

Population-Scalable Multi-Agent World Modeling

Renjie Zhao^{1,2,*}, Yuxiang Wu^{1,2,*}, Mingyu Zhang^{1,2,◇,✉}, Jiaxin Li^{1,2,◇}, Sisi Li³,
Yimin Sheng³, Tianxi Tan^{1,2}, Zhenkai Zhang², Jianyi Zhu³, Yong-Lu Li^{1,2,†,✉}

¹RhOS.ai ²Shanghai Jiao Tong University ³Ophilus.AI

*Equal contribution. ◇Project leaders. †Corresponding author.

✉ mingyuzhang@rhos.ai yonglu_li@rhos.ai

Project Page: <https://rhos.ai/research/khora>

Online Demo: <https://ophilus.ai/khora>

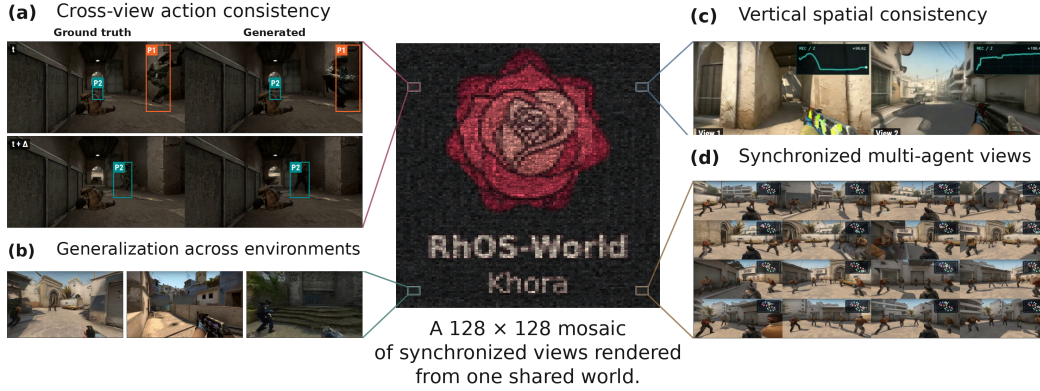


Figure 1: RhOS-World: Khora, a scalable multi-agent world model, maintains spatiotemporal and cross-agent consistency across dynamic agent populations.

Abstract

World models have recently achieved impressive progress in visual prediction and interactive generation, but extending them to multi-agent environments introduces a fundamental scalability challenge. Existing methods generally assume a fixed number of agents during training and inference, which ties the model to a pre-determined agent population and limits inference-time scalability. Our key insight is that cross-view consistency should arise from a shared world state whose evolution does not assume a predefined number of agents, while agent-specific observations should be generated by querying this state through a unified rendering interface. Based on this insight, we propose *Khora*, a scalable multi-agent world model that supports inference-time expansion to arbitrary numbers of agents without retraining. Our framework decouples world-state evolution from visual rendering and introduces a population-agnostic rendering mechanism for incorporating other agent information. This design maintains cross-view consistency through the shared world state rather than through dense interactions among observation streams inside the expensive video generator, enabling approximately linear practical scaling with the number of queried views. Qualitative experiments demonstrate that our approach generalizes to unseen numbers of agents while maintaining visual quality and multi-agent consistency. We further implement a real-time interactive system to demonstrate scalable open-world simulation.

1 Introduction

Imagine a multiplayer battlefield where different players simultaneously fire weapons, throw grenades, and observe the same battle from their own perspectives. To keep the world consistent, the underlying system must maintain a coherent and continuously evolving world state despite these concurrent interactions. A world model for such a setting does more than generate plausible egocentric videos. It should maintain a shared state that remains consistent across agents, viewpoints, and actions. This capability is relevant to embodied AI, robotics, autonomous driving, interactive games, and social simulation. Recent visual world models have made progress in action-conditioned prediction and interactive generation [Wang et al., 2023, Bruce et al., 2024, Hafner et al., 2023]. Emerging multi-agent world models further expose the need for cross-view and action consistency [Savva et al., 2026, Wu et al., 2026, Hu et al., 2026]. Yet a general simulator requires more than consistency among a fixed set of agents: new agents should be able to enter the world at inference time without retraining or changing the model interface.

However, existing multi-agent world models generally assume a fixed agent population. Some methods [Savva et al., 2026, Hu et al., 2026, Liu et al., 2026] introduce cross-agent communication directly inside the video generator, which will lead to dense interactions among observation streams and substantially increase the cost of neural rendering as the population grows. Other approaches [Po et al., 2026, Odyssey, 2026] maintain shared latent representations or external memories, but still organize generation around a predefined number or layout of view slots. Despite their different designs, these methods share a common assumption: the number of participating agents is predetermined during training. Consequently, adding agents at inference time often requires retraining, architectural changes, or a predefined view layout. We argue that this dependence on a fixed population reflects a deeper mismatch between the model interface and the structure of the underlying world.

The governing rules of a physical world are population agnostic. Although the resulting state evolution depends on the entities present and their interactions, the underlying transition rules do not need to be redesigned when the population changes. The underlying dynamics of the environment should follow the same physical rules. Population expansion should therefore be supported as an inference-time operation, rather than encoded through a fixed collection of agent-specific model components.

Motivated by this principle, we propose *Khora*, a scalable multi-agent world model designed for open-world simulation. *Khora* is initialized using agents’ initial observations and a coarse static representation of the environment. During rollout, an action-conditioned state model predicts agent poses and updates a shared STBoard containing persistent scene memory and dynamic entity states. To render a target observation, *Khora* transforms the relevant entities into the target camera coordinate frame and rasterizes them into a fixed-dimensional spatial condition. The neural renderer processes one target view at a time through an interface. And this method does not dimensionality depend on the total number of agents. Additional agents therefore introduce additional state entries and rendering queries without changing the renderer architecture.

We study *Khora* through a set of controlled qualitative evaluations and an interactive system prototype. The results illustrate cross-view action consistency, persistence under partial occlusion, coherent rendering from multiple synchronized viewpoints, and operation with agent populations not observed during training. We additionally evaluate the same shared-state and rendering interfaces across multiple environments with different scene-specific memories.

Our contributions are summarized as follows:

1. We formulate **inference-time population scalability** as an explicit objective for multi-agent world models: the model interface should support dynamically varying agent populations without architectural modification or retraining.
2. We introduce *Khora*, a scalable world model architecture that decouples world-state evolution from visual rendering. It combines a persistent shared STBoard, an action-conditioned pose and state transition model, and a projection-based per-view rendering interface.
3. We demonstrate that *Khora* generalizes to dynamically varying numbers of agents while maintaining visual quality, multi-view consistency and action consistency. It achieves scalable real-time simulation with dominant rendering cost scaling linearly with the number of agents.

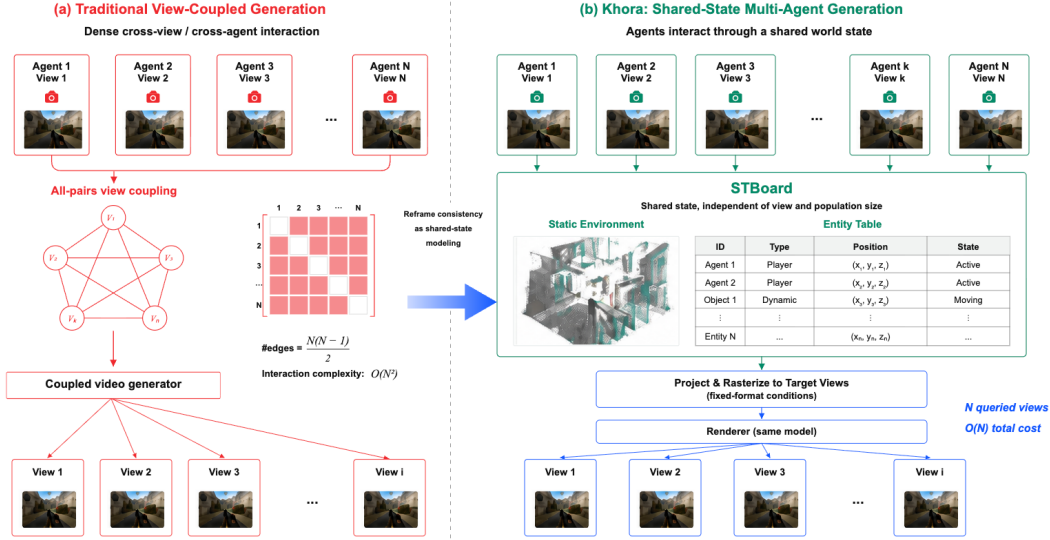


Figure 2: **Conceptual comparison of Khora with existing multi-agent world models.** Existing models often couple consistency to a predefined set of agents or views, tied to an all-pairs interaction graph. Khora instead reframes consistency as shared-state modeling: agents write actions and observations to STBoard and query it for view-conditioned rendering. This shared-state interface allows new agents to be introduced at inference time without retraining, while preserving synchronized observations from the same evolving world.

2 Related Works

World models and interactive generation. World models were originally studied as compact predictive models of environment dynamics for planning and control [Ha and Schmidhuber, 2018, Hafner et al., 2023]. Recent progress in generative modeling has shifted this direction from low-dimensional latent dynamics toward visually rich interactive simulation. Diffusion objectives and transformer backbones have improved high-fidelity video prediction and controllable generation [Ho et al., 2020, Peebles and Xie, 2023, Dosovitskiy et al., 2021]. Building on these advances, diffusion-based and latent-action world models have been used to synthesize action-conditioned futures in game-like environments [Alonso et al., 2024, Gao et al., 2025], while large video models and pretrained video diffusion models have been adapted into interactive world models through action-conditioned generation [Bruce et al., 2024, Huang et al., 2026]. This line of work demonstrates that video generators can provide compelling interactive feedback, but most systems are still organized around a single egocentric stream or a fixed video interface. Khora uses a diffusion-transformer renderer as an observation decoder, but treats rendering as only one component of a stateful multi-agent world model rather than as the world model itself.

Multi-agent video world models. Extending video world models from one agent to many agents introduces additional requirements: shared environment dynamics, concurrent actions, viewpoint-dependent observations, and cross-view agreement. Solaris [Savva et al., 2026] makes an important step in this direction by studying two-player video generation in Minecraft, using cross-attention between views to improve cross-view consistency. MultiGen [Po et al., 2026] moves beyond two-player settings by introducing external memory for editable multiplayer worlds in diffusion game engines, allowing world edits to persist beyond the diffusion model’s local context window. MultiWorld [Wu et al., 2026] further studies scalable multi-agent, multi-view video synthesis with a multi-agent conditioning module and a global state encoder. Gamma-World [Liu et al., 2026] focuses on the cost of modeling more than two players, using permutation-symmetric agent encodings and sparse hub attention to reduce the cost of dense interactions. These works establish multi-agent control and cross-view consistency as core problems for interactive world models, but often remain tied to fixed view layouts or population-dependent interactions. Khora instead decouples shared-state

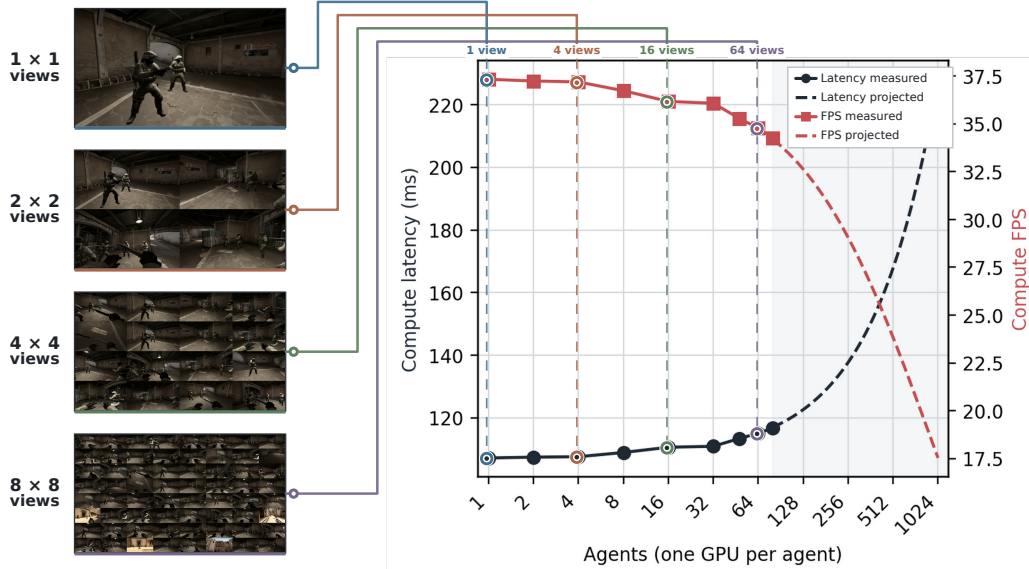


Figure 3: **Inference-time population scalability of Khora.** Left: representative outputs for 1, 4, 16, and 64 synchronized views. Right: compute-only latency and FPS using one GPU per agent. Across the measured range, Khora maintains nearly constant latency with only a modest decrease in FPS as the agent population grows; dashed curves show projections beyond the measured regime.

Table 1: **Comparison of representative multi-agent video world models.** ✓ indicates explicit support, △ partial or limited support, and – indicates that the capability is not explicitly established.

Method	Persistent shared state	Dynamic population	Unseen-count generalization	Population-time scaling
Solaris [Savva et al., 2026]	–	–	–	$O(N^2)$
MultiGen [Po et al., 2026]	✓	△	–	$O(N)$
MultiWorld [Wu et al., 2026]	△	✓	△	$O(N^2)$
Gamma-World [Liu et al., 2026]	–	✓	△	$O(N)$
Khora (ours)	✓	✓	✓	$\approx O(N)$

evolution from per-view rendering through a persistent STBoard, enabling inference-time population expansion. Table 1 shows the difference between different methods.

Evaluation of visual world models. Beyond architectural support for multi-agent interaction and scalability, evaluating whether these models actually represent a coherent shared world remains another important challenge. Visual world models are commonly evaluated using frame-level reconstruction metrics such as PSNR and SSIM [Wang et al., 2004], perceptual similarity metrics such as LPIPS [Zhang et al., 2018], and distribution-level metrics such as FID [Heusel et al., 2018] and FVD [Unterthiner et al., 2019]. These metrics characterize the fidelity, perceptual quality, and temporal realism of individual generated views, but do not directly determine whether multiple views describe the same evolving world. Recent multi-agent world models have therefore increasingly emphasized shared-world consistency. Solaris [Savva et al., 2026] introduces multiplayer evaluations covering movement, memory, grounding, building, and view consistency; MultiGen [Po et al., 2026] highlights coherent viewpoints and consistent cross-player interactions through persistent shared memory; MultiWorld [Wu et al., 2026] evaluates multi-view consistency together with visual quality and action following; Agora-1 [Odyssey, 2026] demonstrates real-time interaction among multiple human or AI participants within a common generated world; and MIRA [Hu et al., 2026] further develops targeted evaluations of multiplayer dynamics and physical behavior beyond visual appearance. Motivated by these developments, we evaluate multi-agent consistency along two complementary dimensions: *cross-view action consistency*, which measures whether action

consequences remain compatible across observers, and *cross-view world consistency*, which measures whether synchronized views agree on entity states, spatial relationships, identity, and persistence.

3 Preliminary

Consider a world with a variable set of agents I_t . At inference time, the model receives a coarse static world prior G , a collection of initial observations $O_0 = \{o_0^i\}_{i \in I_0}$. At each subsequent time step t , the model receives a set of actions $A_t = \{a_t^i\}_{i \in I_t}$. The initial observations and static prior are used to initialize a shared world state and the initial agent poses:

$$S_0, \{p_0^i\}_{i \in I_0} = E(G, O_0). \quad (1)$$

During autoregressive rollout, no ground-truth future observations or poses are provided. The transition model predicts the next poses and updates the shared state from the current state and agent actions:

$$S_{t+1}, \{p_{t+1}^i\}_{i \in I_t} = F_\theta(S_t, \{p_t^i, a_t^i\}_{i \in I_t}; G). \quad (2)$$

Each requested observation is then independently decoded from the updated shared state:

$$o_{t+1}^i = R(S_{t+1}, \{p_{t+1}^i\}_{i \in I_t}, G). \quad (3)$$

This formulation emphasizes three properties of a multi-agent world model. First, **persistence**: the shared state preserves scene and entity information across agents, viewpoints, and temporary occlusions. Second, **update**: agent actions modify both the predicted agent poses and the shared dynamic state. Third, **scalability**: observations are generated through independent view queries whose interface does not assume a predefined number of agents.

In this paper, we study inference-time population scalability. Given a model trained with finite agent populations, our objective is to support dynamically varying numbers of agents without changing the transition or rendering architecture.

4 Method

Given a shared world state initialized from initial observations and a coarse static scene prior, Khora autoregressively predicts future agent poses and observations from agent actions. The number of active agents and the number of requested views may change during inference.

Khora is a scalable world model that decouples world-state evolution from visual rendering. As illustrated in Figure 4, it consists of three components. The **STBoard** maintains persistent scene memory and a dynamically sized set of entity states in shared world coordinates (Section 4.1). The **Action-conditioned World Evolution** model predicts pose changes and updates dynamic entities from the incoming action streams (Section 4.2). The **Geometry-guided View Synthesis** transforms the updated entity states into fixed-dimensional target-view conditions and independently synthesizes each requested observation (Section 4.3). This decomposition keeps the model interface independent of a predefined population and removes cross-agent interaction from the expensive neural renderer.

4.1 STBoard

Existing multi-agent world models usually organize latent representations around a predefined set of views or observers. Consequently, both the learned representation and the rendering interface implicitly depend on the number of participating agents. Such population-dependent representations limit the scalability of the model and prevent arbitrary-agent expansion during inference.

Khora instead maintains a world-centric shared representation called the **Spatio-temporal Board (STBoard)**. It contains two components: a static scene memory representing persistent environmental context and a dynamic entity table representing the current states of active agents. The static memory is initialized from the coarse scene prior and refined using visual trajectories. Each dynamic entity is represented in a shared world coordinate system and stores attributes such as identity, pose, velocity, orientation, occupancy, and task-dependent state.

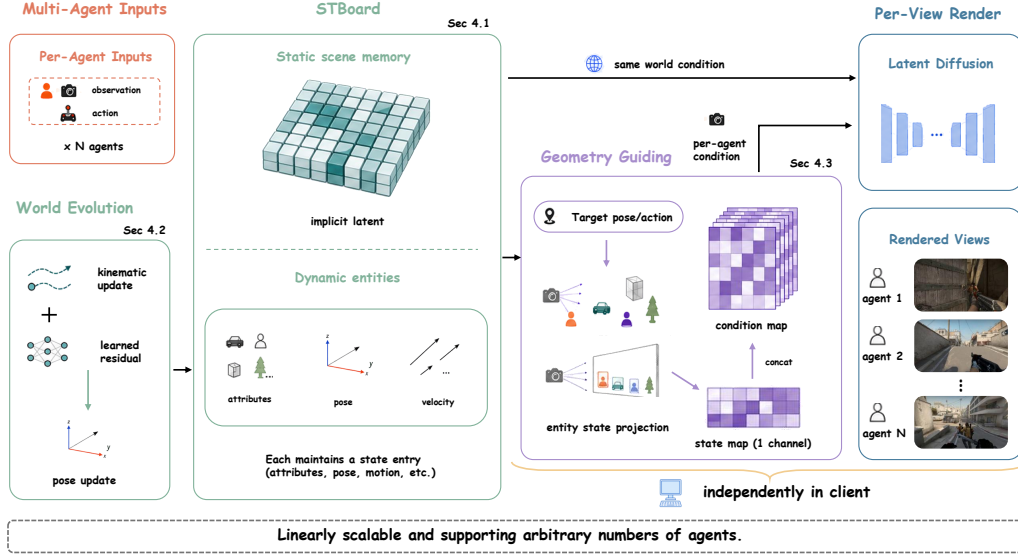


Figure 4: **Overview of Khora.** Initial agent observations and a coarse scene prior initialize a shared STBoard. During autoregressive rollout, agent actions drive an action-conditioned transition model that predicts agent poses and updates dynamic entity states; no ground-truth future poses or observations are required. For each requested view, the predicted entities are transformed into the target camera frame and rasterized into a fixed-dimensional spatial condition. A shared neural renderer independently decodes each condition into a synchronized first-person observation.

Initial observations are encoded and written into the STBoard to initialize agent-specific appearance and state information. During rollout, the dynamic entity table is updated from predicted poses, incoming actions, and the previous world state. The static scene memory remains shared across all agents and provides environmental context for pose transition and rendering. Agents are stored as entries in a dynamic table rather than as fixed view slots. Thus, adding a new agent requires initializing an entity state from its initial observation, appending the state to the dynamic entity table, and issuing an additional rendering query.

The STBoard therefore serves two roles. First, it provides a persistent communication channel through which the effects of one agent’s action can influence the observations of other agents. Second, it separates world-state maintenance from camera layout, allowing multiple views to query the same evolving state without directly communicating inside the renderer.

4.2 Action-conditioned World Evolution

In a physically consistent world simulator, interactions among agents should be reflected in the world state before visual observations are synthesized. Predicting future images directly from image history places unnecessary burden on the renderer and makes interaction modeling difficult. Khora instead models pose and state evolution explicitly before querying the visual renderer.

For each agent, we first obtain a deterministic motion proposal from the action controls:

$$\hat{p}_{t+1}^i = K(p_t^i, a_t^i), \quad (4)$$

where K encodes the known action-to-motion relationship. A learned residual corrects this proposal using the shared world state and static scene context:

$$p_{t+1}^i = \hat{p}_{t+1}^i + \Delta(S_t, G, \hat{p}_{t+1}^i, a_t^i). \quad (5)$$

The predicted poses and actions are then used to update the shared dynamic state:

$$S_{t+1} = U(S_t, \{p_{t+1}^i, a_t^i\}_{i \in I_t}; G). \quad (6)$$

The explicit kinematic model captures deterministic motion induced by agent controls, such as forward movement or camera rotation. The learned residual accounts for environment-dependent

dynamics that cannot be inferred from actions alone. Examples include collisions, terrain-dependent motion, and interactions with other agents. Since these effects depend on the current world state rather than the control signal itself, modeling them as residual updates allows the transition model to incorporate rich environmental context while preserving the efficiency and interpretability of explicit action modeling.

4.3 Geometry-guided View Synthesis

The STBoard explicitly maintains the 3D states of all agents in a shared coordinate system. Instead of requiring the renderer to infer the spatial configuration of multiple agents from latent tokens, Khora directly projects their world coordinates into the target view using camera geometry. This transforms explicit geometric information into rendering conditions, thus reducing the burden of the renderer.

For a target agent i , each potentially visible entity e_{t+1}^j is projected into the image plane using the predicted target camera pose:

$$u_{t+1}^{i,j} = \Pi(p_{t+1}^i, p_{t+1}^j), \quad (7)$$

where Π denotes the camera projection operator and $u_{t+1}^{i,j}$ is the projected image coordinate of entity j in the target view i .

The projected entity set is rasterized into a fixed-resolution spatial conditioning map

$$C_{t+1}^i \in \mathbb{R}^{H \times W \times C}. \quad (8)$$

Its channels encode structured attributes such as occupancy, depth, orientation, identity, and task-dependent entity state. Depth-aware rasterization resolves overlapping projections and suppresses entities that are occluded by scene geometry. The resulting representation has a fixed spatial shape regardless of the number of active agents.

The target observation is synthesized by a shared neural renderer conditioned on the updated world state, predicted camera pose, static scene context, and rasterized entity map:

$$o_{t+1}^i = R(S_{t+1}, C_{t+1}^i, p_{t+1}^i; G). \quad (9)$$

The same renderer is reused for every target agent, and additional views are generated by issuing additional independent queries rather than changing the renderer architecture.

Rendering complexity. Let N_a denote the number of active agents and N_v the number of requested views. In view-coupled generators, dense cross-agent communication inside the neural renderer can require $O(N_a N_v)$ expensive neural interactions, which becomes $O(N^2)$ when every agent receives a rendered view. Khora invokes the neural renderer independently for each requested view, resulting in $O(N_v)$ renderer evaluations. Constructing the conditioning maps still requires up to $O(N_a N_v)$ entity-to-view projections. However, the practical cost can be expressed as

$$T_{\text{Khora}} \approx N_v C_{\text{render}} + N_a N_v C_{\text{proj}}, \quad (10)$$

where $C_{\text{render}} \gg C_{\text{proj}}$. For up to 80 agents, projection and rasterization account for less than 8% of total inference latency.

4.4 Training Data and Inference Protocol

Having described the model architecture, we next summarize the supervision used for training and the information available during inference. Khora is trained on synchronized multi-agent trajectories containing aligned visual observations, actions, and structured state supervision. A coarse static world prior provides the shared coordinate frame and initializes the persistent scene representation. At inference time, only the world prior, initial observations, and subsequent actions are provided; future poses, dynamic states, and observations are generated autoregressively.

5 Experiments

We evaluate Khora from four complementary perspectives: visual quality, multi-agent consistency, computational scalability, and inference-time population dynamics. Visual quality measures whether

Table 2: **Quantitative comparison of visual quality.** All methods are evaluated using the same trajectories, resolution, and prediction horizon within each view setting. “—” indicates that the method does not support the corresponding multi-view setting.

Method	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	FID $\downarrow$	FVD $\downarrow$
<i>2 Views (10,000 cases / 20,000 videos)</i>					
Khora w/o Cross-Agent State	26.0427	0.7125	0.1826	8.7262	51.6657
Khora	26.2425	0.7130	0.1789	7.7098	36.5529
<i>4 Views (10,000 cases / 40,000 videos)</i>					
Khora w/o Cross-Agent State	24.9196	0.7017	0.1851	11.3010	73.4917
Khora	25.2825	0.7020	0.1772	7.9821	38.6849
<i>8 Views (159 cases / 1,272 videos)</i>					
Khora w/o Cross-Agent State	26.2282	0.7573	0.1544	30.1706	104.9082
Khora	26.5775	0.7573	0.1477	27.0311	53.6148

the per-view renderer preserves fidelity as the number of agents increases. Multi-agent consistency examines whether synchronized views agree on action consequences and the underlying world state. Scalability measures the runtime contribution of each system component and compares real-time performance with representative baselines. Finally, dynamic-population evaluation tests whether agents can enter and leave a shared rollout without retraining or architectural modification.

To isolate the contribution of cross-agent information in the STBoard, we introduce an agent-isolated variant, denoted as **Khora w/o Cross-Agent State**. For each target agent, this variant retains the static scene memory, the target agent’s own pose and action history, and the same neural renderer, but masks all dynamic entity entries associated with other agents. Consequently, different views no longer exchange dynamic information through the STBoard. All other model components, training data, renderer parameters, and inference settings remain unchanged.

5.1 Visual Quality

We first evaluate the fidelity of the generated observations using both paired reconstruction metrics and distribution-level video metrics. PSNR and SSIM measure pixel-level reconstruction quality, while LPIPS evaluates perceptual similarity in a learned feature space. We additionally report FID over generated frames and FVD over generated video clips to measure image- and video-level distribution alignment. PSNR and SSIM are better when higher, whereas LPIPS, FID, and FVD are better when lower.

All methods are evaluated on the same held-out trajectories, output resolution, prediction horizon, and agent-view queries. Paired metrics are computed between generated observations and their synchronized ground-truth frames and are averaged over agents and timesteps. FID is computed over individual frames, while FVD is computed over complete generated clips. We use one generated rollout per conditioning sequence and do not perform best-of- K sample selection.

Table 2 summarizes the quantitative results. Khora preserves competitive per-view visual quality while using a rendering interface whose dimensionality does not depend on the number of active agents. This indicates that moving cross-agent interaction from the neural renderer to the shared world state does not require sacrificing visual fidelity.

5.2 Multi-Agent Consistency

Visual quality alone does not determine whether multiple observations describe the same evolving world. We therefore conduct a user study that separately evaluates *cross-view action consistency* and *cross-view world consistency*. Participants are shown synchronized multi-view clips in randomized and anonymized order and rate each sample on a five-point scale. Each clip is evaluated by multiple participants, and we report the mean score together with a 95% confidence interval.

Cross-view action consistency. This criterion measures whether an action performed by one agent produces temporally and spatially compatible consequences in other agents’ observations. Participants consider whether actor motion, interaction events, and action timing agree across

Table 3: **User study of multi-agent consistency.** Scores use a five-point scale, where higher is better.

Method	Action consistency $\uparrow$	World consistency $\uparrow$	Overall $\uparrow$
Khora w/o Cross-Agent State	2.0750	4.0650	3.0700
Khora	3.5700	4.2133	3.8917

synchronized viewpoints. A low score indicates failures such as an entity moving in one view but remaining stationary in another, or the same event occurring at inconsistent times.

Cross-view world consistency. This criterion measures whether synchronized views agree on the persistent state of the environment. Participants evaluate entity identity, relative position, orientation, vertical relationships, visibility, and persistence through occlusion. A low score indicates contradictions such as identity changes, inconsistent geometry, incorrect relative height, or entities disappearing from the shared world.

As shown in Table 3, Khora receives higher ratings for both action consistency and world consistency. The improvement in action consistency reflects the use of a common action-conditioned state update, while the improvement in world consistency reflects the persistent entity representation maintained by the STBoard.

5.3 Scalability and Real-Time Performance

Figure 5 shows the runtime scaling behavior as the agent population increases. Within the measured range from 1 to 80 agents, the total compute latency increases only modestly from 107.16 ms to 116.73 ms per rollout step. View synthesis remains the dominant component, while the additional cost of STBoard update and geometric projection grows gradually with population size. Consequently, per-view FPS decreases only slightly from 37.33 to 34.27.

Importantly, peak VRAM usage per GPU remains nearly constant as the population grows, since each view is rendered through the same agent-independent interface. With additional views distributed across GPUs, aggregate throughput therefore increases from 37.3 view-fps for one agent to 2741.3 view-fps for 80 agents. The projected curves illustrate how the lightweight state-update and projection costs may become increasingly visible at substantially larger populations, while avoiding population-dependent growth in the neural renderer itself.

5.4 Inference-Time Dynamic Population

We finally evaluate whether the active population can change within a single autoregressive rollout. The sequence begins with two agents. Additional agents are introduced at later timesteps, increasing the population from two to four and then from four to eight. Several agents subsequently leave the environment, reducing the population again. Newly introduced agents are initialized from their first observations and appended to the dynamic entity table, while departing agents are removed from the active set. The transition model, STBoard interface, and neural renderer remain unchanged throughout the rollout.

Figure 6 illustrates that Khora can accommodate population changes during an ongoing simulation. Existing agents preserve their trajectories and observations when new agents are introduced, while the new agents become visible from relevant viewpoints after being inserted into the shared state. This experiment demonstrates that population scalability is an inference-time capability rather than a fixed configuration determined during training.

6 Limitations

Despite encouraging results, several limitations remain.

First, Khora currently assumes access to a coarse static prior for each environment. Although the point-cloud representation is incomplete and cannot directly provide the rendered appearance, it supplies the global coordinate scaffold used for pose prediction, projection, and visibility reasoning.

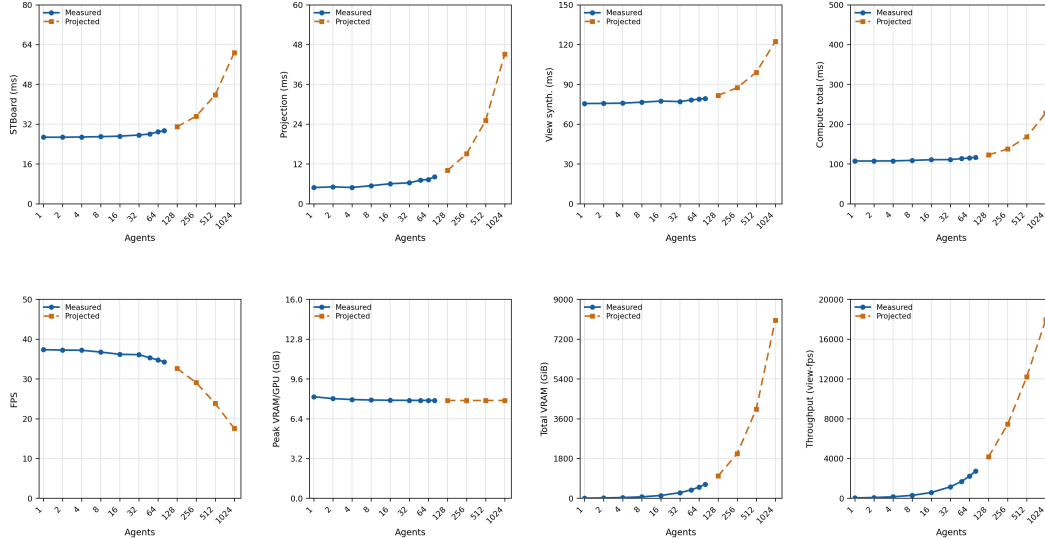


Figure 5: **Runtime scalability with increasing agent population.** Solid blue curves show measurements for up to 80 agents, while dashed orange curves show projected trends at larger populations. We report the latency of STBoard update, geometric projection, and view synthesis, together with total compute latency, per-view FPS, peak VRAM per GPU, total VRAM, and aggregate rendering throughput. Each requested view is synthesized on its respective GPU.

The static scene memory is currently adapted to each map, and the system therefore does not yet support zero-shot deployment in a completely unseen environment.

Second, the current evaluation is primarily qualitative and is intended to demonstrate the behavior of the system rather than establish a comprehensive benchmark. More systematic measurements of cross-view consistency, long-horizon drift, runtime, and memory usage would provide a more complete characterization.

Finally, population scalability is also not equivalent to zero-cost generation. Khora removes quadratic interaction from the expensive neural renderer, but each requested view still requires a rendering pass, and entity projection has a worst-case $O(N_a N_v)$ cost. This geometric cost is negligible in the current operating range but may become relevant for substantially larger populations in online generation.

7 Conclusion

We introduced Khora, a scalable multi-agent world model that decouples shared world-state evolution from view-conditioned rendering. Instead of organizing generation around a fixed set of observers, Khora models a persistent shared world that can be queried independently by dynamically varying agents. This population-agnostic rendering interface enables inference-time expansion without modifying or retraining the renderer, while the dominant neural rendering cost scales linearly with the number of queried views.

Beyond the proposed architecture, we argue that multi-agent world models should be evaluated as persistent shared worlds rather than collections of synchronized video streams. This perspective shifts the focus from cross-view generation toward maintaining a coherent world state that supports concurrent interaction, viewpoint consistency, and scalable simulation.

We hope this work provides a practical step toward scalable open-world simulators capable of supporting large numbers of interacting agents, and encourages future research on world-centric representations, persistent memory, and scalable interactive world models.

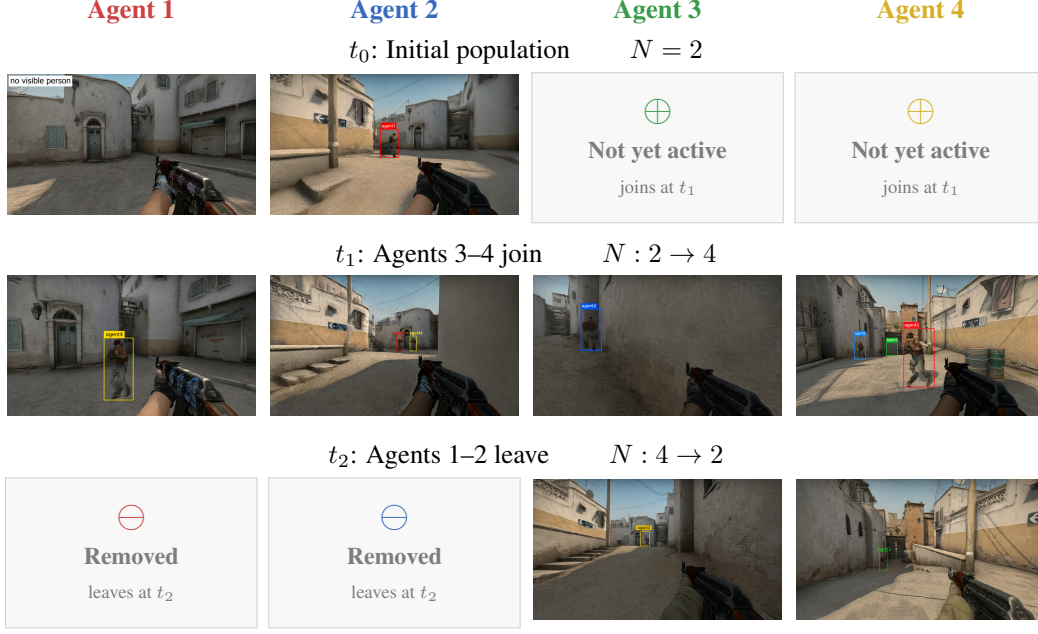


Figure 6: **Dynamic population within a single rollout.** The rollout starts with Agents 1 and 2 at t_0 . At t_1 , Agents 3 and 4 are initialized and inserted into the STBoard, expanding the active population from two to four. At t_2 , Agents 1 and 2 are removed, while Agents 3 and 4 continue from the same evolving shared world state. Agent insertion and removal require neither retraining nor modification of the transition or rendering architecture.

References

- Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari, 2024. URL <https://arxiv.org/abs/2405.12399>.
- Jake Bruce, Michael D. Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. *arXiv preprint arXiv:2402.15391*, 2024.
- Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.
- Shenyuan Gao, Siyuan Zhou, Yilun Du, Jun Zhang, and Chuang Gan. Adaworld: Learning adaptable world models with latent actions, 2025. URL <https://arxiv.org/abs/2503.18938>.
- David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.
- Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.
- Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium, 2018. URL <https://arxiv.org/abs/1706.08500>.
- Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.
- Anthony Hu, Václav Volhejn, Adrien Ramanana Rahary, Chris Mulder, Aditya Makkar, Amélie Royer, Manu Orsini, Alyx Liao, Adam Jelley, Eloi Alonso, Florian Laurent, Fredrik Norén, James Swingos, Jan Hünemann, Kent Rollins, Lucas Hosseini, Matthieu Le Cauchois, Maxim Peter,

- Pim de Witte, Tim Brown, Vincent Micheli, Moritz Böhle, Gabriel de Marmiesse, Viktoriia Sharmanska, Lucia Specia, Michael Black, and Patrick Pérez. Multiplayer interactive world models with representation autoencoders. *arXiv preprint arXiv:2607.05352*, 2026. URL <https://arxiv.org/abs/2607.05352>.
- Siqiao Huang, Jialong Wu, Qixing Zhou, Shangchen Miao, and Mingsheng Long. Vid2world: Crafting video diffusion models to interactive world models, 2026. URL <https://arxiv.org/abs/2505.14357>.
- Fangfu Liu, Kai He, Tianchang Shen, Tianshi Cao, Sanja Fidler, Yueqi Duan, Jun Gao, Igor Gilitschenki, Zian Wang, and Xuanchi Ren. Gamma-World: Generative multi-agent world modeling beyond two players. *arXiv preprint arXiv:2605.28816*, 2026. URL <https://arxiv.org/abs/2605.28816>.
- Odyssey. Agora-1: The multi-agent world model. <https://odyssey.ml/introducing-agora-1>, 2026. Accessed 2026-07-09.
- William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.
- Ryan Po, David Junhao Zhang, Amir Hertz, Gordon Wetzstein, Neal Wadhwa, and Nataniel Ruiz. Multigen: Level-design for editable multiplayer worlds in diffusion game engines. *arXiv preprint arXiv:2603.06679*, 2026. URL <https://arxiv.org/abs/2603.06679>.
- Georgy Savva, Oscar Michel, Daohan Lu, Suppakit Waiwitlikhit, Timothy Meehan, Dhairya Mishra, Srivats Poddar, Jack Lu, and Saining Xie. Solaris: Building a multiplayer video world model in minecraft. *arXiv preprint arXiv:2602.22208*, 2026. URL <https://arxiv.org/abs/2602.22208>.
- Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges, 2019. URL <https://arxiv.org/abs/1812.01717>.
- Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. Drivedreamer: Towards real-world-driven world models for autonomous driving, 2023. URL <https://arxiv.org/abs/2309.09777>.
- Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.
- Haoyu Wu, Jiwen Yu, Yingtian Zou, and Xihui Liu. MultiWorld: Scalable multi-agent multi-view video world models. *arXiv preprint arXiv:2604.18564*, 2026. URL <https://arxiv.org/abs/2604.18564>.
- Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric, 2018. URL <https://arxiv.org/abs/1801.03924>.