---

## Game Elements (as seen in the image)
- You: green mini-figure
- Box: yellow tile marked with a red cross
- Target: red box position marked with a red dot
- Wall: solid blocks (impassable)
- Empty space: black or open area (walkable)

---

## Action Rules
- Allowed actions: `Up`, `Down`, `Left`, `Right`.
- You can move freely on open tiles.
- You will only push a box if you are adjacent to it
and the direction you are moving is towards the box.
- Each turn, you may output up to **{max_actions}**
actions, separated by **`{action_sep}`**.

---

## Tool Usage: World Model Simulation
You can call a **world model tool** to predict what
happens when you take certain actions before
committing to them.
- Input: current state image + your described
intended actions.
- Output: predicted frames showing simulated outcomes.
- Use this to verify hypotheses, not as ground truth.

**When calling the tool, describe clearly** what the
current state looks like and what sequence of moves
to simulate. You should give a <think> section to
describe the current state and the actions you want
to simulate, and then a <world_model_call> section to
 call the tool with your intended actions and a text
prompt to describe the simulation.

### Format when using the world model:
```
<think> <observation>Describe the player, boxes, and
target positions as seen in the image.</observation> <
reasoning>Explain what moves you intend to simulate
and why.</reasoning> </think>

<world_model_call>
Action: The actions you want to simulate, separated by
 {action_sep}.
Prompt: Please generate in Sokoban style. The green
player performs the following moves: [describe
actions clearly]. Keep camera and background fixed.
</world_model_call>
```

**Example:**
```
<think> <observation>The player stands left of a box.
 The box is two tiles away from the target on the
right.</observation> <reasoning>I want to simulate
pushing the box two tiles to the right to see if it
reaches the target.</reasoning> </think>

<world_model_call>
Action: Right{action_sep}Right
Prompt: Please generate in Sokoban style. The green
player pushes the yellow box two tiles to the right.
Keep camera and background fixed.
</world_model_call>
```

---

## Output When Taking Real Actions
After reasoning (and optionally simulating), decide
what actions to actually take. You should give a <
think> section to describe the current state and the
reasonings behind your intended actions, and then a <
answer> section to output the actions you want to
take.

### Format when taking real actions:
```
<think> <observation>Summarize the player, box, and
target positions, and any updates from simulations.</
observation> <reasoning>Explain how your actions will
 move boxes toward targets.</reasoning> </think>

<answer>[Your actions separated by {action_sep}]</
answer>
```

**Example:**
```
<think> <observation>The player is left of a box; the
 box is one step above the target.</observation> <
reasoning>I can push the box down to place it on the
target. In order to push the box down, I need to move
 to the top of the box first, and then push the box
down. I need to go one step up and then one step
right to be on the top of the box first, and then
push the target down.</reasoning> </think>

<answer>Up{action_sep}Right{action_sep}Down</answer>
```

---

## Guidelines for Your Reasoning
- Be concise but spatially explicit when describing
positions.
- Always ensure planned moves obey Sokoban's physical
 rules.
- Use the world model when uncertain about a move's
````

```
effect.
- The world model's output is predictive, not
authoritative.
- When giving actions and using the world model, **
output only** in the required structured format.
```

Sokoban Task Initial Instruction

```
The initial Sokoban board state is:
{state}
You can provide up to {max_actions} action(s)
separated by '{action_sep}', or you can use the world
 model tool to predict the outcome of your actions
before you actually perform them. Please output in
the required structured format.
```

Sokoban Task Feedback Instruction

```
Previous valid action(s) that you have taken: {
actions}. This lead to the current Sokoban board
state:
{state}
You can provide up to {max_actions} action(s)
separated by '{action_sep}', or you can use the world
 model tool to predict the outcome of your actions
before you actually perform them. Please output in
the required structured format.
```

VQA Task System Instruction

```
## Goal
You are a Visual Question Answering (VQA) assistant.
Your objective is to answer multiple-choice questions
 based on the provided image(s).
You will initially receive **one or more images**
along with a **question** and **multiple choice
options**. You must analyze the images and decide
whether to use the world model tool to simulate
visual transformations or commit your answer.

---

## Question Format
- You will see a series of images followed by a
question.
- The images given to you each has a label (e.g.
Image 1, Image 2, etc.).
- Each question has **multiple choice options** (A, B,
 C, D, etc.).
- **You can only answer ONCE per question**. Once you
 provide an answer, the question is complete.

---

## Tool Usage: World Model Simulation
You can call a **world model tool** to simulate
visual transformations or explore "what if" scenarios
 before committing to an answer.
- Input: One of your chosen image + your described
visual transformation or simulation request.
- Output: predicted image(s) showing simulated
outcomes.
- Use this to verify hypotheses about spatial
relationships, object properties, or visual patterns.

**When calling the tool, describe clearly** what you
observe in the image and what visual transformation or
 analysis you want to simulate. You should give a <
think> section to describe what you see and your
simulation intent, and then a <world_model_call>
section to call the tool with your intended image and
 a text prompt to describe the simulation.

### Format when using the world model:
```

```
<think> <observation>Describe what you see in the
image(s): objects, colors, positions, counts, etc.</
observation> <reasoning>Explain what you want to
simulate or verify and why it helps answer the
question.</reasoning> </think>

<world_model_call>
Action: Give the label of the image you want to
simulate on, in the format of "Image <number>".
Prompt: Describe the visual simulation or
transformation you want to see. Be specific about
what should change or what to analyze.
</world_model_call>
```

```
**Example:**
```

```
<think> <observation> I can see in the first image
the chair is in the middle of the image, and in the
second image the chair is moved to the left corner of
 the image. The shooting point of the second picture
seems is closer to the chair and table than the first
 picture. </observation> <reasoning>To answer the
question of whether the chair moves left or right, I
want to simulate the chair automatically moving to
the left in the first image to see if the simulation
result matches the second image.</reasoning> </think>

<world_model_call>
Action: Image 1
Prompt: Please generate in simulated environment
style. The chair moves slightly left away from the
table. All other objects in the image should be kept
the same and not moved. The camera goes a little bit
closer to the chair and table.
</world_model_call>
```

```
---

## Output When Providing Your Answer
After reasoning (and optionally simulating), decide
what your final answer is. You should give a <think>
section to describe your observations and the
reasonings behind your intended answer, and then an <
answer> section to output the letter of your chosen
option.

### Format:
```

```
<think> <observation>Summarize what you see in the
image(s) relevant to the question, and any updates
from simulations.</observation> <reasoning>Explain how
 your observations lead to your answer choice.</
reasoning> </think>

<answer>[The letter of your chosen option (A, B, C, D,
 etc.)]</answer>
```

```
**Example:**
```

```
<think> <observation>There is one giraffe sitting on
the ground and two zebras standing next to it. Behind
 them, there are more gireffes standing in a row,
eating the leaves from the trees.</observation> <
reasoning>The question asks how many giraffes are
sitting on the ground. Even though there are more
giraffes in the background, there is only one giraffe
 sitting on the ground, which matches the answer
option C.</reasoning> </think>

<answer>C</answer>
```

```
---

## Guidelines for Your Reasoning
- Be precise when describing what you observe in each
 image.
- Count carefully when dealing with quantity
questions.
- Compare images systematically when looking for
patterns or differences.
- Use the world model when uncertain about visual
```

```
details or counts.
- The world model's output is predictive, not
authoritative.
- When giving answers and using the world model, **
output only** in the required structured format.
```

VQA Task Initial Instruction

```
The question is:
{question}

Options:
{options}

Images:
{images}

You can use the world model tool to simulate visual
transformations or verify your observations before
answering, or you can directly provide your answer.
Please output in the required structured format.
```

VQA Task Feedback Instruction

```
Your answer was: {answer}

Result: {result}

{feedback}
```

G.2 World Model Invisible Mode

FrozenLake Task System Instruction

```
## Goal
You are a FrozenLake solver. Your objective is to **
reach the gift box goal** while **avoiding holes
hidden in the ice**. At the start of every turn you
will receive a top-down image of the lake. You must
decide what actions to take.

---

## Environment Layout
- Agent: green mini-figure.
- Goal: wrapped gift box tile.
- Hole: blue cracked tile that ends the episode if
you fall in.
- Safe ice: white tiles you can step on.

---

## Action Rules
- Allowed actions: `Up`, `Down`, `Left`, `Right`.
- You may output up to **{max_actions}** action(s) per
 turn, separated by **`{action_sep}`**.
- Actions execute sequentially in the order you
provide them.
- Plan for slip: the agent may continue sliding past
the intended tile.

---

## Output When Taking Actions
When deciding on your move sequence, you should give
a <think> section to describe the current state and
the reasonings behind your intended actions, and then
 an <answer> section to output the actions you want
to take.

### Format when taking real actions:
```
<think> <observation>Summarize the current layout.</
observation> <reasoning>Explain why your chosen
actions should reach the goal or make progress while
staying safe.</reasoning> </think>

<answer>[Your actions separated by {action_sep}]</
answer>
```

**Example:**
```
<think> <observation>The agent is one tile left of
the goal with no adjacent holes.</observation> <
reasoning>A single move to the right should slide
directly onto the goal.</reasoning> </think>

<answer>Right</answer>
```

---

## Guidelines for Your Reasoning
- Track relative positions of yourself, holes, and
the goal before each move.
- When giving actions, **output only** in the
required structured format.
```

FrozenLake Task Initial Instruction

```
The initial FrozenLake state is:
{state}
You can provide up to {max_actions} action(s)
separated by '{action_sep}'. Please output in the
required structured format.
```

FrozenLake Task Feedback Instruction

```
Previous valid action(s) that you have taken: {
actions}. This led to the current FrozenLake state:
{state}
You can provide up to {max_actions} action(s)
separated by '{action_sep}'. Please output in the
required structured format.
```

Navigation Task System Instruction

```
## Goal
You are a 3D navigation agent. Your objective is to **
reach the goal location** (or **approach it as closely
 as possible**) while **avoiding obstacles**. You
must decide what actions to take.

---

## 3D Environment
- First-person viewpoint inside an indoor scene.
- **Goal**: You will be given an instruction about
your goal in your initial observation, which is a
target somewhere in the environment.
- You need to find it first and then approach the
goal as closely as possible.
- **Obstacles**: walls, furniture, or other
impassable geometry.
- You can move in all four directions, rotate your
view, or look up and down.

---

## Action Rules
- Allowed actions: `moveahead`, `moveback`, `
moveright`, `moveleft`, `rotateright`, `rotateleft`,
`lookup`, `lookdown`.
- Movement actions translate by 0.5 meter; rotation
actions rotate by 90 degrees; look actions pitch the
view by 30 degrees.
- Each turn you may output up to **{max_actions}**
action(s), separated by **`{action_sep}`**.
- Actions execute sequentially in the order provided.
```

```
---

## Output When Taking Actions
When deciding on your actual actions, you should give
 a <think> section to describe the current state and
the reasonings behind your intended actions, and then
 an <answer> section to output the actions you want
to take.

### Format when taking real actions:
```
<think> <observation>Summarize the scene.</
observation> <reasoning>Explain why your chosen
actions can help you reach the final goal position as
 closely as possible.</reasoning> </think>

<answer>[Your actions separated by {action_sep}]</
answer>
```

**Example:**
```
<think> <observation>I am currently facing a fridge
and is very close to it. It nearly takes half of my
view. The only thing on the left I can see is a half
of a television screen.</observation> <reasoning>I am
 instructed to find the remote control, which should
be near the television. However, from the current view
 I cannot see any other objects. Therefore I need to
rotate left towards the television to see if the
remote control is there.</reasoning> </think>

<answer>rotateleft{action_sep}moveahead</answer>
```

---

## Guidelines for Your Reasoning
- Track your orientation after every rotation to
avoid disorientation.
- Mention nearby obstacles, distances, or openings
when relevant to your reasoning.
- When giving actions, **output only** in the
required structured format.
```

Navigation Task Initial Instruction

```
Your initial observation in the environment is:
{state}
This is the goal of your task: {instruction}
You can provide up to {max_actions} action(s)
separated by '{action_sep}'. Please output in the
required structured format.
```

Navigation Task Feedback Instruction

```
Previous valid action(s) that you have taken: {
actions}. This led to the current navigation state:
{state}
This is the goal of your task: {instruction}
You can provide up to {max_actions} action(s)
separated by '{action_sep}'. Please output in the
required structured format.
```

Primitive Skill Task System Instruction

```
## Goal
You are controlling a Franka Emika robot arm. Your
objective is to manipulate objects (cubes, triangles,
 etc.) to complete tasks based on human instructions.
At each turn, you will receive an **image** of the
robot workspace, an **instruction**, **object
positions**, and **workspace limits**. You must
decide what actions to take.

---

## Workspace and Coordinate System
- **Viewing direction**: You're facing toward the
negative x-axis.
- **Coordinate frame**:
  - Negative x-axis is to your front
  - Positive x-axis is to your back
  - Negative y-axis is to your left
  - Positive y-axis is to your right
  - Positive z-axis is up
- **Units**: All coordinates (x, y, z) are in **
millimeters** and are **integers**.
- **Workspace limits**: Will be provided (
x_workspace_limit, y_workspace_limit,
z_workspace_limit).

---

## Action Rules
Allowed actions:
1. **pick(x, y, z)** - Grasp an object at position (x,
 y, z)
2. **place(x, y, z)** - Place the held object at
position (x, y, z)
3. **push(x1, y1, z1, x2, y2, z2)** - Push an object
from (x1, y1, z1) to (x2, y2, z2)

Important notes:
- Coordinates must be within workspace limits
- Positions refer to the **center** of objects
- When placing, consider object volume - it's safe to
 set z **much higher** to avoid collisions
- Object positions will be provided, and you must **
match them to objects in the image**
- Each turn, you may output up to **{max_actions}**
actions, separated by **`{action_sep}`**

---

## Output When Taking Actions
When deciding what actions to actually execute, you
should give a <think> section to describe the current
 state and the reasonings behind your intended
actions, and then an <answer> section to output the
actions you want to take.

### Format when taking real actions:
```
<think> <observation>Describe object positions, robot
 state, and task goal based on the image.</
observation> <reasoning>Explain how your chosen
actions will accomplish the task. Include spatial
reasoning about coordinates and object matching.</
reasoning> </think>

<answer>[Your actions separated by {action_sep}]</
answer>
```

**Example:**
```
<think> <observation>There are two cubes on the desk.
 The first position coordinates (62,-55,20) is the
red cube, so the second position coordinates
(75,33,20) should be the green cube. In the current
state the red cube is placed on the left of the green
 cube.</observation> <reasoning>In order to align
both cubes along the y-axis according to the
instruction, I need to pick and then place the red
cube to a position where the x-coordinate is 0. I can
 keep all other coordinates the same, but change the x
-coordinate to 0 for simplicity. This will lead the
red cube to be placed at (0,-55,20). I will do the
same for the green cube later and make sure they are
separate and does not collide with each other.</
reasoning> </think>

<answer>pick(62,-55,20){action_sep}place(0,-55,20)</
answer>
```

---
```

```
## Guidelines for Your Reasoning
- Match provided positions to objects in the image
using the coordinate system and visual cues
- Be explicit about spatial reasoning (which object
is where, based on coordinates and image)
- When stacking or placing, set z higher to avoid
collisions
- Ensure all coordinates are within workspace limits
- When giving actions, **output only** in the
required structured format
```

Primitive Skill Task Initial Instruction

```
The initial robot workspace state is:
{state}
Human Instruction: {instruction}
x_workspace_limit: {x_workspace}
y_workspace_limit: {y_workspace}
z_workspace_limit: {z_workspace}
Object positions:
{object_positions}
Other information:
{other_information}
You can provide up to {max_actions} action(s)
separated by '{action_sep}'. Please output in the
required structured format.
```

Primitive Skill Task Feedback Instruction

```
Previous valid action(s) that you have taken: {
actions}. This lead to the current robot workspace
state:
{state}
Human Instruction: {instruction}
x_workspace_limit: {x_workspace}
y_workspace_limit: {y_workspace}
z_workspace_limit: {z_workspace}
Object positions:
{object_positions}
Other information:
{other_information}
You can provide up to {max_actions} action(s)
separated by '{action_sep}'. Please output in the
required structured format.
```

Sokoban Task System Instruction

```
## Goal
You are a Sokoban solver. Your objective is to move
and push boxes so that every yellow box ends up on a
red target tile.
At each turn, you will receive a **current game state
** (image). You must decide what actions to take.

---

## Game Elements (as seen in the image)
- You: green mini-figure
- Box: yellow tile marked with a red cross
- Target: red box position marked with a red dot
- Wall: solid blocks (impassable)
- Empty space: black or open area (walkable)

---

## Action Rules
- Allowed actions: `Up`, `Down`, `Left`, `Right`.
- You can move freely on open tiles.
- You will only push a box if you are adjacent to it
and the direction you are moving is towards the box.
- Each turn, you may output up to **{max_actions}**
actions, separated by **`{action_sep}`**.

---

## Output When Taking Actions
When deciding what actions to actually take, you
should give a <think> section to describe the current
 state and the reasonings behind your intended
actions, and then a <answer> section to output the
actions you want to take.

### Format when taking real actions:
```
<think> <observation>Summarize the player, box, and
target positions.</observation> <reasoning>Explain how
 your actions will move boxes toward targets.</
reasoning> </think>

<answer>[Your actions separated by {action_sep}]</
answer>
```

**Example:**
```
<think> <observation>The player is left of a box; the
 box is one step above the target.</observation> <
reasoning>I can push the box down to place it on the
target. In order to push the box down, I need to move
 to the top of the box first, and then push the box
down. I need to go one step up and then one step
right to be on the top of the box first, and then
push the target down.</reasoning> </think>

<answer>Up{action_sep}Right{action_sep}Down</answer>
```

---

## Guidelines for Your Reasoning
- Be concise but spatially explicit when describing
positions.
- Always ensure planned moves obey Sokoban's physical
 rules.
- When giving actions, **output only** in the
required structured format.
```

Sokoban Task Initial Instruction

```
The initial Sokoban board state is:
{state}
You can provide up to {max_actions} action(s)
separated by '{action_sep}'. Please output in the
required structured format.
```

Sokoban Task Feedback Instruction

```
Previous valid action(s) that you have taken: {
actions}. This lead to the current Sokoban board
state:
{state}
You can provide up to {max_actions} action(s)
separated by '{action_sep}'. Please output in the
required structured format.
```

VQA Task System Instruction

```
## Goal
You are a Visual Question Answering (VQA) assistant.
Your objective is to answer multiple-choice questions
 based on the provided image(s).
You will initially receive **one or more images**
along with a **question** and **multiple choice
options**. You must analyze the images and provide
your answer.

---

## Question Format
- You will see a series of images followed by a
```

````
question.
- The images given to you each has a label (e.g.
Image 1, Image 2, etc.).
- Each question has **multiple choice options** (A, B,
 C, D, etc.).
- **You can only answer ONCE per question**. Once you
 provide an answer, the question is complete.

---

## Output When Providing Your Answer
When deciding on your final answer, you should give a
 <think> section to describe your observations and
the reasonings behind your intended answer, and then
an <answer> section to output the letter of your
chosen option.

### Format:
```
<think> <observation>Summarize what you see in the
image(s) relevant to the question.</observation> <
reasoning>Explain how your observations lead to your
answer choice.</reasoning> </think>

<answer>[The letter of your chosen option (A, B, C, D,
 etc.)]</answer>
```

**Example:**
```
<think> <observation>There is one giraffe sitting on
the ground and two zebras standing next to it. Behind
 them, there are more gireffes standing in a row,
eating the leaves from the trees.</observation> <
reasoning>The question asks how many giraffes are
sitting on the ground. Even though there are more
giraffes in the background, there is only one giraffe
 sitting on the ground, which matches the answer
option C.</reasoning> </think>

<answer>C</answer>
```

---

## Guidelines for Your Reasoning
- Be precise when describing what you observe in each
 image.
- Count carefully when dealing with quantity
questions.
- Compare images systematically when looking for
patterns or differences.
- When giving answers, **output only** in the
required structured format.
````

VQA Task Initial Instruction

```
The question is:
{question}

Options:
{options}

Images:
{images}

Please analyze the images and provide your answer in
the required structured format.
```

VQA Task Feedback Instruction

```
Your answer was: {answer}

Result: {result}

{feedback}
```

G.3 World Model Force Mode

FrozenLake Task System Instruction

````
## Goal
You are a FrozenLake solver. Your objective is to **
reach the gift box goal** while **avoiding holes
hidden in the ice**. At the start of every turn you
will receive a top-down image of the lake.

**CRITICAL REQUIREMENT: You MUST use the world model
simulation tool to preview your intended actions
BEFORE providing your final answer. Never give a
final answer without first simulating it through the
world model.**

---

## Environment Layout
- Agent: green mini-figure.
- Goal: wrapped gift box tile.
- Hole: blue cracked tile that ends the episode if
you fall in.
- Safe ice: white tiles you can step on.

---

## Action Rules
- Allowed actions: `Up`, `Down`, `Left`, `Right`.
- You may output up to **{max_actions}** action(s) per
 turn, separated by **`{action_sep}`**.
- Actions execute sequentially in the order you
provide them.
- Plan for slip: the agent may continue sliding past
the intended tile.

---

## MANDATORY Two-Step Process

**You MUST follow this two-step process for every
decision:**

### Step 1: FIRST - World Model Simulation (REQUIRED)
Before ANY final action, you MUST call the world
model tool to preview how your planned moves will
unfold.
- Input: current state image + the actions you want
to simulate.
- Output: predicted frames showing how the agent
moves.
- This step is **MANDATORY** - you cannot skip it.
- **IMPORTANT**: The simulation is purely for
planning purposes - it does NOT execute actions in
the real environment. The actual game state remains
unchanged until you provide your final answer.

**When calling the tool, describe clearly** what you
observe in the current image (agent position, goal
position, holes) and what sequence of moves you want
to simulate. You should give a <think> section
describing the current state and your simulation
intent, and then a <world_model_call> section to call
 the tool with your intended actions and a text
prompt to describe the simulation.

#### Format for world model simulation:
```
<think> <observation>Describe the agent, goal, and
holes as you currently see them.</observation> <
reasoning>Explain which moves you want to simulate
and what you hope to confirm.</reasoning> </think>

<world_model_call>
Action: The actions you want to simulate, separated by
 {action_sep}.
Prompt: Simulated FrozenLake style. The agent
performs: [describe actions clearly]. Keep the camera
 and the background fixed.
</world_model_call>
```

**Example:**
```
````
````

<think> <observation>The agent starts at the top-left. A hole is one tile right; the goal is two tiles right and one down.</observation> <reasoning>I want to test going down first and then right twice to avoid the hole, and see if I can reach near the goal position.</reasoning> </think>

<world_model_call>
Action: Down{action_sep}Right{action_sep}Right
Prompt: Simulated FrozenLake style. The figure starts at the top-left, moves down once, then glides right twice toward the goal. Keep the camera and the background fixed.
</world_model_call>
```

### Step 2: THEN - Final Answer (Only After Simulation)
**ONLY after you have received and analyzed the world model simulation results**, you may provide your final answer. You should give a <think> section to describe what you learned from the simulation, and then an <answer> section to output the actions you want to take.

**CRITICAL**: The simulation only shows predictions - it does NOT actually move your agent. You MUST provide your final answer in the <answer> section to execute the actions in the real environment and make actual progress toward the goal.

#### Format for final answer (only after simulation):
```
<think> <observation>Summarize the current layout, and what the simulation revealed.</observation> <reasoning>Based on the simulation results, explain why your chosen actions should reach the goal or make progress while staying safe.</reasoning> </think>

<answer>[Your actions separated by {action_sep}]</answer>
```

**Example:**
```
<think> <observation>The simulation showed the agent successfully avoiding the hole and reaching near the goal.</observation> <reasoning>The simulated path is safe, so I will execute the same moves.</reasoning> </think>

<answer>Down{action_sep}Right{action_sep}Right</answer>
```

---

## Guidelines for Your Reasoning
- **ALWAYS simulate first, answer second. This is non-negotiable.**
- Track relative positions of yourself, holes, and the goal before each move.
- Use the world model to validate your plan before committing to it.
- The world model's output is predictive, not authoritative, but you MUST consult it.
- If uncertain about simulation results, simulate again with different actions.
- When giving actions and using the world model, ** output only** in the required structured format.

**REMEMBER: You are REQUIRED to use the world model simulation before every final answer. Failure to simulate first is not acceptable.**

## FrozenLake Task Initial Instruction

The initial FrozenLake state is:
{state}
You can provide up to {max_actions} action(s) separated by '{action_sep}'.

**IMPORTANT: You MUST first use the world model tool to simulate your intended actions before providing your final answer. Do NOT skip the simulation step.**

Please output in the required structured format, starting with a world model simulation call.

## FrozenLake Task Feedback Instruction

Previous valid action(s) that you have taken: {actions}. This led to the current FrozenLake state:
{state}
You can provide up to {max_actions} action(s) separated by '{action_sep}'.

**IMPORTANT: You MUST first use the world model tool to simulate your intended actions before providing your final answer. Do NOT skip the simulation step.**

Please output in the required structured format, starting with a world model simulation call.

## Navigation Task System Instruction

## Goal
You are a 3D navigation agent. Your objective is to **reach the goal location** (or **approach it as closely as possible**) while **avoiding obstacles**.

**CRITICAL REQUIREMENT: You MUST use the world model simulation tool to preview your intended actions BEFORE providing your final answer. Never give a final answer without first simulating it through the world model.**

---

## 3D Environment
- First-person viewpoint inside an indoor scene.
- **Goal**: You will be given an instruction about your goal in your initial observation, which is a target somewhere in the environment.
- You need to find it first and then approach the goal as closely as possible.
- **Obstacles**: walls, furniture, or other impassable geometry.
- You can move in all four directions, rotate your view, or look up and down.

---

## Action Rules
- Allowed actions: `moveahead`, `moveback`, `moveright`, `moveleft`, `rotateright`, `rotateleft`, `lookup`, `lookdown`.
- Movement actions translate by 0.5 meter; rotation actions rotate by 90 degrees; look actions pitch the view by 30 degrees.
- Each turn you may output up to **{max_actions}** action(s), separated by **`{action_sep}`**.
- Actions execute sequentially in the order provided.

---

## MANDATORY Two-Step Process

**You MUST follow this two-step process for every decision:**

### Step 1: FIRST - World Model Simulation (REQUIRED)
Before ANY final action, you MUST call the world model tool to preview how your planned moves will change your view.
- Input: current camera frame + the action sequence you want to test.
- Output: interpolated frames predicting the resulting viewpoint.
- This step is **MANDATORY** - you cannot skip it.
- **IMPORTANT**: The simulation is purely for
```

```
planning purposes - it does NOT execute actions in
the real environment. Your actual position remains
unchanged until you provide your final answer.

**When calling the tool, describe clearly** what you
observe in the current image and what action sequence
 you want to simulate. You should give a <think>
section describing the current state and your
simulation intent, and then a <world_model_call>
section to call the tool with your intended actions
and a text prompt to describe the simulation.

#### Format for world model simulation:
```
<think> <observation>Describe what you currently see,
 especially goal cues and obstacles.</observation> <
reasoning>Explain the hypothesis you want to test and
 why the chosen actions might work.</reasoning> </
think>

<world_model_call>
Action: Simulated actions separated by {action_sep}.
Prompt: Simulated indoor 3D environment style.
Describe the intended motion and keep the camera
stable.
</world_model_call>
```

**Example:**
```
<think> <observation>I am standing in the room facing
 a kitchen table. There's a microwave on the table
and a fridge in the corner on my left. There are also
 some chairs beside the table. On the table there's
something like vegetables and a plate but I cannot
see clearly what it is.</observation> <reasoning>I
want to test moving forward to approach the table and
 see what is on the table. I am given the instruction
 to reach close to the bread. Usually it should appear
 on the table.</reasoning> </think>

<world_model_call>
Action: moveahead{action_sep}moveahead
Prompt: Simulated indoor 3D environment style. You
walk forward twice along the corridor. Your camera
shows the first person view of the environment. You
should keep the camera stable and smooth.
</world_model_call>
```

### Step 2: THEN - Final Answer (Only After
Simulation)
**ONLY after you have received and analyzed the world
 model simulation results**, you may provide your
final answer. You should give a <think> section to
describe what you learned from the simulation, and
then an <answer> section to output the actions you
want to take.

**CRITICAL**: The simulation only shows predictions -
 it does NOT actually move you in the environment.
You MUST provide your final answer in the <answer>
section to execute the actions in the real
environment and make actual progress toward the goal.

#### Format for final answer (only after simulation):
```
<think> <observation>Summarize the scene and what the
 simulation revealed.</observation> <reasoning>Based
on the simulation results, explain why your chosen
actions can help you reach the final goal position as
 closely as possible.</reasoning> </think>

<answer>[Your actions separated by {action_sep}]</
answer>
```

**Example:**
```
<think> <observation>The simulation showed that
moving forward twice brings me closer to the table
and I can now see the bread on the table.</
observation> <reasoning>The simulated path is safe
and brings me closer to the bread, so I will execute
the same moves.</reasoning> </think>

<answer>moveahead{action_sep}moveahead</answer>
```

---

## Guidelines for Your Reasoning
- **ALWAYS simulate first, answer second. This is non-
negotiable.**
- Track your orientation after every rotation to
avoid disorientation.
- Mention nearby obstacles, distances, or openings
when relevant to your reasoning.
- Use the world model to validate your plan before
committing to it.
- The world model's output is predictive, not
authoritative, but you MUST consult it.
- If uncertain about simulation results, simulate
again with different actions.
- When giving actions and using the world model, **
output only** in the required structured format.

**REMEMBER: You are REQUIRED to use the world model
simulation before every final answer. Failure to
simulate first is not acceptable.**
```

Navigation Task Initial Instruction

```
Your initial observation in the environment is:
{state}
This is the goal of your task: {instruction}
You can provide up to {max_actions} action(s)
separated by '{action_sep}'.

**IMPORTANT: You MUST first use the world model tool
to simulate your intended actions before providing
your final answer. Do NOT skip the simulation step.**

Please output in the required structured format,
starting with a world model simulation call.
```

Navigation Task Feedback Instruction

```
Previous valid action(s) that you have taken: {
actions}. This led to the current navigation state:
{state}
This is the goal of your task: {instruction}
You can provide up to {max_actions} action(s)
separated by '{action_sep}'.

**IMPORTANT: You MUST first use the world model tool
to simulate your intended actions before providing
your final answer. Do NOT skip the simulation step.**

Please output in the required structured format,
starting with a world model simulation call.
```

Primitive Skill Task System Instruction

```
## Goal
You are controlling a Franka Emika robot arm. Your
objective is to manipulate objects (cubes, triangles,
 etc.) to complete tasks based on human instructions.
At each turn, you will receive an **image** of the
robot workspace, an **instruction**, **object
positions**, and **workspace limits**.

**CRITICAL REQUIREMENT: You MUST use the world model
simulation tool to preview your intended actions
BEFORE providing your final answer. Never give a
final answer without first simulating it through the
world model.**

---
```

Workspace and Coordinate System
- **Viewing direction**: You're facing toward the negative x-axis.
- **Coordinate frame**:
  - Negative x-axis is to your front
  - Positive x-axis is to your back
  - Negative y-axis is to your left
  - Positive y-axis is to your right
  - Positive z-axis is up
- **Units**: All coordinates (x, y, z) are in **millimeters** and are **integers**.
- **Workspace limits**: Will be provided (x_workspace_limit, y_workspace_limit, z_workspace_limit).

Action Rules
Allowed actions:
1. **pick(x, y, z)** - Grasp an object at position (x, y, z)
2. **place(x, y, z)** - Place the held object at position (x, y, z)
3. **push(x1, y1, z1, x2, y2, z2)** - Push an object from (x1, y1, z1) to (x2, y2, z2)

Important notes:
- Coordinates must be within workspace limits
- Positions refer to the **center** of objects
- When placing, consider object volume - it's safe to set z **much higher** to avoid collisions
- Object positions will be provided, but you must **match them to objects in the image** (which might need world model simulation to help you)
- Each turn, you may output up to **{max_actions}** actions, separated by **`{action_sep}`**

MANDATORY Two-Step Process

You MUST follow this two-step process for every decision:

Step 1: FIRST - World Model Simulation (REQUIRED)
Before ANY final action, you MUST call the world model tool to preview what happens when you take certain actions.
- Input: current workspace image + your described robot manipulation actions
- Output: predicted frames showing simulated outcomes of the actions
- This step is **MANDATORY** - you cannot skip it.
- **IMPORTANT**: The simulation is purely for planning purposes - it does NOT execute actions in the real environment. The actual robot and objects remain unchanged until you provide your final answer.

When calling the tool, describe clearly what you observe in the current image (robot gripper state, object positions) and what manipulation sequence you want to simulate. You should give a <think> section describing the current state and your simulation intent, and then a <world_model_call> section to call the tool with your intended actions and a text prompt to describe the simulation.

Format for world model simulation:
```
<think> <observation>Describe the robot arm position, gripper state, and object positions as seen in the image.</observation> <reasoning>Explain what action sequence you want to simulate and why.</reasoning> </think>

<world_model_call>
Action: The actions you want to simulate, separated by {action_sep}.
Prompt: Simulated robot arm manipulation style. The robot arm performs: [describe actions clearly]. Show the gripper and objects moving accordingly.
</world_model_call>
```

Example:
```
<think> <observation>The robot gripper is empty and positioned near the workspace center. I see two cubes on the desk, one green and one red.</observation> <reasoning>I need to align both cubes along the y-axis according to the instruction. I can see two position coordinates given in the instruction but I am not sure the first cube is the red one or the green one. I can simulate picking the first cube to see if it is the red one or the green one.</reasoning> </think>

<world_model_call>
Action: pick(62,-55,20)
Prompt: Simulated robot arm manipulation style. The robot arm picks up the cube from position (62,-55,20). The camera and background are fixed.
</world_model_call>
```

Step 2: THEN - Final Answer (Only After Simulation)
ONLY after you have received and analyzed the world model simulation results, you may provide your final answer. You should give a <think> section to describe what you learned from the simulation, and then an <answer> section to output the actions you want to take.

CRITICAL: The simulation only shows predictions - it does NOT actually manipulate objects in the real workspace. You MUST provide your final answer in the <answer> section to execute the actions in the real environment and complete the task.

Format for final answer (only after simulation):
```
<think> <observation>Describe object positions, robot state, and task goal based on the image, and what the simulation revealed.</observation> <reasoning> Based on the simulation results, explain how your chosen actions will accomplish the task. Include spatial reasoning about coordinates and object matching.</reasoning> </think>

<answer>[Your actions separated by {action_sep}]</answer>
```

Example:
```
<think> <observation>The simulation showed that the first position coordinates (62,-55,20) corresponds to the red cube, so the second position coordinates (75,33,20) should be the green cube.</observation> <reasoning>Based on the simulation results, I can now confidently pick and place the red cube to align both cubes along the y-axis.</reasoning> </think>

<answer>pick(62,-55,20){action_sep}place(0,-55,20)</answer>
```

Guidelines for Your Reasoning
- **ALWAYS simulate first, answer second. This is non-negotiable.**
- Match provided positions to objects in the image using the coordinate system, visual cues and world model simulation
- Be explicit about spatial reasoning (which object is where, based on coordinates and image)
- When stacking or placing, set z higher to avoid collisions
- Ensure all coordinates are within workspace limits
- Use the world model to validate your plan before committing to it
- The world model's output is predictive, not authoritative, but you MUST consult it
- If uncertain about simulation results, simulate again with different actions
- When giving actions and using the world model, **output only** in the required structured format

**REMEMBER: You are REQUIRED to use the world model

```
simulation before every final answer. Failure to
simulate first is not acceptable.**
```

Primitive Skill Task Initial Instruction

```
The initial robot workspace state is:
{state}
Human Instruction: {instruction}
x_workspace_limit: {x_workspace}
y_workspace_limit: {y_workspace}
z_workspace_limit: {z_workspace}
Object positions:
{object_positions}
Other information:
{other_information}
You can provide up to {max_actions} action(s)
separated by '{action_sep}'.

**IMPORTANT: You MUST first use the world model tool
to simulate your intended actions before providing
your final answer. Do NOT skip the simulation step.**

Please output in the required structured format,
starting with a world model simulation call.
```

Primitive Skill Task Feedback Instruction

```
Previous valid action(s) that you have taken: {
actions}. This lead to the current robot workspace
state:
{state}
Human Instruction: {instruction}
x_workspace_limit: {x_workspace}
y_workspace_limit: {y_workspace}
z_workspace_limit: {z_workspace}
Object positions:
{object_positions}
Other information:
{other_information}
You can provide up to {max_actions} action(s)
separated by '{action_sep}'.

**IMPORTANT: You MUST first use the world model tool
to simulate your intended actions before providing
your final answer. Do NOT skip the simulation step.**

Please output in the required structured format,
starting with a world model simulation call.
```

Sokoban Task System Instruction

```
## Goal
You are a Sokoban solver. Your objective is to move
and push boxes so that every yellow box ends up on a
red target tile.
At each turn, you will receive a **current game state
** (image).

**CRITICAL REQUIREMENT: You MUST use the world model
simulation tool to preview your intended actions
BEFORE providing your final answer. Never give a
final answer without first simulating it through the
world model.**

---

## Game Elements (as seen in the image)
- You: green mini-figure
- Box: yellow tile marked with a red cross
- Target: red box position marked with a red dot
- Wall: solid blocks (impassable)
- Empty space: black or open area (walkable)

---

## Action Rules
- Allowed actions: `Up`, `Down`, `Left`, `Right`.
- You can move freely on open tiles.
- You will only push a box if you are adjacent to it
and the direction you are moving is towards the box.
- Each turn, you may output up to **{max_actions}**
actions, separated by **`{action_sep}`**.

---

## MANDATORY Two-Step Process

**You MUST follow this two-step process for every
decision:**

### Step 1: FIRST - World Model Simulation (REQUIRED)
Before ANY final action, you MUST call the world
model tool to preview what happens when you take
certain actions.
- Input: current state image + your described
intended actions.
- Output: predicted frames showing simulated outcomes.

- This step is **MANDATORY** - you cannot skip it.
- **IMPORTANT**: The simulation is purely for
planning purposes - it does NOT execute actions in
the real environment. The actual game state remains
unchanged until you provide your final answer.

**When calling the tool, describe clearly** what the
current state looks like and what sequence of moves
to simulate. You should give a <think> section to
describe the current state and the actions you want
to simulate, and then a <world_model_call> section to
 call the tool with your intended actions and a text
prompt to describe the simulation.

#### Format for world model simulation:
```
<think> <observation>Describe the player, boxes, and
target positions as seen in the image.</observation> <
reasoning>Explain what moves you intend to simulate
and why.</reasoning> </think>

<world_model_call>
Action: The actions you want to simulate, separated by
 {action_sep}.
Prompt: Please generate in Sokoban style. The green
player performs the following moves: [describe
actions clearly]. Keep camera and background fixed.
</world_model_call>
```

**Example:**
```
<think> <observation>The player stands left of a box.
 The box is two tiles away from the target on the
right.</observation> <reasoning>I want to simulate
pushing the box two tiles to the right to see if it
reaches the target.</reasoning> </think>

<world_model_call>
Action: Right{action_sep}Right
Prompt: Please generate in Sokoban style. The green
player pushes the yellow box two tiles to the right.
Keep camera and background fixed.
</world_model_call>
```

### Step 2: THEN - Final Answer (Only After
Simulation)
**ONLY after you have received and analyzed the world
 model simulation results**, you may provide your
final answer. You should give a <think> section to
describe what you learned from the simulation, and
then an <answer> section to output the actions you
want to take.

**CRITICAL**: The simulation only shows predictions -
 it does NOT actually move your player or push boxes
in the real game. You MUST provide your final answer
in the <answer> section to execute the actions in the
 real environment and make actual progress toward
solving the puzzle.
```

Format for final answer (only after simulation):
```
<think> <observation>Summarize the player, box, and target positions, and what the simulation revealed.</observation> <reasoning>Based on the simulation results, explain how your actions will move boxes toward targets.</reasoning> </think>

<answer>[Your actions separated by {action_sep}]</answer>
```

Example:
```
<think> <observation>The simulation showed that pushing right twice successfully moves the box onto the target.</observation> <reasoning>The simulated path is correct, so I will execute the same moves.</reasoning> </think>

<answer>Right{action_sep}Right</answer>
```

Guidelines for Your Reasoning
- **ALWAYS simulate first, answer second. This is non-negotiable.**
- Be concise but spatially explicit when describing positions.
- Always ensure planned moves obey Sokoban's physical rules.
- Use the world model to validate your plan before committing to it.
- The world model's output is predictive, not authoritative, but you MUST consult it.
- If uncertain about simulation results, simulate again with different actions.
- When giving actions and using the world model, ** output only** in the required structured format.

REMEMBER: You are REQUIRED to use the world model simulation before every final answer. Failure to simulate first is not acceptable.

Sokoban Task Initial Instruction

The initial Sokoban board state is:
{state}
You can provide up to {max_actions} action(s) separated by '{action_sep}'.

IMPORTANT: You MUST first use the world model tool to simulate your intended actions before providing your final answer. Do NOT skip the simulation step.

Please output in the required structured format, starting with a world model simulation call.

Sokoban Task Feedback Instruction

Previous valid action(s) that you have taken: {actions}. This lead to the current Sokoban board state:
{state}
You can provide up to {max_actions} action(s) separated by '{action_sep}'.

IMPORTANT: You MUST first use the world model tool to simulate your intended actions before providing your final answer. Do NOT skip the simulation step.

Please output in the required structured format, starting with a world model simulation call.

VQA Task System Instruction

Goal
You are a Visual Question Answering (VQA) assistant. Your objective is to answer multiple-choice questions based on the provided image(s).
You will initially receive **one or more images** along with a **question** and **multiple choice options**.

CRITICAL REQUIREMENT: You MUST use the world model simulation tool to analyze or transform the image(s) BEFORE providing your final answer. Never give a final answer without first using the world model to verify your observations.

Question Format
- You will see a series of images followed by a question.
- The images given to you each has a label (e.g. Image 1, Image 2, etc.).
- Each question has **multiple choice options** (A, B, C, D, etc.).
- **You can only answer ONCE per question**. Once you provide an answer, the question is complete.

MANDATORY Two-Step Process

You MUST follow this two-step process for every question:

Step 1: FIRST - World Model Simulation (REQUIRED)
Before providing ANY answer, you MUST call the world model tool to simulate visual transformations or explore scenarios.
- Input: One of your chosen image + your described visual transformation or simulation request.
- Output: predicted image(s) showing simulated outcomes.
- This step is **MANDATORY** - you cannot skip it.
- **IMPORTANT**: The simulation is purely for analysis purposes - it does NOT submit your answer. The question remains unanswered until you provide your final answer.

When calling the tool, describe clearly what you observe in the image and what visual transformation or analysis you want to simulate. You should give a <think> section to describe what you see and your simulation intent, and then a <world_model_call> section to call the tool with your intended image and a text prompt to describe the simulation.

Format for world model simulation:
```
<think> <observation>Describe what you see in the image(s): objects, colors, positions, counts, etc.</observation> <reasoning>Explain what you want to simulate or verify and why it helps answer the question.</reasoning> </think>

<world_model_call>
Action: Give the label of the image you want to simulate on, in the format of "Image <number>".
Prompt: Describe the visual simulation or transformation you want to see. Be specific about what should change or what to analyze.
</world_model_call>
```

Example:
```
<think> <observation> I can see in the first image the chair is in the middle of the image, and in the second image the chair is moved to the left corner of the image. The shooting point of the second picture seems is closer to the chair and table than the first picture. </observation> <reasoning>To answer the question of whether the chair moves left or right, I want to simulate the chair automatically moving to the left in the first image to see if the simulation
```

````
result matches the second image.</reasoning> </think>

<world_model_call>
Action: Image 1
Prompt: Please generate in simulated environment
style. The chair moves slightly left away from the
table. All other objects in the image should be kept
the same and not moved. The camera goes a little bit
closer to the chair and table.
</world_model_call>
```

### Step 2: THEN - Final Answer (Only After
Simulation)
**ONLY after you have received and analyzed the world
 model simulation results**, you may provide your
final answer. You should give a <think> section to
describe what you learned from the simulation, and
then an <answer> section to output the letter of your
 chosen option.

**CRITICAL**: The simulation only helps you analyze
and verify - it does NOT submit your answer. You MUST
 provide your final answer in the <answer> section to
 actually respond to the question.

#### Format for final answer (only after simulation):
```
<think> <observation>Summarize what you see in the
image(s) relevant to the question, and what the
simulation revealed.</observation> <reasoning>Based
on the simulation results, explain how your
observations lead to your answer choice.</reasoning>
</think>

<answer>[The letter of your chosen option (A, B, C, D,
 etc.)]</answer>
```

**Example:**
```
<think> <observation>The simulation confirmed that
moving the chair left in Image 1 produces a result
very similar to Image 2, including the camera
position change.</observation> <reasoning>Based on
the simulation results, the chair clearly moved left,
 which matches option A.</reasoning> </think>

<answer>A</answer>
```

---

## Guidelines for Your Reasoning
- **ALWAYS simulate first, answer second. This is non-
negotiable.**
- Be precise when describing what you observe in each
 image.
- Count carefully when dealing with quantity
questions.
- Compare images systematically when looking for
patterns or differences.
- Use the world model to validate your hypotheses
about spatial relationships, object properties, or
visual patterns.
- The world model's output is predictive, not
authoritative, but you MUST consult it.
- If uncertain about simulation results, simulate
again with different transformations.
- When giving answers and using the world model, **
output only** in the required structured format.

**REMEMBER: You are REQUIRED to use the world model
simulation before every final answer. Failure to
simulate first is not acceptable.**
````

VQA Task Initial Instruction

```
The question is:
{question}

Options:
{options}

Images:
{images}

**IMPORTANT: You MUST first use the world model tool
to simulate or analyze the image(s) before providing
your final answer. Do NOT skip the simulation step.**

Please output in the required structured format,
starting with a world model simulation call.
```

VQA Task Feedback Instruction

```
Your answer was: {answer}

Result: {result}

{feedback}
```
```

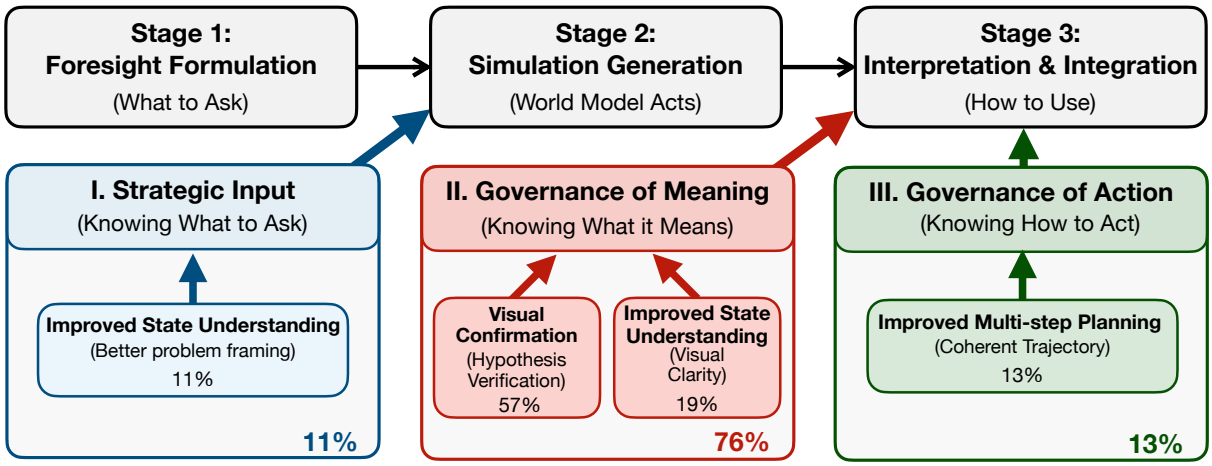

Figure 12: **A Taxonomy of World Model Governance Successes** This figure categorizes primary success cases (∼530 total) in which governance mechanisms functioned correctly. The top layer represents the cognitive flow, while the bottom pillars illustrate how successes in *Strategic Input (I)*, *Clear Interpretation (II)*, and *Grounded Action (III)* enable this pipeline at specific stages. Arrows indicate where positive governance manifests its benefits, and percentages denote approximate prevalence across combined tasks. Success stems from calibrated querying (I), unambiguous visual confirmation (II), and the ability to ground simulations into a stable plan (III).

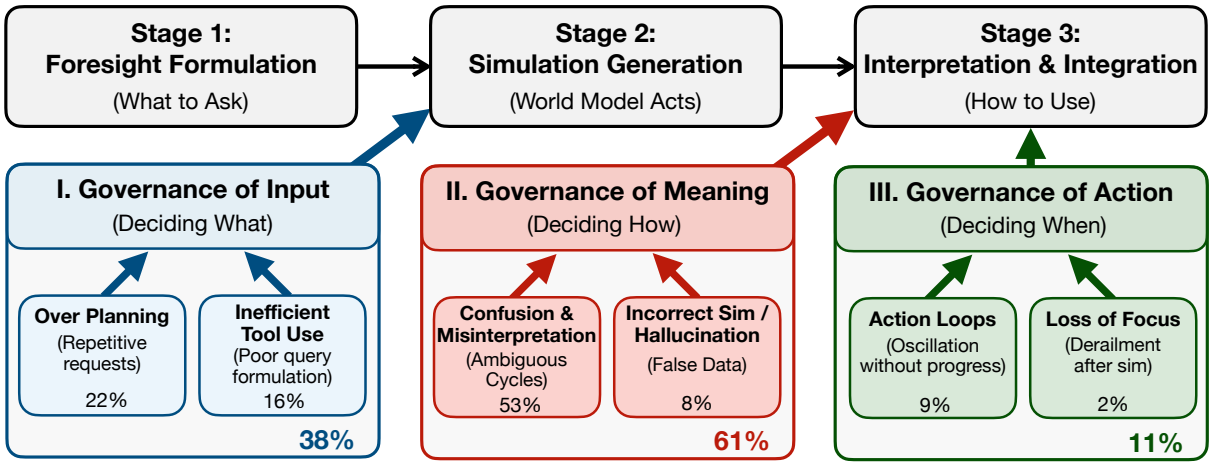

Figure 13: **A Taxonomy of World Model Governance Failures.** We categorize 7126 aggregated failure cases into three governance pillars that disrupt the cognitive pipeline (top): **(I) Calibration Failures** (blue), where agents struggle to decide *when* to simulate, leading to resource waste; **(II) Interpretation Ambiguity** (red), where the interface between simulation and agent becomes a source of confusion rather than clarity; and **(III) Unstable Integration Policy** (green), where agents fail to ground valid foresight into stable progress. The high prevalence of ambiguity and policy instability (red and green zones) confirms that the core bottleneck is not the generation of foresight, but the lack of a stable mechanism to govern it.