ENGRAM: Reliability-Weighted Latent Fusion and Occlusion-Aware Layered Readout for Video World Models

Chenhao Wu[1,*] Zhaolong Su[2,*] Yang Liu[3,*] Tianrui Su[4]
Yingguang Yang[3] Shang Luo[3] Yancheng Chen[1] Chongyang Zhang[5]
Hao Zheng[5] Kefu Xu[3] Congjing Ran[6] Bin Chong[3,†]

[1]University of the Chinese Academy of Sciences [2]Jilin University [3]Peking University
[4]University of New South Wales [5]Fullive-AI [6]Wuhan University

[*]Equal contribution. [†]Corresponding author.

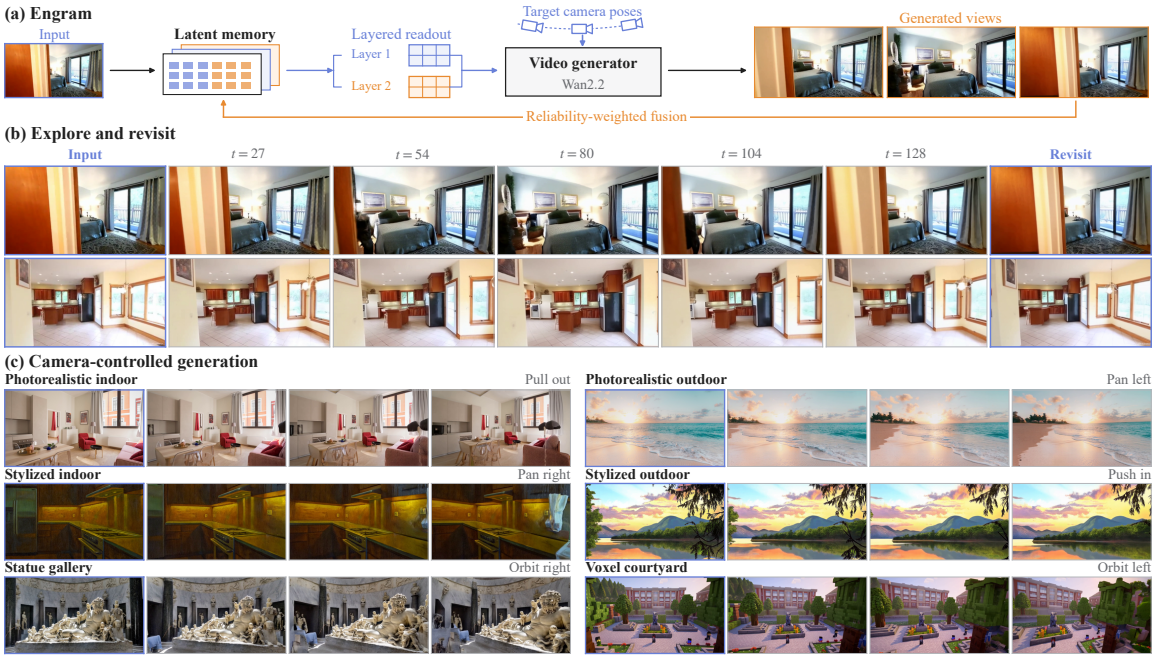

Figure 1. **Explore, remember, and revisit with ENGRAM.** (a) Persistent latent memory with layered readout and reliability-weighted writeback. (b) Closed-loop revisits on RealEstate10K. (c) Camera-controlled generation across diverse WorldScore scenes.

Abstract

Camera-conditioned video world models aim to generate 3D-consistent videos from a single image along specified camera trajectories. However, set-union updates in latent spatial memory leave repeated observations unconsolidated, while frontmost-only readout limits simultaneous access to stored evidence at different depths. To address these limitations, we present ENGRAM*, a latent spatial memory framework combining* reliability-weighted latent fusion *and* occlusion-aware layered readout*. Specifically, recursive fusion integrates geometrically and visually compatible observations in a channel-normalized latent space, consolidating repeated evidence into a shared scene representation. Layered readout exposes multiple depth-ordered latent candidates with their reliability and availability at each target cell. This gives the generator access to information from multiple surfaces as visibility changes. Explicit camera rays provide viewpoint information independently of memory coverage, while geometric consistency constraints guide memory updates. Memory retrieval uses a single projection at latent resolution, avoiding intermediate RGB rendering and VAE re-encoding. We adapt a pretrained video generator using only 256 training scenes. On 100 RealEstate10K test scenes,* ENGRAM *improves novel-view-synthesis and closed-loop PSNRs over Spatia by 0.82 and 4.48 dB, respectively. On a 200-case WorldScore evaluation subset, it achieves an average score of 67.78, exceeding Spatia by 3.31 points. Project page:* https://wchkoutian.github.io/engram-website/.

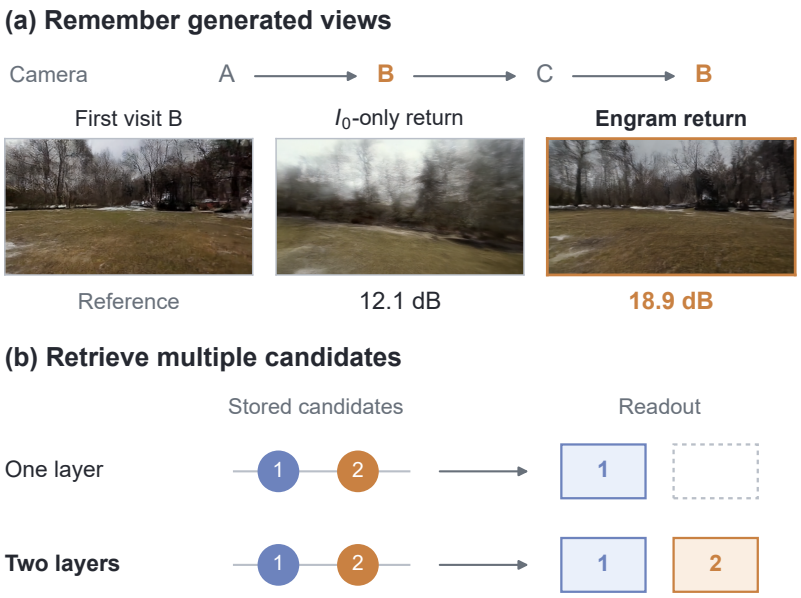

Figure 2. **Retaining and retrieving scene history.** (a) Retaining generated history improves consistency when revisiting a previously generated view. PSNR is measured against the first generated visit in one development-set example. (b) Layered readout exposes multiple stored candidates. This schematic does not attribute the improvement in (a) specifically to layered readout.

1. Introduction

Large-scale video diffusion models synthesize photorealistic sequences and provide a foundation for generative world models [8, 19, 22]. Geometric drift is what separates a world model from a per-frame generator: driven along a camera trajectory, a model produces frames that are individually plausible and jointly inconsistent, and the inconsistency compounds from chunk to chunk. Explicit spatial memory removes the drift by preserving observed geometry and appearance and by showing each newly generated chunk a view of that memory from the target camera.

Existing systems keep explicit 3D representations, including colored point clouds [11, 25] and Gaussian scene representations [2, 24], render them at every step, and pass the render back into the conditioning path through a variational autoencoder (VAE) [2, 11, 25], paying for the render and for a second lossy encode. LSM-World [20] removes the intermediate by storing scene features in the diffusion latent space, lifting latent tokens to 3D with depth-guided back-projection and reading them back with a projection at latent resolution. We retain this direct latent-space projection. Removing the render, however, does not decide what such a store should hold. *What* does a slot represent once the same surface has been observed many times? And *which* stored slot should the generator be shown when several surfaces project to the same target cell?

Set-union accumulation, the natural update for such a store, makes every valid back-projection its own record. A surface seen across a hundred overlapping views therefore occupies a hundred competing entries whose features differ only by depth noise and per-view estimation error. Nothing in the store decides which of them is right, so the decision falls to readout, and readout is where it fails: a small change in estimated depth moves the choice to a different record of the same surface, and a revisit renders that surface inconsistently with the visit that already recorded it. The failure is structural rather than occasional, because overlapping views are precisely what a memory is built to collect. Fusion resolves the competition where it can still be resolved, at write time against a shared scene representation, instead of deferring it to every query. It must also decide what not to store: a generated record can project in front of an initial surface that it never observed, and writing it there destroys the only evidence of that surface.

A second difficulty concerns which records the generator can address. Frontmost selection exposes only the nearest surface at each target cell, leaving other projected candidates unavailable to the generator in that readout. A revisit at a shallower angle, or a ray that grazes an occlusion boundary, exposes the gap, and a single-layer readout has no place to put the answer. Figure 2 illustrates two requirements: retaining generated history for revisits and exposing multiple stored candidates to the generator.

A third challenge is learning spatial-memory conditioning. Training a generator from scratch would require learning both

video synthesis and memory use. Our two-stage adaptation reuses pretrained synthesis capabilities, first learning the memory interface and then adapting the generator. Spatia also adapts a pretrained generator, using 50,000 training videos [27]. Counting optimization steps times effective batch size, our schedule uses 48,000 sample presentations versus 832,000 calculated from Spatia's reported schedule: 5.8% as many, or a 17.3-fold difference in this count. Supplementary Sec. B.1 gives the calculation and counting convention.

We introduce ENGRAM, a latent spatial memory that resolves both questions inside the read–generate–update cycle illustrated in Figure 1. *Reliability-weighted latent fusion* consolidates compatible observations within a voxel into a single slot, weighted by depth reliability and gated so that a generated record cannot enter in front of an initial surface it never observed. *Occlusion-aware layered readout* returns depth-ordered candidates per cell together with their reliability and availability, so the answer stays addressable after the nearest surface stops being the right one. Explicit *Plücker camera rays* carry camera information independently of memory coverage. Generated frames are written back through the same cycle, so a view survives its departure from the context window and is still available when the camera returns. We adapt a pretrained Wan2.2 backbone using 256 training scenes. We evaluate ENGRAM on RealEstate10K [30] and WorldScore [5], covering novel-view synthesis, closed-loop revisits, and camera-controlled world generation.

Contributions. (i) We introduce a *reliability-weighted latent fusion* operator that consolidates compatible multi-view observations of a surface into a single slot, weighted by depth reliability and gated against writing a generated record in front of an initial surface. The store stops accumulating competing records, and readout stops having to adjudicate among them. (ii) We propose an *occlusion-aware layered readout* that returns multiple depth-ordered candidates per cell together with their reliability and availability, so that evidence behind the nearest surface remains addressable to the generator as visibility changes. (iii) We demonstrate *data-efficient spatial-memory adaptation* through a two-stage strategy that learns the memory interface before adapting the generator, enabling consistent generation with limited training data.

2. Related Work

Video diffusion world models. Video diffusion transformers [8, 17, 19, 22] generate temporally coherent video and are increasingly treated as world models when conditioned on a camera trajectory. Without a persistent representation they drift geometrically over long rollouts, which motivates explicit spatial memory.

Spatial memory for consistent generation. Existing methods maintain explicit 3D representations and render target views for generation [2, 3, 6, 11, 23–25], while view-memory methods retrieve past frames [14, 29]. Volumetric and learned latent fusion have been studied for reconstruction [4, 10, 16, 21]. Our closest baseline, LSM-World [20], stores diffusion features in 3D using set-union updates and frontmost readout. We retain direct latent-space projection and introduce reliability-weighted latent fusion and occlusion-aware layered readout within the generation loop.

Camera conditioning and calibration. Plücker coordinates [18] are used for camera conditioning [9]. We use these coordinates to provide viewpoint information independently of memory coverage. Confidence calibration aligns classification probabilities with empirical correctness [7]. Our memory fusion uses geometric and temporal precision proxies to weight observations.

3. Preliminaries

Chunked autoregressive video world models. Given an initial image I_0 and a camera trajectory $\{E_k\}_{k=0}^{N}$, a video world model generates a sequence $\{I_k\}_{k=0}^{N}$ consistent with the prescribed viewpoints. Here, $E_k \in \mathrm{SE}(3)$ maps world coordinates to camera coordinates, and K_l denotes the camera intrinsics at latent resolution, with the frame index omitted for simplicity. Generation proceeds in temporally overlapping chunks; we index chunks by t and reserve k for frames, so that $E_{t,j}$ is the j-th camera of chunk t. For each chunk, the model queries a persistent spatial memory $\mathcal{M}$ from the target viewpoints, generates the corresponding video, and integrates new observations into the memory.

Backbone and conditional generation. We use Wan2.2-TI2V-5B [19] as the pretrained video generation backbone. Its VAE has a temporal compression factor of four and $C = 48$ latent channels. The VAE downsamples each spatial dimension by a factor of 16. A generated chunk comprises several latent frames $Z_{t,j}$, indexed by the same j as its cameras. We optimize a conditional velocity field using flow matching [15]. For clean target latents x and noise $\varepsilon \sim \mathcal{N}(0, I)$, the interpolation is

$x_\sigma = (1-\sigma)x + \sigma\varepsilon$, where $\sigma \in [0,1]$. The training objective is

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}_{\sigma,x,\varepsilon}\left[\|v_\theta(x_\sigma,\sigma,c) - (\varepsilon - x)\|_\Omega^2\right], \tag{1}$$

where Ω contains the valid prediction targets and $\|\cdot\|_\Omega^2$ denotes their mean squared error. The conditioning c comprises spatial-memory features, camera information, a clean prefix, and text when provided. The prefix is retained as unperturbed context. Spatial conditioning enters through a ControlNet-style side branch [12, 26].

Latent spatial memory and baseline operators. Let $z \in \mathbb{R}^{C\times H_l\times W_l}$ denote a latent frame, with depth $D \in \mathbb{R}^{H_l\times W_l}$, whose entries we write D_{uv}, and intrinsics K_l at the same resolution. A valid latent cell (u,v) of a frame captured by a camera E is lifted to a world point $\mathbf{p}_{uv} = E^{-1}\pi^{-1}(u,v,D_{uv};K_l)$ and associated with its feature $\mathbf{f}_{uv} = z_{:,v,u}$, where π^{-1} denotes camera back-projection. LSM-World [20] accumulates these observations through set union,

$$\mathcal{M} \leftarrow \mathcal{M} \cup \{(\mathbf{p}_{uv}, \mathbf{f}_{uv}) : (u,v) \in \Lambda\}, \tag{2}$$

where Λ is the set of valid cells admitted into the baseline memory. For a target camera E, let $\mathcal{P}(u,v)$ contain the cached points that project to cell (u,v) with positive camera depth. For nonempty $\mathcal{P}(u,v)$, the readout selects the frontmost feature,

$$\hat{\mathbf{f}}_{u,v} = \mathbf{f}_{j^\star}, \qquad j^\star = \operatorname*{arg\,min}_{j\in\mathcal{P}(u,v)} d_j, \tag{3}$$

where d_j is the depth of point j in the target camera. A validity mask identifies cells without projected observations. ENGRAM replaces set-union accumulation and frontmost selection with reliability-weighted latent fusion and occlusion-aware layered readout, while retaining direct projection at latent resolution.

Notation. For ENGRAM, $w_i \geq 0$ denotes the reliability weight of observation i, $W_a \geq 0$ the accumulated weight of memory slot a, and $\gamma \in (0,1]$ the temporal discount base for generated observations. Slots are grouped by the voxel index $\mathbf{k} = \lfloor \mathbf{p}/h_{\mathrm{vox}} \rfloor$, where h_{vox} is expressed in the same scene-normalized coordinates as $\mathbf{p}$. Each voxel holds a bounded number of slots, at most L, which may correspond to different depths or different provenance, while the occupied voxel set expands as new regions are observed. We use (u,v) exclusively for latent-cell coordinates.

4. The ENGRAM Model

4.1. Overview

ENGRAM initializes a persistent latent spatial memory and a clean prefix from the input image. It then generates successive video chunks through a read–generate–update cycle (Algorithm 1). At each chunk, the memory provides depth-ordered scene features for the target viewpoints, while explicit camera rays supply viewpoint information independently of memory coverage. The conditioned video generator synthesizes the next chunk, whose latent observations are integrated into memory using estimated geometry and reliability weights. The two central operators are reliability-weighted latent fusion and occlusion-aware layered readout, supported by geometric consistency constraints and a two-stage adaptation strategy. Figure 3 shows the full cycle, while Fig. 4 details the two operators.

4.2. Reliability-weighted latent fusion

Observation reliability. For each generated observation i, we estimate a unit surface normal $\mathbf{n}_i$ from finite differences of the back-projected depth and obtain its unit viewing direction $\mathbf{d}_i$. Both vectors are expressed in the same camera coordinate system. We define

$$g_i = \mathrm{clip}\big(|\mathbf{n}_i^\top \mathbf{d}_i|, \epsilon, 1\big), \qquad w_i = \sqrt{g_i}\,\gamma^{r_i}, \tag{4}$$

where $\epsilon > 0$ is a small positive floor and r_i is the observation's temporal index at latent resolution, measured from the start of the rollout. The angular factor down-weights grazing observations, while the temporal factor moderates the contribution of later generated observations. Both factors are precision proxies rather than fitted error variances: g_i is the grazing-angle precision of the observation, computed from the same normal estimate, and γ^{r_i} encodes the known autoregressive bias of generated frames.

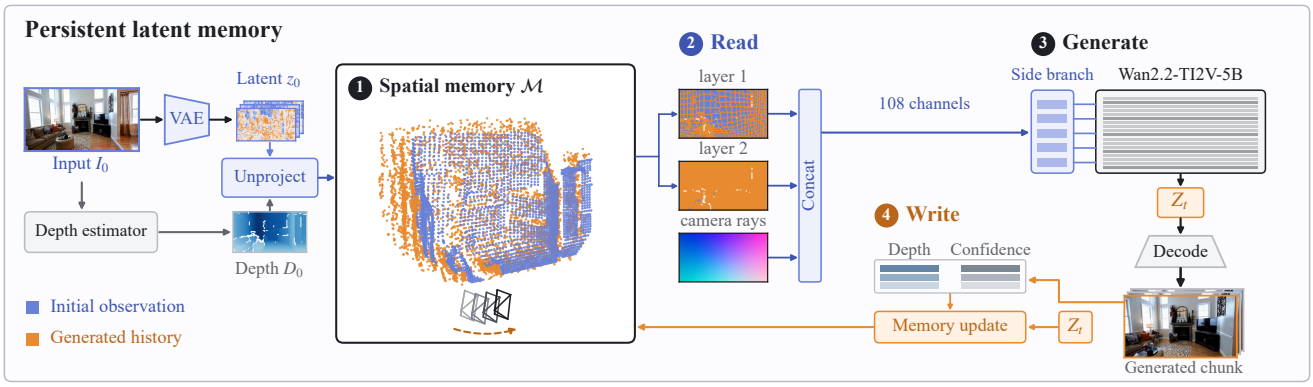

Figure 3. **Overview of ENGRAM.** The input image initializes spatial latent memory $\mathcal{M}$. Two readout layers and camera rays condition the video generator through a side branch. Generated latents are written back using geometry estimated from decoded frames; matching Z_t ports denote the same output tensor. Blue indicates initial observations and orange indicates generated history. The cloud is subsampled from a recorded memory, and the layer maps are recomputed at a target pose. Module and depth/confidence glyphs are schematic.

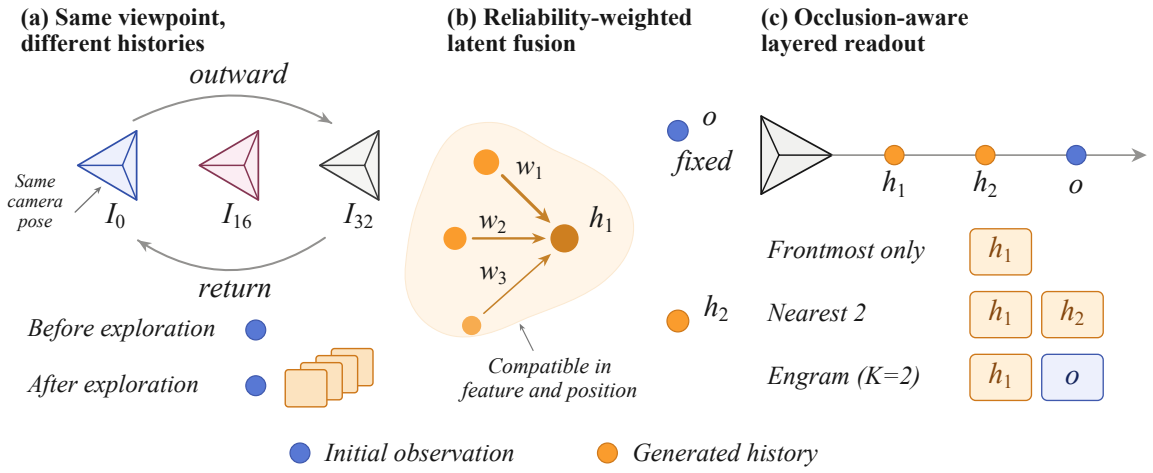

Figure 4. **Reliability-weighted latent fusion and occlusion-aware layered readout.** (a) Revisiting the same viewpoint after exploration provides additional observations. (b) Compatible generated observations are fused using reliability weights, while the initial observation is preserved. (c) Layered readout exposes depth-ordered candidates and retains initial-view evidence when available.

Feature normalization. We normalize latent features using fixed channel statistics $(\boldsymbol{\mu}, \Sigma)$, where Σ is a positive diagonal matrix of channel variances with numerical stabilization:

$$\tilde{\mathbf{f}}_i = \Sigma^{-1/2}(\mathbf{f}_i - \boldsymbol{\mu}). \tag{5}$$

This normalization places latent channels on comparable scales for appearance compatibility testing. The statistics remain fixed throughout memory construction and retrieval. During readout, we restore the original latent coordinates through $\mathbf{f} = \Sigma^{1/2}\tilde{\mathbf{f}} + \boldsymbol{\mu}$.

Before fusion, we aggregate accepted observations within each voxel of the current view using their reliability weights. In the update below, $\bar{\mathbf{f}}_i$ and $\bar{\mathbf{p}}_i$ denote the aggregated feature and position of that voxel, and $\bar{w}_i$ its summed weight. We match each candidate to the closest compatible slot in feature space within the same voxel. Compatibility requires $\|\bar{\mathbf{f}}_i - \tilde{\mathbf{f}}_a\|_2 < \tau_{\mathrm{f}}$ and $\|\bar{\mathbf{p}}_i - \mathbf{p}_a\|_2 / h_{\mathrm{vox}} < \tau_{\mathrm{p}}$. Generated candidates are matched only to slots formed from generated observations, preserving

Algorithm 1 ENGRAM rollout. Initialization runs once, before the first chunk; the remaining steps run once per generation chunk. The memory $\mathcal{M}$ and the clean prefix z_{prefix} persist across chunks.

Require: Input image I_0 and its camera E_0, target cameras $\mathcal{E}_t = \{E_{t,j}\}$ for the current chunk t, latent-resolution intrinsics K_l, and readout depth K.

Initialize (once): estimate the depth map D_0 of I_0, back-project its valid pixels through (E_0, K_l) to seed the spatial memory $\mathcal{M}$ with initial observations, and encode I_0 to obtain the clean prefix z_{prefix}.
Read: $R_t \leftarrow \text{LAYEREDREADOUT}(\mathcal{M}, \mathcal{E}_t, K_l, K)$. Sec. 4.3
Camera: construct Plücker rays $\boldsymbol{\ell}_t$ relative to $E_{t,0}$ and form the spatial condition $c_t^{\text{sp}} = [R_t; \boldsymbol{\ell}_t]$. Sec. 4.4
Generate: $Z_t \leftarrow \text{GENERATE}(c_t^{\text{sp}}, z_{\text{prefix}}, \text{text})$.
Decode and estimate geometry: $X_t \leftarrow \mathcal{D}(Z_t)$; estimate depth maps at the timestamps of the latent frames. Sec. 4.5
for all new latent frames j in the chunk **do**
    Back-project valid observations from $(Z_{t,j}, D_{t,j}, E_{t,j}, K_l)$.
    Compute observation weights using Eq. (4).
    Apply the geometric admission rule. Sec. 4.5
    Aggregate accepted observations within each voxel and match candidates to compatible slots. Sec. 4.2
    Fuse matched candidates using Eq. (6) and allocate slots for unmatched candidates when capacity permits.
end for
Handoff: refine the encoding of the last generated frame to obtain the next clean prefix. Eq. (8)

the initial observation. For a matched pair, we apply

$$\begin{aligned} \tilde{\mathbf{f}}_a &\leftarrow \frac{W_a \tilde{\mathbf{f}}_a + \bar{w}_i \bar{\mathbf{f}}_i}{W_a + \bar{w}_i}, \\ W_a &\leftarrow \min(W_a + \bar{w}_i, W_{\max}). \end{aligned} \tag{6}$$

The weight cap limits the accumulated influence of earlier observations. An unmatched candidate initializes a new slot with weight $\min(\bar{w}_i, W_{\max})$, subject to a per-voxel limit of L slots. Unmatched candidates are discarded when the voxel has reached capacity.

4.3. Occlusion-aware layered readout

For each target view, we project memory slots with positive camera depth onto the latent grid and order them by depth within each cell. We first select up to K frontmost slots. If none originates from the initial observation, the nearest projected slot from that observation replaces the last selected slot, when available.

Each selected layer provides a latent feature (C channels), a bounded reliability value, a validity mask, and a normalized inverse depth. Features are restored to their original latent coordinates before conditioning, and missing layers are filled with zeros and marked invalid. The resulting memory conditioning has $K(C+3)$ channels, giving 102 channels for $C = 48$ and $K = 2$. This layered representation provides the generator with stored information at multiple depths as visibility changes.

4.4. Explicit Plücker camera conditioning

Explicit camera rays provide a dense viewpoint signal independently of memory coverage. For each target camera $E_{t,j}$, we use the relative extrinsic $E_{t,j}^{\text{rel}} = E_{t,j} E_{t,0}^{-1}$, where $E_{t,0}$ is the first camera of the current chunk. For each latent cell (u, v), we compute the unit ray direction $\mathbf{r}_{uv}$ and camera center $\mathbf{o}$ in this reference camera coordinate system. The Plücker representation is

$$\boldsymbol{\ell}_{uv} = (\mathbf{r}_{uv}, \mathbf{m}_{uv}), \qquad \mathbf{m}_{uv} = \mathbf{o} \times \mathbf{r}_{uv}, \tag{7}$$

following [9, 18]. The six ray channels are concatenated with the memory readout and injected through the conditioning branch. For $C = 48$ and $K = 2$, the combined spatial conditioning contains 108 channels. Cells without projected memory therefore retain explicit information about the target viewpoint.

4.5. Continuous generation and memory update

After generation, we decode the chunk and estimate depth at the timestamps of its latent frames. The estimated geometry is aligned with the latent grid and memory coordinate system, then paired with the original generated latent features. Only

Table 1. **Evaluation on 200 WorldScore cases.** Static averages the seven control and quality metrics; Dynamic also includes the three motion metrics; Avg. is their mean. All scores are higher-is-better. A dash indicates a metric or aggregate not applicable to the method. Best and second-best scores are shown in bold and underlined, respectively.

Method	Overall			Control and content			Geometry and perception				Motion		
	Avg.	Static	Dynamic	Cam.	Obj.	Content	3D	Photo.	Style	Subj.	Acc.	Mag.	Smooth.
Models with 3D cache													
WonderWorld [24]	–	63.82	–	**67.61**	40.75	**68.78**	79.43	84.53	62.10	43.56	–	–	–
Spatia [27]	<u>64.47</u>	66.65	62.29	55.67	**65.75**	37.75	85.83	89.49	82.09	49.96	<u>58.27</u>	26.48	71.64
LSM-World [20]	56.59	59.70	53.47	57.27	38.25	42.64	83.59	64.41	80.58	51.17	49.08	26.74	40.94
General video models													
Wan2.2-TI2V-5B [19]	64.27	<u>67.33</u>	61.21	33.53	54.59	46.84	<u>90.01</u>	89.33	**95.12**	**61.88**	55.01	<u>28.93</u>	56.86
LTX-Video [8]	63.48	64.58	<u>62.39</u>	30.82	<u>61.25</u>	49.34	**91.23**	**94.94**	72.81	51.67	57.85	27.11	<u>86.83</u>
CogVideoX [22]	60.34	62.66	58.02	44.80	42.00	<u>55.09</u>	81.00	81.82	74.46	59.44	**59.81**	**30.70**	51.13
ENGRAM (ours)	**67.78**	**69.97**	**65.60**	<u>66.92</u>	57.00	41.89	85.84	<u>91.90</u>	<u>86.67</u>	<u>59.57</u>	52.47	24.76	**88.95**

observations with finite, positive depth are eligible for writing. The geometric admission rule filters generated candidates against the initial-view depth before memory insertion. Accepted observations are integrated through the fusion procedure in Sec. 4.2.

At each chunk boundary, we encode the last generated frame I_{last} and refine its latent through the fixed VAE decoder:

$$z^{\star}_{\text{prefix}} \approx \arg\min_{z} \|\mathcal{D}(z) - I_{\text{last}}\|_1 . \tag{8}$$

The refined latent serves as the clean prefix for the next generation chunk.

4.6. Training and implementation

We adapt the pretrained generator in two stages. Stage 1 trains the conditioning branch while keeping the generation backbone and VAE fixed. Stage 2 freezes the conditioning branch and optimizes all parameters of the generation Transformer. The VAE remains fixed throughout. Both stages use the flow-matching objective in Eq. (1), with the clean prefix retained as unperturbed context and excluded from the prediction loss.

Training uses 256 scenes, each providing five forward windows and four return windows, yielding 2,304 training windows. Training memories are constructed from observed video history using precomputed geometry. Each window includes a text caption, which is dropped with probability 0.1 during training.

The two stages run for 4,000 and 8,000 optimization steps, respectively, with an effective batch size of four across four NVIDIA H200 GPUs. We use AdamW, bfloat16 mixed precision, and gradient checkpointing. At inference, depth is estimated from decoded frames using MapAnything [13], and video chunks are sampled with UniPC [28] using 40 steps.

4.7. Why fusion improves on selection

An idealized observation model illustrates the benefit of combining repeated evidence. Consider observations of a fixed scene location with independent, zero-mean latent errors and covariance $\sigma_i^2 I$. Fixed inverse-variance weighting is unbiased and yields covariance $\left(\sum_i \sigma_i^{-2}\right)^{-1} I$, which is no larger than that of any individual observation, and is therefore also no worse than selecting any candidate fixed in advance.

Frontmost selection uses the depth-argmin over the candidates. For repeated observations of the same surface with i.i.d., zero-mean, non-degenerate depth errors, the minimum observed depth is biased toward the camera (Appendix A). Fusion aggregates compatible candidates through a weighted mean.

Our practical operator uses geometric and temporal reliability weights with compatibility matching and bounded accumulation. Appendix A analyzes weighted averaging, depth-based selection, and the effects of correlated errors and observation-dependent weights.

Table 2. **Evaluation on 100 RealEstate10K scenes.** NVS evaluates frames 1–80; closed-loop revisit compares frame 152 with the initial image. All methods use the same evaluation protocol. Best and second-best scores are bold and underlined, respectively.

	NVS			Closed-loop revisit		
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
ViewCrafter [25]	8.494	0.3405	0.7852	9.540	0.3467	0.7468
FlexWorld [2]	11.167	0.5805	0.6684	12.191	0.6089	0.5612
Spatia [27]	<u>16.745</u>	<u>0.6832</u>	<u>0.3525</u>	<u>15.978</u>	<u>0.6827</u>	<u>0.3373</u>
LSM-World [20]	15.624	0.6311	0.4148	12.235	0.3716	0.7487
ENGRAM (ours)	**17.568**	**0.6975**	**0.3212**	**20.455**	**0.7692**	**0.2200**

5. Experiments

We evaluate complete-system performance on WorldScore and RealEstate10K, and isolate the effects of memory writing, layered readout, and two-stage adaptation through ablations.

5.1. Main comparisons and shared protocol

We compare complete systems using released checkpoints, except for LSM-World [20], which we reproduce. Wan2.2-TI2V-5B [19] serves as a backbone reference with image and text conditioning rather than pose tensors; its comparison with ENGRAM is not an isolated memory ablation.

WorldScore. We evaluate on a 200-case WorldScore subset using its ten official metrics [5]. WonderWorld's motion metrics and dependent aggregates are not applicable. General video models receive image and camera-text prompts, whereas pose-conditioned methods receive numeric camera poses. Table 1 shows that ENGRAM achieves the highest aggregate score of 67.78. Relative to Spatia, motion smoothness, camera control, and subjective quality improve by 17.31, 11.25, and 9.61 points, respectively. Its 3D consistency is nearly unchanged, while motion accuracy and magnitude remain lower. Detailed protocols are provided in Supplementary Sec. C.1.

RealEstate10K. We evaluate all methods on the same 100 held-out scenes [27, 30]. NVS scores frames 1–80 against ground truth; closed-loop evaluation compares frame 152 with the initial image. Methods share target camera trajectories, field of view, frame times, and scoring resolution. Each task starts from the real input image and uses only generated observations as subsequent visual history. Table 2 reports mean per-scene scores.

Table 2 shows that ENGRAM achieves the best PSNR, SSIM, and LPIPS on both tasks among the compared methods. Relative to Spatia, NVS PSNR improves by 0.82 dB to 17.568 dB, while closed-loop PSNR improves by 4.48 dB to 20.455 dB. Closed-loop SSIM increases from 0.6827 to 0.7692, and LPIPS falls from 0.3373 to 0.2200, showing gains in both structural and perceptual similarity. The larger improvements on revisits highlight the model's ability to recover previously observed views after exploration. This pattern aligns with the ablations in Sec. 5.3, where disabling history writing or masking the second readout layer produces larger PSNR drops on closed-loop reconstruction than on NVS.

5.2. Computational cost

Figure 5 summarizes the recorded quality–cost trade-off and adaptation cost. Our two-stage adaptation uses 80.8 GPU hours of logged updates: 17.0 for Stage 1 and 63.8 for Stage 2. Direct latent-memory readout avoids intermediate RGB rendering and VAE encoding when constructing spatial conditions.

5.3. Core operator ablations

Table 3 examines memory writing, memory conditioning, the second readout layer, and Stage 2 adaptation.

No-write keeps the branch, the camera conditioning and the initial-image cache, and disables only the write-back of generated observations, so memory never grows beyond the first view. This isolates what generated history contributes on top of the initial observation. NVS PSNR drops by about 0.5 dB. Closed-loop PSNR drops by 3.2 decibels, and LPIPS rises

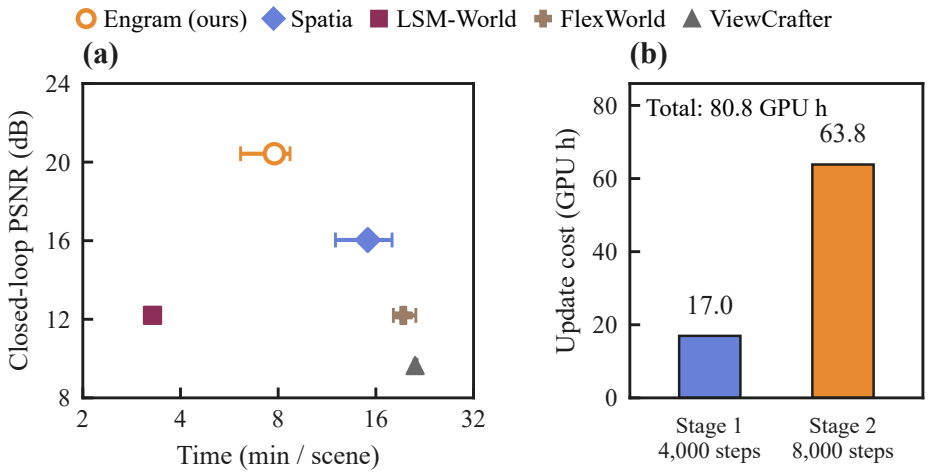

Figure 5. **Reconstruction quality and computational cost.** (a) Closed-loop PSNR versus median inference time; bars show runtime interquartile ranges. Hardware and sampling settings vary across methods. (b) Training-update GPU hours for the two adaptation stages.

Table 3. **Memory and adaptation ablations on 100 RealEstate10K scenes.** All variants use the same scenes and evaluation protocol. Best and second-best scores are bold and underlined, respectively.

	NVS			Closed-loop revisit		
Variant	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
ENGRAM (full)	**17.568**	**0.6975**	**0.3212**	**20.455**	**0.7692**	**0.2200**
No-write	17.052	0.6857	0.3302	17.284	0.7167	0.3473
No-read	12.213	0.5795	0.5999	10.633	0.5721	0.7608
Second-layer masked	<u>17.321</u>	<u>0.6924</u>	<u>0.3253</u>	<u>19.200</u>	<u>0.7423</u>	<u>0.2678</u>
Stage 1 only	16.777	0.6769	0.3423	18.570	0.7253	0.2681

from 0.220 to 0.347. The larger drop in closed-loop reconstruction indicates a stronger dependence on generated-history writing.

No-read removes the memory condition from the architecture entirely and is trained separately under the same recipe, with the 102 memory channels set to zero while the six camera channels are retained. NVS and closed-loop PSNR decrease to 12.213 and 10.633 dB, respectively. This highlights the importance of memory conditioning for reconstruction quality on both tasks.

Second-layer masked zeroes the second-layer channels (51:102) of the same checkpoint and leaves the remaining operators untouched. Relative to the Full results reported in Table 3, NVS and closed-loop PSNR are lower by 0.25 and 1.25 dB, respectively. The original candidate-selection and initial-view reservation rules remain active.

Stage 1 only evaluates the Stage 1 checkpoint with the pretrained backbone under the same inference configuration. Compared with the full model, NVS and closed-loop PSNR decrease by 0.79 and 1.89 dB, respectively, showing the benefit of Stage 2 adaptation.

5.4. Revisiting generated content

Closed-loop scoring returns to the real initial image, which can reward initial-view anchoring rather than memory. We therefore also run an $A \to B \to C \to B$ route on static development scenes, in which a first visit B exposes content absent from the seed A and is itself generated rather than observed. The ENGRAM model is compared with a control that keeps the initial-image cache and disables all later writes, holding camera conditioning and the generated-prefix rule fixed.

On the 20 development scenes in Table S1, the control reaches 17.63 dB when returning to A and 12.47 dB when revisiting B, against 20.88 and 16.87 dB for ENGRAM with SSIM of 0.7496 and 0.6636. The margin is therefore larger at the generated

view ($+4.4$ dB) than at the observed one ($+3.2$ dB), which is the ordering an initial-view-anchoring explanation does not predict: a model that only replays the first image has less to lose at the pose it was conditioned on. Because B was itself generated, the revisit reference is the first-generated frame at that pose and not a ground-truth image, so these numbers measure persistence rather than accuracy against an unseen view, and we treat them as development-scale evidence.

5.5. Discussion and limitations

Both NVS and closed-loop evaluation use multi-chunk generation. Removing generated-history writing or masking the second readout layer causes a larger PSNR drop in closed-loop reconstruction than in NVS. These results highlight the role of persistent memory when returning to earlier viewpoints. History writing preserves earlier observations across chunks, while layered readout supplies candidates from multiple surfaces as visibility changes. We report the two tasks separately to distinguish novel-view reconstruction from revisit consistency.

ENGRAM relies on estimated geometry. Depth is predicted from decoded frames and determines where latent observations are written in the world coordinate system. Errors in these estimates can therefore propagate through later memory queries, displacing stored observations and altering the depth ordering of readout candidates. Compatibility checks and bounded accumulation regulate memory updates, but systematic depth bias remains a limitation. Memory grows as new regions are explored: the per-voxel slot limit bounds local density, not total storage. Moving surfaces are not explicitly modelled.

6. Conclusion

We presented ENGRAM, a latent spatial memory framework combining reliability-weighted latent fusion and occlusion-aware layered readout. The framework consolidates compatible observations and exposes multiple depth-ordered candidates within a read–generate–update cycle. Generated-history write-back retains scene evidence beyond the current context window, while explicit camera rays provide viewpoint information independently of memory coverage. Adapted with 256 training scenes, ENGRAM improves novel-view synthesis and closed-loop reconstruction on RealEstate10K and achieves the highest aggregate score among the compared systems on the 200-case WorldScore subset. The gains are particularly pronounced for closed-loop revisits. Ablation comparisons highlight the role of generated-history writing and layered memory conditioning in closed-loop reconstruction, as well as the benefit of second-stage adaptation. Future work will extend the framework to dynamic scenes and memory management under fixed storage budgets.

LLM Usage Statement

OpenAI Codex assisted with literature collection, auxiliary code development, and manuscript editing. The authors designed the study, verified the experimental results and all revisions, and take responsibility for all scientific claims and interpretations.

References

[1] George Casella and Roger L. Berger. *Statistical Inference*. Duxbury, Thomson Learning, 2 edition, 2002. 13

[2] Luxi Chen, Zihan Zhou, Min Zhao, Yikai Wang, Ge Zhang, Wenhao Huang, Hao Sun, Ji-Rong Wen, and Chongxuan Li. FlexWorld: Progressively expanding 3D scenes for flexiable-view synthesis. *arXiv preprint arXiv:2503.13265*, 2025. 2, 3, 8

[3] Jaeyoung Chung, Suyoung Lee, Hyeongjin Nam, Jaerin Lee, and Kyoung Mu Lee. LucidDreamer: Domain-free generation of 3D Gaussian splatting scenes. *arXiv preprint arXiv:2311.13384*, 2023. 3

[4] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pages 303–312. ACM, 1996. 3

[5] Haoyi Duan, Hong-Xing Yu, Sirui Chen, Li Fei-Fei, and Jiajun Wu. WorldScore: A unified evaluation benchmark for world generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 27713–27724, 2025. 3, 8, 16

[6] Paul Engstler, Andrea Vedaldi, Iro Laina, and Christian Rupprecht. Invisible Stitch: Generating smooth 3D scenes with depth inpainting. In *2025 International Conference on 3D Vision (3DV)*, pages 457–468. IEEE, 2025. 3

[7] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017. 3

[8] Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, et al. LTX-Video: Realtime video latent diffusion. *arXiv preprint arXiv:2501.00103*, 2024. 2, 3, 7

[9] 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. 3, 6

[10] Jiahui Huang, Shi-Sheng Huang, Haoxuan Song, and Shi-Min Hu. DI-Fusion: Online implicit 3D reconstruction with deep priors. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8932–8941, 2021. 3

[11] Tianyu Huang, Wangguandong Zheng, Tengfei Wang, Yuhao Liu, Zhenwei Wang, Junta Wu, Jie Jiang, Hui Li, Rynson Lau, Wangmeng Zuo, et al. Voyager: Long-range and world-consistent video diffusion for explorable 3D scene generation. *ACM Transactions on Graphics (TOG)*, 44(6):1–15, 2025. 2, 3

[12] Zeyinzi Jiang, Zhen Han, Chaojie Mao, Jingfeng Zhang, Yulin Pan, and Yu Liu. VACE: All-in-one video creation and editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17191–17202, 2025. 4

[13] Nikhil Keetha, Norman Müller, Johannes Schönberger, Lorenzo Porzi, Yuchen Zhang, Tobias Fischer, Arno Knapitsch, Duncan Zauss, Ethan Weber, Nelson Antunes, Jonathon Luiten, Manuel Lopez-Antequera, Samuel Rota Bulò, Christian Richardt, Deva Ramanan, Sebastian Scherer, and Peter Kontschieder. MapAnything: Universal feed-forward metric 3D reconstruction. In *International Conference on 3D Vision (3DV)*. IEEE, 2026. 7, 15

[14] Runjia Li, Philip Torr, Andrea Vedaldi, and Tomas Jakab. VMem: Consistent interactive video scene generation with surfel-indexed view memory. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 25690–25699, 2025. 3

[15] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The eleventh international conference on learning representations*, 2023. 3

[16] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality*, pages 127–136. Ieee, 2011. 3

[17] 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. 3

[18] Julius Plucker. XVII. on a new geometry of space. *Philosophical Transactions of the Royal Society of London*, (155):725–791, 1865. 3, 6

[19] 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. 2, 3, 7, 8

[20] Weijie Wang, Haoyu Zhao, Yifan Yang, Feng Chen, Zeyu Zhang, Yefei He, Zicheng Duan, Donny Y Chen, Yuqing Yang, and Bohan Zhuang. Latent spatial memory for video world models. *arXiv preprint arXiv:2606.09828*, 2026. 2, 3, 4, 7, 8

[21] Silvan Weder, Johannes L Schonberger, Marc Pollefeys, and Martin R Oswald. NeuralFusion: Online depth fusion in latent space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3162–3172, 2021. 3

[22] 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*, pages 83048–83077, 2025. 2, 3, 7

[23] 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

[24] Hong-Xing Yu, Haoyi Duan, Charles Herrmann, William T Freeman, and Jiajun Wu. WonderWorld: Interactive 3D scene generation from a single image. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5916–5926, 2025. 2, 7

[25] Wangbo Yu, Jinbo Xing, Li Yuan, Wenbo Hu, Xiaoyu Li, Zhipeng Huang, Xiangjun Gao, Tien-Tsin Wong, Ying Shan, and Yonghong Tian. ViewCrafter: Taming video diffusion models for high-fidelity novel view synthesis. *arXiv preprint arXiv:2409.02048*, 2024. 2, 3, 8

[26] 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. 4

[27] 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*, pages 4245–4257, 2026. 3, 7, 8, 15, 16

[28] Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. UniPC: A unified predictor-corrector framework for fast sampling of diffusion models. *Advances in Neural Information Processing Systems*, 36:49842–49869, 2023. 7, 15

[29] Jensen Zhou, Hang Gao, Vikram Voleti, Aaryaman Vasishta, Chun-Han Yao, Mark Boss, Philip Torr, Christian Rupprecht, and Varun Jampani. Stable Virtual Camera: Generative view synthesis with diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12405–12414, 2025. 3

[30] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Transactions on Graphics*, 37(4):1–12, 2018. 3, 8, 16

ENGRAM: Reliability-Weighted Latent Fusion and Occlusion-Aware Layered Readout for Video World Models

Supplementary Material

A. Analysis of the Weighted Fusion Rule

We analyze repeated observations of a fixed scene location in the normalized latent coordinates of Sec. 4.2. Let $\mathbf{f}_i = \mathbf{f}^\star + \boldsymbol{\varepsilon}_i$, $i = 1, \dots, n$, where $\mathbf{f}^\star \in \mathbb{R}^C$ is the unknown latent feature. For the independent-noise analysis, we assume $\mathbb{E}[\boldsymbol{\varepsilon}_i] = 0$, $\mathrm{Cov}(\boldsymbol{\varepsilon}_i) = \sigma_i^2 I$, $0 < \sigma_i^2 < \infty$, and mutually independent errors. These are assumptions of the observation model. The correlated case is considered separately below.

A.1. Inverse-variance weighting

Consider one scalar channel and a linear estimator $\hat{f} = \sum_i c_i f_i$ with fixed coefficients. Unbiasedness for every $f^\star$ requires $\sum_i c_i = 1$. Under independence,

$$\mathrm{Var}(\hat{f}) = \sum_i c_i^2 \sigma_i^2. \tag{S1}$$

Minimizing this variance subject to $\sum_i c_i = 1$ with a Lagrange multiplier λ gives $2c_i\sigma_i^2 = \lambda$. Hence the minimizing coefficients are

$$c_i^\star = \frac{\sigma_i^{-2}}{\sum_j \sigma_j^{-2}}. \tag{S2}$$

Let $\omega_i = \sigma_i^{-2}$ denote the inverse variance. Applying the same coefficients to all latent channels gives

$$\begin{aligned} \hat{\mathbf{f}} &= \frac{\sum_i \omega_i \mathbf{f}_i}{\sum_i \omega_i}, \\ \mathrm{Cov}(\hat{\mathbf{f}}) &= \Big(\sum_i \sigma_i^{-2}\Big)^{-1} I \preceq \min_i \sigma_i^2 \, I. \end{aligned} \tag{S3}$$

The estimator is unbiased because its coefficients sum to one. The covariance inequality is strict for $n \geq 2$ under the positive, finite variance assumptions above. This establishes the best linear unbiased estimator under the stated model [1].

A.2. Uncapped online accumulation

Let $S_t = \sum_{i=1}^t \omega_i$ and $\hat{\mathbf{f}}_t = S_t^{-1} \sum_{i=1}^t \omega_i \mathbf{f}_i$. For $t \geq 2$,

$$\hat{\mathbf{f}}_t = \frac{S_{t-1}\hat{\mathbf{f}}_{t-1} + \omega_t \mathbf{f}_t}{S_{t-1} + \omega_t}. \tag{S4}$$

Thus, uncapped accumulation equals the batch estimator. Permuting the same observation–weight pairs leaves the estimate unchanged because its numerator and denominator are sums. Writing $\alpha_t = \omega_t / S_t$ gives $\hat{\mathbf{f}}_t = (1 - \alpha_t)\hat{\mathbf{f}}_{t-1} + \alpha_t \mathbf{f}_t$.

Under inverse-variance weighting, $\mathrm{Cov}(\hat{\mathbf{f}}_t) = S_t^{-1} I$. This covariance tends to zero when $\sum_i \omega_i = \infty$. A sufficient condition is $\omega_i \geq \omega_{\min} > 0$ for all observations.

The practical update in Eq. (6) caps the stored weight at $W_{\max}$. Once the cap alters the stored weight, the batch-estimator identity and order independence generally no longer hold. The limiting behavior then depends on the incoming observations and their weights.

A.3. Single-observation selection

For an index j fixed in advance, selecting $\mathbf{f}_j$ gives an unbiased estimator with covariance $\sigma_j^2 I$. Equation (S3) therefore gives

$$\mathrm{Cov}(\hat{\mathbf{f}}) \preceq \mathrm{Cov}(\mathbf{f}_j), \tag{S5}$$

with strict inequality for $n \geq 2$ under the stated assumptions.

Depth-based selection instead uses an index determined by the observed depths. For repeated measurements of the same surface, let $d_i = d^\star + \eta_i$ and $J = \arg\min_i d_i$. Assume $n \geq 2$ and i.i.d., integrable, non-degenerate depth errors with $\mathbb{E}[\eta_i] = 0$. Then

$$\begin{aligned} \mathbb{E}[d_J - d^\star] &= \mathbb{E}\big[\min_i \eta_i\big] \\ &\leq \mathbb{E}\big[\min(\eta_1, \eta_2)\big] \\ &= -\frac{1}{2}\mathbb{E}\big[|\eta_1 - \eta_2|\big] < 0. \end{aligned} \tag{S6}$$

Thus, minimum-depth selection biases the selected depth toward the camera under this model.

The bias of the associated latent feature is determined by

$$\mathbb{E}[\mathbf{f}_J - \mathbf{f}^\star] = \sum_j \Pr(J = j)\, \mathbb{E}[\boldsymbol{\varepsilon}_j \mid J = j]. \tag{S7}$$

When depth ordering is independent of all latent errors, this expectation is zero. Latent-feature bias therefore depends on the relationship between the selection event and the feature errors.

A.4. Correlation and imperfect weights

For one scalar channel, suppose $\mathrm{Var}(\varepsilon_i) = \sigma^2$ and $\mathrm{Cov}(\varepsilon_i, \varepsilon_j) = \rho\sigma^2$ for $i \neq j$, with $\rho \in [0, 1)$. The equally weighted mean has variance

$$\begin{aligned} \mathrm{Var}(\hat{f}) &= \frac{1}{n^2}\Big(n\sigma^2 + n(n-1)\rho\sigma^2\Big) \\ &= \frac{\sigma^2}{n}\big(1 + (n-1)\rho\big) = \frac{\sigma^2}{n_{\mathrm{eff}}}, \end{aligned} \tag{S8}$$

where

$$n_{\mathrm{eff}} = \frac{n}{1 + (n-1)\rho}. \tag{S9}$$

For $\rho > 0$ and $n > 1$, $n_{\mathrm{eff}} < n$. As $n \to \infty$, the effective sample size approaches $1/\rho$ and the variance approaches $\rho\sigma^2$.

More generally, let V be the covariance matrix of the scalar observation errors. For fixed coefficients $\mathbf{c}$ satisfying $\mathbf{1}^\top \mathbf{c} = 1$,

$$\mathbb{E}[\hat{f}] = f^\star, \qquad \mathrm{Var}(\hat{f}) = \mathbf{c}^\top V \mathbf{c}. \tag{S10}$$

Under independence, this reduces to $\sum_i c_i^2 \sigma_i^2$. Fixed weights that differ from inverse variances remain unbiased under the zero-mean model, but need not minimize the variance.

For observation-dependent coefficients, unbiasedness requires

$$\mathbb{E}\Big[\sum_i c_i \varepsilon_i\Big] = 0. \tag{S11}$$

A.5. Affine normalization and weighted averaging

Let $g(\mathbf{f}) = \Sigma^{-1/2}(\mathbf{f} - \boldsymbol{\mu})$, with fixed $\boldsymbol{\mu}$ and positive diagonal Σ. For the same observations and coefficients satisfying $\sum_i c_i = 1$,

$$\begin{aligned} \sum_i c_i g(\mathbf{f}_i) &= \Sigma^{-1/2}\Big(\sum_i c_i \mathbf{f}_i - \boldsymbol{\mu}\Big) \\ &= g\Big(\sum_i c_i \mathbf{f}_i\Big). \end{aligned} \tag{S12}$$

Weighted averaging therefore commutes with this affine map. Applying $g^{-1}(\tilde{\mathbf{f}}) = \Sigma^{1/2}\tilde{\mathbf{f}} + \boldsymbol{\mu}$ restores the corresponding average in the original latent coordinates.

This identity holds for the same observations and weights. In the memory operator, normalization also determines the feature distances used for compatibility testing. Its role is to place channels on comparable scales; the noise assumptions used above are specified separately.

B. Implementation and Training Details

Memory configuration. The latent has $C = 48$ channels, and each voxel accommodates at most $L = 8$ slots. The compatibility thresholds are $\tau_{\mathrm{f}} = 3$ and $\tau_{\mathrm{p}} = 1$, with position distances divided by the voxel side length. The accumulated weight is capped at $W_{\max} = 20$. Scene coordinates are normalized by the median metric depth s_0 of the initial image. A voxel side length of 0.005 metres therefore becomes $h_{\mathrm{vox}} = (0.005\,\mathrm{m})/s_0$ in normalized coordinates. A generated slot adopts the incoming candidate position only when the candidate weight exceeds the stored weight before the update.

Normalization and reliability. Channel normalization uses fixed statistics and a variance stabilizer of 10^{-5}. The angular factor in Eq. (4) uses $\epsilon = 10^{-3}$, and the temporal discount base is $\gamma = 0.98$. The angular exponent is $1/2$, and the temporal factor is γ^{r_i}. Here, $r_i = \lfloor s_i/4 \rfloor$, where s_i is the global RGB-frame index measured from the rollout start. Initial-observation weights are multiplied by four. Generated observations are fused into separate slots, while the initial-observation slots remain fixed.

Layered conditioning. The readout returns up to $K = 2$ layers. Each layer contains C feature channels, one reliability channel, one validity mask, and one inverse-depth channel. The memory representation therefore has 102 channels; appending the six Plücker ray channels gives 108. The reliability channel is

$$q_a = \frac{\min(W_a, 5)}{1 + \min(W_a, 5)}. \tag{S13}$$

The readout cap of five is applied separately from the fusion weight cap $W_{\max} = 20$. The inverse-depth channel is $1/(1+d)$, where d is the positive camera depth in scene-normalized coordinates. Missing layers are filled with zeros and marked invalid. The conditioning branch uses blocks copied from backbone layers $\{0, 4, 8, 12, 16, 20, 24, 28\}$.

Generation and memory updates. Each chunk contains nine latent frames corresponding to 33 RGB frames. Consecutive chunks overlap by one RGB frame, giving a stride of 32 frames. Sampling uses UniPC [28] with 40 steps. Depth is estimated from decoded frames using MapAnything [13]. Writes require finite positive depth. A candidate projecting onto valid initial-view depth is rejected when its depth in that camera is below 0.9 times the reference depth. At each chunk boundary, the latent refinement in Eq. (8) uses Adam for 50 steps with learning rate 0.02 and mean absolute RGB reconstruction error.

Optimization details. Both training stages use a learning rate of 5×10^{-6}. The learning rate is constant in Stage 1; Stage 2 uses a 100-step linear warm-up followed by a constant learning rate.

Recorded training cost. The logged training-update costs are 17.0 GPU hours for Stage 1 and 63.8 GPU hours for Stage 2, totaling 80.8 GPU hours. These costs exclude data preparation, initialization, and checkpoint saving.

B.1. Training-sample accounting

We count training-sample presentations as $P = \sum_s T_s B_s$, where T_s is the number of optimization steps and B_s is the effective batch size in stage s. Repeated draws of the same sample each count as a presentation. For ENGRAM, both stages use an effective batch size of four:

$$P_{\mathrm{Engram}} = 4{,}000 \times 4 + 8{,}000 \times 4 = 48{,}000.$$

Spatia reports 8,000 ControlNet updates followed by 5,000 LoRA updates, with a batch size of 64 in both stages [27]. Interpreting the reported batch size as the effective batch size gives

$$P_{\mathrm{Spatia}} = 8{,}000 \times 64 + 5{,}000 \times 64 = 832{,}000.$$

Thus, $P_{\mathrm{Engram}}/P_{\mathrm{Spatia}} \approx 5.8\%$, and the reciprocal is approximately 17.3. Spatia's 50,000 source videos describe the dataset size and do not enter this ratio. These are schedule-derived presentation counts; sample duration, resolution, and trainable parameters differ between methods, so the ratio does not measure unique-data requirements, processed frames or tokens, GPU hours, or computational speedup.

C. Evaluation and Ablation Protocols

C.1. Benchmark evaluation protocols

RealEstate10K evaluates reconstruction fidelity, while WorldScore provides complementary measures of camera control and motion quality.

WorldScore. Table 1 reports the ten official metrics and the WorldScore-Static and WorldScore-Dynamic aggregates [5], with Avg. defined as their mean. Static averages seven control and quality metrics; Dynamic averages those seven plus three motion metrics. These aggregates are distinct from the static and dynamic sample counts. WonderWorld generates static scenes only, so its motion metrics and the aggregates that depend on them are not applicable and are left blank rather than filled with the benchmark's placeholder zeros. General video models receive image and text conditioning with benchmark camera prompts, whereas camera-controlled systems receive numeric poses; this input difference is constant across their rows.

RealEstate10K. Table 2 evaluates novel-view synthesis (NVS) and closed-loop return on a common held-out scene list of 100 scenes [27, 30]. NVS generates 81 frames and scores frames 1–80 against their ground-truth counterparts. Closed-loop generation follows a fixed 153-pose trajectory and scores frame 152 against the initial image. Each task starts independently from the same real image and prescribed camera annotations. Subsequent visual history contains only generated observations; held-out RGB frames are used solely for scoring, including when adapting camera-input interfaces. All methods generate the full evaluation trajectories. Target field of view, camera conventions, frame times, and scoring resolution are matched across methods. NVS scores are averaged within each scene and then across scenes. We use a fixed seed policy and no per-metric checkpoint selection.

C.2. Ablation settings

Table 3 includes inference interventions, a separately trained No-read model, and a Stage 1 checkpoint comparison.

Inference interventions. *No-write* and *second-layer masked* reuse the same ENGRAM checkpoint as the full row and change only the runtime path. No-write retains the initial-image cache, the camera conditioning and the generated-prefix rule, and suppresses the write-back of generated observations, so the memory never grows beyond the first view. Second-layer masked zeroes the second-layer conditioning channels (51:102) while leaving selection depth $K = 2$, the first layer, the camera channels and the reservation of real initial-view candidates intact. It is therefore not a retrained single-layer model, and the residual gap it produces combines the second layer with the reservation rule rather than isolating either. Because both rows reuse the full checkpoint, their differences from the full model are attributable to the removed component without retraining variance, but they cannot be read as the performance of a system designed without that component.

Retrained variant. *No-read* removes the memory condition from the architecture and is trained separately under the same data, schedule and objective, with the 102 memory channels set to zero and the six camera channels retained.

Stage 2 adaptation. *Stage 1 only* loads the Stage 1 side-branch checkpoint on the unmodified pretrained backbone, retaining the full model's memory operators, inference path, hyperparameters and score normalization. This comparison evaluates the benefit of Stage 2 adaptation.

Sharing. All five rows use the same 100 test scenes, camera trajectories, target resolution, sampler and scoring code. Scenes are scored individually and then averaged, so no row benefits from a different subset.

D. Generated-Content Revisit

Closed-loop scoring in Table 2 returns to the *real* initial image, which can reward initial-view anchoring rather than memory. We therefore also evaluate a route in which the return target is itself generated (Fig. S1): starting from a real seed A, the model generates a first visit B that exposes content absent from A, leaves toward C, and then returns to the pose of B.

The comparison holds camera conditioning and the generated-prefix rule fixed and differs only in whether generated observations are written back. Because B was never observed, the revisit reference is the first-generated frame at that pose and not a ground-truth image, so the revisit column measures persistence of generated content rather than accuracy against an unseen view. The return to A is scored against the real input image and does measure accuracy.

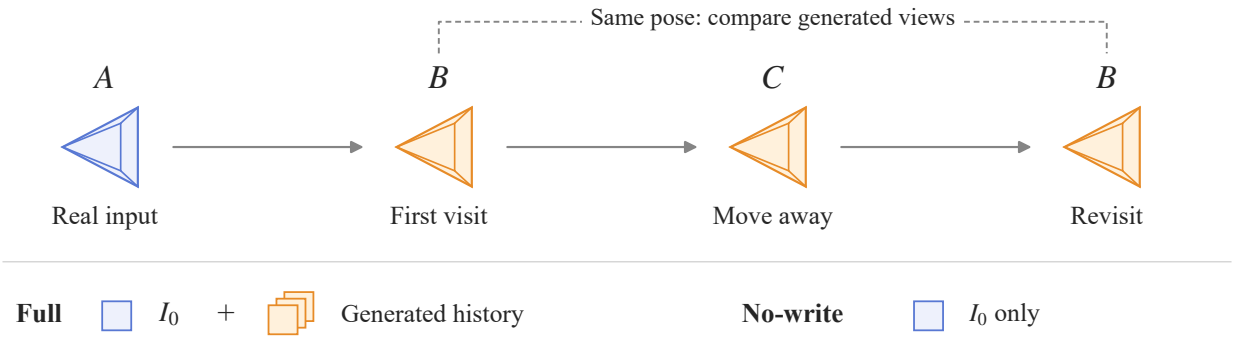

Figure S1. Generated-content revisit protocol. View B reveals content absent from the seed A and is generated rather than observed, so the return at B is scored against the first-generated frame at that pose and not against ground truth. Camera conditioning and the generated-prefix continuation rule are held fixed between the two rows.

Table S1. **Development-scale revisit comparison.** Evaluation on 20 static development scenes, disjoint from test-100. Return to A uses the real input image as reference; revisit B uses the first-generated frame at the same pose (Fig. S1). No-write retains the initial-image cache and disables later writes. Both variants share the camera trajectories, sampler, and scoring code.

	Return to A			Revisit B		
Variant	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
ENGRAM (full)	20.876	0.7496	0.2131	16.866	0.6636	0.4799
No-write	17.629	0.6921	0.3275	12.465	0.5172	0.6971

Table S1 reports both columns on 20 static development scenes held out from the test-100 evaluation. The margin is larger at the generated view than at the observed one. A model that only replays the initial image has less to lose at the pose it was conditioned on, so this ordering is not what initial-view anchoring predicts. We treat these scenes as development-scale evidence: they are fewer than the test-100 set and were not reserved for a final comparison.

E. Qualitative Results

This section complements Tables 2 and 1 with archived rollouts. Images are uniformly resized without enhancement or retouching. LSM-World results use our reproduction, as no checkpoint was released.

E.1. Closed-loop revisit

Figure S2 compares ViewCrafter, FlexWorld, LSM-World, Spatia, and ENGRAM on the same four scenes. The camera leaves the input view and returns; each method's frame 152 is compared with the real input image. The PSNR labels report individual scene scores rather than dataset averages.

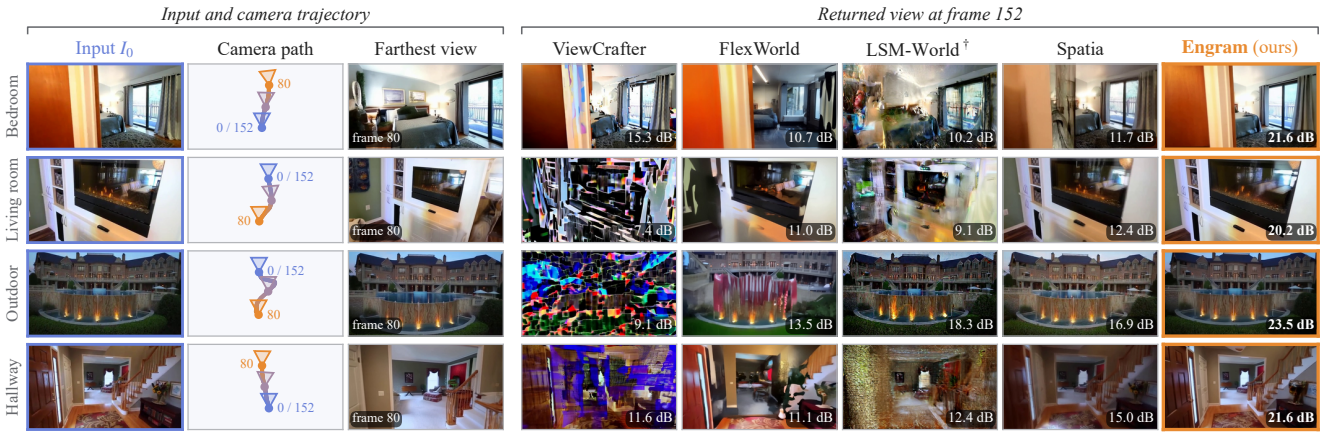

Figure S2. **Closed-loop revisit on RealEstate10K.** Starting from the input I_0, the camera follows the dataset trajectory for 81 frames and then returns; frame 152 should reproduce the input view. *Camera path* is a top-down view of the outbound route (blue: start, orange: farthest pose; triangles show heading), and *farthest view* is ENGRAM's frame 80, showing that the camera has left the input view. The five right-hand columns show each method's returned frame with its PSNR against I_0. All frames are archived outputs, uniformly resized, without enhancement. Scenes were chosen after inspecting 26 test scenes to cover different scene types and to make the differences visible. Table 2 reports all scenes. †Our reproduction.

E.2. Novel-view synthesis

Figure S3 follows the target camera trajectory through four RealEstate10K scenes. Ground-truth frames and the five methods are aligned at eight frames, from 10 to 80 in steps of 10. Each method's PSNR is averaged over frames 1–80 of that scene, using the same scoring protocol as Table 2.

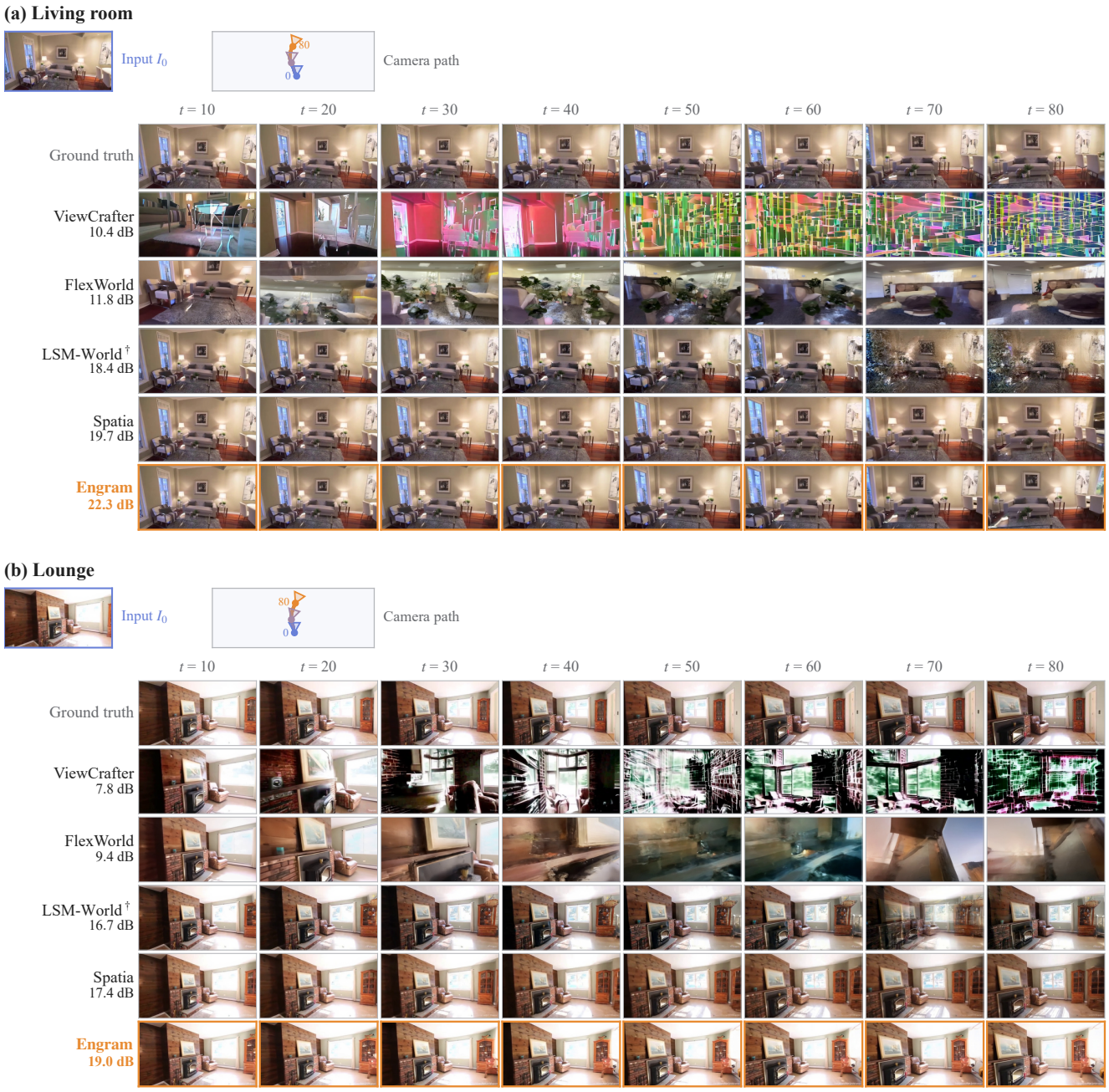

Figure S3. **Novel-view synthesis along the dataset trajectory.** Each block shows the input, the top-down camera path, and eight frames ($t = 10, 20, \ldots, 80$) of the ground truth and five methods for one RealEstate10K test scene. Panels (a)–(b) are shown here; (c)–(d) continue on the next page. The value under each method name is that scene's mean PSNR over frames 1–80. Differences in this setting are smaller than in the closed-loop setting, consistent with Table 2. †Our reproduction.

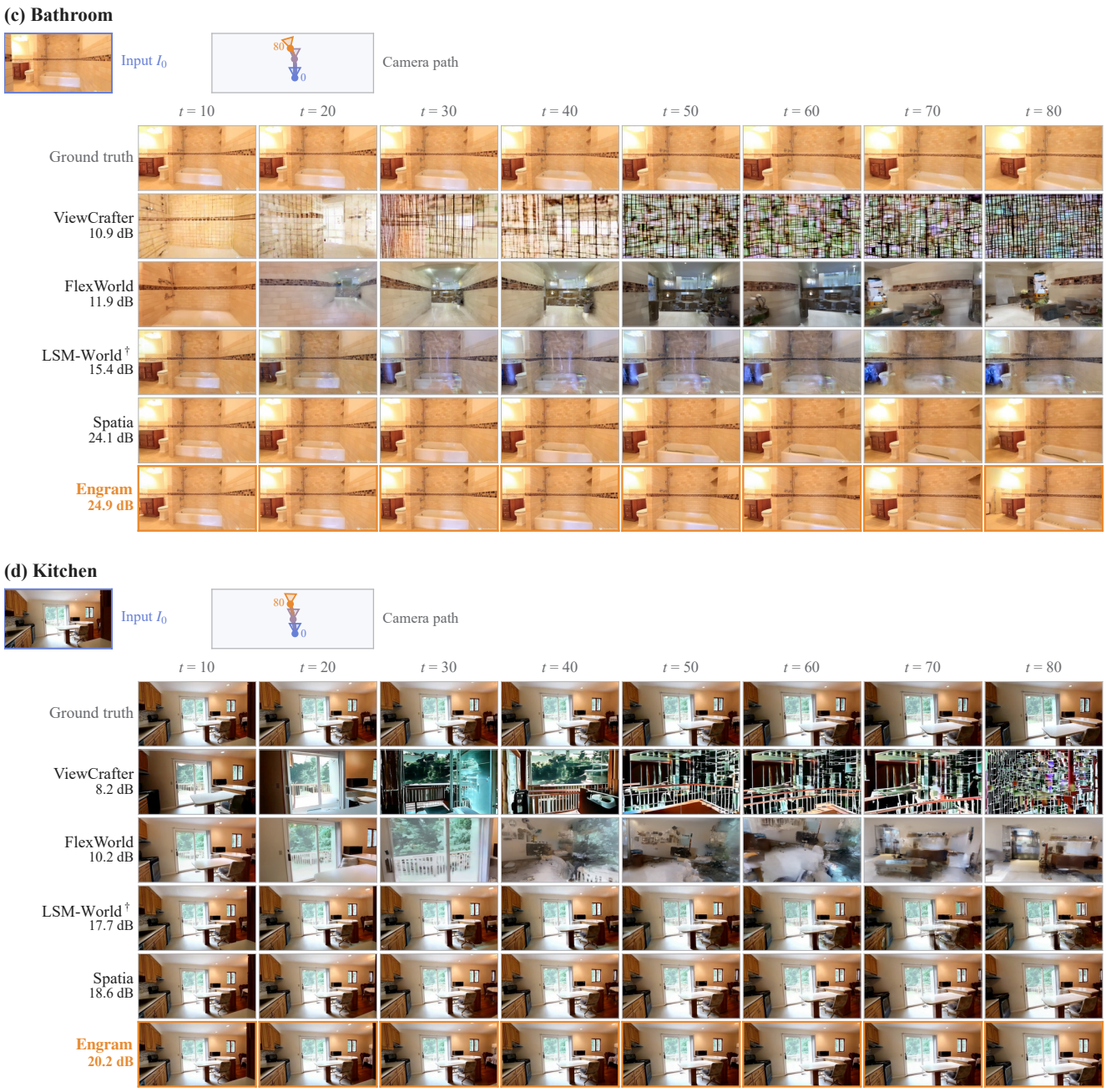

Figure S3. **Novel-view synthesis (continued).** (c) Bathroom and (d) kitchen, using the same methods, frame indices, and scoring protocol as panels (a)–(b). †Our reproduction.

E.3. WorldScore generation

Figure S4 compares Wan2.2-TI2V-5B, CogVideoX, LTX-Video, Spatia, LSM-World, and ENGRAM on five benchmark inputs. The scenes span photorealistic and stylized, indoor and outdoor environments, and a dynamic case. For each scene, two aligned rows show how the six rollouts evolve under the indicated camera command.

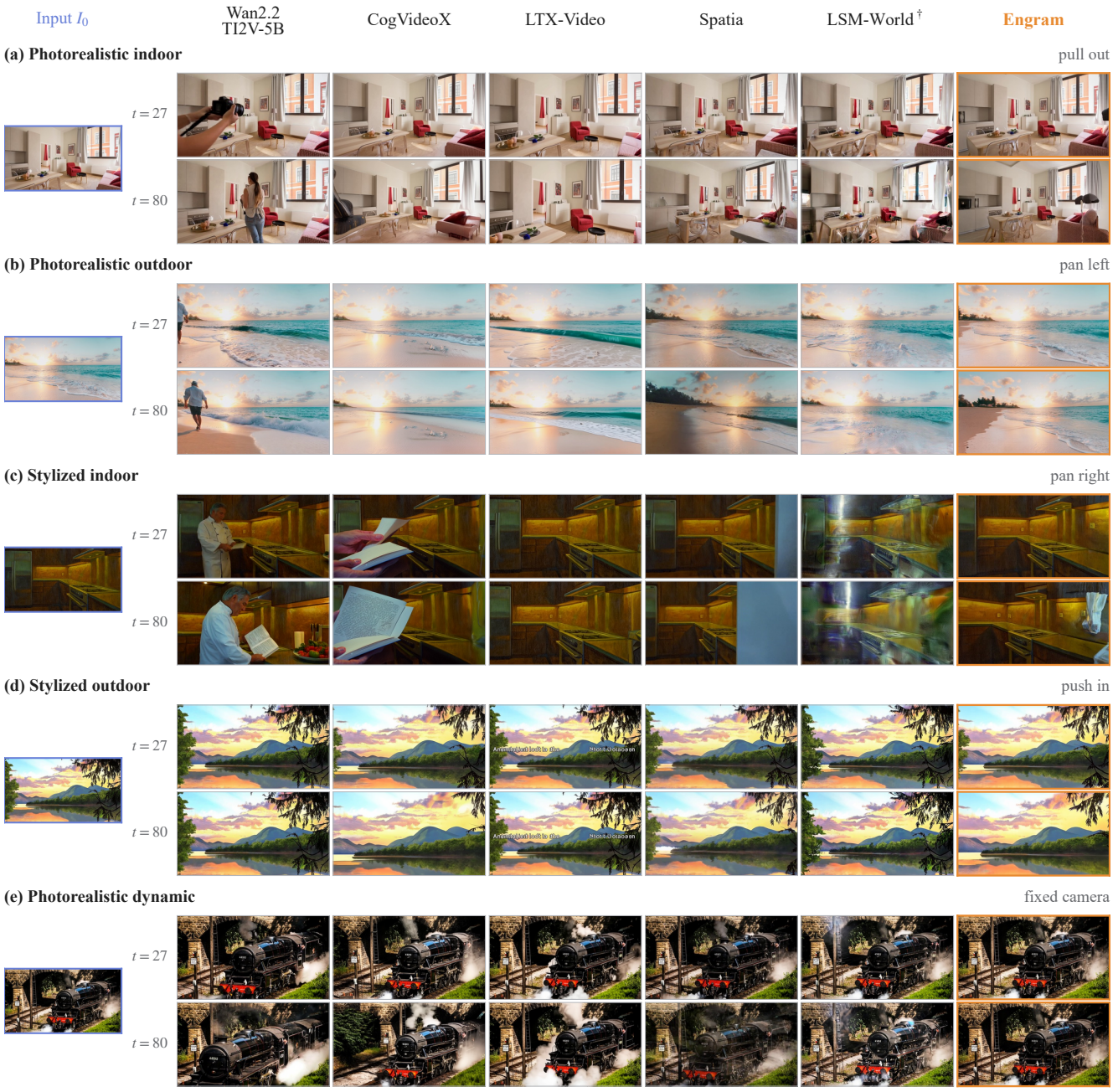

Figure S4. **WorldScore generation across six methods.** Each scene block shows one shared input and frames 27 and 80 of each 81-frame rollout, with methods aligned in columns. All images are archived outputs, uniformly resized. †Our reproduction.

E.4. Failure cases

Figure S5 shows two limitations on the same RealEstate10K test protocol. In the first case, newly revealed content differs from the ground truth; in the second, an unsupported dark occluder appears late in the rollout. The ground truth and all five methods are shown at matching frames.

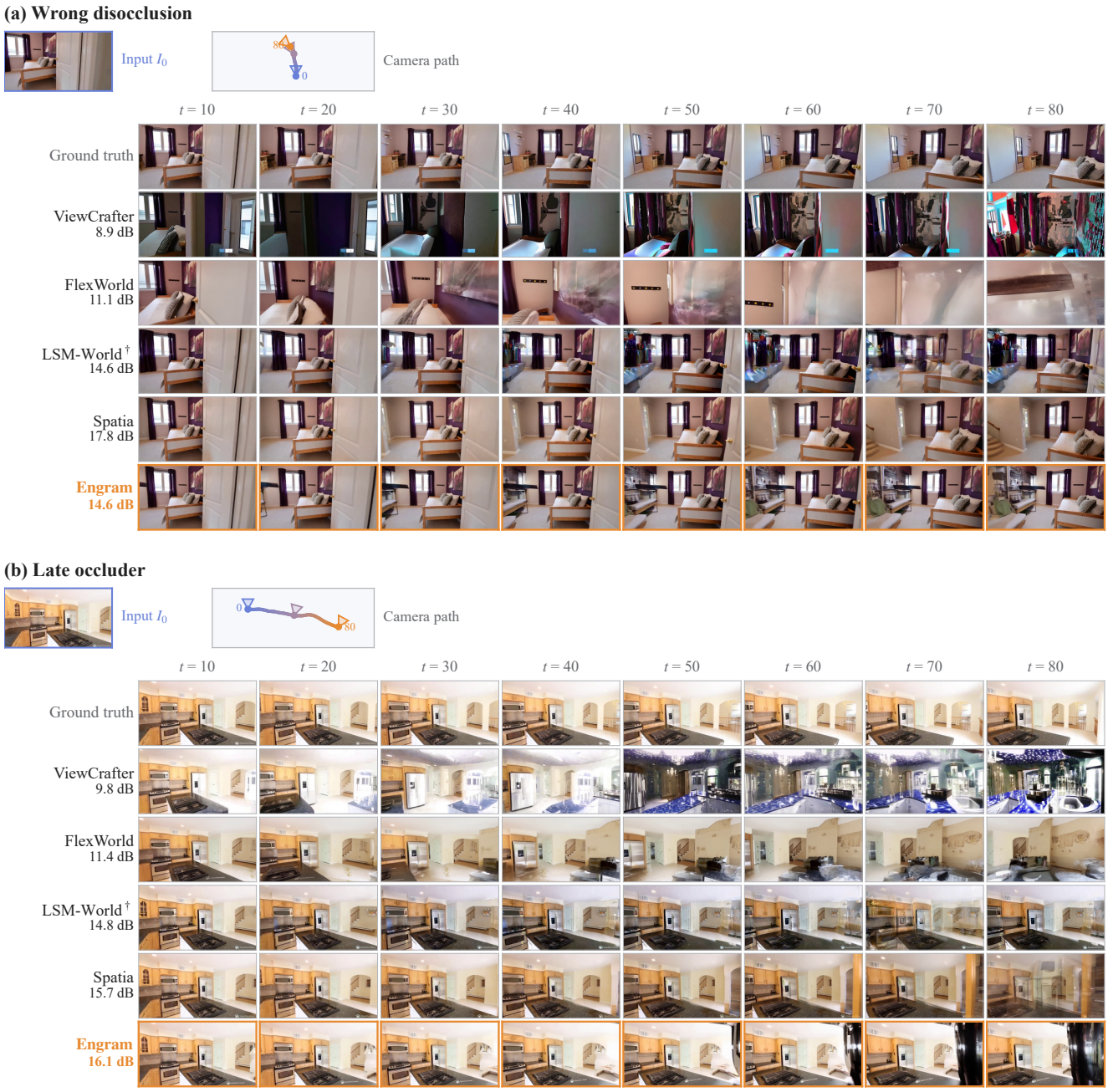

Figure S5. **Failure cases of ENGRAM.** (a) When the path opens onto a room that the input never showed, ENGRAM generates content that disagrees with the ground truth and scores below Spatia on this scene. (b) Late in the rollout a dark occluder appears that is not in the scene. Rows show eight aligned frames ($t = 10, 20, \ldots, 80$) for the ground truth and five methods. Both examples come from the same test set and protocol as Fig. S3. †Our reproduction.