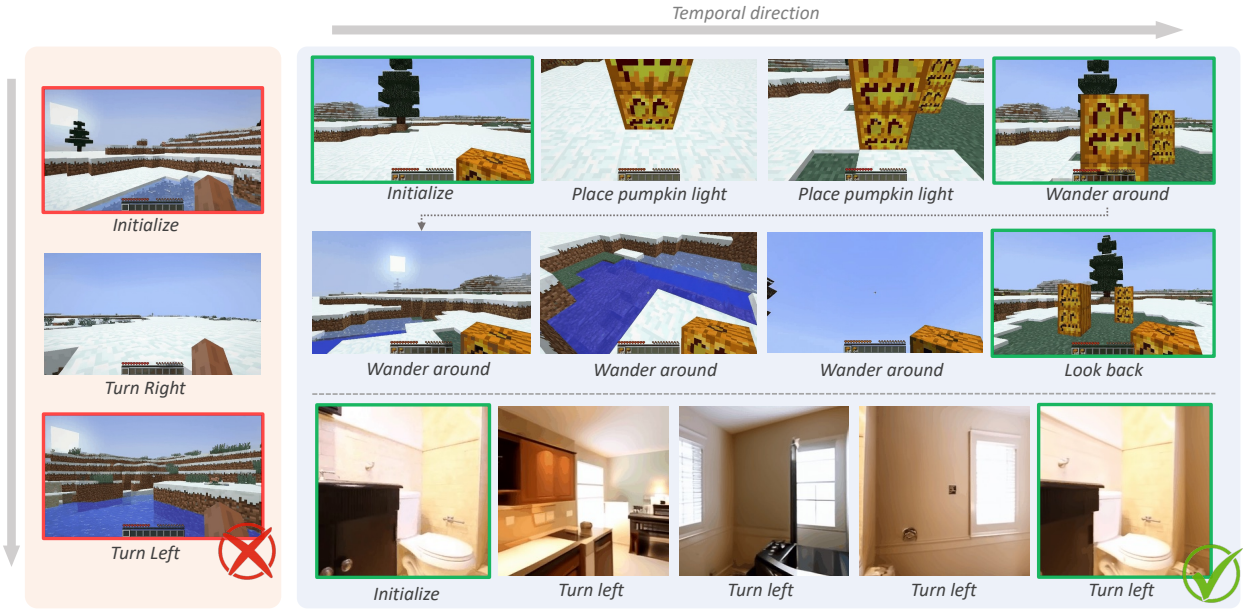
WORLDMEM: Long-term Consistent World Simulation with Memory

Zeqi Xiao[1] Yushi Lan[1] Yifan Zhou[1] Wenqi Ouyang[1],
Shuai Yang[2] Yanhong Zeng[3] Xingang Pan[1]

[1]S-Lab, Nanyang Technological University,
[2]Wangxuan Institute of Computer Technology, Peking University
[3]Shanghai AI Laboratory

{zeqi001, yushi001, yifan006, wenqi.ouyang, xingang.pan}@ntu.edu.sg

williamyang@pku.edu.cn, zengyh1900@gmail.com



(a) without Memory (b) with Memory

Figure 1. **WORLDMEM enables long-term consistent world simulation with an integrated memory mechanism.** (a) Previous world simulation methods typically face the problem of inconsistent world due to limited temporal context window size. (b) WORLDMEM empowers the agent to explore diverse and consistent worlds with an expansive action space, *e.g.*, crafting environments by placing objects like pumpkin light or freely roaming around. Most importantly, after exploring for a while and glancing back, we find the objects we placed are still there, with the inspiring sight of the light melting surrounding snow, testifying to the passage of time. Red and green boxes indicate scenes that should be consistent. Project page at https://xizaoqu.github.io/worldmem.

Abstract

*World simulation has gained increasing popularity due to its ability to model virtual environments and predict the consequences of actions. However, the limited temporal context window often leads to failures in maintaining long-term consistency, particularly in preserving 3D spatial consistency. In this work, we present WorldMem, a framework that enhances scene generation with a memory bank consisting of memory units that store memory frames and states (*e.g.*, poses and timestamps). By employing a memory attention mechanism that effectively extracts relevant information from these memory frames based on their states, our method is capable of accurately reconstructing previously observed scenes, even under significant viewpoint or temporal gaps. Furthermore, by incorporating timestamps into the states, our framework not only models a static world but also captures its dynamic evolution over time, enabling both perception and interaction within the simulated world. Extensive experiments in both virtual and real scenarios validate the effectiveness of our approach.*

1. Introduction

World simulation has gained significant attention for its ability to model environments and predict the outcomes of actions [1, 2, 5, 7, 27, 34]. Recent advances in video diffusion models have further propelled this field, enabling high-fidelity rollouts of potential future scenarios based on user actions, such as navigating through an environment or interacting with objects. These capabilities make world simulators particularly promising for applications in autonomous navigation [2, 7] and as an alternative to traditional game engines [5, 27].

Despite impressive progress, existing methods suffer from a key limitation: they struggle to maintain long-term consistency. A prominent issue is the lack of 3D spatial consistency: As illustrated in Figure 1(a), when the viewpoint moves in one direction and then returns, the content of the environment has changed. This inconsistency arises because video diffusion models operate within limited temporal windows, restricting their ability to retain information over extended time periods. Ideally, the model should be able to recall the generated environment and past events even across long temporal gaps. However, simply increasing the temporal window is impractical due to the prohibitive memory and computational costs.

To address this limitation, we introduce WORLDMEM, a novel framework that enhances long-term consistency in video-based world simulators through a dedicated memory mechanism. Our core idea is to *continuously store visual and state information in an external memory bank and retrieve the most relevant memories for generating new frames.* By removing the constraint of a bounded time horizon, this memory mechanism allows the model to capture and reuse historical cues, ensuring better preservation of environment details and past events.

WORLDMEM is built upon a Conditional Diffusion Transformer (CDiT) [28] that integrates external action signals to guide first-person viewpoint generation. By leveraging Diffusion Forcing (DF) [3] during training, the model supports autoregressive generation, enabling extended temporal simulation. Unlike previous methods that store past content either as explicit 3D reconstructions [24, 36] or within implicit network weights [1, 17], we instead maintain a *memory bank* composed of memory units – each containing a past frame and its corresponding state. This approach offers both flexibility and precision: We avoid rigid 3D reconstructions while retaining the fine-grained visual details necessary for accurate scene synthesis. The memory bank is continuously updated throughout generation, and a dedicated retrieval strategy effectively retrieves relevant units to guide future generation steps.

To effectively incorporate the retrieved memory units, we adopt the flexible DF paradigm, treating memory frames as "clear" latents alongside noisy latents in the denoising loop. This design obviates the need for additional conditioning techniques [15, 43, 48] and prevents information loss that may occur when passing through separate conditioning modules. Following [8, 38], we model information exchange between memory frames and the frames currently being generated through attention [35], where the current frames act as queries and the memory frames serve as keys and values. To ensure robust information retrieval, we represent states with specialized embeddings that enrich both queries and keys. Specifically, we employ Plücker embeddings [31] for dense pose representation and incorporate relative embeddings to facilitate efficient learning.

We evaluate WORLDMEM on a customized Minecraft benchmark [6] and on RealEstate10K [51]. The Minecraft benchmark includes diverse terrains (*e.g.*, plains, savannas, and deserts) and various action modalities (movement, viewpoint control, and event triggers). Extensive experiments show that WORLDMEM significantly improves 3D spatial consistency, enabling robust viewpoint reasoning and high-fidelity scene reconstructions, as shown in Figure 1(b). Furthermore, in dynamic environments, WORLDMEM accurately tracks and follows evolving events and environment changes, demonstrating its ability to both perceive and interact with the generated world.

Overall, WORLDMEM's enhanced capacity to maintain long-term consistency represents a critical advancement for interactive world simulators, paving the way for more accurate, persistent, and immersive virtual environments. We hope our promising results will inspire future research on memory-based consistent world simulation.

2. Related Work

Video diffusion model. With the rapid advancement of diffusion models [3, 28, 33], video generation has made significant strides [4, 9, 20, 26, 37, 39, 44]. The field has evolved from traditional U-Net-based architectures [4, 9, 37] to Transformer-based frameworks [25, 26, 50], enabling video diffusion models to generate highly realistic and temporally coherent videos. Recently, autoregressive video generation [3, 15, 21] has emerged as a promising approach to extend video length, theoretically indefinitely. Notably, Diffusion Forcing [3] introduces a per-frame noise-level denoising paradigm. Unlike the full-sequence paradigm, which applies a uniform noise level across all frames, per-frame noise-level denoising offers a more flexible approach, enabling autoregressive generation.

Interactive world simulation. World simulation aims to model an environment by predicting the next state given the current state and action. This concept has been extensively explored in the construction of world models [11] for agent learning [10, 12, 13, 18]. With advances in video generation, high-quality world simulation with robust control has become feasible, leading to numerous works focusing on in-

teractive world simulation [1, 2, 5, 7, 27, 34, 47]. These approaches enable agents to navigate generated environments and interact with them based on external commands.

However, due to context window limitations, such methods tend to forget previously generated content, leading to inconsistencies in the simulated world, particularly in maintaining 3D spatial coherence.

Consistent world simulation. Ensuring the consistency of a generated world is crucial for effective world simulation. Existing approaches can be broadly categorized into two types: explicit reconstructions and implicit conditioning.

One mainstream approach to ensuring consistency is to explicitly reconstruct the generated world into a 3D representation [8, 23, 24, 29, 36, 45, 46]. While this strategy can reliably maintain consistency, it imposes strict constraints on flexibility: Once the world is reconstructed, modifying or interacting with it becomes challenging.

Another research direction focuses on implicit learning. Methods like [1, 34] ensure consistency by overfitting to predefined scenarios (*e.g.*, specific CS:GO or DOOM maps), limiting scalability. StreamingT2V [15] maintains long-term consistency by continuing on both global and local visual contexts from previous frames, while SlowFastGen [17] progressively trains LoRA [19] modules for memory recall. However, these methods rely on abstract representations, making accurate scene reconstruction challenging. In contrast, our approach retrieves information from previously generated frames and their states, ensuring world consistency without overfitting to specific scenarios.

3. WORLDMEM

This section details the methodology of WORLDMEM. Sec. 3.1 introduces the relevant preliminaries, while Sec. 3.2 describes the interactive world simulator serving as our baseline. Sec. 3.3 presents the core of our proposed memory mechanism, followed by Sec. 3.4 covering the memory retrieval strategies and Sec. 3.5 discussing the key designs for state embeddings in memory blocks.

3.1. Preliminary

Video diffusion models. Video diffusion models generate video sequences by iteratively denoising Gaussian noise through a learned reverse process:

$$p_\theta(\mathbf{x}_t^{k-1}|\mathbf{x}_t^k) = \mathcal{N}(\mathbf{x}_t^{k-1}; \mu_\theta(\mathbf{x}_t^k, k), \sigma_k^2\mathbf{I}), \quad (1)$$

where all frames $(\mathbf{x}_t^k)_{1\le t\le T}$ share the same noise level k, and T is the context window length. This *full-sequence* approach enables global guidance but lacks flexibility in sequence length and autoregressive generation.

Autoregressive video generation. Autoregressive video generation aims to extend videos over the long term by predicting frames sequentially [22, 41]. While various methods exist for autoregressive generation, Diffusion Forcing (DF) [3] provides a neat and effective approach to achieve this. Specifically, DF introduces *per-frame noise levels* k_t:

$$p_\theta(\mathbf{x}_t^{k_t-1}|\mathbf{x}_t^{k_t}) = \mathcal{N}(\mathbf{x}_t^{k_t-1}; \mu_\theta(\mathbf{x}_t^{k_t}, k_t), \sigma_{k_t}^2\mathbf{I}), \quad (2)$$

Unlike full-sequence diffusion, DF generates video flexibly and stably beyond the training horizon. Autoregressive generation is a special case when only the last one or a few frames are noisy. With autoregressive video generation, long-term interactive world simulation becomes feasible.

3.2. Interactive World Simulation

Before introducing the memory mechanism, we first outline the construction of a world simulator that incorporates actions as additional inputs within a conditional diffusion transformer framework. Following previous works [5], we adopt a conditional diffusion transformer as the baseline for interactive world simulation. Specifically, we adopt a DiT [28] architecture for video generation and use DF [3] for autoregressive generation. As illustrated in Figure 2 (a), our model comprises multiple DiT blocks with spatial and temporal modules for spatiotemporal reasoning. In the temporal module, causal attention ensures each frame only attends to preceding frames.

To further support interactive world simulation, we integrate interactive modules that incorporate external control signals into the models. The primary control signal is action [5, 27, 47], which is crucial in guiding the simulation. In our Minecraft experiments, the action space consists of 25 dimensions, including movements (left, right, forward, back), view controls (look up, look down), and event triggers (object placement, item switching). To condition the model on these actions, we employ a multi-layer perceptron (MLP) to project the control signals into the embedding space. As shown in Figure 2 (c), we add the action embeddings and denoising timestep embedding together and inject the information to temporal blocks by AdaLN [42], following the paradigm of previous approaches [2, 5]. Please note that we also inject timestep embeddings to spatial blocks in the same way, but we do not highlight that in Figure 2 for simplicity. Likewise, default attention settings like residual connections, multi-head attention, and feedforward networks are also omitted in Figure 2 for simplicity.

With the combination of conditional DiT [28] and DF [3] as a baseline, we can effectively simulate a long-term interactive world. However, due to the high computational cost of video generation, the context window remains limited. Consequently, content outside this window is forgotten, leading to inconsistencies in long-term generation [5].

3.3. Condition on Memory

We introduce a *memory mechanism* to prevent the model from forgetting content outside the context window. Intuitively, this mechanism functions as a persistent external

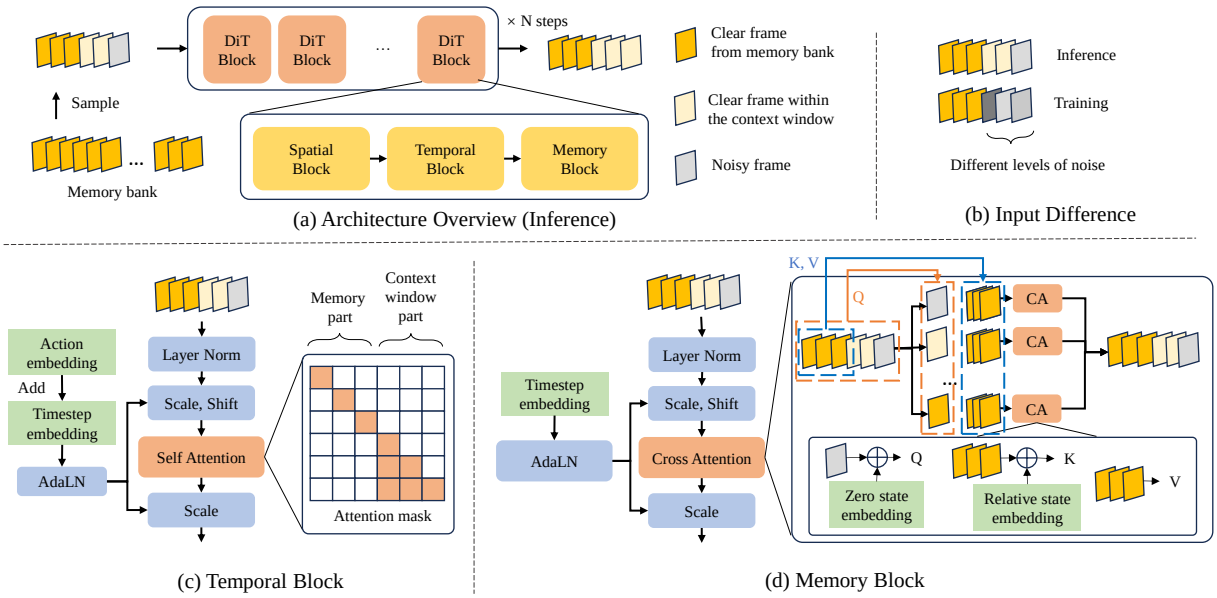


Figure 2. **Comprehensive overview of WORLDMEM.** The framework comprises a conditional diffusion transformer integrated with memory blocks, with a dedicated memory bank storing memory units from previously generated content. By retrieving these memory units from the memory bank and incorporating the information by memory blocks to guide generation, our approach ensures long-term consistency in world simulation.

buffer, allowing the model to revisit past frames and maintain a continuous record of the evolving scene. Specifically, we design a *memory bank* that continuously stores previously generated results. The memory is structured as a series of *memory units* $\mathbf{M} = \{\mathbf{I}, \mathbf{p}, \mathbf{t}\}$, which include visual content in the form of memory frames $\mathbf{I}$, along with state information such as poses $\mathbf{p}$ and timestamps $\mathbf{t}$.

However, effectively utilizing these memory units to guide generation is non-trivial. Unlike existing conditioning methods [15, 17], which condition on high-level and abstract features, our objective is to accurately reconstruct previously seen scenarios, even under significant viewpoint changes or scene changes. This requires the conditioning module to reason about the relationships between past results and the current generation.

To achieve this, we adopt the cross-attention mechanism [35] to guide information exchange between the denoising frames and the memory frames. Let $\mathbf{X} \in \mathbb{R}^{l_1 \times d}$ be the denoising frames (queries) and $\mathbf{M} \in \mathbb{R}^{l_2 \times d}$ be the memory frames (keys and values), where l_1 and l_2 are the flattened lengths of $\mathbf{X}$ and $\mathbf{M}$, respectively, and d is the channel dimension. We first incorporate the state embeddings $\mathbf{E}_X$ and $\mathbf{E}_M$ by summation:

$$\tilde{\mathbf{X}} = \mathbf{X} + \mathbf{E}_X, \quad \tilde{\mathbf{M}} = \mathbf{M} + \mathbf{E}_M. \tag{3}$$

We then compute cross-attention as:

$$\mathbf{X}' = \text{CrossAttn}(\mathrm{Q} = p_q(\tilde{\mathbf{X}}), \mathrm{K} = p_k(\tilde{\mathbf{M}}), \mathrm{V} = p_v(\mathbf{M})), \tag{4}$$

where $\mathbf{X}'$ is the output after attending to relevant information in the memory frames, p_q, p_k, and p_v are attention projectors. This allows the network to focus on memory content that is most pertinent to the current generation step. We will further elaborate on how $\mathbf{E}_X$ and $\mathbf{E}_M$ are computed in Sec. 3.5.

To incorporate memory units into different DiT blocks, we leverage the advantages of DF by treating memory frames as clear frames and feeding them into the diffusion pipeline alongside input noisy frames, as shown in Figure 2 (b). For the training process, all frames in the temporal context window have different levels of noise, and these memory frames have a clear level of noise (indicated as k_{min}):

$$\mathbf{x}_i^k, k \begin{cases} \sim \mathcal{P}(\mathbf{k}), & i \in \mathbf{TF} \\ = k_{\text{min}}, & i \in \mathbf{MF} \end{cases} \tag{5}$$

where $\mathbf{x}_i^k$ indicates the i^{th} frame with noise level k, $\mathbf{k}$ is the set of possible noise levels, $\mathbf{TF}$ is the set of frame indices within temporal context window, $\mathbf{MF}$ is the set of frame indices in the memory frame part. For inference,

$$\mathbf{x}_i^k, k = \begin{cases} k_{\text{max}}, & i \in \mathbf{GF} \\ k_{\text{min}}, & \text{otherwise} \end{cases} \tag{6}$$

where $\mathbf{GF}$ is the set of indices of generated frames, k_{max} indicates the max level of noise.

We ensure that these memory frames influence only the memory block by masking them in the temporal block. As

Algorithm 1: Memory Retrieval Algorithm

Input:
- A memory bank of n historical states $\{(\mathbf{I}_i, \mathbf{p}_i, t_i)\}_{i=1}^{n}$.
- Current state $(\mathbf{I}_c, \mathbf{p}_c, t_c)$.
- Memory condition length L.
- Similarity threshold tr.
- Weights w_o, w_t.

Output: A list of selected state indices S.

begin
  Δ **Compute Confidence Score**
  Compute FOV overlap ratio $\mathbf{o}$ with the current state via Monte Carlo sampling.
  Compute time difference $\mathbf{d} = \text{Concat}(\{|t_i - t_c|)\}_{i=1}^{n}$.
  Compute confidence $\boldsymbol{\alpha} = \mathbf{o} \cdot w_o$ - $\mathbf{d} \cdot w_t$.
  Δ **Selection with Similarity Filtering**
  Initialize $S = \varnothing$.
  for $m = 1$ **to** L **do**
    Select the state i^* with the highest α_{i^*}.
    Append i^* to S.
    Remove all states j whose feature similarity with i^* exceeds tr.
  end
  Return S.
end

shown in Figure 2 (c), given the memory frame length as L_M, the mask can be noted as:

$$A_{\text{mask}}(i,j) = \begin{cases} 1, & \text{if } i \le L_M \text{ and } j = i \\ 1, & \text{if } i > L_M \text{ and } j \le i \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

Within the memory block, we use cross-attention to retrieve information from memory frames and guide the generation of the current frame.

3.4. Memory Retrieve

Since the number of memory frames available for conditioning is limited, an efficient strategy is required to sample memory units from the memory bank. We adopt a greedy matching algorithm based on frame-pair similarity, where similarity is defined using the field-of-view (FOV) overlap ratio and timestamp differences as confidence measures. Algorithm 1 presents our approach to memory retrieval. Additionally, we apply a similarity filter to eliminate redundant units, ensuring that only the most relevant frames are retrieved for reference in the current generation. Figure 3 visualizes the overlapping metric with the Monte Carlo sampling.

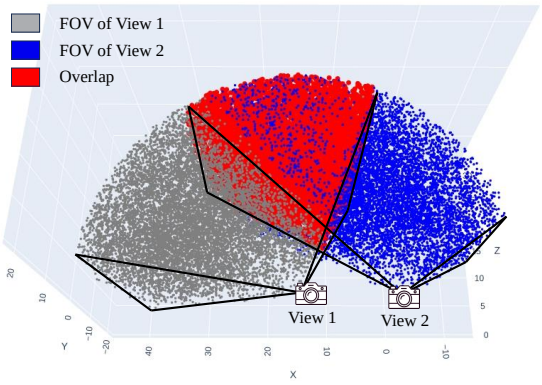


Figure 3. Two-view FOV overlapping visualization

3.5. State Embedding Design

The design of state embedding is critical for the memory block to retrieve valid information effectively.

A key aspect of this process is capturing spatial information, which primarily depends on pose embeddings. Inspired by [8, 14], which transforms frame-wise poses into dense positional embeddings to encode fine-grained spatial details, we adopt Plücker embedding [30] as our pose representation. For timestamp embeddings, a simple MLP-based mapping is sufficient. The state embedding $\mathbf{E}$ is then computed as the sum of pose and timestamp embeddings:

$$\mathbf{E} = G_p(\text{PE}(\mathbf{p})) + G_t(t), \tag{8}$$

where G_p and G_t are MLP layers mapping vectors to embeddings, and PE denotes the Plücker embedding function, which maps $\mathbf{p} \in \mathbb{R}^5$ (x, y, z, pitch, yaw) to $\mathbf{p} \in \mathbb{R}^{h \times w \times 6}$.

Additionally, we observe that relative embeddings are more effective than absolute embeddings for spatial reasoning, as they significantly reduce the model's learning complexity. To leverage this, we introduce relative embedding in the memory attention block, where the Query embedding is always set to zero, and the Key embedding is derived from the relative pose. To integrate relative embeddings into our architecture, we separate the cross-attention into frame-wise querying, where input frames attend to memory frames independently. This design is illustrated in Figure 2 (d).

4. Experiments

Datasets. We use MineDojo [6] to create diverse training and evaluation datasets in Minecraft, configuring diverse environments (*e.g.*, plains, savannas, ice plains, and deserts), agent actions, and interactions. For real-world scenes, we utilize RealEstate10K [51] with camera pose annotations, designing trajectories that incorporate past scenes to evaluate long-term world consistency.

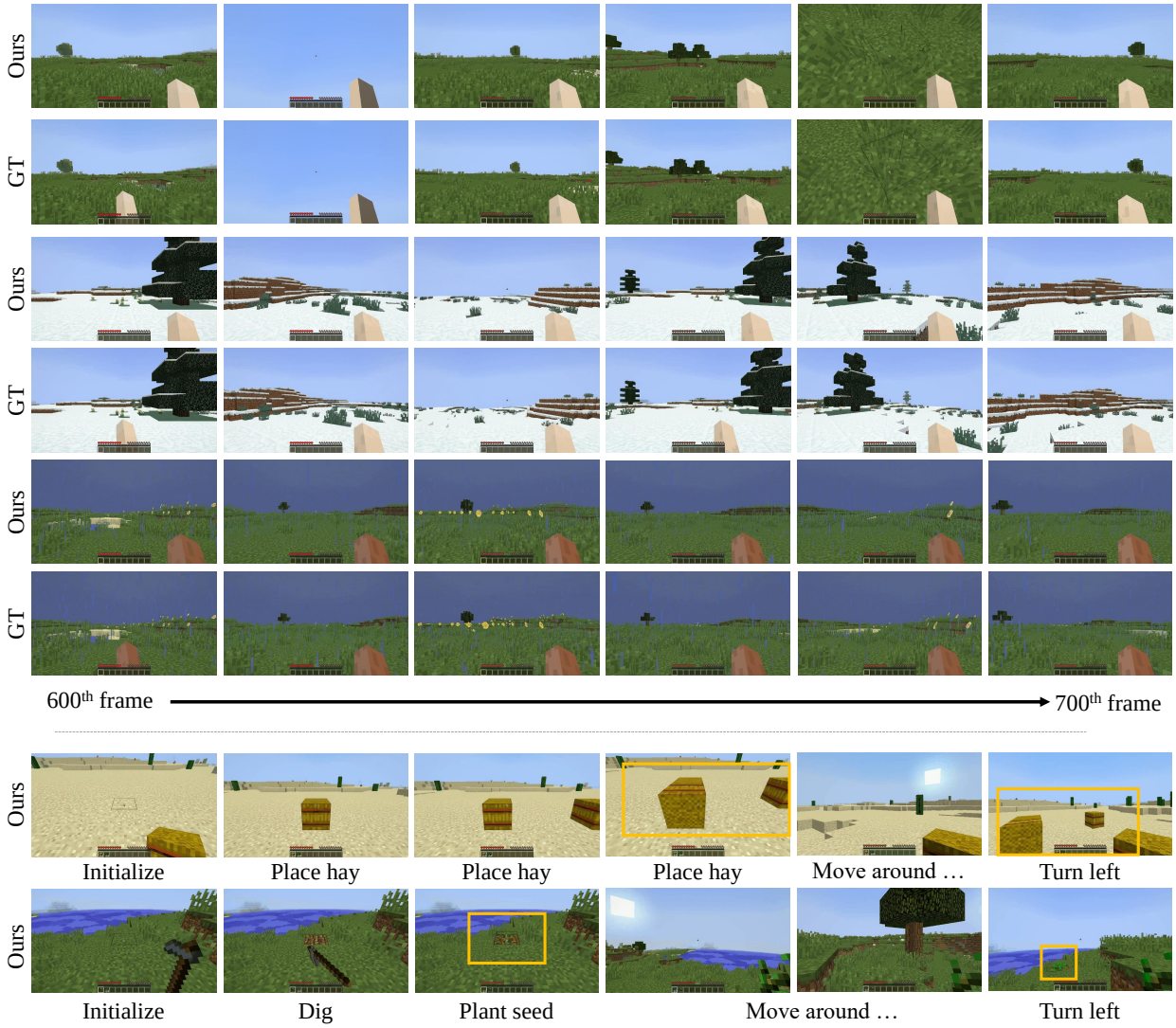


Figure 4. **Qualitative results.** We showcase WORLDMEM's capabilities through two sets of examples. **Top**: A comparison with Ground Truth (GT). WORLDMEM accurately models diverse dynamics (e.g., rain) by conditioning on 600 past frames, ensuring temporal consistency. **Bottom**: Interaction with the world. Objects like hay in the desert or wheat in the plains persist over time, with wheat visibly growing. For the best experience, see the supplementary videos.

Metrics. For quantitative evaluation, we employ reconstruction metrics, where the method of obtaining ground truth (GT) varies by specific settings. We then assess the consistency and quality of the generated videos using PSNR, LPIPS [49], and reconstruction FID (rFID) [16], which collectively measure pixel-level fidelity, perceptual similarity, and overall realism.

4.1. Results on Generation Benchmark

Comparisons on Minecraft Benchmark. We compare our approach with a standard full-sequence training method [14, 40] and Diffusion Forcing (DF) [3]. The key differences are as follows: the full-sequence conditional diffusion transformer [28] maintains the same noise level during training and inference, DF introduces different noise levels for training and inference, and our method incorporates a memory mechanism. To assess both short-term and long-term world consistency, we conduct evaluations within and beyond the context window. We evaluate both settings on 300 test videos and use 4,800 frames to compute rFID. In the following experiments, the agent's poses are generated by the game simulator as ground truth. However, in real-world scenarios, only the action input is available, and the pose is not directly observable. In such cases, the next-frame pose can be predicted based on the previous scenes, past states, and the upcoming action. We explore this design choice in the supplementary material.

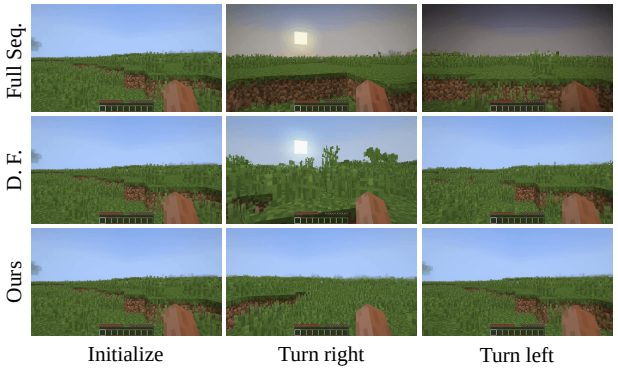


Figure 5. **Within context window evaluation examples.** It illustrates an example where the motion sequence first involves turning right and then returning to the original position, demonstrating methods' ability to maintain self-contained consistency.

Within context window. For this experiment, all methods use a context window of 16, while our approach additionally maintains a memory window of 8. We test on customized motion scenarios (*e.g.*, turn left, then turn right or move forward, then backward) to assess self-contained consistency, where the ground truth consists of previously generated frames at the same positions. As shown in Table 1 and Figure 5, the full-sequence baseline suffers from inconsistencies even within its own context window. DF improves consistency by enabling greater information exchange among generated frames. Our memory-based approach achieves the best performance, demonstrating the effectiveness of integrating a dedicated memory mechanism.

Beyond context window. In this setting, all methods use a context window of 8 and generate 100 future frames; our method further employs a memory window of 8 while initializing a 600-frame memory bank. We compute the reconstruction error using the subsequent 100 ground truth frames after 600 frames. Full-sequence methods can not roll out that long so we exclude it. DF exhibits poor PSNR and LPIPS scores, indicating severe inconsistency with the ground truth beyond the context window. Additionally, its low rFID suggests notable quality degradation. In contrast, our memory-augmented approach consistently outperforms others across all metrics, demonstrating superior long-term consistency and quality preservation. Figure 6 further substantiates these findings.

Comparisons on Real Scenarios. We further compare our approach with DFoT [32] on the RealEstate dataset [51]. Similar to DF [3], DFoT discards previously generated content that lies outside its context window and fails to maintain global world consistency.

To verify this, we currate 100 samples and test the 360-degree consistency of both DFoT and our method by rotating the scene a full 360 degrees and checking whether it returns to the same position. We measure the quality of reconstruction by comparing the first frame and the final frame (after a full rotation). As shown in Table 2, our method outperforms DFoT in all metrics, indicating that our integrated memory mechanism substantially improves consistency. Qualitative comparisons in Figure 7 further support these findings.

Table 1. Evaluation on Minecraft benchmark [6]

Within context window			
Methods	PSNR ↑	LPIPS ↓	rFID ↓
Full Sequence	20.35	0.0691	13.87
Diffusion Forcing [3]	26.56	0.0094	13.88
Ours	**27.01**	**0.0072**	**13.73**
Beyond context window			
Methods	PSNR ↑	LPIPS ↓	rFID ↓
Full Sequence	/	/	/
Diffusion Forcing [3]	18.04	0.4376	51.28
Ours	**25.32**	**0.1429**	**15.37**

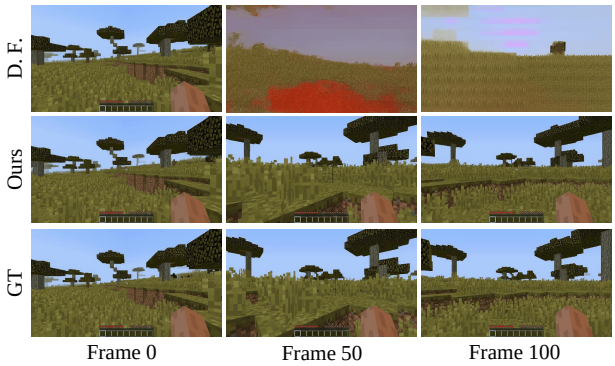


Figure 6. **Beyond context window evaluation examples.** It shows that Diffusion-Forcing suffers from inconsistency and quality degradation after generating a certain number of frames. In contrast, our method maintains high quality and faithfully reconstructs previously observed scenarios.

Table 2. Evaluation on RealEstate10K [51]

Methods	PSNR ↑	LPIPS ↓	rFID ↓
DFoT [32]	8.396	0.6676	156.74
Ours	**20.19**	**0.1773**	**67.14**

Qualitative results. Figure 4 showcases WORLDMEM's capabilities. The top section demonstrates its ability to operate in a free action space across diverse environments. Given a 600-frame memory bank, our model generates 100 future frames while preserving the ground truth's actions and poses, ensuring strong world consistency. The bottom section highlights dynamic environment interaction. By us-

Figure 7. **Examples on RealEstate [51].** DFoT [32] discards content beyond its context window, losing 360-degree consistency. In contrast, our method preserves details and accurately returns to the original location.

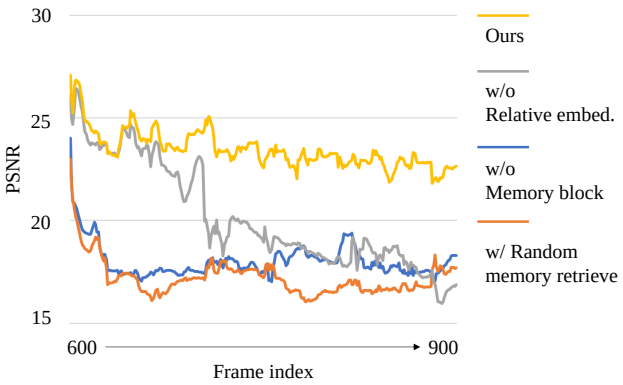


Figure 8. **Long-term Generation Comparison.** This figure presents the PSNR of different ablation methods compared to the ground truth over a 300-frame sequence. The results show that our method without memory blocks or using random memory retrieval exhibits immediate inconsistencies with the ground truth. Additionally, the model lacking relative embeddings begins to degrade significantly beyond 100 frames. In contrast, our full method maintains strong consistency even beyond 300 frames.

ing timestamps as embeddings, the model remembers environmental changes and captures natural event evolution, such as plant growth over time.

4.2. Ablation

Embedding designs. The design of embeddings within the memory block is crucial for effective cross-frame relationship modeling. We evaluate three strategies (Table 3): (1) sparse pose embedding with absolute encoding, (2) dense pose embedding with absolute encoding, and (3) dense pose embedding with relative encoding. Results show that dense pose embeddings (Plücker embedding) significantly enhance all metrics, emphasizing the benefits of richer pose representations. Switching from absolute to relative encoding further improves performance, particularly in LPIPS and rFID, by facilitating relationship reasoning and information retrieval. As illustrated in Figure 8, absolute embeddings accumulate errors over time, while relative embeddings maintain stability even beyond 300 frames.

Table 3. Ablation on embedding designs

Pose type	Embed. type	PSNR ↑	LPIPS ↓	rFID ↓
Sparse	Absolute	20.67	0.2887	39.23
Dense	Absolute	23.63	0.1830	29.34
Dense	Relative	**25.32**	**0.1429**	**15.37**

Figure 9. **Results w/o and w/ time condition.** Without timestamps, the model fails to differentiate memory units from the same location at different times, causing errors. With time conditioning, it aligns with the updated world state, ensuring consistency.

Table 4. Ablation on time condition

Time embedding	PSNR ↑	LPIPS ↓	rFID ↓
w/o	23.17	0.1989	23.89
w/	**25.12**	**0.1613**	**16.53**

Time condition. We ablate the effectiveness of timestamp embedding in Table 4. For this experiment, we curate 100 video samples featuring placing events and evaluate whether future generations align with event progression. As shown in the table, incorporating time embeddings significantly improves PSNR and LPIPS, indicating that adding temporal information helps the model faithfully reproduce event changes in world simulation. Since events like plant growth are inherently unpredictable, we do not conduct quantitative evaluations on such cases but instead provide qualitative illustrations in Figure 9.

Memory retrieve strategy. We analyze memory retrieval strategies in Table 5. Random sampling from the memory bank leads to poor performance and severe quality degradation, as evidenced by a sharp drop in rFID and rapid divergence from the ground truth (Figure 8). The confidence-based filtering significantly enhances consistency and generation quality. Additionally, we refine retrieval by filtering out redundant memory units based on similarity, further improving all evaluation metrics and demonstrating the effectiveness of our approach.

Table 5. Ablation on memory retrieve strategy

Strategy	PSNR ↑	LPIPS ↓	rFID ↓
Random	18.32	0.3224	47.35
+ Confidence Filter	23.12	0.1863	24.33
+ Similarity Filter	**25.32**	**0.1429**	**15.37**

5. Conclusion

In conclusion, WORLDMEM tackles the longstanding challenge of maintaining long-term consistency in world simulation by employing a memory bank of past frames and associated states. Its memory attention mechanism enables accurate reconstruction of previously observed scenes – even under large viewpoints or temporal gaps – and effectively models dynamic changes over time. Extensive experiments in both virtual and real settings confirm WORLDMEM's capacity for robust, immersive world simulation. We hope our work will encourage further research on the design and applications of memory-based world simulators.

References

[1] Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos J Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari. *Advances in Neural Information Processing Systems*, 37: 58757–58791, 2025. 2, 3

[2] Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models, 2024. 2, 3

[3] Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2025. 2, 3, 6, 7

[4] Haoxin Chen, Menghan Xia, Yingqing He, Yong Zhang, Xiaodong Cun, Shaoshu Yang, Jinbo Xing, Yaofang Liu, Qifeng Chen, Xintao Wang, et al. Videocrafter1: Open diffusion models for high-quality video generation. *arXiv preprint arXiv:2310.19512*, 2023. 2

[5] Decart, Julian Quevedo, Quinn McIntyre, Spruce Campbell, Xinlei Chen, and Robert Wachen. Oasis: A universe in a transformer. 2024. Project website. 2, 3, 12

[6] Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. *Advances in Neural Information Processing Systems*, 35: 18343–18362, 2022. 2, 5, 7, 12

[7] Ruili Feng, Han Zhang, Zhantao Yang, Jie Xiao, Zhilei Shu, Zhiheng Liu, Andy Zheng, Yukun Huang, Yu Liu, and Hongyang Zhang. The matrix: Infinite-horizon world generation with real-time moving control. *arXiv preprint arXiv:2412.03568*, 2024. 2, 3

[8] Ruiqi Gao, Aleksander Holynski, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul Srinivasan, Jonathan T Barron, and Ben Poole. Cat3d: Create anything in 3d with multi-view diffusion models. *arXiv preprint arXiv:2405.10314*, 2024. 2, 3, 5

[9] Yuwei Guo, Ceyuan Yang, Anyi Rao, Zhengyang Liang, Yaohui Wang, Yu Qiao, Maneesh Agrawala, Dahua Lin, and Bo Dai. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. *arXiv preprint arXiv:2307.04725*, 2023. 2

[10] David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. *Advances in neural information processing systems*, 31, 2018. 2

[11] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018. 2

[12] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019. 2

[13] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020. 2

[14] Hao He, Yinghao Xu, Yuwei Guo, Gordon Wetzstein, Bo Dai, Hongsheng Li, and Ceyuan Yang. Cameractrl: Enabling camera control for text-to-video generation. *arXiv preprint arXiv:2404.02101*, 2024. 5, 6

[15] Roberto Henschel, Levon Khachatryan, Daniil Hayrapetyan, Hayk Poghosyan, Vahram Tadevosyan, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. Streamingt2v: Consistent, dynamic, and extendable long video generation from text. *arXiv preprint arXiv:2403.14773*, 2024. 2, 3, 4

[16] 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. *Advances in neural information processing systems*, 30, 2017. 6

[17] Yining Hong, Beide Liu, Maxine Wu, Yuanhao Zhai, Kai-Wei Chang, Linjie Li, Kevin Lin, Chung-Ching Lin, Jianfeng Wang, Zhengyuan Yang, Ying Nian Wu, and Lijuan Wang Wang. Slowfast-vgen: Slow-fast learning for action-driven long video generation. *arXiv preprint arXiv:2410.23277*, 2024. 2, 3, 4

[18] Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023. 2

[19] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022. 3

[20] Yang Jin, Zhicheng Sun, Ningyuan Li, Kun Xu, Hao Jiang, Nan Zhuang, Quzhe Huang, Yang Song, Yadong Mu, and Zhouchen Lin. Pyramidal flow matching for efficient video generative modeling. *arXiv preprint arXiv:2410.05954*, 2024. 2

[21] Jihwan Kim, Junoh Kang, Jinyoung Choi, and Bohyung Han. FIFO-diffusion: Generating infinite videos from text without training. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 2

[22] Dan Kondratyuk, Lijun Yu, Xiuye Gu, José Lezama, Jonathan Huang, Grant Schindler, Rachel Hornung, Vighnesh Birodkar, Jimmy Yan, Ming-Chang Chiu, Krishna Somandepalli, Hassan Akbari, Yair Alon, Yong Cheng, Josh Dillon, Agrim Gupta, Meera Hahn, Anja Hauth, David Hendon, Alonso Martinez, David Minnen, Mikhail Sirotenko, Kihyuk Sohn, Xuan Yang, Hartwig Adam, Ming-Hsuan Yang, Irfan Essa, Huisheng Wang, David A. Ross, Bryan Seybold, and Lu Jiang. Videopoet: A large language model for zero-shot video generation, 2024. 3

[23] Hanwen Liang, Junli Cao, Vidit Goel, Guocheng Qian, Sergei Korolev, Demetri Terzopoulos, Konstantinos N Plataniotis, Sergey Tulyakov, and Jian Ren. Wonderland: Navigating 3d scenes from a single image. *arXiv preprint arXiv:2412.12091*, 2024. 3

[24] Fangfu Liu, Wenqiang Sun, Hanyang Wang, Yikai Wang, Haowen Sun, Junliang Ye, Jun Zhang, and Yueqi Duan. Reconx: Reconstruct any scene from sparse views with video diffusion model. *arXiv preprint arXiv:2408.16767*, 2024. 2, 3

[25] Xin Ma, Yaohui Wang, Gengyun Jia, Xinyuan Chen, Ziwei Liu, Yuan-Fang Li, Cunjian Chen, and Yu Qiao. Latte: Latent diffusion transformer for video generation. *arXiv preprint arXiv:2401.03048*, 2024. 2

[26] OpenAI. Video generation models as world simulators. https://openai.com/research/video-generation-models-as-world-simulators, 2024. 2

[27] Jack Parker-Holder, Philip Ball, Jake Bruce, Vibhavari Dasagi, Kristian Holsheimer, Christos Kaplanis, Alexandre Moufarek, Guy Scully, Jeremy Shar, Jimmy Shi, Stephen Spencer, Jessica Yung, Michael Dennis, Sultan Kenjeyev, Shangbang Long, Vlad Mnih, Harris Chan, Maxime Gazeau, Bonnie Li, Fabio Pardo, Luyu Wang, Lei Zhang, Frederic Besse, Tim Harley, Anna Mitenkova, Jane Wang, Jeff Clune, Demis Hassabis, Raia Hadsell, Adrian Bolton, Satinder Singh, and Tim Rocktäschel. Genie 2: A large-scale foundation world model. 2024. 2, 3

[28] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023. 2, 3, 6

[29] Xuanchi Ren, Tianchang Shen, Jiahui Huang, Huan Ling, Yifan Lu, Merlin Nimier-David, Thomas Müller, Alexander Keller, Sanja Fidler, and Jun Gao. Gen3c: 3d-informed world-consistent video generation with precise camera control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. 3

[30] Vincent Sitzmann, Semon Rezchikov, Bill Freeman, Josh Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. *Advances in Neural Information Processing Systems*, 34: 19313–19325, 2021. 5

[31] Vincent Sitzmann, Semon Rezchikov, William T. Freeman, Joshua B. Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. In *NeurIPS*, 2021. 2

[32] Kiwhan Song, Boyuan Chen, Max Simchowitz, Yilun Du, Russ Tedrake, and Vincent Sitzmann. History-guided video diffusion. *arXiv preprint arXiv:2502.06764*, 2025. 7, 8, 12

[33] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. 2

[34] Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter. Diffusion models are real-time game engines. *arXiv preprint arXiv:2408.14837*, 2024. 2, 3

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. 2, 4

[36] Hengyi Wang and Lourdes Agapito. 3d reconstruction with spatial memory. *arXiv preprint arXiv:2408.16061*, 2024. 2, 3

[37] Jiuniu Wang, Hangjie Yuan, Dayou Chen, Yingya Zhang, Xiang Wang, and Shiwei Zhang. Modelscope text-to-video technical report. *arXiv preprint arXiv:2308.06571*, 2023. 2

[38] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20697–20709, 2024. 2

[39] Yaohui Wang, Xinyuan Chen, Xin Ma, Shangchen Zhou, Ziqi Huang, Yi Wang, Ceyuan Yang, Yinan He, Jiashuo Yu, Peiqing Yang, et al. Lavie: High-quality video generation with cascaded latent diffusion models. *arXiv preprint arXiv:2309.15103*, 2023. 2

[40] Zhouxia Wang, Ziyang Yuan, Xintao Wang, Yaowei Li, Tianshui Chen, Menghan Xia, Ping Luo, and Ying Shan. Motionctrl: A unified and flexible motion controller for video generation. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 6

[41] Tong Wu, Zhihao Fan, Xiao Liu, Yeyun Gong, Yelong Shen, Jian Jiao, Hai-Tao Zheng, Juntao Li, Zhongyu Wei, Jian Guo, Nan Duan, and Weizhu Chen. Ar-diffusion: Auto-regressive diffusion model for text generation, 2023. 3

[42] Jingjing Xu, Xu Sun, Zhiyuan Zhang, Guangxiang Zhao, and Junyang Lin. Understanding and improving layer normalization. *Advances in neural information processing systems*, 32, 2019. 3

[43] Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models. *arXiv preprint arXiv:2308.06721*, 2023. 2

[44] Tianwei Yin, Qiang Zhang, Richard Zhang, William T Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast causal video generators. *arXiv preprint arXiv:2412.07772*, 2024. 2

[45] Hong-Xing Yu, Haoyi Duan, Charles Herrmann, William T Freeman, and Jiajun Wu. Wonderworld: Interactive 3d scene generation from a single image. *arXiv preprint arXiv:2406.09394*, 2024. 3

[46] Hong-Xing Yu, Haoyi Duan, Junhwa Hur, Kyle Sargent, Michael Rubinstein, William T Freeman, Forrester Cole, Deqing Sun, Noah Snavely, Jiajun Wu, et al. Wonderjourney:

Going from anywhere to everywhere. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6658–6667, 2024. 3

[47] Jiwen Yu, Yiran Qin, Xintao Wang, Pengfei Wan, Di Zhang, and Xihui Liu. Gamefactory: Creating new games with generative interactive videos. *arXiv preprint arXiv:2501.08325*, 2025. 3

[48] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3836–3847, 2023. 2

[49] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 6

[50] Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-sora: Democratizing efficient video production for all, 2024. 2

[51] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. In *SIGGRAPH*, 2018. 2, 5, 7, 8, 12

6. Supplementary Materials

6.1. Limitations

Despite the effectiveness of our approach, certain issues warrant further exploration. First, we cannot guarantee that that we can always retrieve all necessary information from memory bank In some corner cases (*e.g.*, when views are blocked by obstacles), relying solely on view overlap may be insufficient. Second, our current interaction with the environment lacks diversity and realism. In future work, we plan to extend our models to real-world scenarios with more realistic and varied interactions. Lastly, our memory design still entails linearly increasing memory usage, which may impose limitations when handling extremely long sequences.

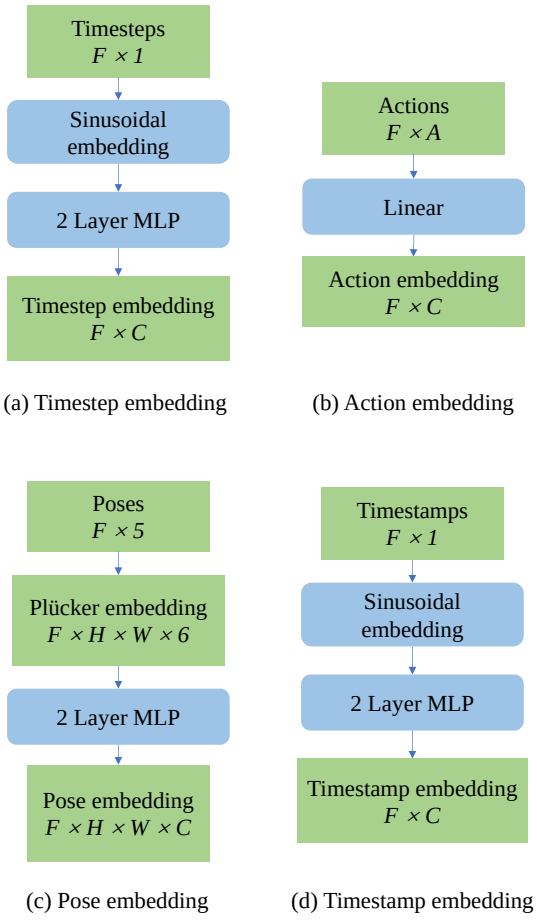
6.2. Details and Experiments

Embedding designs. We present the detailed designs of embeddings for timesteps, actions, poses, and timestamps in Figure 10, where F, C, H, W, A denote the frame number, channel count, height, width, and action count, respectively.

The input pose is parameterized by position (x, z, y) and orientation (pitch θ and yaw ϕ). The extrinsic matrix $\mathbf{E} \in \mathbb{R}^{4\times4}$ is formed as:

$$\mathbf{E} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}, \tag{9}$$

where $\mathbf{t} = (x, z, y)^T$ and $\mathbf{R} = \mathbf{R}_y(\phi)\mathbf{R}_x(\theta)$.



(a) Timestep embedding (b) Action embedding

(c) Pose embedding (d) Timestamp embedding

Figure 10. Illustration of different embeddings.

To encode camera pose, we adopt the Plücker embedding. Given a pixel (u, v) with normalized camera coordinates:

$$\mathbf{p}_c = \mathbf{K}^{-1}[u, v, 1]^T, \tag{10}$$

its world direction is:

$$\mathbf{d}_{u,v} = \mathbf{R}\mathbf{p}_c + \mathbf{t}. \tag{11}$$

Setting the camera center $\mathbf{o} = \mathbf{t}$, the Plücker embedding is:

$$\mathbf{p}_{u,v} = (\mathbf{o} \times \mathbf{d}_{u,v}, \mathbf{d}_{u,v}) \in \mathbb{R}^6. \tag{12}$$

For a frame of size $H \times W$, the full embedding is:

$$\mathbf{P}_i \in \mathbb{R}^{H\times W\times 6}. \tag{13}$$

Sampling strategy for training. We compare different sampling strategies during training in the Minecraft benchmark. Small-range sampling restricts memory conditioning to frames within 2m in the Minecraft world, while large-range sampling extends this range to 8m. Progressive sampling, on the other hand, begins with small-range samples

Table 6. Ablation on sampling strategy for training

Sampling strategy	PSNR ↑	LPIPS ↓	rFID ↓
Small-range	19.23	0.3786	46.55
Large-range	21.11	0.3855	42.96
Progressive	**25.32**	**0.1429**	**15.37**

Table 7. Ablation on length of memory context length

Length	PSNR ↑	LPIPS ↓	rFID ↓
1	22.18	0.1899	20.47
4	24.68	0.1568	16.54
8	**25.32**	**0.1429**	**15.37**
16	23.14	0.1687	18.33

for initial training steps and then gradually expands to large-range samples.

As shown in Table 6, both small-range and large-range sampling struggle with consistency and quality, whereas progressive sampling significantly improves all metrics. This suggests that gradually increasing difficulty during training helps the model learn to reason and effectively query information from memory blocks.

Memory context length. We evaluate how different memory context lengths affect performance in the Minecraft benchmark. Table 7 shows that increasing the context length from 1 to 8 steadily boosts PSNR, lowers LPIPS, and reduces rFID. However, extending the length to 16 deteriorates results, indicating that excessive memory frames may introduce noise or reduce retrieval precision. A context length of 8 provides the best trade-off, yielding the highest PSNR and the lowest LPIPS and rFID.

Pose prediction. For interactive play, ground truth poses are not accessible. To address this, we designed a lightweight pose prediction module that estimates the pose of the next frame. As illustrated in Figure 11, the predictor takes the previous image, the previous pose, and the upcoming action as inputs and outputs the predicted next pose. This module enables the system to operate using actions alone, eliminating the need for ground truth poses during inference. In Table 8, we compare the performance of using predicted poses versus ground truth poses. While using ground truth poses yields better results across all metrics, the performance drop with predicted poses is acceptable. This is because our method does not rely heavily on precise pose predictions – new frames are generated based on these predictions – and the ground truth poses generated by the Minecraft simulator also contain a certain degree of randomness.

Experimental details. For our experiments on Minecraft [6], we utilize the Oasis [5] and a base model. The model

Table 8. Comparison between using predicted poses and ground truth poses

Pose Type	PSNR ↑	LPIPS ↓	rFID ↓
Ground truth	25.32	0.1429	15.37
Predicted	23.13	0.1786	20.36

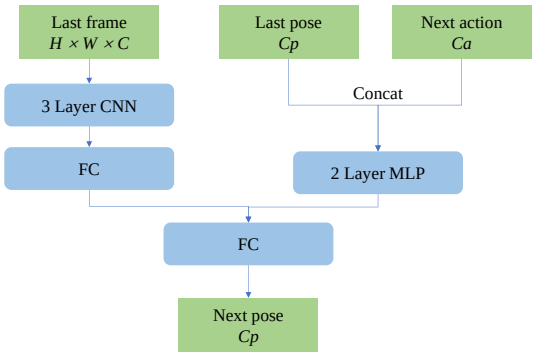


Figure 11. Structure of pose predictor.

is trained using the Adam optimizer with a fixed learning rate of 2×10^{-5}. Training is conducted at a resolution of 640×320, where frames are first encoded into a latent space via a VAE at a resolution of 32×18, then further patchified to 16×9. Our training dataset comprises approximately 20K long videos, each containing 1500 frames, sourced from [6]. During training, we employ an 8-frame temporal context window alongside an 8-frame memory window. The model is trained for approximately 200K steps using 8 GPUs, with a batch size of 2 per GPU. For the hyperparameters specified in Algorithm 1 of the main paper, we set the similarity threshold tr to 0.9, w_o to 1, and w_t to $0.2t_c$. For the noise levels in Eq. (5) and Eq. (6), we set k_{min} to 15 and k_{max} to 1000.

For our experiments on RealEstate10K [51], we adopt DFoT [32] as the base model. The RealEstate10K dataset provides a training set of approximately 65K short video clips. Training is conducted at a resolution of 256×256, with frames patchified to 128×128. To incorporate an additional memory block, we freeze all other components and fine-tune only the memory module. Training is performed with a 2-frame temporal context window and a 1-frame memory window. The model is trained for approximately 50K steps using 4 GPUs, with a batch size of 8 per GPU.

FOV Overlapping Computation. We present the details of Monte Carlo-based FOV overlapping computation in Alg. 2.

Algorithm 2: Monte Carlo-based FOV Overlapping Computation

Input:

- $P_{\text{mem}} \in \mathbb{R}^{F\times 5}$: memory-bank poses (x,y,z,pitch,yaw), F is the number of stored poses.
- $P_{\text{curr}} \in \mathbb{R}^{5}$: pose of the current frame being generated.
- n_s: number of 3D sample points (default 10,000).
- r_s: radius of the sampling sphere (default 30 m).
- FOV_h, FOV_v: horizontal/vertical field-of-view angles (in degrees).

Output:

- $\mathbf{o} \in \mathbb{R}^{F}$: overlapping ratios between each memory pose and the current pose.

begin

Δ Step 1: Random Sampling in a Sphere

Generate n_s points $\mathbf{p}$ uniformly in a 3D sphere of radius r_s:

$$\mathbf{p} \leftarrow \text{PointSampling}\left(n_s,\, r_s\right).$$

Δ Step 2: Translate Points to P_{curr} as Center

Let $P_{\text{curr}}(x, y, z)$ be the 3D coordinates of the current camera pose. Shift all sampled points:

$$\mathbf{p} \leftarrow \mathbf{p} + P_{\text{curr}}(x, y, z).$$

Δ Step 3: FOV Checks

Compute a boolean matrix $\mathbf{b}_{\text{mem}} \in \{0,1\}^{F\times n_s}$, where each entry indicates if a point in $\mathbf{p}$ lies in the FOV of a memory-bank pose:

$$\mathbf{b}_{\text{mem}} \leftarrow \text{IsInsideFOV}\Big(\mathbf{p},\, P_{\text{mem}},\, FOV_h,\, FOV_v\Big).$$

Similarly, compute a boolean vector $\mathbf{b}_{\text{curr}} \in \{0,1\}^{n_s}$ for the current pose:

$$\mathbf{b}_{\text{curr}} \leftarrow \text{IsInsideFOV}\Big(\mathbf{p},\, P_{\text{curr}},\, FOV_h,\, FOV_v\Big).$$

Δ Step 4: Overlapping Ratio Computation

Obtain the final overlapping ratio vector $\mathbf{o} \in \mathbb{R}^{F}$ by combining $\mathbf{b}_{\text{mem}}$ and $\mathbf{b}_{\text{curr}}$. For instance,

$$\mathbf{o}[i] \;=\; \frac{1}{n_s}\sum_{j=1}^{n_s}\Big(\mathbf{b}_{\text{mem}}[i,j]\cdot\mathbf{b}_{\text{curr}}[j]\Big),$$

to measure the fraction of sampled points that are visible in both the i-th memory-bank pose and the current pose.

Return $\mathbf{o}$.

end

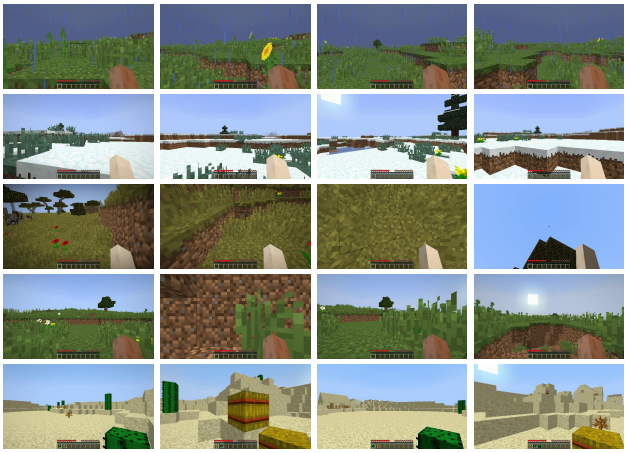


Figure 12. **Training Examples.** Our training environments encompass diverse terrains, action spaces, and weather conditions, providing a comprehensive setting for learning.

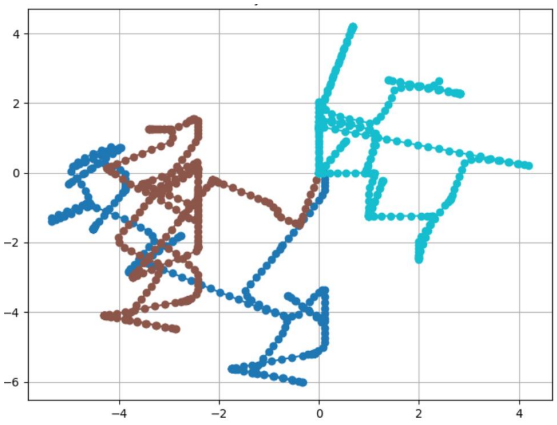


Figure 13. **Visualization of Trajectory Examples in the X-Z Space.** The axis scales represent distances within the Minecraft environment.

6.3. Visualizations

In this section, we provide more visualization of different aspects to facilitate understanding.

Minecraft Training Examples. We present a diverse set of training environments that include various terrain types, action spaces, and weather conditions, as shown in Figure 12. These variations help enhance the model's adaptability and robustness in different scenarios.

Trajectory Examples in Minecraft. Figure 13 illustrates trajectory examples in the x-z space over 100 frames. The agent's movement exhibits a random action pattern, ensuring diverse learning objectives and a broad range of sampled experiences.

Pose Distribution. We collect and visualize 800 samples within a sampling range of 8, as shown in Figure 14. The

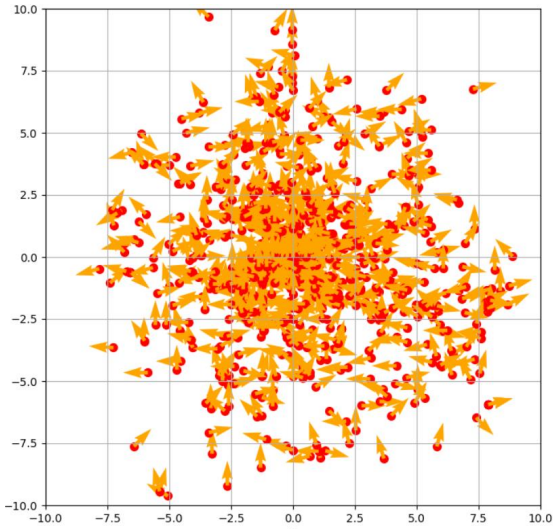


Figure 14. **Visualization of Relative Pose Distribution for Training in X-Z Space.** Red dots indicate positions, while yellow arrows represent directions.

random pattern observed in Figure 13 ensures a diverse distribution of sampled poses in space, which is beneficial for learning the reasoning process within the memory blocks.

More Qualitative Results. For additional qualitative examples, we recommend consulting the project page, which offers enhanced visualizations.