L3G-WORLD: REAL-TIME INFINITE WORLD GENERATION WITH LATENT 3D GAUSSIAN MEMORY

Anonymous authors
Paper under double-blind review

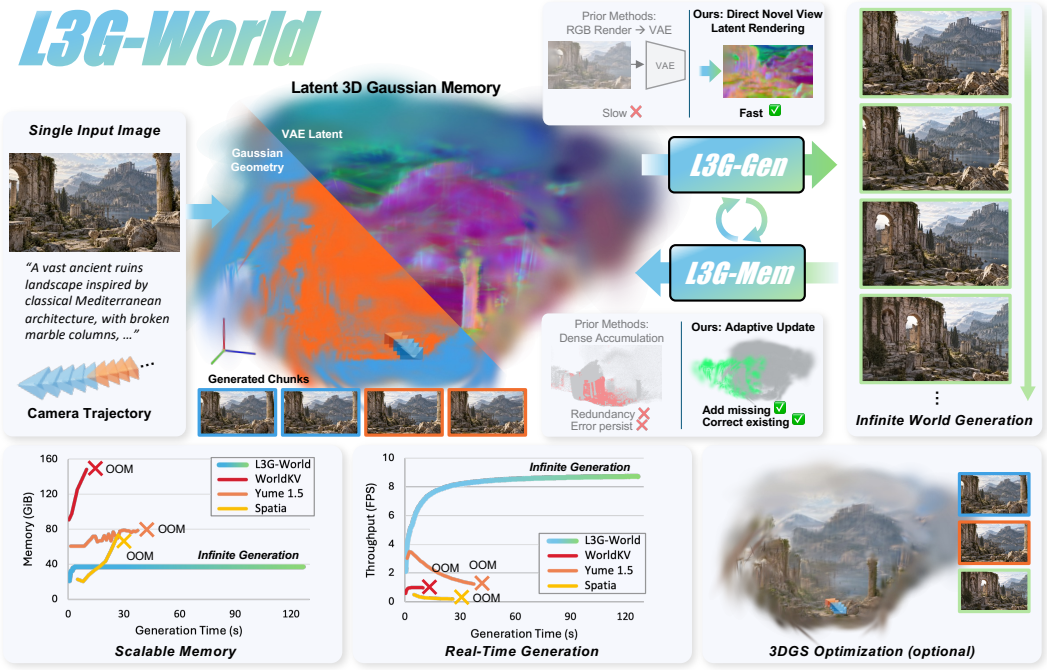


Figure 1: **Teaser. L3G-World** generates an explorable, spatially consistent world in real time from a single image and a user-controlled camera trajectory. To this end, L3G-Mem maintains a compact latent 3D Gaussian memory by adding Gaussians only for missing regions and correcting existing ones, and L3G-Gen generates video conditioned on VAE latents rendered directly from the memory.

ABSTRACT

Interactive world models generate visual content conditioned on user actions, yet maintaining spatial consistency over long-horizon exploration in real time remains challenging. Existing approaches rely on either frame retrieval or explicit 3D memories, both of which scale poorly as memory grows with the generated stream. Moreover, explicit 3D memories condition latent video generators by re-encoding RGB renderings, which adds computation and propagates rendering artifacts. In this paper, we introduce a **L**atent **3D** **G**aussian Memory (**L3G**) that combines the efficiency of feed-forward reconstruction with a compact, rapidly renderable scene representation. Building on this memory, we propose **L3G-World**, in which a memory updater and a video diffusion transformer alternate autoregressively to generate spatially consistent worlds. The memory updater spawns Gaussians in newly observed regions and refines existing ones in explored regions, so that the memory grows with scene extent rather than stream length. Each Gaussian further stores a VAE latent feature, allowing the memory to be rendered directly into the DiT's latent space without VAE re-encoding. Experiments show that **L3G-World** achieves state-of-the-art generation quality and long-horizon spatial consistency, with up to $20.5\times$ fewer 3D primitives, $10.3\times$ faster memory update, and $307\times$ faster memory rendering than prior 3D memory-based methods, enabling real-time world generation at 9 FPS.

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

1 INTRODUCTION

Interactive video world models (Seo et al., 2026; Nam et al., 2026; Bruce et al., 2024; Team et al., 2026; Mao et al., 2026; Sun et al., 2025; Wang et al., 2026; Gao et al., 2026; Basant et al., 2026), which generate visual content conditioned on user-provided actions, have recently emerged as a promising framework for real-world simulation (Seo et al., 2026; Mao et al., 2026; Basant et al., 2026), interactive game generation (Bruce et al., 2024; Team et al., 2026; Sun et al., 2025; Wang et al., 2026; Gao et al., 2026), and embodied simulation (Agarwal et al., 2026; Chen et al., 2026b). A central challenge lies in maintaining spatial consistency during long-horizon exploration and loop closures while still supporting real-time generation.

One line of work employ temporal contexts, by retrieving frames or KV caches at each self-attention layer of diffusion models from previously generated chunks during autoregressive generation (Team et al., 2026; Wang et al., 2026; Mao et al., 2026; Yi et al., 2026). While such retrieval methods can preserve information from earlier generations, its memory cost grows with the length of the generated stream and spatial resolution. As shown in Tab. 1, this linear growth makes long-horizon generation increasingly expensive and can eventually lead to out-of-memory failures.

Another line of work explicitly stores scene information in persistent 3D representations. These approaches preserve previously generated content by reconstructing frames into point-cloud-based memories (Ren et al., 2025; Zhao et al., 2026; Wu et al., 2026a; Shen et al., 2026; Lee et al., 2026) or 3D Gaussian primitives (De Lutio et al., 2026; Wu et al., 2025; Yang et al., 2026). Such spatially anchored representations improve consistency across revisited regions while enabling fine-grained camera control. However, as shown in Tab. 1, existing approaches exhibit a fundamental trade-off between reconstruction efficiency and latency. Feed-forward methods based on per-pixel point clouds or Gaussian primitives enable fast scene reconstruction, but predicting dense 3D representations for every pixel introduces substantial memory overhead. As generation proceeds, these dense primitives are repeatedly accumulated over the same spatial regions, resulting in increasingly redundant scene representations and slower target-view rendering. Moreover, since previously stored primitives are not adaptively updated, geometrically inaccurate regions may persist over time. In contrast, optimization-based methods (De Lutio et al., 2026; Wu et al., 2025) maintain more compact representations whose memory footprint primarily scales with scene extent, but their iterative optimization introduces substantial latency, making them unsuitable for real-time generation.

To address these challenges, we introduce a compact **L**atent **3**D **G**aussian Memory (**L3G**), which is continually read and updated online, and whose size grows primarily with the spatial extent of the explored scene rather than with the length of the generated stream. Building on this memory, we propose **L3G-World**, a world-generation framework composed of two modules: a feed-forward memory updater **L3G-Mem**, and a memory-conditioned video diffusion transformer **L3G-Gen**. As the two modules alternate autoregressively, L3G-Mem spawns new Gaussians only in uncovered regions, and refines the existing Gaussians in explored regions through residual updates. The memory is therefore not a buffer that accumulates primitives, but a persistent scene representation revised over time, which avoids redundant accumulation and progressively corrects geometrically inaccurate Gaussians. Since each Gaussian additionally stores a VAE latent feature, the memory is rendered directly into the latent space of L3G-Gen under the target cameras, conditioning the generation of the next video chunk without VAE re-encoding. The generated chunk is then fed back to L3G-Mem, which again updates the memory, closing the autoregressive loop. Combined with few-step distillation of L3G-Gen, this design enables real-time, long-horizon world generation.

Experiments demonstrate that L3G-World achieves state-of-the-art long-horizon consistency and generation quality while using 16.5–$20.5\times$ fewer 3D primitives, performing memory reconstruction 5.8–$10.3\times$ faster, and providing 246–$307\times$ higher VAE-space rendering throughput than the strongest prior explicit 3D-memory baselines. Combined with few-step distillation of L3G-Gen, this enables real-time world generation at 9.01 FPS on two H100 GPUs.

2 RELATED WORKS

Video Diffusion and World Models. Video diffusion models have made substantial progress in high-quality video generation (Yang et al., 2025; Wan et al., 2025; Ali et al., 2025; Agarwal et al., 2026; HaCohen et al., 2026). Most adopt latent diffusion (Rombach et al., 2021; Blattmann et al.,

Table 1: **Computational efficiency of memory analysis.** We compare memory scaling across methods and report memory and latency for constructing and rendering explicit scene representations. For memory scaling, T denotes the number of generated frames, R denotes the resolution size, and S denotes the spatial coverage of the scene.

Method	Resolution	Manner	Refinement	Memory Scaling	Memory ↓	Recon. / Scene ↓	Rendering / Frame ↓	Total Latency ↓	Total FPS ↑	Real-time
KV Cache (Yi et al., 2026)	832 ×480	–	–	$\mathcal{O}(TR)$	46.0 GiB	–	–	OOM	OOM	✓
Point Cloud (Zhao et al., 2026)	1248 ×704	Feed-forward	✗	$\mathcal{O}(TR)$	33.3 GiB	246.7 s	1,550 ms	1720.4 s	0.24 fps	✗
3D Gaussian (Yang et al., 2026)	560 ×336	Feed-forward	✗	$\mathcal{O}(TR)$	36.8 GiB	10.3 s	6.0 ms	208.7 s	1.92 fps	✗
3D Gaussian (De Lutio et al., 2026)	960 ×540	Optimization	✓	$\mathcal{O}(S)$	21.32 GiB	3,508.4 s	7.0 ms	3843.2 s	0.10 fps	✗
Latent 3D Gaussian (L3G-World)	832 ×480	**Feed-forward**	✓	$\mathcal{O}(S)$	**7.4 GiB**	**2.4 s**	**0.48 ms**	**45.8 s**	**9.01 fps**	✓

2023), where raw videos are compressed into a latent space using a VAE. Reducing the number of sampling steps, together with causal generation, has further enabled low-latency video streaming (Yin et al., 2025; Huang et al., 2026; Lin et al., 2026; Nie et al., 2026; Gu et al., 2026). They improve few-step generation by training on model-generated histories to mitigate exposure bias, or by combining flow-based transitions with on-policy distillation. Building on these advances, interactive world models (Seo et al., 2026; Nam et al., 2026; Bruce et al., 2024; Team et al., 2026; Mao et al., 2026; Sun et al., 2025; Wang et al., 2026; Gao et al., 2026; Basant et al., 2026) generate interactive environments conditioned on user or agent actions.

Feed-forward 3D Reconstruction. Traditional reconstruction methods (Mildenhall et al., 2020; Kerbl et al., 2023) rely on per-scene optimization, whereas recent feed-forward approaches directly predict 3D geometry from input images in a single forward pass. They typically predict per-pixel pointmap (Wang et al., 2025a; Lin et al., 2025) or 3D Gaussians (Charatan et al., 2024; Chen et al., 2024; Xu et al., 2025). Recent works (Wang et al., 2025b; Zhuo et al., 2025; Li et al., 2025; Chen et al., 2025) further introduce streaming formulations that sequentially process incoming images for online reconstruction. Complementarily, token-based 3DGS methods (An et al., 2025; Veicht et al., 2026; Ren et al., 2026; Yoo et al., 2026) decode compact sets of Gaussian primitives from visual or learnable tokens rather than predicting dense per-pixel Gaussians.

Consistent World Generation. To improve long-term spatial consistency, several world models maintain explicit 3D scene memories, typically represented as point clouds (Ren et al., 2025; Zhao et al., 2026; Wu et al., 2026a; Shen et al., 2026; Lee et al., 2026) or 3D Gaussians (Yang et al., 2026; De Lutio et al., 2026; Wu et al., 2025). These methods reconstruct 3D representations from generated frames and render them into subsequent target views to provide spatially grounded conditioning. Such explicit 3D memories improve consistency and camera controllability, but feed-forward approaches typically rely on dense per-frame reconstruction and continuously accumulate new 3D primitives, causing memory consumption and rendering cost to grow with the generation horizon. Some methods (De Lutio et al., 2026; Wu et al., 2025) use optimization-based reconstruction to maintain a scene-level 3D representation rather than simply accumulating per-frame primitives, but their slow per-scene optimization introduces substantial latency and is difficult to reconcile with real-time generation. In contrast, L3G-World maintains compact 3D Gaussians as scene memory, by adaptively spawning new primitives or refining existing ones. Moreover, each Gaussian directly stores VAE latent features, enabling efficient conditioning of the video diffusion model for real-time world generation.

3 METHODOLOGY

Real-time interactive world generation that maintains geometric consistency places two competing demands on a 3D memory. Consistency over a long horizon requires storing previously generated content, while keeping pace with generation requires a memory that stays compact and fast to read and write. Dense per-frame accumulation stores everything but grows unboundedly, since redundant primitives pile up over the same regions and slow down rendering.

We therefore design latent 3D Gaussian memory around two decisions. First, the memory is updated selectively: it *spawns* new Gaussians only for newly observed geometry and *refines* existing ones to correct reconstruction errors from earlier steps, so that memory usage remains largely independent of the number of generated frames. Second, each Gaussian stores a VAE latent feature, so that the memory is rendered directly in the latent space of the video generator and conditions the generator without VAE re-encoding.

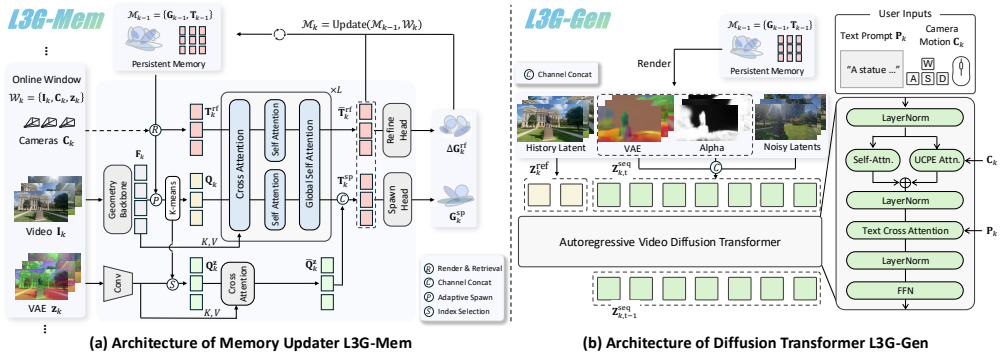


Figure 2: **Main Architecture.** (a) L3G-Mem takes the current window $\mathcal{W}_k$ and the memory $\mathcal{M}_{k-1}$. Rendering $\mathcal{M}_{k-1}$ into the current cameras determines which geometry features are clustered by K-means into spawning queries, and identifies the visible Gaussians whose memory tokens are retrieved as refinement tokens. Both token sets are processed by a joint transformer and decoded into new Gaussians and residual updates, respectively. The new Gaussians are appended to the memory and the retrieved ones are updated in place, yielding $\mathcal{M}_k$. (b) L3G-Gen renders the VAE features and opacity of the memory under the target cameras, concatenates them with the noisy latents along the channel dimension, and denoises the next chunk conditioned on the history latents.

As illustrated in Fig. 2, L3G-World realizes these decisions with two modules that alternate autoregressively: a memory updater, L3G-Mem, which reads and updates the latent 3D Gaussian memory with each generated chunk, and a video generator, L3G-Gen, which generates the next chunk conditioned on the memory rendered in its latent space. We first present L3G-Mem and the latent 3D Gaussian memory (Sec. 3.1), then describe how L3G-Gen is conditioned on the memory and distilled for real-time inference (Sec. 3.2). Implementation details are provided in Sec. B.1 and Sec. B.2, respectively.

3.1 3D Gaussians as World Memory

Online Window-Based Memory Update. L3G-Mem processes a stream of images $\mathrm{I}^i \in \mathbb{R}^{3\times h\times w}$ and their corresponding camera parameters $\mathcal{C}^i$ in windows of size W. At autoregressive step k, we define the input window as $\mathcal{W}_k = \{(\mathrm{I}_k^i, \mathcal{C}_k^i, z_k^i)\}_{i=1}^W = \{\mathbf{I}_k, \mathbf{C}_k, \mathbf{z}_k\}$, where $\mathbf{z}_k$ indicates VAE latent of $\mathbf{I}_k$. The latent 3D Gaussian memory after processing this window is represented as:

$$\mathcal{M}_k = \{(\mathcal{G}_j, \mathrm{T}_j)\}_{j=1}^{N_k} = \{\mathbf{G}_k, \mathbf{T}_k\}, \tag{1}$$

where T_j denotes a memory token, $\mathcal{G}_j$ contains the corresponding Gaussian parameters, and N_k is the number of memory tokens at k-th window. The Gaussian parameters are decoded from each memory token using an MLP-based Gaussian head, $\mathcal{G}_j = \mathrm{Head}(\mathrm{T}_j)$.

Given the previous memory $\mathcal{M}_{k-1}$ and the current input window $\mathcal{W}_k$, our model updates the memory autoregressively as:

$$\mathcal{M}_k = \mathrm{Update}\left(\mathcal{M}_{k-1}, \mathcal{W}_k\right). \tag{2}$$

Adaptive Gaussian Spawning. To keep latent 3D Gaussian memory compact, L3G-Mem selectively spawns new Gaussians only where additional scene geometry is needed. This requires identifying regions that are insufficiently represented by the current memory and initializing spawning query tokens from those regions, leaving well-represented regions to the existing Gaussians. To identify such regions, we render the Gaussians stored in $\mathcal{M}_{k-1}$ from the current camera views $\mathbf{C}_k$ to obtain RGB images and opacity maps. The opacity indicates whether each pixel is covered by the memory, and the per-pixel RGB error against $\mathbf{I}_k$ indicates whether the memory reproduces that pixel faithfully. We mark pixels with low opacity or high error as insufficiently represented, and collect them into a spawn mask $\mathbf{B}_k$.

The spawn mask then determines where new Gaussians are spawned: we draw the spawning queries from the masked observation, so that the new Gaussians fill the regions marked by $\mathbf{B}_k$. Specifically, we encode $\mathbf{I}_k$ with a geometric foundation model (Lin et al., 2025) to obtain geometry features $\mathbf{F}_k = \{\mathrm{F}_k^l\}_{l=1}^L$, where $L = 3$ denotes the selected feature layers, and retain only the features

of F_k^1 inside $\mathbf{B}_k$ as spawning candidates. Following ZipSplat (Veicht et al., 2026), we apply K-means to these candidates to initialize a compact set of spawning query tokens $\mathbf{Q}_k$. Thus, rather than clustering over all input geometry features, our updater allocates its spawning budget only to content missing from the memory.

Gaussian Refinement. The refinement stream of L3G-Mem updates existing Gaussians that are visible in the current window, rather than leaving previously stored content fixed. We render the Gaussians stored in $\mathcal{M}_{k-1}$ from the current camera views $\mathbf{C}_k$, retrieve the memory tokens associated with the visible Gaussians, and use them as refinement tokens, denoted by $\mathbf{T}_k^{\mathrm{rf}}$. These tokens and their corresponding Gaussian parameters are subsequently updated using observations from the current window $\mathcal{W}_k$.

Joint Token Processing. L3G-Mem concatenates the spawning and refinement tokens along the token dimension and processes them with a single transformer $\mathcal{F}_\theta$:

$$\left(\bar{\mathbf{Q}}_k, \bar{\mathbf{T}}_k^{\mathrm{rf}}\right) = \mathcal{F}_\theta\left(\left[\mathbf{Q}_k; \mathbf{T}_k^{\mathrm{rf}}\right], \mathbf{F}_k\right), \tag{3}$$

where $[\cdot;\cdot]$ denotes concatenation along the token dimension. The transformer consists of L blocks. In each block l, the two streams first cross-attend to the geometric features F_k^l, followed by stream-wise self-attention. A global self-attention layer is then applied over the concatenated spawning query and refinement tokens, enabling information exchange between newly observed and previously reconstructed scene content.

VAE Latent Lifting. Since video diffusion models operate in the VAE latent space, we additionally lift VAE features into latent 3D Gaussian memory, allowing it to be rendered directly in the latent space of the diffusion model. A dedicated feature-lifting branch initializes latent queries $\mathbf{Q}_k^{\mathbf{z}}$ from $\mathbf{z}_k$ using the same spatial indices as the spawning query tokens, and cross-attends to $\mathbf{z}_k$. This produces VAE latent tokens that are spatially aligned with the corresponding query tokens, and can therefore be jointly decoded as $\mathbf{T}_k^{\mathrm{sp}} = \mathrm{Concat}_c(\bar{\mathbf{Q}}_k; \bar{\mathbf{Q}}_k^{\mathbf{z}})$, where $\mathrm{Concat}_c(\cdot)$ is channel-wise concatenation.

Gaussian Decoding and Memory Update. The processed spawning tokens are concatenated with their corresponding VAE latent tokens and decoded into new Gaussian primitives, while the refinement tokens are decoded into residual updates for the retrieved Gaussians:

$$\mathbf{G}_k^{\mathrm{sp}} = \mathrm{Head}_{\mathrm{sp}}\left(\mathbf{T}_k^{\mathrm{sp}}\right), \qquad \Delta\mathbf{G}_k^{\mathrm{rf}} = \mathrm{Head}_{\mathrm{rf}}\left(\bar{\mathbf{T}}_k^{\mathrm{rf}}\right). \tag{4}$$

To obtain $\mathcal{M}_k$, L3G-Mem updates $\mathcal{M}_{k-1}$ in two ways. It appends the newly spawned Gaussians together with their memory tokens, and updates the retrieved Gaussians in place by applying the predicted residuals and replacing their memory tokens with the refinement outputs. All memory entries not selected for refinement remain unchanged.

Autoregressive Training. We train over K consecutive windows, starting from an empty memory $\mathcal{M} = \emptyset$. We apply stop gradients through $\mathcal{M}_{k-1}$ at each step k to ensure causality. At step k, we supervise renderings of $\mathcal{M}_k$ from the target views of all windows up to k, with the overall objective

$$\mathcal{L} = \frac{1}{K}\sum_{k=1}^{K}\left(\mathcal{L}_{\mathrm{rgb}}^k + \lambda_{\mathrm{lpips}}\mathcal{L}_{\mathrm{lpips}}^k + \lambda_{\mathrm{depth}}\mathcal{L}_{\mathrm{depth}}^k + \lambda_{\mathrm{vae}}\mathcal{L}_{\mathrm{vae}}^k + \lambda_{\mathrm{vis}}\mathcal{L}_{\mathrm{vis}}^k\right). \tag{5}$$

The terms denote RGB ℓ_1, LPIPS, depth ℓ_1, VAE latent MSE, and visibility losses, respectively. The visibility loss (Ren et al., 2026) encourages each Gaussian to lie within at least one input-view frustum.

3.2 Reconstruction-Guided Auto-Regressive Video Generation

We introduce a reconstruction-guided autoregressive video diffusion model for long-horizon world generation. At each autoregressive step, the model retrieves previously generated information from the persistent 3D Gaussian memory to guide the generation of the next video chunk. This provides spatially aligned conditioning for preserving scene consistency when revisiting previously explored regions. To support real-time interaction, we further distill the model into a few-step generator.

Conditioning for Generation. At autoregressive step k, we render the VAE features stored in $\mathcal{M}_{k-1}$ under the target cameras $\mathbf{C}_k$ to obtain spatially aligned rendered VAE features $\hat{\mathbf{z}}_k^{\mathrm{mem}}$. We

concatenate these features with the noisy video latents $\mathbf{z}_{k,t}$ along the channel dimension, providing geometric conditioning without introducing additional memory tokens. The previously generated reference latents $\mathbf{z}_k^{\text{ref}}$ are concatenated with zero-filled channels to match the channel dimension of the target input. The input sequence is formed by token-wise concatenation:

$$\mathbf{z}_{k,t}^{\text{seq}} = \left[\mathbf{z}_k^{\text{ref}}; \text{Concat}_c(\mathbf{z}_{k,t}; \hat{\mathbf{z}}_k^{\text{mem}})\right]. \tag{6}$$

Only the target video latents are denoised, while the history features remain fixed. Following (Zhang et al., 2026), we add the residual camera ray rotary positional embedding (RayRoPE (Wu et al., 2026b)) on the self attention in each DiT layer.

Temporal Positional Encoding. Following (Seo et al., 2026), we assign the reference latent a temporal rotary positional embedding (RoPE (Su et al., 2021)) beyond the current generation chunk. The temporal positions associated with the input sequence are

$$\mathbf{p}_k^{\text{seq}} = \left[\underbrace{-128, \ldots, -128 + H}_{\text{reference}}; \underbrace{0, \ldots, L_z}_{\text{target}}\right], \tag{7}$$

where H and L_z denote the numbers of history latents and target chunk, respectively. All spatial tokens within each latent frame share its temporal RoPE position. The history latents serve as conditioning rather than a future prediction target, and remain accessible to all target tokens.

On-Policy Distillation. We first train the reconstruction-conditioned teacher $\mathbf{v}_\phi$ using flow matching (Lipman et al., 2022), as described in Sec. A.1. To reduce the sampling cost, we distill the teacher into a few-step auto-regressive student following AnyFlow (Gu et al., 2026). The student learns flow-map transitions between arbitrary noise levels, allowing it to traverse the denoising trajectory using fewer model evaluations. After flow-map initialization, we jointly optimize flow-map training and on-policy distribution matching. Specifically, flow-map backward simulation produces student-generated samples, which are used for distribution matching under the guidance of the frozen teacher. Training on the student's own rollouts, rather than relying solely on noisy ground-truth samples, helps mitigate the train–test mismatch in few-step autoregressive generation.

4 Experiments

4.1 Experimental Setup

Implementation Details. We implement our memory updater based on ZipSplat (Veicht et al., 2026), using Depth Anything 3 (DA3) (Lin et al., 2025) as the geometry backbone. For the video generation model, we adopt Cosmos-Predict2.5 (NVIDIA et al., 2025). We train our memory updater on DL3DV (Ling et al., 2024), and our video generation model on DL3DV and RealEstate10K (Zhou et al., 2018). The memory updater is fine-tuned including the geometry backbone. We fine-tune Diffusion Transformer (DiT) backbone of the video generator using LoRA (Hu et al., 2022), and we adopt AnyFlow (Gu et al., 2026) for few-step distillation. All models are optimized using AdamW (Loshchilov & Hutter, 2017) on four NVIDIA A100 GPUs, with a batch size of 8 and 16 for the memory updater and world model, respectively. The implementation and training details of the memory updater and video DiT are provided in Sec. B.1 and B.2.

Baselines. We compare L3G-World against state-of-the-art world models with either explicit or implicit memory, including point-cloud-based methods, Lyra 2.0 (Shen et al., 2026), GEN3C (Ren et al., 2025), Spatia (Zhao et al., 2026), SPMem (Wu et al., 2026a), and 3DScenePrompt (Lee et al., 2026); the Gaussian-based method NeoVerse (Yang et al., 2026); and temporal-context-based methods, Yume 1.5 (Mao et al., 2026) and Matrix-Game 3.0 (Wang et al., 2026). For a fair comparison in our single-image setting, we use Depth Anything 3 GIANT-1.1-NESTED (Lin et al., 2025) to obtain metric depth for all explicit 3D-memory methods, aligning the scale of the reconstructed scene representation with the evaluation camera trajectory. For Gaussian-based methods, including NeoVerse and L3G-World, we align the initially predicted Gaussian geometry to the same metric depth. All methods are evaluated on a single NVIDIA A100 GPU, including the efficiency measurements. Detailed baseline configurations are provided in Sec. B.3.

Evaluation Benchmark. We evaluate L3G-World and all baselines on DL3DV (Ling et al., 2024) and RealEstate10K (Zhou et al., 2018). To evaluate long-horizon spatial consistency, we construct

Table 2: **Video generation evaluation.** We evaluate generation quality, self-consistency, and efficiency on DL3DV and RealEstate10K. The best and second-best results for each metric within each dataset are shown in **bold** and underline, respectively.

	Self-Consistency			Efficiency					Generation Quality						Pose Error		3D Consistency
Method	PSNR ↑	SSIM ↑	LPIPS ↓	# Primitives (K) ↓	Memory (MiB) ↓	Recon. s/chunk ↓	Render FPS (RGB) ↑	Render FPS (VAE) ↑	LPIPS ↓	FID ↓	IQ ↑	AQ ↑	SC ↑	BC ↑	TransErr ↓	RotErr (°) ↓	MEt3R ↓
DL3DV																	
Temporal Context																	
Yume 1.5	13.71	0.487	0.655	–	–	–	–	–	0.699	50.88	0.601	0.514	0.805	**0.888**	0.904	11.09	**0.050**
Matrix Game 3.0	12.26	0.401	0.540	–	–	–	–	–	0.681	67.78	0.609	0.431	0.774	0.865	0.376	**3.11**	0.070
Explicit 3D Memory																	
GEN3C	13.54	0.464	0.474	1,802	47.27	20.513	1.70	0.40	0.618	48.65	0.510	0.480	0.813	0.869	**0.210**	4.24	0.075
Spatia	13.96	0.398	0.571	9,647	220.80	29.743	0.29	0.08	0.638	89.39	0.546	0.382	0.719	0.831	0.738	15.92	0.122
SpMem	12.53	0.513	0.637	1,667	38.16	62.654	0.22	0.06	0.711	74.23	0.504	0.448	0.758	0.868	0.799	10.10	0.056
lyra 2.0	13.60	0.382	0.493	1,273	92.27	2.337	11.36	1.69	0.618	52.26	0.651	0.481	0.801	0.882	0.348	4.79	0.074
3DScenePrompt	13.63	**0.528**	0.609	149,880	6292	7.098	0.24	0.06	0.706	87.99	0.544	0.400	0.704	0.810	0.759	11.22	0.080
NeoVerse	13.53	0.525	0.652	8,766	501.57	5.682	55.56	3.11	0.755	152.25	0.417	0.305	0.649	0.797	0.508	7.80	0.089
L3G-World	15.11	0.431	**0.435**	77	20.65	**0.227**	336.6	**765.0**	**0.583**	48.09	0.619	0.486	**0.814**	**0.888**	0.278	5.22	0.081
L3G-World Distill	**15.23**	0.423	0.447	**40**	**14.57**	0.268	**381.3**	697.9	**0.583**	**45.74**	**0.653**	**0.516**	0.811	0.887	0.249	5.98	0.082
RealEstate10K																	
Temporal Context																	
Yume 1.5	15.22	0.652	0.634	–	–	–	–	–	0.698	48.01	0.618	0.490	0.796	0.881	0.818	10.19	0.048
Matrix Game 3.0	17.95	0.657	0.437	–	–	–	–	–	0.752	178.49	0.475	0.340	0.647	0.803	0.618	3.37	0.076
Explicit 3D Memory																	
GEN3C	16.62	0.653	0.358	1,802	42.27	19.125	1.89	0.44	0.537	43.76	0.572	0.472	0.819	0.905	0.112	0.46	0.027
Spatia	16.93	0.628	0.388	2,849	67.52	19.560	0.88	0.22	0.578	59.70	0.553	0.414	0.797	0.889	0.399	2.43	0.052
SpMem	13.76	0.660	0.577	655	14.99	41.166	0.78	0.19	0.531	54.12	0.617	0.480	0.789	0.857	0.672	4.08	**0.023**
lyra 2.0	17.30	0.607	0.313	1,273	92.27	2.299	11.63	1.70	0.550	37.36	**0.720**	**0.512**	0.824	0.893	**0.091**	**0.20**	0.025
3DScenePrompt	15.46	**0.700**	0.470	149,880	6292	7.090	0.28	0.07	0.648	64.81	0.484	0.418	0.773	0.858	0.572	3.45	0.033
NeoVerse	12.87	0.542	0.638	10,437	597.21	5.238	55.56	3.07	0.737	142.75	0.515	0.361	0.652	0.799	0.759	5.98	0.061
L3G-World	**19.42**	0.691	**0.219**	32	11.56	0.394	536.3	**943.2**	**0.524**	**36.89**	0.648	0.464	**0.838**	**0.922**	0.208	0.29	**0.023**
L3G-World Distill	15.82	0.651	0.522	**29**	**10.59**	**0.263**	**619.5**	719.2	0.628	78.31	0.506	0.414	0.768	0.881	0.685	2.14	**0.023**

481-frame loop-closure trajectories that explore away from the initial view and then retrace the same camera path in reverse. We measure *self-consistency* by comparing generated frames at corresponding camera poses along the outgoing and returning trajectories using PSNR, SSIM, and LPIPS. For generation quality, we report LPIPS, FID (Heusel et al., 2017), and four VBench (Huang et al., 2024) metrics: Imaging Quality (IQ), Aesthetic Quality (AQ), Subject Consistency (SC), and Background Consistency (BC). Following (Seo et al., 2026), we evaluate camera controllability using translation error (TransErr) and rotation error (RotErr), and evaluate geometric consistency using MEt3R (Asim et al., 2025), following (Jang et al., 2026).

To evaluate the scalability of explicit 3D memory, we measure the number of stored primitives, memory footprint, reconstruction latency, and rendering throughput. We separately report RGB rendering FPS and VAE-space conditioning FPS, where the latter includes VAE encoding after RGB rendering for methods without directly renderable VAE features. For temporal-context-based methods, we instead report the size of the retained context. Further evaluation details are provided in Sec. B.4.

4.2 Quantitative Results

Tab. 2 compares L3G-World with the baselines in terms of long-horizon self-consistency, memory efficiency, generation quality, and camera pose and 3D consistency.

Long-Horizon Self-Consistency. On DL3DV, the two L3G-World variants achieve the best and second-best PSNR and LPIPS, with L3G-World obtaining the lowest LPIPS. On RealEstate10K, L3G-World further achieves the best PSNR and LPIPS, and the second-best SSIM. These results support our motivation that a persistent, spatially anchored scene memory can preserve previously observed content when the camera revisits a region after long-horizon exploration.

Memory Efficiency. The gains in efficiency are even larger. Compared with NeoVerse, a Gaussian-based method, L3G-World uses 114–326× fewer primitives and 24–52× less memory across the two datasets. Its compact representation already enables 6.1–9.7× faster RGB rendering, while directly rendering the stored VAE latent features increases the VAE-conditioning throughput by 246–307×. The advantage also holds against point-cloud-based memories. Compared with Lyra 2.0, a particularly efficient point-cloud baseline, L3G-World performs memory reconstruction 5.8–10.3× faster while using 4.5–8.0× less memory, and achieves 453–555× higher VAE-rendering throughput. These gains directly reflect our two design choices: instead of accumulating dense primitives, L3G-Mem adds Gaussians only for previously uncovered or high-error regions and updates existing ones in revisited regions; moreover, storing VAE features in each Gaussian eliminates the costly VAE re-encoding required for conditioning the DiT.

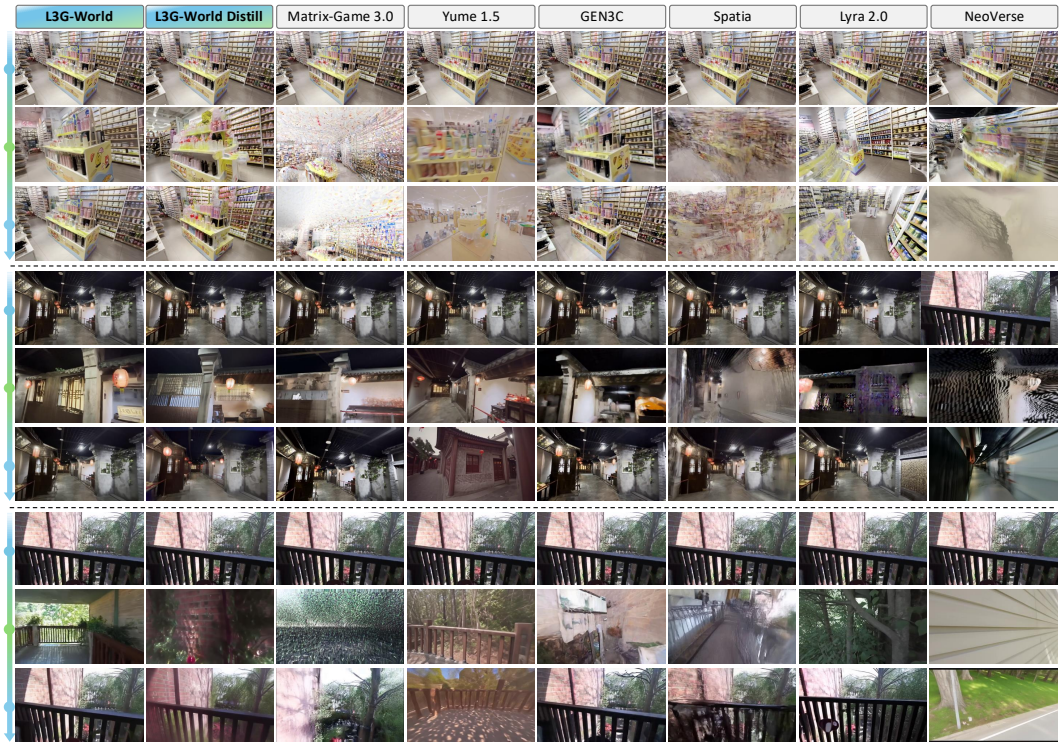


Figure 3: **Qualitative results of video generation on DL3DV and RealEstate10K.** For each scene, all methods start from the same input view (top), explore unseen regions (middle), and return to the initial camera location (bottom). L3G-World preserves scene structure and appearance upon loop closure, whereas the baselines exhibit drift, blur, or rendering artifacts.

Generation Quality and 3D Consistency. Importantly, this efficiency does not come at the cost of generation quality. On DL3DV, L3G-World achieves the best SC and ties for the best BC, while L3G-World Distill obtains the best FID, IQ, and AQ and ties with L3G-World for the best LPIPS. On RealEstate10K, L3G-World achieves the best LPIPS, FID, SC, and BC, together with the second-best IQ. For camera control and 3D consistency, L3G-World achieves pose errors and MEt3R scores comparable to those of the strongest baselines.

Overall, these results show that the proposed latent 3D Gaussian memory simultaneously improves long-horizon consistency and removes the reconstruction and rendering bottlenecks of existing explicit 3D memories, enabling scalable real-time world generation.

4.3 Qualitative Results

Fig. 3 presents qualitative comparisons under long-horizon loop-closure trajectories. For each scene, all methods start from the same input view, explore previously unseen regions, and eventually return to the initial camera location; thus, the final frame directly reveals whether the generated world remains consistent after long-horizon exploration. Frame-retrieval-based methods often exhibit noticeable appearance or geometric drift when returning to previously observed regions. Among explicit 3D-memory approaches, Spatia and NeoVerse show substantial blur and rendering artifacts as generation progresses, while GEN3C and Lyra 2.0 produce comparatively stronger results but often struggle to extrapolate plausible geometry and appearance far beyond the single input view. In contrast, L3G-World preserves scene structure and appearance throughout long-horizon exploration and recovers a visually consistent scene upon loop closure. L3G-World Distill also retains this behavior while enabling few-step generation. Additional qualitative comparisons with all baselines are provided in Sec. C.

Table 3: **Ablation study of L3G-Mem on DL3DV.** We report RGB and VAE latent NVS quality and efficiency. The best and second-best results are shown in **bold** and underline, respectively.

Variant	VAE MSE ↓	VAE Cosine ↑	# Gaussians (K) ↓	Recon. Latency (ms) ↓	Render FPS (VAE) ↑	PSNR ↑	SSIM ↑	LPIPS ↓
w/o Adaptive Spawning	0.1821	0.7762	75	230.31	742.4	16.262	**0.4085**	0.5351
w/o Refinement	0.2130	0.7428	45	**205.38**	937.5	15.515	0.3815	0.5710
w/o Global Self-Attn.	0.1841	0.7747	**41**	219.59	938.5	16.364	0.3972	0.5346
w/o RGB Loss	0.2506	0.6960	74	227.58	**1199.0**	10.256	0.2724	0.7512
w/o VAE Lifting	0.5700	0.4669	–	–	2.97	–	–	–
L3G-Mem (Full)	**0.1770**	**0.7824**	**41**	224.81	943.6	**16.516**	0.4061	**0.5338**

Table 4: **Ablation study of memory refinement in L3G-World**. The best and second-best results are shown in **bold**.

	Self-Consistency			Generation Quality						Pose Error		3D Consistency
Method	PSNR ↑	SSIM ↑	LPIPS ↓	LPIPS ↓	FID ↓	IQ ↑	AQ ↑	SC ↑	BC ↑	TransErr ↓	RotErr (°) ↓	MEt3R ↓
w/o Memory Refinement	13.99	0.373	0.515	0.594	45.99	0.647	0.503	0.796	0.877	0.387	9.35	0.102
L3G-World (Full)	**15.23**	**0.423**	**0.447**	**0.583**	**45.74**	**0.653**	**0.516**	**0.811**	**0.887**	**0.249**	**5.98**	**0.082**

4.4 Ablation Study

Memory Updater. Tab. 3 ablates the key components of L3G-Mem on novel view synthesis (NVS) on DL3DV, with all variants trained using a batch size of 2. We provide details of evaluation of L3G-Mem in Sec. D. Without adaptive spawning, by applying K-means to all geometry features, the model uses $\sim$1.8$\times$ more Gaussians while degrading both RGB and VAE NVS, showing that selective spawning reduces redundant primitives. Removing the refinement stream causes the largest quality drop, demonstrating the importance of correcting existing Gaussians through residual updates. Removing global self-attention between the spawn and refine streams further degrades performance, highlighting the benefit of coordinating newly spawned and existing scene content. We also ablate RGB supervision by setting $\lambda^k_{\text{lpips}}{=}\lambda^k_{\text{rgb}}{=}0$. Although L3G-Gen is conditioned on directly rendered VAE features, RGB supervision remains important for learning accurate 3D geometry. Finally, encoding RGB renderings with the VAE instead of lifting VAE features not only substantially slows down rendering but also degrades the rendered latent quality. Overall, the full model achieves the strongest reconstruction quality across most RGB and VAE metrics while retaining a compact Gaussian memory and comparable inference latency.

Memory Refinement in World Generation. We further evaluate whether adaptively refining the latent 3D Gaussian memory, rather than simply accumulating primitives, improves long-horizon world generation. We disable the residual refinement in L3G-Mem, such that Gaussians remain fixed after being spawned and the memory evolves only through accumulation. As shown in Tab. 4, removing memory refinement consistently degrades generation quality, with particularly large drops in self-consistency, pose accuracy, and 3D consistency. This demonstrates that a persistent scene memory should not only preserve previously generated content, but also progressively correct imperfect 3D primitives as the scene is revisited. By continually refining the accumulated scene representation, L3G-World reduces errors propagated over autoregressive generation and provides more reliable spatial conditioning for consistent and controllable world generation.

5 Conclusion

We presented L3G-World, a compact 3D Gaussian memory for real-time, spatially consistent world generation. Instead of accumulating dense primitives from every generated chunk, our feed-forward memory updater spawns new Gaussians only in regions insufficiently covered by the memory and refines existing ones in previously reconstructed regions, keeping the memory compact and fast to render. Each Gaussian further carries a VAE-space feature, so the memory can be rendered directly into the latent space of the DiT without VAE re-encoding, which makes memory conditioning efficient enough for real-time generation. Experiments show that L3G-World achieves state-of-the-art generation quality in terms of long-horizon spatial consistency, while substantially reducing the cost of memory reconstruction, rendering, and storage. We believe that compact, latent-space 3D memories offer a practical foundation for scalable interactive world models.

AI USE STATEMENT

In this work, we have not used generative AI tools for any task requiring disclosure: research ideation, citation retrieval, experimental design, and analysis of results were all carried out by the authors, and generative AI tools do not form a component of the proposed method. Additionally, we used generative AI tools for writing assistance, specifically polishing the phrasing of author-written text, and for polishing author-made figures. We have reviewed all AI-assisted work: all polished text and figures were verified by the authors to preserve the intended meaning. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Niket Agarwal, Arslan Ali, Jon Allen, Martin Antolini, Adeline Aubame, Alisson Azzolini, Junjie Bai, Maciej Bala, Yogesh Balaji, Josh Bapst, et al. Cosmos 3: Omnimodal world models for physical ai. *arXiv preprint arXiv:2606.02800*, 2026.

Arslan Ali, Junjie Bai, Maciej Bala, Yogesh Balaji, Aaron Blakeman, Tiffany Cai, Jiaxin Cao, Tianshi Cao, Elizabeth Cha, Yu-Wei Chao, et al. World simulation with video foundation models for physical ai. *arXiv preprint arXiv:2511.00062*, 2025.

Honggyu An, Jaewoo Jung, Mungyeom Kim, Sunghwan Hong, Chaehyun Kim, Kazumi Fukuda, Minkyeong Jeon, Jisang Han, Takuya Narihira, Hyuna Ko, et al. C3g: Learning compact 3d representations with 2k gaussians. *arXiv preprint arXiv:2512.04021*, 2025.

Mohammad Asim, Christopher Wewer, Thomas Wimmer, Bernt Schiele, and Jan Eric Lenssen. Met3r: Measuring multi-view consistency in generated images. In *IEEE/CVF Computer Vision and Pattern Recognition (CVPR)*, 2025.

Aarti Basant, Amlan Kar, Despoina Paschalidou, Fangyin Wei, Francesco Ferroni, Guillermo Garcia Cobo, Haithem Turki, Huan Ling, Jaewoo Seo, James Lucas, et al. Nvidia omnidreams: Real-time generative world model for closed-loop autonomous vehicle simulation. *arXiv preprint arXiv:2606.03159*, 2026.

Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Nicholas M Boffi, Michael S Albergo, and Eric Vanden-Eijnden. Flow map matching with stochastic interpolants: A mathematical framework for consistency models. *arXiv preprint arXiv:2406.07507*, 2024.

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. In *Forty-first International Conference on Machine Learning*, 2024.

David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19457–19467, June 2024.

Ruiyang Chen, Feiran Li, Chu Zhou, Zonglin Li, Zhanyu Ma, and Heng Guo. Of3gs: On-the-fly feed-forward 3d gaussian splatting from unposed images. *arXiv preprint arXiv:2606.03254*, 2026a.

Xingyu Chen, Yue Chen, Yuliang Xiu, Andreas Geiger, and Anpei Chen. Ttt3r: 3d reconstruction as test-time training. *arXiv preprint arXiv:2509.26645*, 2025.

Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images. *arXiv preprint arXiv:2403.14627*, 2024.

Yuzhi Chen, Ronghan Chen, Dongjie Huo, Yandan Yang, Dekang Qi, Haoyun Liu, Tong Lin, Shuang Zeng, Junjin Xiao, Xinyuan Chang, et al. Abot-physworld: Interactive world foundation model for robotic manipulation with physics alignment. *arXiv preprint arXiv:2603.23376*, 2026b.

Freeman Cheng, Botao Ye, Xueting Li, Junqi You, Fangneng Zhan, and Ming-Hsuan Yang. Recosplat: Online feed-forward gaussian splatting via render-and-compare. *arXiv preprint arXiv:2603.09968*, 2026.

Riccardo De Lutio, Tobias Fischer, Yen-Yu Chang, Yuxuan Zhang, Zhangjie Wu, Xuanchi Ren, Tianchang Shen, Katarína Tóthová, Zan Gojcic, and Haithem Turki. Artifixer: Enhancing and extending 3d reconstruction with auto-regressive diffusion models. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–12, 2026.

Zelin Gao, Qiuyu Wang, Jiapeng Zhu, Jingye Chen, Zichen Liu, Qingyan Bai, Jiahao Wang, Yufeng Yuan, Hanlin Wang, Yichong Lu, et al. Infinite worlds with versatile interactions. *arXiv preprint arXiv:2607.07534*, 2026.

Yuchao Gu, Guian Fang, Yuxin Jiang, Weijia Mao, Song Han, Han Cai, and Mike Zheng Shou. Anyflow: Any-step video diffusion model with on-policy flow map distillation. *arXiv preprint arXiv:2605.13724*, 2026.

Yoav HaCohen, Benny Brazowski, Nisan Chiprut, Yaki Bitterman, Andrew Kvochko, Avishai Berkowitz, Daniel Shalem, Daphna Lifschitz, Dudu Moshe, Eitan Porat, et al. Ltx-2: Efficient joint audio-visual foundation model. *arXiv preprint arXiv:2601.03233*, 2026.

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.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *Advances in Neural Information Processing Systems*, 38:167283–167308, 2026.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Wooseok Jang, Seonghu Jeon, Jisang Han, Jinhyeok Choi, Minkyung Kwon, Seungryong Kim, Saining Xie, and Sainan Liu. Repurposing geometric foundation models for multi-view diffusion. In *European Conference on Computer Vision (ECCV)*, 2026.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. URL https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/.

JoungBin Lee, Jaewoo Jung, Jisang Han, Takuya Narihira, Kazumi Fukuda, Junyoung Seo, Sunghwan Hong, Yuki Mitsufuji, and Seungryong Kim. 3d scene prompting for scene-consistent camera-controllable video generation. In *International Conference on Learning Representations*, volume 2026, pp. 73753–73777, 2026.

Zizun Li, Jianjun Zhou, Yifan Wang, Haoyu Guo, Wenzheng Chang, Yang Zhou, Haoyi Zhu, Junyi Chen, Chunhua Shen, and Tong He. Wint3r: Window-based streaming reconstruction with camera token pool, 2025. URL https://arxiv.org/abs/2509.05296.

Haotong Lin, Sili Chen, Jun Hao Liew, Donny Y. Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth anything 3: recovering the visual space from any views. *arXiv preprint arXiv:2511.10647*, 2025.

Shanchuan Lin, Ceyuan Yang, Hao He, Jianwen Jiang, Yuxi Ren, Xin Xia, Yang Zhao, Xuefeng Xiao, and Lu Jiang. Autoregressive adversarial post-training for real-time interactive video generation. *Advances in Neural Information Processing Systems*, 38:41061–41086, 2026.

Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22160–22169, 2024.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Xiaofeng Mao, Zhen Li, Chuanhao Li, Xiaojie Xu, Kaining Ying, and Kaipeng Zhang. Yume1. 5: A text-controlled interactive world generation model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7752–7761, 2026.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

Jisu Nam, Yicong Hong, Chun-Hao Paul Huang, Feng Liu, JoungBin Lee, Jiyoung Kim, Siyoon Jin, Yunsung Lee, Jaeyoon Jung, Suhwan Choi, et al. Worldcam: Interactive autoregressive 3d gaming worlds with camera pose as a unifying geometric representation. *arXiv preprint arXiv:2603.16871*, 2026.

Weili Nie, Julius Berner, Nanye Ma, Chao Liu, Saining Xie, and Arash Vahdat. Transition matching distillation for fast video generation. *arXiv preprint arXiv:2601.09881*, 2026.

NVIDIA, Arslan Ali, Junjie Bai, Maciej Bala, Yogesh Balaji, Aaron Blakeman, Tiffany Cai, Jiaxin Cao, Tianshi Cao, Elizabeth Cha, Yu-Wei Chao, Prithvijit Chattopadhyay, Mike Chen, Yongxin Chen, Yu Chen, Shuai Cheng, Yin Cui, Jenna Diamond, Yifan Ding, Jiaojiao Fan, Linxi Fan, Liang Feng, Francesco Ferroni, Sanja Fidler, Xiao Fu, Ruiyuan Gao, Yunhao Ge, Jinwei Gu, Aryaman Gupta, Siddharth Gururani, Imad El Hanafi, Ali Hassani, Zekun Hao, Jacob Huffman, Joel Jang, Pooya Jannaty, Jan Kautz, Grace Lam, Xuan Li, Zhaoshuo Li, Maosheng Liao, Chen-Hsuan Lin, Tsung-Yi Lin, Yen-Chen Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Yifan Lu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun Mo, Seungjun Nah, Yashraj Narang, Abhijeet Panaskar, Lindsey Pavao, Trung Pham, Morteza Ramezanali, Fitsum Reda, Scott Reed, Xuanchi Ren, Haonan Shao, Yue Shen, Stella Shi, Shuran Song, Bartosz Stefaniak, Shangkun Sun, Shitao Tang, Sameena Tasmeen, Lyne Tchapmi, Wei-Cheng Tseng, Jibin Varghese, Andrew Z. Wang, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Jiashu Xu, Dinghao Yang, Xiaodong Yang, Haotian Ye, Seonghyeon Ye, Xiaohui Zeng, Jing Zhang, Qinsheng Zhang, Kaiwen Zheng, Andrew Zhu, and Yuke Zhu. World simulation with video foundation models for physical ai. *arXiv preprint arXiv:2511.00062*, 2025.

Jiawei Ren, Michal Tyszkiewicz, Jiahui Huang, and Zan Gojcic. Tokengs: Decoupling 3d gaussian prediction from pixels with learnable tokens. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6121–6132. IEEE, 2025.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

Amirmojtaba Sabour, Sanja Fidler, and Karsten Kreis. Align your flow: Scaling continuous-time flow map distillation. *Advances in Neural Information Processing Systems*, 38:146459–146512, 2026.

Junyoung Seo, Hyunwook Choi, Minkyung Kwon, Jinhyeok Choi, Siyoon Jin, Gayoung Lee, Junho Kim, JoungBin Lee, Geonmo Gu, Dongyoon Han, Sangdoo Yun, Seungryong Kim, and Jin-Hwa Kim. Seoul World Model: Grounding World Simulation Models in a Real-World Metropolis. In *European Conference on Computer Vision (ECCV)*, 2026.

Tianchang Shen, Sherwin Bahmani, Kai He, Sangeetha Grama Srinivasan, Tianshi Cao, Jiawei Ren, Ruilong Li, Zian Wang, Nicholas Sharp, Zan Gojcic, et al. Lyra 2.0: Explorable generative 3d worlds. *arXiv preprint arXiv:2604.13036*, 2026.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2021.

Wenqiang Sun, Haiyu Zhang, Haoyuan Wang, Junta Wu, Zehan Wang, Zhenwei Wang, Yunhong Wang, Jun Zhang, Tengfei Wang, and Chunchao Guo. Worldplay: Towards long-term geometric consistency for real-time interactive world modeling. *arXiv preprint arXiv:2512.14614*, 2025.

Robbyant Team, Zelin Gao, Qiuyu Wang, Yanhong Zeng, Jiapeng Zhu, Ka Leong Cheng, Yixuan Li, Hanlin Wang, Yinghao Xu, Shuailei Ma, Yihang Chen, Jie Liu, Yansong Cheng, Yao Yao, Jiayi Zhu, Yihao Meng, Kecheng Zheng, Qingyan Bai, Jingye Chen, Zehong Shen, Yue Yu, Xing Zhu, Yujun Shen, and Hao Ouyang. Advancing open-source world models, 2026. URL https://arxiv.org/abs/2601.20540.

Alexander Veicht, Sunghwan Hong, Daniel Barath, and Marc Pollefeys. Zipsplat: Fewer gaussians, better splats. *arXiv preprint arXiv:2606.05102*, 2026.

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

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025a.

Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025b.

Zile Wang, Zexiang Liu, Jiaxing Li, Kaichen Huang, Baixin Xu, Fei Kang, Mengyin An, Peiyu Wang, Biao Jiang, Yichen Wei, et al. Matrix-game 3.0: Real-time and streaming interactive world model with long-horizon memory. *arXiv preprint arXiv:2604.08995*, 2026.

Jay Zhangjie Wu, Yuxuan Zhang, Haithem Turki, Xuanchi Ren, Jun Gao, Mike Zheng Shou, Sanja Fidler, Zan Gojcic, and Huan Ling. Difix3d+: Improving 3d reconstructions with single-step diffusion models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26024–26035. IEEE, 2025.

Tong Wu, Shuai Yang, Ryan Po, Yinghao Xu, Ziwei Liu, Dahua Lin, and Gordon Wetzstein. Video world models with long-term spatial memory. *Advances in Neural Information Processing Systems*, 38:49371–49393, 2026a.

Yu Wu, Minsik Jeon, Jen-Hao Rick Chang, Oncel Tuzel, and Shubham Tulsiani. Rayrope: Projective ray positional encoding for multi-view attention. In *European Conference on Computer Vision (ECCV)*, 2026b.

Haofei Xu, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. Depthsplat: Connecting gaussian splatting and depth. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Yuxue Yang, Lue Fan, Ziqi Shi, Junran Peng, Feng Wang, and Zhaoxiang Zhang. Neoverse: Enhancing 4d world model with in-the-wild monocular videos. *arXiv preprint arXiv:2601.00393*, 2026.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations*, volume 2025, pp. 83048–83077, 2025.

Jung Yi, Minjae Kim, Paul Hyunbin Cho, Wooseok Jang, Sangdoo Yun, and Seungryong Kim. Worldkv: Efficient world memory with world retrieval and compression, 2026. URL https://arxiv.org/abs/2605.22718.

Tianwei Yin, Qiang Zhang, Richard Zhang, William T Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast autoregressive video diffusion models. In *CVPR*, 2025.

Mijin Yoo, In Cho, Subin Jeon, Jiwoo Lee, Eunbyung Park, and Seon Joo Kim. Scenes as objects, not primitives: Instance-structured 3d tokenization from unposed views, 2026. URL https://arxiv.org/abs/2606.29513.

Cheng Zhang, Boying Li, Meng Wei, Yan-Pei Cao, Camilo Gambardella, Dinh Phung, and Jianfei Cai. Unified camera positional encoding for controlled video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 38027–38037, 2026.

Jinjing Zhao, Fangyun Wei, Zhening Liu, Hongyang Zhang, Chang Xu, and Yan Lu. Spatia: Video generation with updatable spatial memory. In *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, 2026.

Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images, 2018. URL https://arxiv.org/abs/1805.09817.

Dong Zhuo, Wenzhao Zheng, Jiahe Guo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Streaming 4d visual geometry transformer. *arXiv preprint arXiv:2507.11539*, 2025.

APPENDIX

A PRELIMINARIES

A.1 VIDEO DIFFUSION MODELS

Flow Map Formulation.. Video diffusion models (Yang et al., 2025; Wan et al., 2025; Ali et al., 2025; Agarwal et al., 2026; HaCohen et al., 2026) estimate the latent space of pretrained 3D VAE from pure Gaussian noise. Given video $x \sim p_{\text{data}}$, the 3D VAE encoder $\mathcal{E}$ and decoder $\mathcal{D}$ yield $z_0 = \mathcal{E}(x)$ and $\hat{x} = \mathcal{D}(z_0)$ where z_0 is the clean latent of video sample and $\hat{x}$ is decoded video. The latent state follows the probability-flow ODE $\frac{\mathrm{d}z_t}{\mathrm{d}t} = \mathbf{v}(z_t, t)$, whose flow map (Sabour et al., 2026) transports states from time t to r:

$$\begin{aligned} \Phi_{r \leftarrow t}(z_t) &= z_t - \int_r^t \mathbf{v}(z_\tau, \tau)\, \mathrm{d}\tau \\ &= z_t - (t-r)\mathbf{u}(z_t, r, t), \end{aligned} \tag{8}$$

where $0 \le r < t \le 1$ and u denotes the average velocity. In the infinitesimal limit $r \to t$, $u(z_t, r, t) \to v(z_t, t)$, recovering the instantaneous velocity modeled by flow matching. For training samples $z_t = (1-t)z_0 + t\epsilon$, with $\epsilon \sim \mathcal{N}(0, I)$ and $t \sim \mathcal{U}(0, 1)$, a velocity predictor v_θ is trained using

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{x,\epsilon,t}\left[\|\mathbf{v}_\theta(z_t, t) - (\epsilon - z_0)\|_2^2\right]. \tag{9}$$

On-Policy Distillation.. On-policy distillation (Gu et al., 2026) alleviates the train–test discrepancy by optimizing the student on samples generated from its own inference trajectory under the guidance of a pretrained teacher. Given Gaussian noise z, the student first generates $\hat{z}_0 = f_\theta(z)$ through its inference process. Following DMD-style reverse-KL optimization, the generated sample is re-noised at $t \in [0, T]$ as $z_t = (1-t)\hat{z}_0 + t\epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$, and the student is optimized using the discrepancy between the real and generated score functions:

$$\nabla_\theta \mathcal{L}_{\text{DMD}} = -\mathbb{E}_{t,z}\left[\left(s_{\text{real}}(z_t, t) - s_{\text{fake}}(z_t, t)\right) \frac{\partial f_\theta(z)}{\partial \theta}\right]. \tag{10}$$

A.2 3D GAUSSIAN SPLATTING

3D Gaussian Splatting (Kerbl et al., 2023) represents a scene as a collection of anisotropic 3D Gaussians $\mathcal{G} = \{(\mu_i, \Sigma_i, \alpha_i, c_i)\}_{i=1}^N$, where $\mu_i \in \mathbb{R}^3$ denotes the Gaussian center, $\Sigma_i \in \mathbb{R}^{3\times3}$ its covariance, α_i its opacity, and c_i its view-dependent appearance. The covariance is parameterized by a rotation and scale as $\Sigma_i = R_i S_i S_i^\top R_i^\top$. Given a camera viewpoint, the 3D Gaussians are projected onto the image plane and rendered via depth-ordered alpha compositing. The resulting pixel color is computed as

$$C = \sum_{i=1}^{N} c_i \alpha_i \prod_{j=1}^{i-1}(1 - \alpha_j), \tag{11}$$

where c_i and α_i denote the color and effective opacity of the i-th projected Gaussian at the corresponding pixel.

B IMPLEMENTATION DETAILS

B.1 GAUSSIAN MEMORY UPDATER

Architecture. At step k, the updater L3G-Mem consumes the current window $\mathcal{W}_k = \{\mathbf{I}_k, \mathbf{C}_k, \mathbf{z}_k\}$ and the persistent memory $\mathcal{M}_{k-1} = \{\mathbf{G}_{k-1}, \mathbf{T}_{k-1}\}$.

We feed the frames and cameras to the geometry backbone (Lin et al., 2025) and extract $1{,}536$-dimensional geometry features $\mathbf{F}_k = \{\mathbf{F}_k^l\}_{l=1}^L$ from layers 39, 29, and 19 ($L = 3$); the cross-attention layers attend to geometric tokens in this order. The backbone patch size is 14. In parallel, two convolutional encoders map the VAE latents $\mathbf{z}_k$ to 128-dimensional VAE tokens.

Table 5: **Training configuration** of L3G-Mem, L3G-Gen, and L3G-Gen Distilled. For L3G-Gen Distilled, values separated by "/" denote the flow-map and on-policy distillation stages, respectively.

Component	L3G-Mem	L3G-Gen	L3G-Gen Distilled
Optimizer	AdamW	AdamW	AdamW
Learning rate	1×10^{-4}	1×10^{-4}	5×10^{-5} / 2×10^{-6}
Optimizer betas	$(0.9, 0.95)$	–	$(0.9, 0.95)$ / $(0, 0.999)$
Weight decay	0.05	0.01	-
Batch size	8	16	16 / 8
EMA decay	–	–	0.999 / 0.99
Gradient clipping	1.0	1.0	–
Training steps	66K	1.1K	1.0K / 1.6K
Total parameters	1.7B	2B	2B
Trainable parameters	1.6B	75.8M	210.3M
RGB Resolution	364×210	832×480	832×480
VAE latent resolution	104×60	104×60	104×60

K-means on the $l = 39$ geometry features selects the indices that initialize the spawn queries $\mathbf{Q}_k$ and that gather the aligned queries from the remaining geometry layers, the RGB grid, and the VAE grid. The selected RGB and VAE queries additionally cross-attend to the complete RGB and VAE token sets.

The spawning stream and refinement stream each contain three blocks of $1{,}536$-dimensional cross-attention, self-attention, and global self-attention followed by a feed-forward network. The joint decoder concatenates the $1{,}536$-dimensional scene token, the 128-dimensional RGB token, and the 128-dimensional VAE token, and each resulting memory token is decoded into 32 Gaussians, following (Veicht et al., 2026). The memory tokens $\mathbf{T}$ are cached in the persistent memory set $\mathcal{M}$.

Spawn Mask. We compute the spawn mask $\mathbf{B}_k$ of Sec. 3.1 on the patch grid of F_k^3. For each patch p, we average the accumulated opacity and the per-pixel RGB MSE over its 14×14 pixels to obtain $A_{k,p}$ and $E_{k,p}$, and set:

$$\mathbf{B}_{k,p} = \mathbb{1}(E_{k,p} > \tau_E \ \vee \ A_{k,p} < \tau_\alpha)\,. \tag{12}$$

Low-opacity patches thus propose Gaussians for uncovered regions, whereas high-error patches propose additional Gaussians in regions that are covered but poorly reproduced by the current memory. The number of spawning queries N_{q}, which serves as the number of clusters in K-means, is set in proportion to the fraction of masked patches in the current window, ranging linearly from $N_{\min}$ when no patch is masked to $N_{\max}$ when all patches are masked. Every query produces $S = 32$ Gaussians, and at $k = 1$, where the memory is empty, we use $N_{\max}$.

Gaussian Retrieval. At $k > 1$, we rasterize $\mathbf{G}_{k-1}$ once in the current context cameras to obtain RGB, accumulated opacity, expected depth, and the projected conic of every Gaussian. For each Gaussian G_n projected into view v, we probe the pixels at the ellipse center and along the positive and negative directions of both principal axes. A probe pixel $\mathbf{p}$ is considered visible only if it lies inside the image, the local contribution and the accumulated opacity at $\mathbf{p}$ exceed their visibility thresholds:

$$d_{n,v} \leq D_v(\mathbf{p}) + \epsilon_{\mathrm{abs}} + \epsilon_{\mathrm{rel}} |D_v(\mathbf{p})|, \tag{13}$$

where $d_{n,v}$ is the projected depth of Gaussian G_n in view v and $D_v(\mathbf{p})$ is the rendered expected depth at $\mathbf{p}$. This depth test rejects Gaussians lying behind the current memory surface without requiring input or ground-truth depth. We then retrieve all memory tokens with at least one visible Gaussian and refine them.

Residual Memory Update. For each Gaussian found visible by the retrieval procedure, the residual head predicts an update δ_n and a gate $g_n \in [0, 1]$. Writing a Gaussian as $G_n = (\boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, \mathbf{c}, \mathbf{z})$, whose covariance $\boldsymbol{\Sigma}$ is parameterized by a scale $\mathbf{s}$ and a quaternion $\mathbf{q}$ and whose VAE feature is $\mathbf{z}$, we update $(\boldsymbol{\mu}, \mathbf{s}, \mathbf{q}, \alpha, \mathbf{c}, \mathbf{z})$ as

$$\boldsymbol{\mu}' = \boldsymbol{\mu} + \beta_\mu g \tanh(\delta_\mu), \tag{14}$$
$$\mathbf{s}' = \mathbf{s} \odot \exp(\beta_s g \tanh(\delta_s)), \tag{15}$$
$$\mathbf{q}' = \mathrm{norm}(\mathbf{q} + \beta_q g \tanh(\delta_q)), \tag{16}$$
$$\alpha' = \sigma(\mathrm{logit}(\alpha) + \beta_\alpha g \tanh(\delta_\alpha)), \tag{17}$$
$$\mathbf{c}' = \mathbf{c} + \beta_c g \tanh(\delta_c), \tag{18}$$
$$\mathbf{z}' = \mathbf{z} + \beta_z g_z \tanh(\delta_z), \tag{19}$$

where g_z is an independently predicted VAE-feature gate. The persistent scene and VAE tokens are also updated residually.

VAE Feature Lifting. Using a simple convolutional network, we embed 16-channel VAE latents $\mathbf{z}_k$ into 128-dimensional tokens, resulting in spatially aligned feature with geometry feature. The K-means representative indices from spawning query tokens initialize the VAE queries for newly spawned tokens. A single cross-attention layer lets the VAE queries attend to all current VAE tokens. The joint decoder therefore receives a concatenated vector from scene token and attended VAE token. It emits both the $S = 32$ Gaussians and a 16-dimensional VAE latent feature for each Gaussian. VAE features are rendered using the same geometry, opacity, and compositing as RGB. We detach Gaussian geometry in this feature rasterizer, so the VAE objective supervises the lifted features and shared token representation without moving Gaussian centers through the VAE render.

Training. We train on K consecutive video chunks, without frame overlap, each containing W context views and held-out interpolation views. Ground-truth cameras are provided to both the geometry backbone and renderer. We place every rollout in a single coordinate system whose origin and orientation are defined by the first context camera of chunk 1. A single scale, equal to the reciprocal of the maximum pairwise camera-center distance over the complete clip, is then applied to all camera translations and depths in every chunk. For depth supervision, we use depth maps predicted by Depth Anything 3 (Lin et al., 2025).

We use frozen VAE latents (NVIDIA et al., 2025) as input, where every 4-th RGB frame of VAE latent is used for input video frames. Training follows the online recurrence used at inference. After each chunk, to ensure temporal consistency, the keys and values in cross and local self-attention layer of the spawning stream are cached and remain attached within the sampled rollout. At chunk k, the updated global memory is rendered into the target views of chunks $1, \ldots, K$.

The reconstruction losses are defined as:

$$\begin{aligned}
\mathcal{L}_{\mathrm{rgb}} &= \mathrm{mean}(\|\hat{\mathbf{I}} - \mathbf{I}^{\mathrm{tgt}}\|_1), \\
\mathcal{L}_{\mathrm{lpips}} &= \mathrm{LPIPS}(\hat{\mathbf{I}}, \mathbf{I}^{\mathrm{tgt}}), \\
\mathcal{L}_{\mathrm{depth}} &= \mathrm{mean}(\|\hat{\mathbf{D}} - \mathbf{D}^{\mathrm{tgt}}\|_1), \\
\mathcal{L}_{\mathrm{vae}} &= \mathrm{mean}(\|\hat{\mathbf{z}} - \mathbf{z}^{\mathrm{tgt}}\|_2^2),
\end{aligned} \tag{20}$$

where $\hat{\mathbf{I}}$, $\hat{\mathbf{D}}$, and $\hat{\mathbf{z}}$ are the RGB, depth, and VAE feature renderings of the memory, and $\mathbf{I}^{\mathrm{tgt}}$, $\mathbf{D}^{\mathrm{tgt}}$, and $\mathbf{z}^{\mathrm{tgt}}$ are the corresponding target images, Depth Anything 3 depth maps, and VAE latents. Here, the mean is taken over the corresponding pixels or latent elements across target views, and LPIPS is averaged over target images. Following TokenGS (Ren et al., 2026), we additionally use a visibility loss $\mathcal{L}_{\mathrm{vis}}$ that encourages each Gaussian to lie within at least one observed input-view frustum. We omit the window index from the loss terms for simplicity. At each rollout step, we evaluate the weighted objective in Eq. 5 and minimize its average over the K chunks. Although previous memory states are detached, supervision from earlier target views still constrains the current spawning and refinement outputs, encouraging consistency with previously reconstructed content.

We initialize our memory updater L3G-Mem from ZipSplat (Veicht et al., 2026), while the new refine stream, global self-attention, and convolutional network for VAE latent are trained from scratch. We train L3G-Mem end-to-end, including the geometric foundation model. When widening the Gaussian head for jointly decoding Gaussian primitives and VAE features, we copy all pretrained weights exactly and zero-initialize its new VAE-input columns.

B.2 Video Diffusion Model Details

B.2.1 Teacher Model.

Architecture. We build the teacher model on Cosmos-Predict2.5-2B (Ali et al., 2025) with the Wan VAE latent representation. The pretrained DiT backbone is kept frozen, while LoRA adapters, the UCPE (Zhang et al., 2026) camera-conditioning branches, and the widened input projection are trained. For reconstruction-guided generation, the noisy video latent is channel-concatenated with the VAE feature rendered from the current 3D Gaussian memory, together with its validity mask and conditioning indicator. We additionally retrieve up to three previously observed latents according to camera-pose proximity and provide them as clean reference tokens.

Training. We train the teacher using rectified flow matching on DL3DV (Ling et al., 2024) and RealEstate10K (Zhou et al., 2018). The anchor frame is provided as a clean image latent and excluded from the denoising loss, while the remaining target latents are jointly denoised with full bidirectional attention. During training, we randomly drop text and reconstruction conditioning and vary the number of retrieved reference latents to improve robustness to different conditioning configurations. The teacher is fine-tuned with rank-16 LoRA adapters (Hu et al., 2021) together with the UCPE modules and input projection.

Inference. At inference, the teacher generates one 81-frame clip at a time using 16 denoising steps with classifier-free guidance. After each generated clip, the reconstructed scene memory is updated from the generated observations and rendered under the cameras of the next clip. Reference latents are retrieved only from the generated history, and the final generated frame is used to initialize the following clip.

B.2.2 Student Model.

Architecture. The student retains the same backbone, reconstruction conditioning, reference conditioning, and UCPE camera modules (Zhang et al., 2026) as the teacher, but converts the bidirectional DiT into a chunk-causal autoregressive model. Each training clip is divided into short latent chunks, and the current chunk attends to compressed long-term history, retrieved reference latents, and the most recent full-resolution chunks. Older history is spatially compressed, while the two most recent chunks remain at the original latent resolution. At inference, the corresponding keys and values are cached independently for the standard self-attention and UCPE attention branches.

Flow-Map Distillation. We first initialize the student through flow-map distillation following AnyFlow (Gu et al., 2026). Unlike the teacher, which predicts the instantaneous denoising velocity at a single noise level, the student is conditioned on both the source and destination noise levels and learns transitions between arbitrary points along the denoising trajectory. Training combines standard diffusion, direct consistency, and intermediate flow-map transitions. We initialize the student from the teacher and increase the LoRA rank to 128, while preserving the teacher initialization for the shared camera and reconstruction-conditioning modules.

On-Policy Distillation and Inference. We further train the student with on-policy distribution matching using samples generated from its own autoregressive rollouts. A frozen teacher provides the real-data score, while a trainable copy of the teacher estimates the student distribution, and the flow-map objective (Boffi et al., 2024) is retained during this stage for stable optimization. At inference, the student generates three latent frames per autoregressive step using a few flow-map evaluations without classifier-free guidance. The generated latents are continuously added to the reference pool and periodically used to update the 3D Gaussian memory, which is then rendered to provide reconstruction guidance for subsequent chunks.

B.3 Baselines

Depth and Scale Alignment. Our evaluation considers a single-image-to-video setting, where the explicit 3D-memory baselines (Ren et al., 2025; Zhao et al., 2026; Wu et al., 2026a; Shen et al., 2026; Lee et al., 2026; Yang et al., 2026) must initialize their scene representation from a single input image. This introduces a scale ambiguity between the monocularly reconstructed 3D scene and

the metric camera trajectory used for evaluation, which can lead to incorrectly rendered target views even when the reconstructed geometry is otherwise accurate. To ensure a consistent metric scale across methods, we use Depth Anything 3 GIANT-1.1-NESTED (Lin et al., 2025) as the depth estimator for all explicit 3D-memory baselines. The predicted metric depth is used to place the reconstructed 3D representation at the same scale as the evaluation camera trajectory. For Gaussian-based methods, NeoVerse (Yang et al., 2026) and L3G-World, we additionally align the depth rendered from the initially predicted Gaussians with the metric depth estimated from the input image.

Baseline Implementations. The official implementations of SPMem (Wu et al., 2026a) and 3DScenePrompt (Lee et al., 2026) initialize their first generation chunk using multiple input views or video frames, from which they reconstruct a 3D point-cloud representation and render it under subsequent target cameras. Since our benchmark provides only a single input image, we modify their initialization stage to reconstruct the 3D representation from the first frame alone and render it to the target cameras of the first chunk. For subsequent autoregressive generation chunks, we retain the behavior of the official implementations.

NeoVerse (Yang et al., 2026) is designed for single-image world generation, but its released implementation does not provide the complete autoregressive generation pipeline required by our long-horizon evaluation. We therefore build an autoregressive inference pipeline on top of the official implementation, repeatedly reconstructing and updating its Gaussian representation from generated observations and rendering the resulting memory under the target cameras for subsequent generation.

Yume 1.5 (Mao et al., 2026) and Matrix-Game 3.0 (Wang et al., 2026) are action-conditioned world models that use discrete keyboard-style controls rather than continuous camera parameters. To evaluate them along the same camera trajectories as the other methods, we convert the provided camera motion into the corresponding sequence of discrete action inputs and use these actions for autoregressive generation.

B.4 Evaluation

Datasets. We evaluate L3G-World and all baselines on the test splits of DL3DV (Ling et al., 2024) and RealEstate10K (Zhou et al., 2018). To explicitly evaluate long-horizon consistency under revisitation, we construct 481-frame loop-closure trajectories. Starting from the input view, the camera first follows an exploration trajectory and subsequently retraces the same trajectory in reverse, eventually returning to the initial camera pose.

Self-Consistency. The loop-closure trajectory provides pairs of generated frames corresponding to the same camera pose but generated at different points in the autoregressive sequence. We pair each frame on the outgoing trajectory with its corresponding frame on the returning trajectory and compute PSNR, SSIM, and LPIPS between them. We report the average over all corresponding frame pairs as our self-consistency metrics. This evaluation directly measures whether scene appearance and structure are preserved after long-horizon generation and subsequent loop closure.

Camera Controllability. Following (Seo et al., 2026), we estimate the camera trajectory from each generated video using the feed-forward geometry estimator (Lin et al., 2025), and compare the estimated trajectory against the ground-truth camera trajectory used for generation. We report translation error (TransErr) after resolving the global scale ambiguity of the estimated trajectory, together with rotation error (RotErr) in degrees.

Efficiency. For explicit 3D-memory methods, we measure the number of stored 3D primitives and their memory footprint after long-horizon generation, together with the average reconstruction time per generation chunk. We further measure target-view rendering throughput in two settings: RGB rendering FPS and VAE-space conditioning FPS. For methods whose 3D memory stores only RGB appearance, VAE-space conditioning includes both target-view RGB rendering and subsequent VAE encoding. In contrast, L3G-World directly renders the stored VAE features into the target-view latent space without RGB rendering or re-encoding. All efficiency measurements are performed on a single NVIDIA A100 GPU.

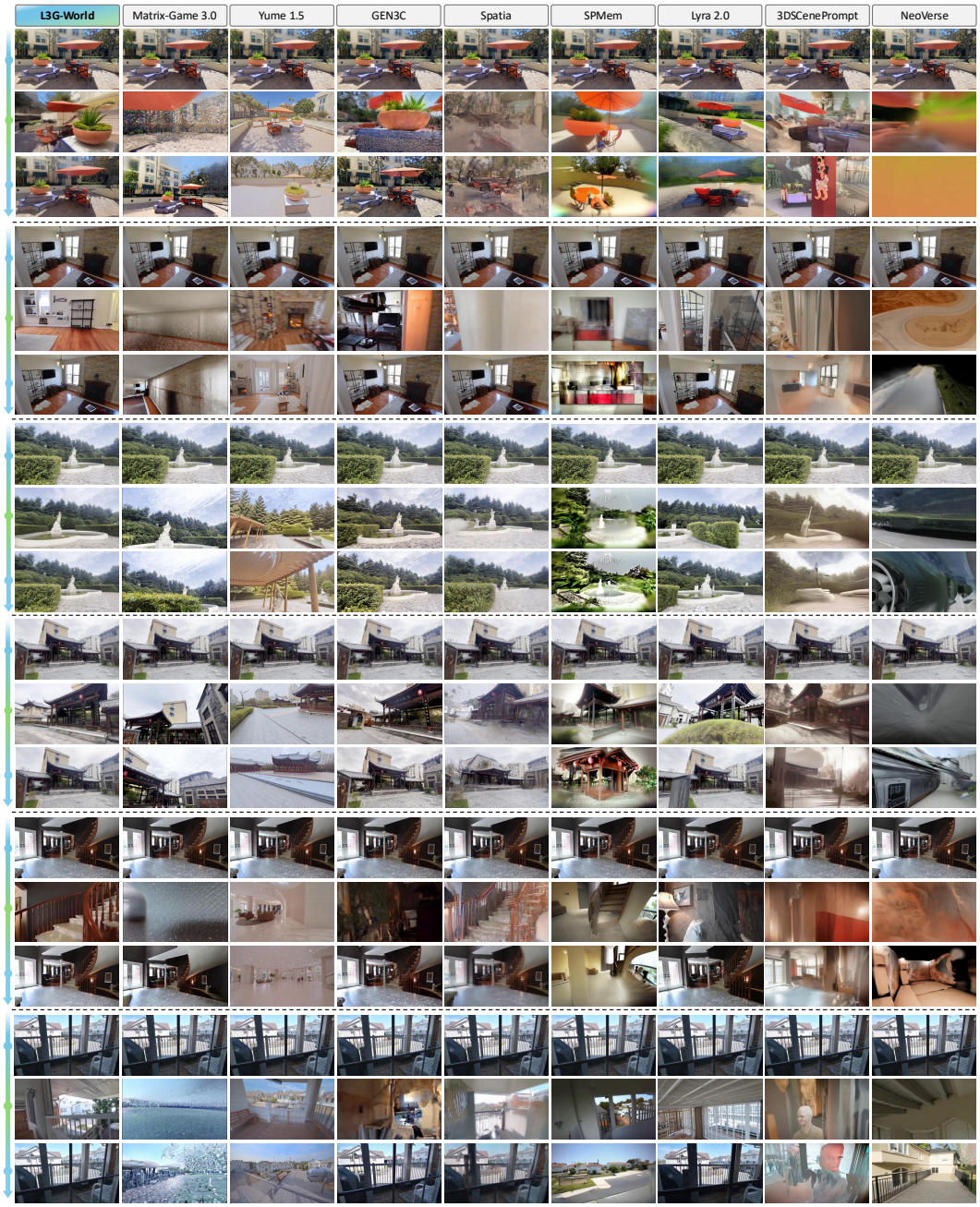


Figure 4: **Additional qualitative results of video generation on DL3DV and RealEstate10K.** For each scene, all methods start from the same input view (top), explore unseen regions (middle), and return to the initial camera location (bottom). L3G-World preserves scene structure and appearance upon loop closure, whereas the baselines exhibit drift, blur, or rendering artifacts.

C Qualitative Results

Fig. 4 extends the comparison in Fig. 3 to additional scenes from DL3DV and RealEstate10K under the same long-horizon loop-closure trajectories. Starting from the same input view, each method explores previously unseen regions and eventually returns to the initial camera location, so that the final frame directly reveals whether the generated world remains consistent after long-horizon exploration. The baselines exhibit distinct failure modes. Temporal-context-based methods, Yume 1.5 and Matrix-Game 3.0, generally follow the camera motion during exploration, but fail to preserve

Table 6: **Online NVS evaluation.** We evaluate online reconstruction on DL3DV and on RealEstate10K, reporting RGB and VAE latent novel view synthesis quality together with memory size, reconstruction latency, and VAE rendering FPS.

Method	Resolution ($H \times W$)	VAE MSE ↓	VAE Cosine Sim. ↑	# Primitives (K) ↓	Memory (MiB) ↓	Recon. s/chunk ↓	VAE Render FPS ↑	PSNR ↑	SSIM ↑	LPIPS ↓
DL3DV										
ReCoSplat	480×480	**0.118**	**0.858**	2,085	198.88	0.809	3.0	**21.53**	**0.707**	**0.318**
ReCoSplat	480×832	0.664	0.344	11,550	1101.48	4.179	2.7	12.41	0.438	0.711
OF^{3}GS	405×832	0.452	0.522	8,987	479.98	2.244	3.0	14.03	0.418	0.606
Ours	480×832	0.142	0.823	**39**	**14.11**	**0.314**	**735.3**	17.52	0.526	0.537
RealEstate10K										
ReCoSplat	480×480	**0.073**	**0.934**	1,237	117.93	0.770	3.1	**24.88**	**0.869**	**0.208**
ReCoSplat	480×832	0.576	0.568	5,818	554.85	4.059	2.9	13.56	0.622	0.640
OF^{3}GS	405×832	0.487	0.634	6,649	355.07	1.876	3.0	14.15	0.590	0.566
Ours	480×832	0.125	0.866	**25**	**8.94**	**0.310**	**1333.3**	19.68	0.706	0.428

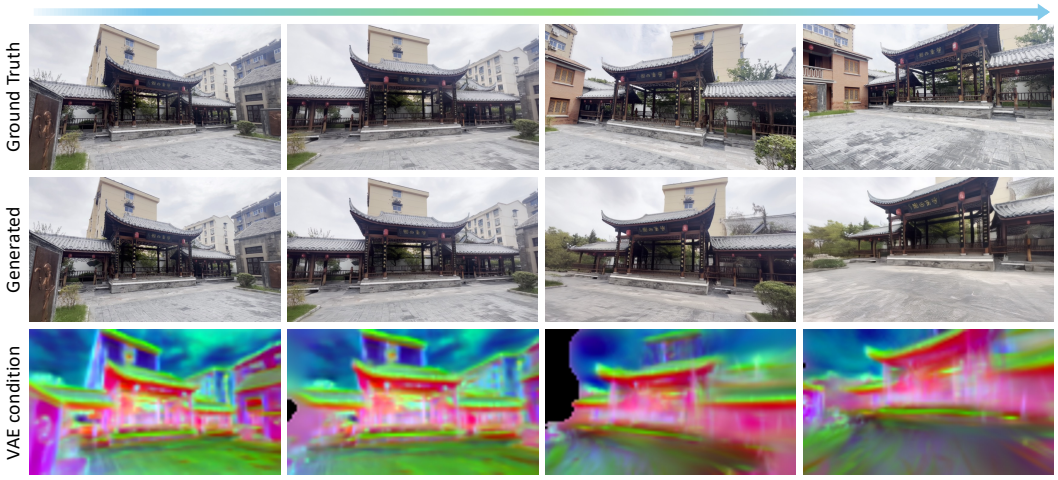


Figure 5: **Visualization of latent 3D Gaussian memory conditioning for video generation.** We visualize the ground-truth and generated video, with target-view rendered VAE condition from latent 3D Gaussian memory, illustrating that spatially aligned latent memory provides consistent conditioning for video generation.

spatial consistency upon return: the final frame often shows a layout or appearance different from the input view, and in some cases even a different viewpoint, indicating that camera control also degrades over the long trajectory. Among explicit 3D memories, SPMem tends to hallucinate inconsistent content once the camera leaves the initially observed region, and NeoVerse progressively collapses into blurry textures over the long trajectory. Among the baselines that return to the starting scene more reliably, GEN3C recovers the input view quite well, but struggles to extrapolate plausible content in regions unseen in the input view, producing distorted or incomplete frames during exploration. Lyra 2.0 returns reasonably close to the input view, but its final frame is still not fully spatially aligned with it, indicating drift from the original scene. In contrast, L3G-World both generates plausible content in unseen regions and returns to a scene spatially aligned with the input view in both structure and appearance, as its latent 3D Gaussian memory anchors previously generated content while continually refining it throughout exploration.

D Comparison with Online 3DGS Methods

Existing online 3D Gaussian Splatting methods provide strong feed-forward reconstruction and could, in principle, serve as an external 3D memory for world generation. However, these methods are primarily designed for high-fidelity novel-view reconstruction, whereas a memory for real-time world generation must satisfy a different operating regime: it must be continuously updated throughout autoregressive generation, remain compact as exploration proceeds, and render the conditioning

signal for every subsequent video chunk with very low latency. In particular, a dense representation that is effective for reconstruction can introduce substantial memory and rendering overhead when repeatedly queried by a video diffusion model. Our goal is therefore not to outperform dedicated online 3DGS methods as a standalone reconstruction system, but to construct a sufficiently faithful 3D memory that can be efficiently updated and directly consumed by L3G-Gen.

To characterize this trade-off quantitatively, we evaluate latent 3D Gaussian memory as an online reconstruction model on DL3DV (Ling et al., 2024) and additionally on RealEstate10K (Zhou et al., 2018). We process each sequence online in streaming chunks of nine frames. After each memory update, we render the current 3D representation from held-out target viewpoints and evaluate both RGB- and VAE-space reconstruction. We report PSNR, SSIM, and LPIPS for RGB reconstruction, and mean squared error (MSE) and cosine similarity between the rendered and ground-truth VAE latents. We additionally measure the number of stored primitives, memory footprint, reconstruction latency per chunk, and VAE rendering throughput to evaluate the efficiency required for repeated memory updates and queries during world generation.

As shown in Tab. 6, latent 3D Gaussian memory is substantially more efficient than dedicated online 3DGS methods (Cheng et al., 2026; Chen et al., 2026a). Compared with ReCoSplat Cheng et al. (2026) at 480×480, latent 3D Gaussian memory uses approximately 49–$53\times$ fewer primitives and 13–$14\times$ less memory across the two datasets, while updating the memory about $2.5\times$ faster. Moreover, since these methods store only RGB Gaussians, conditioning a latent video generator requires rendering RGB images and re-encoding them with the VAE, whereas latent 3D Gaussian memory directly renders its stored VAE features, achieving 245–$430\times$ higher VAE rendering throughput. Despite this compact representation, latent 3D Gaussian memory retains meaningful reconstruction fidelity: at our target resolution of 480×832, it substantially improves both RGB- and VAE-space reconstruction over ReCoSplat at the same resolution on both datasets and remains competitive with OF^{3}GS (Chen et al., 2026a) while operating at a higher resolution. ReCoSplat at the lower 480×480 resolution achieves higher reconstruction fidelity, which is expected from a method specifically optimized for novel-view synthesis. We therefore view these reconstruction metrics not as a standalone reconstruction benchmark, but as an objective measure that latent 3D Gaussian memory preserves sufficiently accurate spatial and latent information while satisfying the much stricter memory and throughput requirements of real-time world generation.

More importantly, the utility of this reconstruction is determined by how the memory conditions subsequent video generation. Fig. 5 visualizes, for each target camera, the ground-truth view, the generated view, and the VAE condition directly rendered from latent 3D Gaussian memory. The rendered latent remains spatially aligned with the target camera and preserves the structure of previously observed regions. By directly conditioning L3G-Gen on these spatially anchored VAE features, the model can generate novel content while remaining consistent with the persistent scene memory. Thus, latent 3D Gaussian memory is not merely an efficient reconstruction module: it provides a compact, continuously updated, and directly renderable latent interface between the accumulated 3D scene and autoregressive video generation.

E Limitations and Future Work

Despite its efficiency, L3G-World has several limitations that suggest directions for future work. First, latent 3D Gaussian memory is primarily designed for static scenes and does not explicitly model independently moving objects. Extending the persistent memory to disentangle dynamic content from static geometry would enable more general interactive environments. Second, although the memory grows with explored spatial extent rather than generation length, its size can still increase under extremely large-scale exploration; adaptive pruning or hierarchical memory representations could further improve scalability. Finally, while the refinement mechanism can progressively correct inaccurate Gaussians, errors introduced during long-horizon autoregressive generation may still propagate to subsequent memory updates.