LEARNING A SPATIAL MEMORY CODEC FOR VIDEO WORLD MODELS

Anonymous authors
Paper under double-blind review

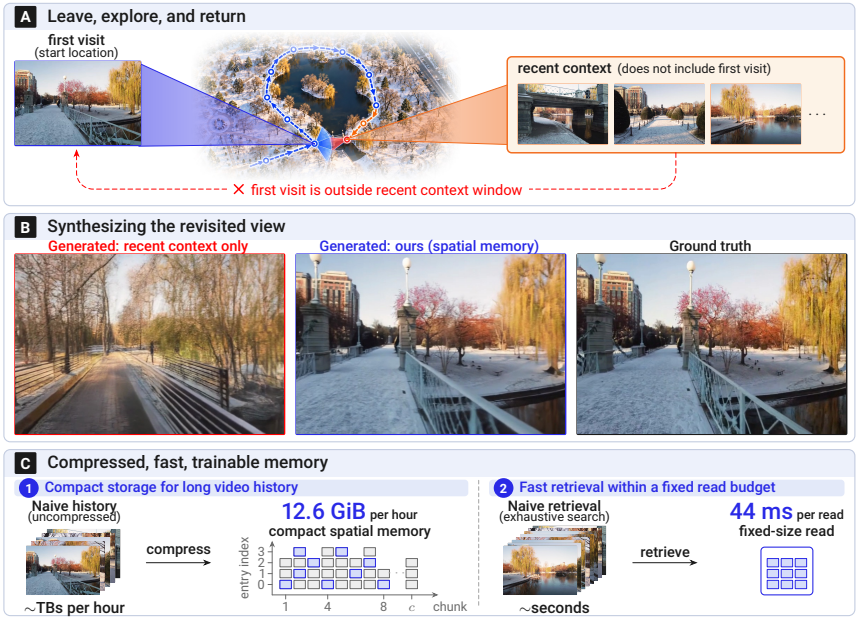

Figure 1: Video world models must remember a place after its earlier views leave recent context **(A)**; otherwise, the scene can change on return **(B)**. Keeping the full history in context is costly. Our learned codec compresses past video into addressable entries and searches compact landmarks by target pose and time **(C)**, supplying relevant evidence within a fixed read budget.

ABSTRACT

Generative video world models are emerging as a foundation for interactive agents, simulators, and creative tools that synthesize visual trajectories through spatial environments. Scaling such models to long trajectories requires more than local temporal coherence: when a place is revisited, the model must recover visual information from much earlier observations. Since keeping all observations in the model's context is not scalable, the model needs a long-term memory that preserves useful visual information while keeping storage compact and retrieval efficient. Existing approaches summarize history into a fixed-size state, retain a growing archive, or rely on explicit 3D reconstruction. These choices can lose important visual information, incur high storage and retrieval costs, or require handcrafted update rules. We introduce a learned spatial memory codec that stores history in compact learned entries. These entries are retrieved by a reader through a hierarchical, pose-indexed mechanism that efficiently selects those most relevant to each generated view. Specifically, each video chunk is compressed into a small set of entries, each paired with a compact index representation that we call a landmark. Each landmark summarizes its corresponding entry and records where and when it was observed. To form the generation context, the reader first selects a fixed-size set of the most relevant landmarks and then uses only their corresponding entries. We show that our memory codec balances compact storage, efficient retrieval, and consistent generation across challenging revisits, including viewpoint changes and full U-turns. We further demonstrate retrieval from nearly two hours of observations, far beyond the training horizon.

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

Recent advances in video models have motivated their use for generating interactive trajectories through visual environments (Parker-Holder & Fruchter, 2025; Qian et al., 2026). Such trajectories require long-term memory: when a place is revisited, the model should remain consistent with what it showed before. In video diffusion transformers, such consistency is typically achieved by attention over the frame context (Wan Team et al., 2025). However, extending this context to all past observations is not practical, because attention cost grows quadratically with context length (Vaswani et al., 2017). This creates the need for a memory mechanism that stores historical visual information outside the generator context and retrieves only what is needed.

A memory module must balance storage, retrieval cost, and consistency. These goals are coupled in several ways. Compressing history reduces storage, but can remove details needed at revisits. A larger archive preserves more information, but increases retrieval cost. Reading a small archive subset is efficient, but effective only if it contains the information needed for the target view.

One way to ease the storage–retrieval tradeoff is to make memory addressable, giving each stored item a lightweight index used to identify relevant content before it is read. Explicit 3D memories are naturally indexed by spatial location but rely on reconstruction pipelines and hand-designed update rules (Ren et al., 2025; Li et al., 2025b). Hybrid memories combine explicit geometric retrieval with learned conditioning but retain predefined spatial representations and retrieval rules (Yu et al., 2026b; Qian et al., 2026). Fully learned memories avoid explicit reconstruction, but fixed-size states may discard information needed for revisits, whereas growing compressed histories make retrieval increasingly costly (Wei et al., 2026; Peng et al., 2026; Hong et al., 2025).

We introduce a method that treats long-term memory as a learned codec for world state: an encoder writes compact representations of past observations, and a reader decodes the subset needed by the video generator. Each observed video chunk is compressed into learned memory entries, reducing the storage required for a growing archive. To allocate storage adaptively, an importance ordering determines how much storage each chunk receives according to a requested reconstruction quality. For efficient retrieval, each entry is paired with a compact index representation, called a landmark, that serves as its address. At query time, given the pose and time of a target view, the reader scores the landmarks to select relevant entries and decodes only those entries into target-view features, forming a hierarchical retrieval mechanism we call Sparse Landmark Attention. Together, compact entries, adaptive storage, and learned landmark retrieval provide a practical balance between storage, retrieval cost, and consistency.

We train the method in two stages. First, we train the memory codec and landmark-based reader through future-view reconstruction. Second, we jointly train the codec and video generator end to end, using retrieved memory features as context. At inference, the memory uses either a fixed archive of observed context or an autoregressively updated archive of generated chunks.

We first evaluate the memory codec independently of the video generator through held-out-view reconstruction, examining compression, sparse retrieval, adaptive storage, and long-context behavior. We then integrate the memory with the video generator, and compare against existing world models. Across these experiments, we consider dynamic environments, revisits from nearby viewpoints, reverse traversals and U-turns, and archives spanning nearly two hours.

Our contributions are threefold:

- **A learned spatial memory architecture.** We store history in compact, adaptively allocated entries, each paired with a learned landmark address for pose-conditioned hierarchical retrieval under a bounded read budget.
- **A training framework for generation-aware memory.** Our two-stage training strategy combines future-view reconstruction pretraining with joint optimization of the memory codec and video generator, enabling memory representations that specialize in video generation.
- **An empirical demonstration of scalable, consistent video generation.** Across 32 clips from six datasets, our bidirectional video model improves mean PSNR by 2.16 dB, reduces LPIPS by 28%, and reduces camera rotation error by 55% relative to the strongest baseline on each metric. The memory also scales to nearly two-hour archives without long-context fine-tuning.

2 Related Work

Explicit and hybrid spatial caches. World models and camera-controlled video generators index observations using reconstructed point clouds or surfels (Ren et al., 2025; Li et al., 2025b), pose- and time-indexed frames (Xiao et al., 2025), or 3D-localized patches (Yu et al., 2026b). Hybrid world models combine camera-aware memory with autoregressive temporal context (Wang et al., 2026c; Qian et al., 2026; AlayaWorld Team et al., 2026). These memories offer interpretable spatial access through explicit geometry or pose-based retrieval. We instead learn the stored entries and their retrieval without maintaining a persistent 3D map.

Implicit and cache-native memory. Long-context models carry compressed, recurrent, or test-time learned states (Rae et al., 2020; Bulatov et al., 2022; Behrouz et al., 2025). Video world models compress history into fixed-size states (Wei et al., 2026), retrieve historical frames or cached features into a bounded context (Sun et al., 2025; Chen et al., 2026), or maintain compressed histories accessed through attention or pruning (Hong et al., 2025; Peng et al., 2026; Yi et al., 2026; Wu et al., 2026b). Learned retrieval methods query the full context in shallow layers (Yu et al., 2026a) or select full-resolution KV chunks using pooled query-key summaries (Xu et al., 2026). Related long-video methods use sparse selection or distant-history reconstruction (Cai et al., 2025; Xue et al., 2026). We instead learn compact entries and dedicated pose- and time-aware landmarks for hierarchical retrieval under a bounded read budget.

Learned and variable-rate codecs. Building on nested dropout (Rippel et al., 2014), flexible visual tokenizers learn ordered, variable-length codes using nested or tail-token dropout (Yan et al., 2025; Bachmann et al., 2025; Li et al., 2026). However, they target visual reconstruction and generation rather than persistent memory for world models. Existing world models instead use fixed-rate codecs (Hu et al., 2026a) or allocate compression using predefined relevance rules (Oshima et al., 2026). We apply learned variable-rate representation to long-term world-model memory, using nested dropout and learned allocation to adapt storage per observed video chunk.

Novel-view synthesis and geometry-aware state. Feed-forward geometry models such as VGGT and VGGT-Ω learn cross-view features for 3D reconstruction (Wang et al., 2025a; 2026b). Learned novel-view synthesis methods adapt such encoders or directly compress fixed source views into latent scene representations queried to render targets (Szymanowicz et al., 2026; Sajjadi et al., 2022; Jin et al., 2025). They primarily reconstruct supplied scenes rather than maintain memory over growing trajectories or generate newly explored content. Related video world models use geometry-aware state to coordinate views or retrieve pose-aligned tokens (Wu et al., 2026a; Chou et al., 2025). Our method instead writes learned entries to a persistent archive for continued generation.

Sparse access. Sparse attention reduces computation by restricting expensive token-level attention to a selected subset of tokens or blocks. HiLS, building on Landmark Attention, applies this hierarchy to long-context language models by using compact landmarks to select relevant token blocks before accessing their full tokens (Mohtashami & Jaggi, 2023; Hu et al., 2026b). We adapt this mechanism to spatial memory for video world models by replacing text blocks with learned memory entries and selecting them using the target pose and time. Each entry has a learned landmark, and PRoPE (Li et al., 2025a) encodes relative camera geometry during selection.

3 Method

We formulate long-term memory as a learned spatial codec: it encodes past video observations into a compact representation and decodes the visual information needed for a new view on demand. The codec supports two operations on an *archive* of learned memory entries: *writing* and *reading*. The writer encodes each observed video chunk into a small set of learned memory entries (Figure 2A). When a memory entry is committed to the archive, it is paired with a compact landmark that summarizes the entry and serves as its retrieval address (Figure 2B). Given the pose and time of a target view requested by a downstream module (e.g., a video generator), the reader scores the landmarks, selects the K most relevant entries, and decodes them into target-view features (Figure 2C).

3.1 Writing Learned Memory Entries

At write step c, the writer (illustrated in Figure 2A) receives an eight-frame observation chunk $o_c = (F_c, \mathcal{P}_c, \tau_c)$, where F_c, $\mathcal{P}_c$, and τ_c denote visual frames, camera poses, and observation times, respectively. Its geometry encoder, initialized from VGGT-Ω (Wang et al., 2026b), jointly processes o_c and $L_{\max}$ learned register token sets (orange grids). The encoder also attends to key-value representations cached from the preceding W observation chunks. The encoder discards its frame

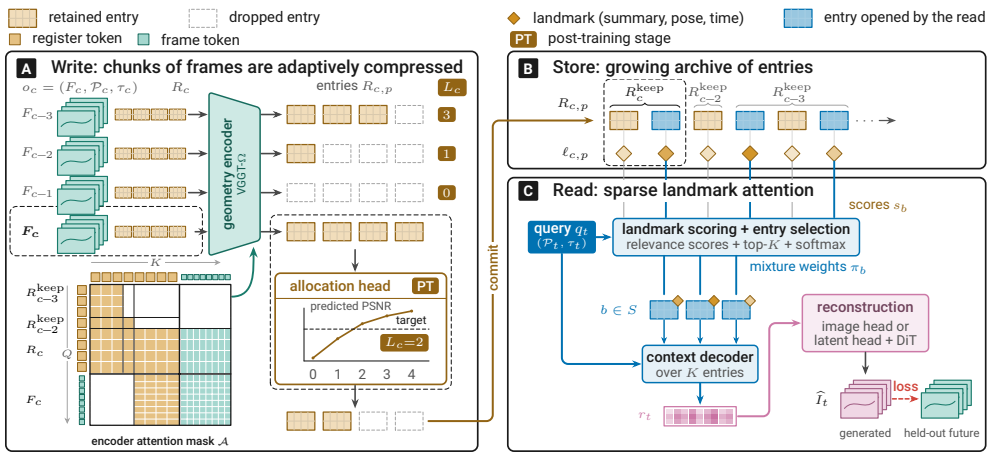

Figure 2: **Overview of our learned spatial memory codec. (A) Write:** the writer compresses each observed video chunk into an adaptively allocated set of memory entries. **(B) Store:** retained entries are committed to a growing archive, and each erntry is paired with a compact landmark. **(C) Read:** given a target pose and time, the reader scores the landmarks, selects the top-K entries, and decodes only the selected entries into target-view features.

outputs and uses the register outputs as candidate memory entries $R_c = (R_{c,0}, \ldots, R_{c,L_{\max}-1})$, where each $R_{c,p} \in \mathbb{R}^{hw \times D}$ is an $h \times w$ grid of D-dimensional tokens. A post-trained allocation head predicts a content-dependent number L_c of entries to retain and commits the corresponding prefix $R_c^{\text{keep}} = (R_{c,0}, \ldots, R_{c,L_c-1})$ to the archive.

Geometry-aware causal encoding. To encode spatially meaningful memory entries, we associate each frame's tokens with its camera pose and observation time. The register token sets have no observed pose or time of their own, so each is assigned to that of a distinct predefined frame in the chunk. To preserve camera relationships under changes to the global coordinate frame, π^3-style (Wang et al., 2025c) pose-injection blocks in the encoder use PRoPE (Li et al., 2025a) to encode *relative* camera geometry and temporal RoPE (Su et al., 2021) to encode observation time. Table E.1 lists the camera assignments of the register token sets.

To account for repeated observations, the encoder attends to a KV-cache of retained entries from recent chunks when encoding a new one. Specifically, it operates over a local window of $W + 1$ chunks with causal attention that prevents access to future chunks. Figure 2A illustrates the attention mask, with queries along the rows and keys along the columns, and Equation D.1 defines it.

Adaptive retention. A fixed allocation wastes storage on redundant chunks that need fewer entries. To reduce this waste, we apply nested dropout (Rippel et al., 2014) during codec pretraining: the decoder receives only a prefix $(R_{c,0}, \ldots, R_{c,L-1})$, where $L \in \{1, \ldots, L_{\max}\}$. Training across prefix lengths encourages early entries to preserve the most useful information and later entries to add finer details. After codec pretraining, we freeze the codec and train an allocation head to predict future PSNR for each possible prefix. At inference, given a requested PSNR, the head selects the shortest prefix that meets the request, or the one with the highest predicted PSNR if none does. Only retained entries are committed to the archive and made available to subsequent writes (Appendix F.1).

Committing and indexing entries. At commit time, we pair each retained entry $R_{c,p}$ with a compact landmark $\ell_{c,p}$ (Figure 2B). The reader scores these landmarks to select the entries to decode. Each landmark records the entry's pose $\mathcal{P}_{c,p}$ and time $\tau_{c,p}$ and contains H heads, each consisting of a d-dimensional summary $m_{c,p,h}$ and an entropy scalar $e_{c,p,h}$:

$$\ell_{c,p} = \left(\{(m_{c,p,h}, e_{c,p,h})\}_{h=1}^{H}, \mathcal{P}_{c,p}, \tau_{c,p}\right). \tag{1}$$

Inspired by HiLS (Hu et al., 2026b), we construct each summary head $m_{c,p,h}$ using a learned query that weights the entry's tokens instead of averaging them uniformly. For each entry position p and head h, the query is shared across chunks, while each entry produces its own summary. PRoPE incorporates the entry's pose into this aggregation. The head entropy $e_{c,p,h}$, computed from the weights that the learned query assigns to the entry tokens, indicates whether the summary draws broadly from the entry or concentrates on a few tokens (Equations D.2–D.3).

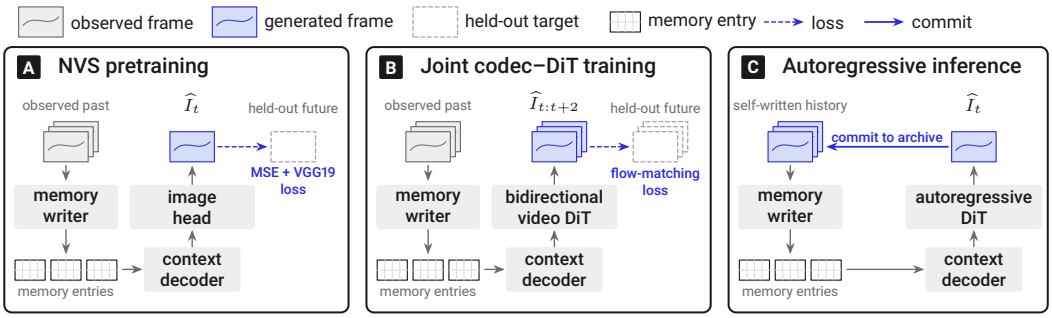

Figure 3: Training and autoregressive generation framework. **(A)** Future-view reconstruction pretrains the codec; VAE-latent adaptation then aligns its output with the generator. **(B)** The codec and bidirectional video DiT train jointly under the flow-matching objective. **(C)** An autoregressive DiT generates from its stored history and commits completed chunks before generating the next chunk.

3.2 Reading Learned Memory Entries

The reader receives a target query $q_t = (\mathcal{P}_t, \tau_t)$ from the downstream module (Figure 2C). Decoding all archived entries into target-view features would require dense attention over the entire archive. Instead, we introduce *Sparse Landmark Attention (SLA)*, which first scores the compact landmarks to select at most K entries and then decodes only the selected entries into target-view features r_t. The decoder therefore processes at most K memory entries per query, while landmark scoring scales with the number of archived entries.

Selecting memory entries. In the first stage of SLA, denoted SLA-Select, the reader scores the archived entries for a given target query q_t and selects the K most relevant ones. Specifically, it constructs learned query vectors from the target camera and compares them via dot products with the learned summary vectors of each landmark, using the head entropies as biases. A learned scoring function then combines these comparisons with explicit geometric and temporal measurements to produce a relevance score $s_{t,(c,p)}$ between q_t and each archived entry $R_{c,p}$. The K entries with the highest scores are passed to the second stage of SLA for decoding (Equations D.4–D.6).

Decoding the selected entries. The second stage of SLA, denoted SLA-Decode, uses the context decoder to transform the selected entries into a spatial grid of target-view features r_t for query q_t. The decoder operates on learned target-view tokens whose attention is conditioned on the target camera pose and time. Appendix E details target-token initialization. In the first four layers, each target-view token attends separately to each selected entry, and the resulting outputs are combined using normalized mixture weights derived from the relevance scores $s_{t,(c,p)}$ (Equation D.7). These weights are shared across the four layers, attention heads, and target-view token positions. In the remaining eight layers, the tokens from all selected entries form a single set over which attention is jointly normalized, allowing their contributions to vary across layers, heads, and target positions. Table 1b compares shared weighting, standard attention, and our hybrid design.

Learning retrieval. Although top-K selection is discrete, the relevance scores $s_{t,(c,p)}$ of the selected entries determine the differentiable mixture weights used in SLA-Decode, allowing gradients to propagate through the scoring mechanism. Moreover, relevance-score computation is normalized over the full archive, so each selected score depends on all landmark logits, allowing gradients to reach unselected landmarks. To additionally train the full token representations of unselected entries, we occasionally replace a selected entry with an unselected one during training. Appendix C provides additional details and analysis.

3.3 Training and Video Generation

Memory-codec pretraining. We pretrain the memory codec independently of the video diffusion model through future-view reconstruction (Figure 3A). The writer constructs an archive from observed video chunks, excluding the held-out target view. Given only the target pose and time, the reader produces target-view features r_t, from which an image head reconstructs the target. An image MSE loss and a VGG19 perceptual loss (Equation D.8) jointly train the writer, reader, and image head. Training targets include delayed revisits and viewpoint changes, requiring the codec to preserve and retrieve long-term visual information. By bypassing the video diffusion model, this pretraining task allows us to evaluate retrieval independently of video generation.

Training the allocation head. With the codec frozen, we evaluate all $L_{\max} + 1$ retention choices, from zero to $L_{\max}$ entries. For each prefix, we commit its entries and decode future views to measure reconstruction PSNR. The allocation head is trained to predict these PSNR values (Appendix F.1).

Video-generation integration. Before connecting the memory codec to the video diffusion transformer (DiT), we attach a latent head that maps the target-view features r_t to predicted memory latents $\widehat{z}_t^{\text{mem}}$ in the generator's VAE latent space. We fine-tune the writer, reader, and latent head with latent and decoded-image losses through the frozen VAE decoder (Figure 3A; Equation D.9). The DiT's patch embedding converts both memory and video latents into tokens, which are concatenated without RGB decoding and re-encoding. We then jointly train the memory codec and DiT using flow matching, while keeping the VAE frozen (Figure 3B; Equation D.10).

For autoregressive generation, we adapt the video model for causal generation using the training procedure described in Appendix E. After each chunk is generated, the writer commits it to the archive before generating the next chunk (Figure 3C).

4 Experiments

In this section, we evaluate how our memory balances revisit consistency, storage, and retrieval efficiency as the observed history grows. We first study the effects of compression and sparse retrieval on memory reconstruction. We then ablate the components of SLA and evaluate adaptive storage allocation. Next, we test generalization and scaling to observation archives spanning nearly two hours without long-context fine-tuning. Finally, we integrate the memory with video models and compare against prior world models. For the memory-codec experiments, we reconstruct held-out views using the image head and report full-frame PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018), averaged within each scene and then across scenes. The project page presents additional qualitative videos, including challenging dynamic scenes, long-gap revisits, reverse traversals, and U-turns. Appendices A, B, and E describe the datasets, protocols, and training details.

4.1 Encoder Compression and Hierarchical Reading

Table 1a compares encoders and read modes variants on our DL3DV test set of 64 two-loop scenes. Each archive is built from 128 first-loop observations, and 32 targets are sampled across the following 96 second-loop frames, excluding targets within 32 frames of memory.

Configurations. We compare three streamable encoders under both dense reading and SLA-Select. All encoders use eight-frame chunks. In Transformer 1-to-1, as in LagerNVS, decoding uses the encoded input image-patch tokens, so each chunk stores $8 \times 1{,}536$ tokens. In the bottleneck encoders, the decoders instead read from additional learned register tokens. The fixed and adaptive bottlenecks store four 2×512-token and 512-token entries per chunk, respectively. For the adaptive bottleneck, we retain all four entries at inference in this comparison. Comp. denotes the ratio of stored tokens relative to Transformer 1-to-1. A dense read processes the entire 128-frame archive of 16 chunks. SLA instead reads K units of 1,536, 1,024, and 512 tokens, respectively; setting $K{=}4/6/12$ therefore fixes the read budget at 6,144 tokens. The separate LagerNVS baseline uses its released checkpoint and re-encodes RGB at query time.

Table 1: **Memory codec ablations. (a)** Encoder compression and dense versus sparse reading. **(b)** Entry-selection and decoding variants. **Bold** and underline indicate the best and second-best results, respectively. The oracle is not ranked.

(a) Encoder and read mode

Encoder	Read	PSNR↑	SSIM↑	LPIPS↓	Comp.	Read tok.
Stored-RGB reference (LagerNVS)						
16 nearest poses	re-encode	17.17	0.530	0.391	RGB	—
All 128 sources	dense	16.92	0.521	0.410	RGB	—
Streamable encoders (eight-frame chunks)						
Transformer 1-to-1	dense	14.60	0.524	0.777	1×	196,608
Transformer 1-to-1	SLA ($K{=}4$)	15.53	0.532	0.675	1×	6,144
Fixed bottleneck	dense	19.82	0.614	<u>0.311</u>	12×	16,384
Fixed bottleneck	SLA ($K{=}6$)	20.12	<u>0.631</u>	0.314	12×	6,144
Adaptive bottleneck	dense	<u>20.24</u>	0.631	**0.308**	6×	32,768
Adaptive bottleneck	SLA ($K{=}12$)	**20.28**	**0.637**	0.312	6×	6,144

(b) Entry selection and decoding

Control	PSNR↑	SSIM↑	LPIPS↓
Entry selection			
Random entries	18.89	0.599	0.392
Uniform stride	19.53	0.610	0.354
Nearest pose	<u>20.17</u>	<u>0.636</u>	<u>0.315</u>
Learned SLA-Select	**20.28**	**0.637**	**0.312**
Attn. oracle*	20.31	0.638	0.309
Reader on selected entries			
12-layer SLA-Decode	19.03	0.593	0.401
12-layer standard attn.	20.27	0.637	0.313

*The oracle ranks entries using dense-attention mass and requires a dense pass.

Effect of encoder compression. Both bottlenecks outperform the per-frame (1-to-1) representation under both dense and sparse reading. For a fixed read budget, compression allows the decoder to access a larger portion of the observed trajectory. Despite compressing the observations and restricting retrieval, the adaptive bottleneck with SLA achieves the highest PSNR and SSIM among the tested variants. The bottlenecks also outperform the LagerNVS reference, while encoding each chunk only once at write time avoids LagerNVS's repeated source encoding at query time.

Sparse versus dense reading. Across all encoders, the sparse variant improves PSNR over its separately trained dense counterpart while reading fewer tokens. We hypothesize that this advantage stems from separating retrieval from decoding. A dense reader must both identify relevant observations and reconstruct the target, whereas SLA-Select first narrows the context, allowing SLA-Decode to focus on reconstruction. Consistent with this hypothesis, the LagerNVS reference also performs better when reading only nearby source views rather than all observations.

Adaptive storage allocation. DL3DV contains more dynamic detail and faster camera motion than SpatialVID, so we expect it to require more storage at the same quality target. The learned allocations reflect this difference: at an 18.5 dB target, DL3DV retains 2.9 entries per chunk, compared with 0.63 on SpatialVID ($6.4\times$ compression; 14/16 clips meet the target). At a 21 dB target, DL3DV retains the full allocation, with reconstruction quality saturating at 21.1 dB. Appendix F reports the full sweep on the 32-clip development set used for checkpoint selection.

4.2 Learning Sparse Retrieval

Entry selection and decoding. Using the same scenes, archives, and targets as Table 1a, we compare different mechanisms for selecting the entries to decode. SLA-Select outperforms random and uniform-stride selection and nearly matches a dense-attention oracle, while nearest-pose selection is also competitive on these repeated-loop scenes. Training SLA-Select requires SLA-Decode, whose relevance-weighted attention provides gradients through the selection scores. However, once entries are selected, standard attention outperforms SLA weighting throughout the decoder. We therefore use a hybrid decoder: SLA weighting in the first four layers enables training the selection mechanism, while standard attention in the remaining eight layers improves decoding.

Reconstruction-driven retrieval. Figures 4A and B evaluates whether reconstruction alone can learn effective retrieval without direct supervision of entry relevance. During training, sparse reconstruction catches up with and eventually outperforms dense reading from the same checkpoint, while agreement with the dense-attention oracle increases. Together, these trends show that reconstruction alone learns both effective entry selection and decoding from the selected subset.

Effect of read budget. In Figure 4C, we vary the read budget (K) at inference using the same trained model. The default $K{=}12$ approaches dense-read quality while opening less than one fifth of the archive, and $K{=}16$ matches it. Quality then saturates as more entries are opened, showing that a small read budget captures most of the useful information without decoding the full archive.

4.3 Generalization to Long Contexts

We test whether the codec generalizes beyond its training horizon by building KITTI-360 archives (Liao et al., 2022) from nearly two hours of observations combined across independent drives, without long-context fine-tuning. During training, the codec sees at most 128 observations.

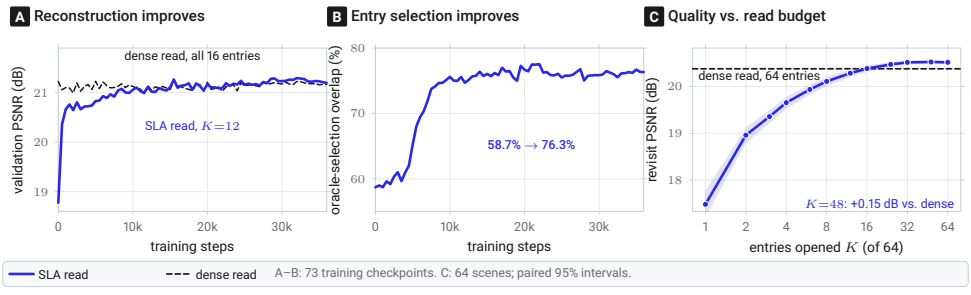

Figure 4: **Learning sparse retrieval.** **(A)** Sparse reconstruction improves during training and approaches dense reading. **(B)** Agreement between SLA-Select and the dense-attention top-K oracle increases during training. **(C)** Reconstruction quality improves with the read budget K and saturates near dense-read performance.

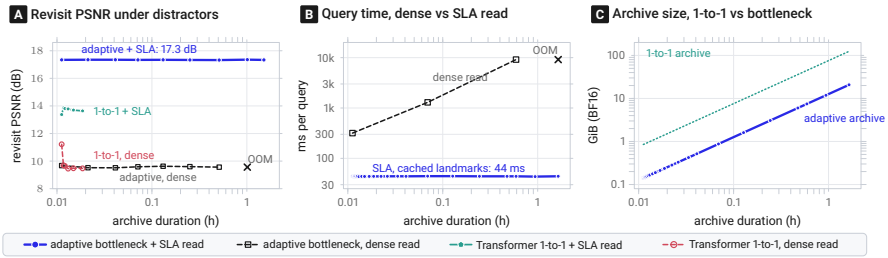

Figure 5: **Long-context retrieval.** Irrelevant observations are added while queries and relevant context remain fixed. **(A)** Revisit PSNR vs. archive duration. **(B)** Dense and sparse query latency. **(C)** Archive storage for Transformer 1-to-1 and the adaptive bottleneck with all entries retained. Archive duration is logarithmic; (B,C) also use logarithmic y-axes. OOM denotes an out-of-memory dense read.

Robustness to irrelevant context. We fix the query views and relevant context, then append unrelated observations while reading 12 entries per query. PSNR remains near 17.3 dB as the archive grows, whereas dense reading eventually runs out of memory (Figure 5A). PSNR is lower than on DL3DV because KITTI-360 is outside the training domain and revisits occur minutes apart, with changes in shadows and scene content. Across 384 strict revisits from six query drives, combining nine independent drives into archives containing 1.87–1.98 hours of observations changes PSNR by at most 0.3 dB relative to reading only relevant observations. Moreover, 92–100% of selected entries come from relevant chunks (Appendix G), showing that unrelated observations are rarely retrieved.

Runtime and storage. Using an optimized implementation on a single NVIDIA B200, with cached landmarks and a resident archive, retrieval and RGB decoding take about 44 ms for archives of up to 1.65 hours (Figure 5B), excluding video generation. Storage grows linearly at 12.6 GiB/hour (Figure 5C), and each chunk is encoded once in about 87 ms.

4.4 End-to-end Video Generation

We compare against implicit state- and cache-based systems SANA-WM, GIM-World, and ReWorld; the point-cloud cache of GEN3C; and the hybrid spatial memories of Matrix-Game 3.5 and AlayaWorld. We evaluate on six datasets with shared past observations and a clean starting frame, scoring only future frames. Our video model is a pose-conditioned Wan2.2-TI2V-5B trained jointly with the memory (Section 3.3). We report PSNR, LPIPS, and observed-anchor rotation error ΔR_Ω (VGGT-Ω). While PSNR is sensitive to moving objects and lighting changes, rotation error measures agreement with the commanded rotation relative to the observed scene.

Results. Both our bidirectional and autoregressive models outperform all baselines in their respective groups on every dataset in PSNR and LPIPS, and achieve the lowest mean rotation error (Table 2). The AR model also outperforms all bidirectional baselines on average, despite generating causally. These results are computed over the fixed evaluation cohort rather than the full datasets. Qualitative comparisons are shown in Figures 6 and H.1.

Table 2: End-to-end comparison on 32 fixed clips: PSNR↑/LPIPS↓ per dataset and averages, with rotation error ΔR_Ω (°, ↓). **Bold**/underline: best/second per group; memory-only unranked.

Method	KITTI-360	Oxford Spires	SpatialVID	Sekai walking	DL3DV	MIND	Average		
							PSNR↑	LPIPS↓	ΔR_Ω(°)↓
Bidirectional window generators									
Gen3C	10.05/.648	20.32/.349	16.94/.343	16.74/.363	15.12/.557	13.71/.482	15.48	0.457	7.52
Matrix-Game 3.5	10.20/.621	18.73/.330	16.54/.321	16.65/.379	15.75/.435	14.24/.485	15.35	0.429	3.17
GIM-World	9.03/.621	10.66/.632	12.66/.580	13.31/.618	12.02/.599	12.56/.575	11.71	0.604	37.70
Ours, bidirectional	**14.74/.386**	**20.72/.232**	**18.28/.247**	**18.13/.297**	**19.35/.288**	**14.65/.393**	**17.64**	**0.307**	**1.42**
Causal / streaming autoregressive generators									
ReWorld-5B	8.83/.687	11.72/.646	13.77/.503	12.46/.666	12.42/.653	12.63/.594	11.97	0.625	23.99
SANA-WM	9.15/.711	14.35/.570	14.42/.506	13.12/.556	12.20/.658	11.91/.640	12.52	0.607	20.13
AlayaWorld v1.1	9.80/.698	15.72/.489	14.31/.485	15.28/.462	12.90/.661	12.05/.654	13.34	0.575	26.27
Ours, AR	**13.60/.455**	**19.51/.309**	**18.03/.294**	**17.17/.356**	**17.78/.341**	**13.63/.475**	**16.62**	**0.372**	**1.73**
Memory-only reconstruction									
Ours, memory-only	16.63/.473	19.64/.443	19.92/.357	18.10/.517	21.19/.372	16.76/.465	18.71	0.438	2.94

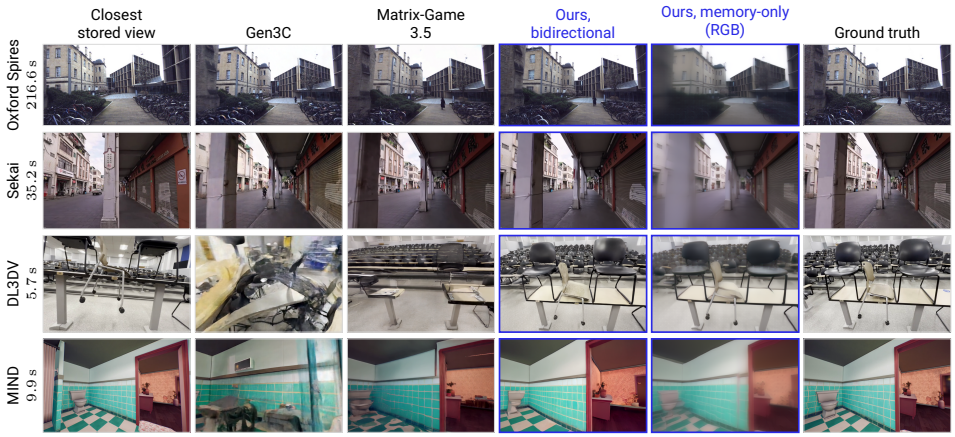

Figure 6: **Qualitative comparison to other world models.** Each row shows the same selected future target across methods. Memory-only uses the separately trained RGB codec. Vertical labels give the capture-time gap from the closest stored view to the target; methods may select other views. Gen3C and Matrix-Game 3.5 show distorted or missing fine structures and poorer geometric alignment with the target view.

Generation versus regression. The memory-only results (last row of Table 2 and second-to-last column of Figure 6) are produced by the separately trained RGB codec. As a deterministic regressor, it achieves the highest mean PSNR but blurs uncertain content. Bidirectional generation produces sharper outputs, improves LPIPS on every dataset, and halves the mean rotation error.

Object persistence. *Can 4D Foundation Models Remember?* (He et al., 2026) introduces Persist-Bench, a targeted memory benchmark that evaluates object permanence, continuity, and appearance against 360° references. Following the protocol in Appendix I, our bidirectional and autoregressive models achieve the best results on static objects: 91.77/86.25/90.28 for the bidirectional model, compared with the published best of 84.06/80.60/82.61 (Argus, Argus, and GEN3C, respectively). Both models trail GEN3C on dynamic objects.

5 Conclusion, Limitations, and Future Work

We introduced a learned spatial memory for video world models that connects a growing history to a bounded generation context. Our approach stores observations as compact learned entries and uses lightweight landmarks for hierarchical retrieval, decoding only the entries relevant to the target view. Adaptive retention adjusts storage to the observed content and requested reconstruction quality, allocating fewer entries when greater compression is sufficient. Unlike explicit 3D memories, our representation and retrieval mechanism are trained end to end with the video generator, allowing the memory to specialize for generation without explicit scene reconstruction. Our results show that this combination supports consistent revisits over long histories while compressing the stored history and keeping decoder context bounded.

Our method is not without limitations. First, our formulation assumes known camera pose and time. We view this primarily as a separation of concerns: our goal is to study long-term visual memory, while pose estimation or action-conditioned pose prediction could be learned jointly with memory and generation within the same end-to-end framework. Second, evaluating memory over long self-generated trajectories requires an autoregressive video generator that remains stable over long rollouts. Otherwise, generation drift becomes difficult to disentangle from memory errors. Reducing such drift is itself an active and important research direction in long-horizon video generation. We therefore isolate long-term memory using observed histories, while evaluating autoregressive generation at horizons where the underlying video model remains reliable.

Looking forward, Sparse Landmark Attention bounds the expensive decoding stage to a fixed number of entries, but landmark scoring still grows linearly with the archive. An interesting direction is to extend the hierarchical principle to landmark retrieval itself, using additional levels of compact indices to search increasingly long histories. Learned consolidation and update mechanisms could further merge repeated observations and revise stored state as environments change, improving storage efficiency and memory in dynamic scenes.

AI USE STATEMENT

Large language models assisted with code development and debugging, internal manuscript review and editing, LaTeX and figure layout, and checks of consistency between the implementation, experimental descriptions and reported results. The authors are responsible for verifying the implementation, references, empirical results and final text, and take full responsibility for the content of this submission.

ETHICS STATEMENT

This work studies memory-conditioned video generation using existing video datasets and synthetic data from a game engine. Improved long-term visual memory can support simulation and interactive content creation, but can also retain sensitive details from observed scenes and produce misleading depictions of real places or events. Memory archives should be treated as potentially sensitive data, with appropriate access and retention controls. Generated videos should be identified as synthetic, and downstream use should respect dataset licenses, privacy and consent.

REPRODUCIBILITY STATEMENT

We will supply all evaluation code to enable independent evaluation of all reported results. Section 3 describes the memory architecture, and Appendix C gives the sparse landmark attention analysis. Appendices A and B document benchmark construction, preprocessing, evaluation cohorts, metrics and model-specific comparison settings. Appendix D collects the formulas used in Section 3. Appendix E describes architecture and implementation details, training stages, data sampling and objectives, and identifies the remaining implementation verification items. Appendix F consolidates adaptive-write allocation and calibration; Appendix G reports additional retrieval evaluations. These descriptions distinguish the held-out codec evaluation from released-model comparisons with differing training histories and protocols.

REFERENCES

AlayaWorld Team, Kaipeng Zhang, Chuanhao Li, Yifan Zhan, Yongtao Ge, et al. Alayaworld: Interactive long-horizon world modeling – full technical report. *arXiv preprint arXiv:2607.18367*, 2026.

Roman Bachmann, Jesse Allardice, David Mizrahi, Enrico Fini, Oğuzhan Fatih Kar, Elmira Amirloo, Alaaeldin El-Nouby, Amir Zamir, and Afshin Dehghan. FlexTok: Resampling images into 1d token sequences of flexible length. In *International Conference on Machine Learning (ICML)*, 2025.

Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Aydar Bulatov, Yuri Kuratov, and Mikhail S. Burtsev. Recurrent memory transformer. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Shengqu Cai, Ceyuan Yang, Lvmin Zhang, Yuwei Guo, et al. Mixture of contexts for long video generation. *arXiv preprint arXiv:2508.21058*, 2025.

Zhifei Chen, Luozhou Wang, Guibao Shen, Dongyu Yan, et al. ReWorld: An interactive world model with long-horizon memory. *arXiv preprint arXiv:2608.23565*, 2026.

Yu-Cheng Chou, Xingrui Wang, Yitong Li, Jiahao Wang, et al. Captain safari: A world engine with pose-aligned 3d memory. *arXiv preprint arXiv:2511.22815*, 2025.

Jiaxing Cui, Jie Wu, Ming Li, Tao Yang, Xiaojie Li, Rui Wang, Andrew Bai, Yuanhao Ban, and Cho-Jui Hsieh. Self-Forcing++: Towards minute-scale high-quality video generation. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/8a30aba6514b56d02976f49797f6338a-Abstract-Conference.html.

Guangzhao He, Hadar Averbuch-Elor, and Wei-Chiu Ma. Can 4D foundation models remember?, 2026. URL https://arxiv.org/abs/2609.20819.

Yicong Hong, Yiqun Mei, Chongjian Ge, Yiran Xu, et al. RELIC: Interactive video world model with long-horizon memory. *arXiv preprint arXiv:2512.04040*, 2025.

Anthony Hu, Václav Volhejn, Adrien Ramanana Rahary, Chris Mulder, et al. Multiplayer interactive world models with representation autoencoders. *arXiv preprint arXiv:2607.05352*, 2026a.

Xiang Hu, Xinyu Wei, Hao Gu, Minshen Zhang, et al. Hierarchical sparse attention done right: Toward infinite context modeling. *arXiv preprint arXiv:2607.02980*, 2026b.

Jiahui Huang, Qunjie Zhou, Hesam Rabeti, Aleksandr Korovko, Huan Ling, Xuanchi Ren, Tianchang Shen, Jun Gao, Dmitry Slepichev, Chen-Hsuan Lin, Jiawei Ren, Kevin Xie, Joydeep Biswas, Laura Leal-Taixe, and Sanja Fidler. ViPE: Video pose engine for 3d geometric perception. *arXiv preprint arXiv:2508.10934*, 2025.

Haian Jin, Hanwen Jiang, Hao Tan, Kai Zhang, et al. LVSM: A large view synthesis model with minimal 3d inductive bias. In *International Conference on Learning Representations (ICLR)*, 2025.

Ruilong Li, Brent Yi, Junchen Liu, Hang Gao, Yi Ma, and Angjoo Kanazawa. Cameras as relative positional encoding. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025a.

Runjia Li, Philip Torr, Andrea Vedaldi, and Tomas Jakab. VMem: Consistent interactive video scene generation with surfel-indexed view memory. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025b.

Yan Li, Changyao Tian, Renqiu Xia, Ning Liao, Weiwei Guo, Hongsheng Li, Jifeng Dai, Hao Li, and Xue Yang. AdapTok: Learning adaptive and temporally causal video tokenization in a 1d latent space. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Zhen Li, Chuanhao Li, Xiaofeng Mao, Shaoheng Lin, et al. Sekai: A video dataset towards world exploration. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025c.

Yiyi Liao, Jun Xie, and Andreas Geiger. KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, et al. DL3DV-10K: A large-scale scene dataset for deep learning-based 3d vision. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Amirkeivan Mohtashami and Martin Jaggi. Random-access infinite context length for transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Yuta Oshima, Yusuke Iwasawa, Masahiro Suzuki, Yutaka Matsuo, and Hiroki Furuta. WorldPack: Dynamic frame compression for long-context video world modeling. *Transactions on Machine Learning Research*, 2026.

Jack Parker-Holder and Shlomi Fruchter. Genie 3: A new frontier for world models. Google DeepMind, `https://deepmind.google/blog/genie-3-a-new-frontier-for-world-models/`, 2025. Blog announcement, August 5; accessed 2026-09-24.

Zhan Peng, Jie Ma, Huiqiang Sun, Chong Gao, et al. Compression and retrieval: Implicit memory retrieval for video world models. *arXiv preprint arXiv:2606.23105*, 2026.

Runjia Qian, Zile Wang, Jihai Zhang, Kai Zou, Wei Yu, Jiaxing Li, Zexiang Liu, Yaokun Li, Fei Kang, Kaichen Huang, Mengyin An, Haobo Zhang, Biao Jiang, Jiahua Wang, Haofeng Sun, Yang Liu, and Yangguang Li. Matrix-game 3.5: Enhancing real-time streaming interactive world models with patch memory. Tech report, `https://matrix-game-v3-5.github.io/`, 2026. arXiv:2608.29910; accessed 2026-09-19.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations (ICLR)*, 2020.

Xuanchi Ren, Tianchang Shen, Jiahui Huang, Huan Ling, et al. GEN3C: 3d-informed world-consistent video generation with precise camera control. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Oren Rippel, Michael A. Gelbart, and Ryan P. Adams. Learning ordered representations with nested dropout. In *International Conference on Machine Learning (ICML)*, 2014.

Mehdi S. M. Sajjadi, Henning Meyer, Etienne Pot, Urs Bergmann, et al. Scene representation transformer: Geometry-free novel view synthesis through set-latent scene representations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

Wenqiang Sun, Haiyu Zhang, Haoyuan Wang, Junta Wu, et al. Worldplay: Towards long-term geometric consistency for real-time interactive world modeling. *arXiv preprint arXiv:2512.14614*, 2025.

Stanislaw Szymanowicz, Minghao Chen, Jianyuan Wang, Christian Rupprecht, and Andrea Vedaldi. LagerNVS: Latent geometry for fully neural real-time novel view synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Yifu Tao, Miguel Ángel Muñoz-Bañón, Lintong Zhang, Jiahao Wang, Lanke Frank Tarimo Fu, and Maurice Fallon. The Oxford Spires dataset: Benchmarking large-scale LiDAR-visual localisation, reconstruction and radiance field methods. *International Journal of Robotics Research*, 2025. doi: 10.1177/02783649251369905.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Wan Team. Wan2.2: Model and code release. `https://github.com/Wan-Video/Wan2.2`, 2025. Includes the TI2V-5B checkpoint, released July 28; accessed 2026-09-24.

Wan Team, Ang Wang, Baole Ai, Bin Wen, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Jiahao Wang, Yufeng Yuan, Rujie Zheng, Youtian Lin, et al. SpatialVID: A large-scale video dataset with spatial annotations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026a.

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

Jianyuan Wang, Minghao Chen, Shangzhan Zhang, Nikita Karaev, Johannes Schönberger, Patrick Labatut, Piotr Bojanowski, David Novotny, Andrea Vedaldi, and Christian Rupprecht. VGGT-Ω. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21486–21499, 2026b.

Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. MoGe-2: Accurate monocular geometry with metric scale and sharp details. *arXiv preprint arXiv:2507.02546*, 2025b.

Yifan Wang et al. Pi3X model and code release. `https://github.com/yyfz/Pi3`, 2025c. Engineering update with pose, intrinsics, and depth conditioning, December 28; accessed 2026-09-24.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4), 2004.

Zile Wang, Zexiang Liu, Jiaxing Li, Kaichen Huang, et al. Matrix-game 3.0: Real-time and streaming interactive world model with long-horizon memory. *arXiv preprint arXiv:2604.08995*, 2026c.

Zhengxuan Wei, Xu Guo, Xinghui Li, Xunzhi Xiang, et al. Geometry-aware implicit memory for video world models. *arXiv preprint arXiv:2606.02436*, 2026.

Haoyu Wu, Jiwen Yu, Yingtian Zou, and Xihui Liu. Multiworld: Scalable multi-agent multi-view video world models. *arXiv preprint arXiv:2604.18564*, 2026a.

Xindi Wu, Sven Elflein, James Lucas, Olga Russakovsky, et al. Addressable memory for video world models. *arXiv preprint arXiv:2608.07408*, 2026b.

Zeqi Xiao, Yushi Lan, Yifan Zhou, Wenqi Ouyang, et al. WorldMem: Long-term consistent world simulation with memory. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Jiacong Xu, Hanwen Jiang, Zhixin Shu, Kalyan Sunkavalli, Vishal M. Patel, and Yiqun Mei. Wonder: Video world model done better. *arXiv preprint arXiv:2607.26037*, 2026.

Bowen Xue, Brandon Y. Feng, Chenguo Lin, Yuchen Lin, et al. Ring forcing: Towards precise long-term memory for autoregressive video diffusion. *arXiv preprint arXiv:2608.26794*, 2026.

Wilson Yan, Volodymyr Mnih, Aleksandra Faust, Matei Zaharia, Pieter Abbeel, and Hao Liu. ElasticTok: Adaptive tokenization for image and video. In *International Conference on Learning Representations (ICLR)*, 2025.

Yixuan Ye, Xuanyu Lu, Yuxin Jiang, Yuchao Gu, et al. MIND: Benchmarking memory consistency and action control in world models. *arXiv preprint arXiv:2602.08025*, 2026.

Jung Yi, Minjae Kim, Paul Hyunbin Cho, Wooseok Jang, et al. WorldKV: Efficient world memory with world retrieval and compression. *arXiv preprint arXiv:2605.22718*, 2026.

Jiwen Yu, Jianxiong Gao, Jianhong Bai, Yiran Qin, et al. MemLearner: Learning to query context memory for video world models. In *European Conference on Computer Vision (ECCV)*, 2026a.

Wei Yu, Runjia Qian, Yumeng Li, Liquan Wang, et al. Mosaicmem: Hybrid spatial memory for controllable video world models. *arXiv preprint arXiv:2603.17117*, 2026b.

Cheng Zhang et al. Unified camera positional encoding for controlled video generation. *arXiv preprint arXiv:2512.07237*, 2025.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Shengjun Zhang, Zhang Zhang, Simin Huang, Zhenyu Tang, et al. MBench: A comprehensive benchmark on memory capability for video world models. *arXiv preprint arXiv:2606.00793*, 2026.

A BENCHMARK CONSTRUCTION

Relation to prior benchmarks. Prior benchmarks test object persistence (He et al., 2026), loop revisits (Wu et al., 2026b), and entity, environment, and causal consistency (Zhang et al., 2026). Our evaluation uses MIND (Ye et al., 2026) and revisit protocols built from KITTI-360, Oxford Spires, SpatialVID, Sekai, and DL3DV (Liao et al., 2022; Tao et al., 2025; Wang et al., 2026a; Li et al., 2025c; Ling et al., 2024), including archives up to nearly two hours; the rest of this appendix describes their construction.

A.1 GEOMETRY ANNOTATION AND REVISIT MINING

We re-annotated the video data used for revisit mining with VGGT-Ω (Wang et al., 2026b) to estimate camera geometry and MoGe-2 (Wang et al., 2025b) to recover metric scale. These annotations provide the camera geometry used by the model; poses and scale remain estimates.

For mining, we used the original ViPE (Huang et al., 2025) annotations as a pre-filter to identify candidate revisit types from camera trajectories. These include U-turns, retraced paths, loop returns, and look-backs (Figure A.1). The original annotations serve to find candidates; the re-annotated geometry supplies their camera conditioning.

Each example pairs earlier observations with later, held-out views of a previously visited scene. Memory and targets come from the same source video, including for synthetic examples. Observation counts, temporal gaps, and target sampling are specified for each evaluation in Appendix B.

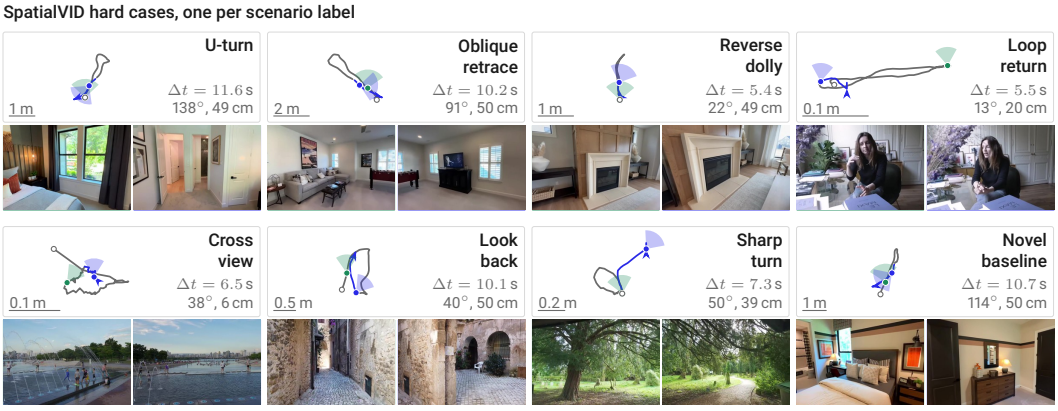

Figure A.1: SpatialVID hard-case mining examples, one per scenario label. Each panel shows an estimated top-down trajectory above a stored frame (left) and a return frame (right). Gray and blue paths denote the stored pass and return; green and blue wedges mark the displayed views' fields of view. Annotations give their time separation Δt, heading difference, and camera distance. Poses and scale are estimated.

B Experimental Protocol and Comparison Scope

B.1 Codec comparison

The codec test set consists of 64 deterministically selected, pose-mined scenes from the DL3DV validation split, none of which are used in training. Each scene provides 128 unique first-loop observations and 32 targets sampled from 96 second-loop frames, excluding targets within 32 frames of memory. Images are resized to 512×768 with bilinear antialiasing. PSNR, SSIM and LPIPS-Alex are measured over the full frame, without visibility masks. Scene-bootstrap intervals use 10,000 resamples. Dense and sparse variants are trained separately; inference controls use the same final RGB codec and fully retained archive.

All three encoders process eight-frame chunks. The 1:1 encoder stores one 1,536-token feature grid per frame. The fixed bottleneck stores 2×512 tokens per chunk, and the nested bottleneck supports 0–4 entries of 512 tokens each. With all entries retained, a 128-frame archive contains 196,608, 16,384 and 32,768 tokens, respectively. A 6,144-token read opens 4, 6 and 12 selection units; the fixed bottleneck groups two entries per unit. Compression ratios count payload tokens and exclude landmarks, address buffers and other runtime state. This comparison retains all candidate entries and does not evaluate adaptive storage allocation.

Random, uniform-stride and nearest-pose controls use ordinary attention over the selected entries. The 12-layer dense reader with learned selection separates the effect of selection from the SLA mixture. The dense-attention oracle ranks entries by attention mass at layers 0, 4, 8 and 11, using at most 64 query positions. It requires a dense pass. Nearest-pose selection combines translation normalized by archive spread with forward-axis disagreement.

The separate LagerNVS baseline uses its released general checkpoint at native resolution, below our 512×768, and its own camera normalization. Its selective variant chooses sources by camera position. Our past-only revisit split differs from the published LagerNVS benchmarks (Szymanowicz et al., 2026), so scores across those benchmarks are not directly comparable.

B.2 Learning curves and long archives

The learning curves evaluate 73 checkpoints from an earlier training run, before the final decoder-weighting schedule. Dense and sparse reads use the same checkpoint at each point. These curves therefore describe retrieval learning under that earlier schedule.

The KITTI read-budget study uses whole-drive archives. The archive-growth study instead starts with chunks relevant to fixed queries and appends unrelated drives, so the two studies have different dense references. Archive growth retains four entries per chunk and reads 12 entries per query. The fused-archive evaluation uses six query drives with 64 strict revisits each. Its nearly two-hour duration sums nine independent drives. Payload curves count BF16 entries and exclude index and metadata storage.

B.3 Timing protocol

Resident-query timing includes landmark scoring, top-K selection, payload gathering and RGB decoding on one NVIDIA B200 at batch one, using the final RGB codec, cached landmarks and geometry, and optimized kernels. At 512×768, each point uses 24 warm-up and 40 timed queries, giving about 44 ms per query. Encoding one eight-frame chunk into entries and landmarks with the same optimized kernels takes about 87 ms. Upload, compilation, warm-up, disk I/O and video diffusion are excluded; these measurements do not establish end-to-end video throughput.

B.4 KV-cache storage baseline

Figure 1 estimates the storage required to retain self-attention keys and values for the full history in Wan2.2-TI2V-5B (Wan Team, 2025). The model has $L = 30$ layers and attention width $d = 3{,}072$. At 512×768 resolution, the VAE's spatial stride of 16 and 2×2 DiT patches yield $n = (512/32)(768/32) = 384$ tokens per independently encoded frame. At 10 frames/s, one hour contains $F = 36{,}000$ frames. Storing both keys and values in BF16 ($b = 2$ bytes per element), for

one sequence and one cache per layer, requires

$$S_{\mathrm{KV}} = 2LdnFb = 5.096 \times 10^{12} \text{ bytes} \approx 5.1 \text{ TB } (4.63 \text{ TiB}). \tag{B.1}$$

This is a payload calculation for framewise history encoding; it excludes model weights, text cross-attention caches, and temporary activations. Using the native video VAE's fourfold temporal compression instead gives approximately 1.27 TB/hour. The figure uses the framewise estimate; its learned-memory storage value comes from the measured KITTI archive.

B.5 Video-generation comparison

Table 2 uses 32 fixed clips: five each from KITTI-360, Oxford Spires, Sekai walking and DL3DV; ten from SpatialVID (five standard and five longer revisits); and two from MIND. Each supplies 128 earlier observations and a clean query RGB anchor. All methods use the same future targets and random seed. We score 128 future frames per clip, except for one DL3DV clip with 120 valid targets. We average frames within clips, clips within datasets, and the six dataset means equally. All 32 clips are retained. These descriptive means are not full-dataset estimates. Released models retain their native resolution, observation consumption and sampling schedule; outputs are mapped to a shared scoring crop per clip. Training data and compute are not matched.

Our bidirectional model uses 50 denoising steps and CFG 1, jointly producing the supplied anchor and 128 future frames. Our autoregressive model uses four-step DMD sampling, CFG 1 and an eight-latent context. Both generate at 768×512 and rebase memory and query cameras to the query anchor. The memory-only row uses the separately trained RGB codec with $K = 16$. Gen3C, Matrix's released bidirectional base and GIM-World jointly denoise windows and may extend them autoregressively; the other group uses causal or streaming autoregressive generators. Bold and underlined values rank methods within each group. RGB reconstruction is unranked.

SANA-WM uses its streaming release with a query-anchor sink and supplied history. AlayaWorld uses its released autoregressive teacher, initialized from the anchor, with the corresponding low-resolution projection weights. GIM uses the original MIND camera/action metadata. MIND intrinsics estimated from observed RGB with VGGT-Ω are mapped back from the estimator grid before adapting to each model's resolution; source camera translations retain the centimeter-to-meter assumption. No future target RGB is used for conditioning.

B.6 Rotation-error measurement

Rotation error uses VGGT-Ω on four fixed target blocks of up to 32 frames. Each estimation batch contains the query anchor, seven pose-selected earlier observations, and the union of 32 chronological generated views (24 in the shorter final block) and eight fixed target views. Only the eight fixed target views are scored; the added views provide temporal context from each method's own predictions. Estimated and commanded orientations are compared in the query-anchor coordinate frame, and the four block means are averaged. All methods use the same view selection and estimator. This diagnostic is sensitive to estimator context and shares a model family with the memory encoder; it is not independent geometric ground truth.

C Analysis of Sparse Landmark Attention

SLA adapts the hierarchical operator and task-loss gradient path of HiLS (Hu et al., 2026b) to a pose-conditioned entry weighting shared across a target view. We analyze allocation, learning, and the restriction imposed by sharing, at fixed attention inputs.

C.1 Entry mixture and budgeted allocation

We use the notation of Eq. D.7: for B eligible entries of P tokens, S is a nonempty selected support, $n = (\ell, h, i)$ indexes a layer, head, and query position, $\rho_{n,b,j}$ and $Z_{n,b}$ are the within-entry attention weights and partition sums, $u_{n,b}$ is the within-entry attention output, $\pi_b = \mathrm{softmax}_{b \in S}(gs_b)$ are the selected-entry mixture weights with learned scale g, and $o_n = o^{(\ell)}$ is the combined output. In the SLA layers ($\ell < 4$), Eq. D.7 gives joint attention $P_n(b, j) = \pi_b \rho_{n,b,j}$, with zero mass outside S.

Each row distributes the fixed entry mass π_b within entry b. In contrast, adding gs_b to token logits before a single softmax gives entry mass proportional to $e^{gs_b} Z_{n,b}$; it agrees with SLA only when $Z_{n,b}$ is constant across selected entries.

Proposition 1 (Top-K allocation)**.** *For finite $s \in \mathbb{R}^B$, $g > 0$, and $1 \le K \le B$, softmax over the K largest scores solves*

$$\max_{\pi \in \Delta^B,\, |\operatorname{supp} \pi| \le K} \big[g\pi^\top s + H(\pi) \big], \qquad H(\pi) = -\sum_b \pi_b \log \pi_b, \tag{C.1}$$

where Δ^B is the probability simplex and $0 \log 0 = 0$. The solution is unique if no score tie crosses the support boundary.

Proof. On any nonempty support A, let $Z_A = \sum_{b \in A} e^{gs_b}$ and $\sigma^A_b = e^{gs_b}/Z_A$. The objective is $\log Z_A - D_{\mathrm{KL}}(\pi \| \sigma^A)$, maximized by σ^A. Adding a entry increases Z_A; replacing a selected score by a larger one also increases it because $g > 0$. Thus a top-K support is optimal, with boundary ties the only ambiguity. □

C.2 Learning the scores

With detached support S, differentiating the mixture over rows $\mathcal{N}$ of layers 0–3 gives

$$\frac{\partial \mathcal{L}}{\partial s_b} = g\pi_b \sum_{n \in \mathcal{N}} \left\langle \frac{\partial \mathcal{L}}{\partial o_n}, u_{n,b} - o_n \right\rangle, \qquad b \in S. \tag{C.2}$$

This follows from the softmax Jacobian and the chain rule; the adjoints include later layers and temporary memory updates. A negative derivative raises a score when shifting mass toward that entry locally reduces loss. Unselected scores receive no direct mixture gradient, but archive-wide normalization of content compatibility can train unselected landmarks; exploration exposes their payloads. At $|S| = 1$ the mixture gradient vanishes. Parameters shared with payload construction also receive gradients through paths outside Eq. C.2.

C.3 Omitted entries and shared weighting

At the same inputs, dense attention has entry marginal $\alpha_{n,b} = Z_{n,b} / \sum_{b'} Z_{n,b'}$. Hard omission makes its KL to SLA infinite, so define omitted mass $1 - M_n(S)$ separately, where $M_n(S) = \sum_{b \in S} \alpha_{n,b}$. On S, write $\alpha^S_{n,b} = \alpha_{n,b}/M_n(S)$ and $P_n^{\mathrm{dense},S}(b,j) = \alpha^S_{n,b}\rho_{n,b,j}$. The common within-entry conditionals cancel: $D_{\mathrm{KL}}(P_n^{\mathrm{dense},S} \| P_n) = D_{\mathrm{KL}}(\alpha^S_n \| \pi)$.

Proposition 2 (Best shared weighting)**.** *For $w_n \ge 0$, $\sum_n w_n = 1$, define $\bar\alpha^S = \sum_n w_n \alpha^S_n$. Every distribution π positive on S satisfies*

$$\sum_n w_n D_{\mathrm{KL}}(\alpha^S_n \| \pi) = \sum_n w_n D_{\mathrm{KL}}(\alpha^S_n \| \bar\alpha^S) + D_{\mathrm{KL}}(\bar\alpha^S \| \pi). \tag{C.3}$$

The minimizer is $\bar\alpha^S$; the minimum is $I(N;J)$ under $P(N=n, J=b) = w_n \alpha^S_{n,b}$.

Proof. Split $\log(\alpha^S_{n,b}/\pi_b)$ into $\log(\alpha^S_{n,b}/\bar\alpha^S_b) + \log(\bar\alpha^S_b/\pi_b)$ and sum. The second term is nonnegative and vanishes at $\pi = \bar\alpha^S$; the first is the stated mutual information. □

Scope. These identities establish neither task-optimal scores nor a reconstruction error bound between independently evolving decoders. Omitted mass and sharing KL are attention diagnostics, and reconstruction need not learn the barycenter. The 4+8 schedule permits row-specific weights after layer 3 but cannot recover omitted entries. Proposition 1 requires $g > 0$ and a top-K support; exploration preserves only the fixed-support optimum.

D Method Formulas

This appendix collects the formulas referenced in Section 3, in the order in which the Method uses them. Implementation settings are in Appendix E.

Writer attention mask (Section 3.1). Frame tokens attend within their chunk; the current registers additionally read the cached key–value states of the retained entries of the preceding chunks. For surviving prefix length $\kappa(a)$ at chunk age a, the entry-level mask is

$$\mathcal{A}\big((c,p) \leftarrow (c',p')\big) = \mathbf{1}\big[c' = c \ \vee \ \big(0 < c - c' < 16 \ \wedge \ p' < \kappa(c-c')\big)\big]\,. \tag{D.1}$$

Landmark summaries and entropies (Section 3.1). Let $P = hw$ and $R_{c,p} = \{r_{c,p,j}\}_{j=1}^{P}$ denote the entry's tokens. Each head h uses a learned query $q_{p,h}$, shared across chunks at entry position p, and projected keys $k_{c,p,j,h} = W_h^K r_{c,p,j}$ with learned $W_h^K \in \mathbb{R}^{d\times D}$; bars denote PRoPE transforms using the entry's pose. The pooling weights and summary are

$$a_{c,p,j,h} = \mathrm{softmax}_j\Big(\bar{q}_{c,p,h}^{\top}\bar{k}_{c,p,j,h}/\sqrt{d}\Big)\,, \qquad m_{c,p,h} = \sum_{j=1}^{P} a_{c,p,j,h}\bar{k}_{c,p,j,h}, \tag{D.2}$$

and the head entropy, stored without a positional transform, is

$$e_{c,p,h} = -\sum_{j=1}^{P} a_{c,p,j,h}\log a_{c,p,j,h}. \tag{D.3}$$

Landmark scoring (Section 3.2). Let $\mathcal{E}$ be the surviving entries. For an entry $b = (c,p)$, write $m_{b,u} = m_{c,p,u}$, $e_{b,u} = e_{c,p,u}$, and $\mathrm{slot}(b) = p$; the landmark heads u number $H_{\mathrm{L}} = H$ with width $d_{\mathrm{L}} = d$. The target view is represented by A camera rays with directions $d_{t,a}$. For ray a and head u, the pose-aware query is

$$\bar{q}_{t,a,u} = \mathcal{T}_{t,a}\big([\zeta + W_f d_{t,a}]_u\big), \tag{D.4}$$

where $\mathcal{T}_{t,a}$ applies PRoPE with the target pose, ζ is learned, and W_f is zero-initialized. The landmark-attention weights and content compatibility are

$$\omega_{t,b}^{a,u} = \mathrm{softmax}_{b\in\mathcal{E}}\left(\frac{\bar{q}_{t,a,u}^{\top}m_{b,u}}{\sqrt{d_{\mathrm{L}}}} + e_{b,u}\right), \qquad C_{t,b} = \log\left(\frac{1}{AH_{\mathrm{L}}}\sum_{a,u}\omega_{t,b}^{a,u}\right). \tag{D.5}$$

For a fixed target, write $C_b = C_{t,b}$. The relevance score combines C_b with explicit geometric and temporal features,

$$s_b = \gamma_g\hat{d}_b + \beta_{\mathrm{slot}(b)} + f_\theta(\mathbf{f}_b) + \gamma_c C_b, \qquad \mathbf{f}_b = \left[\widetilde{d}_b, \log\widetilde{d}_b, \cos\phi_b, \phi_b, \frac{U_t^{\top}(c_b - c_t)}{\|c_b - c_t\|_2}, \widetilde{a}_b\right], \tag{D.6}$$

where $\hat{d}_b$ is the standardized negative camera distance, $\widetilde{d}_b$ the distance relative to the median entry distance, ϕ_b the forward-axis disagreement, $\widetilde{a}_b$ the chunk index linearly normalized to $[0,1]$ over the archive (zero for a single chunk), U_t the target camera-to-world rotation, and c_t, c_b the camera centers; f_θ is the learned scoring network and γ_g, γ_c and the per-slot bias β its scalar coefficients.

Selection and decoding (Section 3.2). The reader selects $S = \mathrm{TopK'}(s)$ with $K' = \min(K, |\mathcal{E}|)$, once per target, with detached indices and live scores, and forms the selected-entry mixture weights $\pi_b = \mathrm{softmax}_{b\in S}(g\, s_b)$ with a learned scale g initialized to 0.1; subtracting the selected-score mean before scaling leaves the softmax unchanged. In the context decoder, let $n = (\ell, h, i)$ index a layer, head, and target-token position. For decoder head width d_{D}, let $z_{n,b,j} = q_n^{\top}k_{n,b,j}/\sqrt{d_{\mathrm{D}}}$ be the token logits (including positional transforms), $Z_{n,b} = \sum_j e^{z_{n,b,j}}$, $\rho_{n,b,j} = e^{z_{n,b,j}}/Z_{n,b}$ the within-entry attention weights, and $u_{n,b} = \sum_j \rho_{n,b,j}v_{n,b,j}$. Writing $u_b^{(\ell)} = u_{n,b}$ and $Z_b^{(\ell)} = Z_{n,b}$ for a fixed head and target token, the decoder combines the selected entries as

$$o^{(\ell)} = \sum_{b\in S} w_b^{(\ell)}u_b^{(\ell)}, \qquad w_b^{(\ell)} = \begin{cases} \pi_b, & 0 \le \ell < 4, \\ \dfrac{Z_b^{(\ell)}}{\sum_{c\in S} Z_c^{(\ell)}}, & 4 \le \ell < 12, \end{cases} \tag{D.7}$$

with $o_n = o^{(\ell)}$. The first four layers share π_b across layers, heads, and target positions; the remaining eight use ordinary attention over the same entries.

Codec reconstruction loss (Section 3.3). The RGB codec trains its writer, reader, and image head with unit-weighted image MSE and frozen VGG19 perceptual losses on the held-out target I_t:

$$\mathcal{L}_{\text{codec}} = \|\widehat{I}_t - I_t\|_2^2 + \mathcal{L}_{\text{VGG19}}(\widehat{I}_t, I_t). \tag{D.8}$$

VAE-latent refinement loss (Section 3.3). With the latent head predicting $\widehat{z}_t^{\text{mem}}$ in the frozen VAE space, the refinement objective is

$$\mathcal{L}_{\text{latent}} = \mathcal{L}_1^{\text{latent}} + 0.25\,\mathcal{L}_{\text{Charbonnier}}^{\text{image}} + 0.50\,\mathcal{L}_{\text{VGG19}}^{\text{image}} + 0.05\,\mathcal{L}_{\text{gradient}}^{\text{image}}, \tag{D.9}$$

where the image terms are computed on targets decoded through the frozen VAE decoder; Appendix E defines each term.

Flow-matching loss (Section 3.3). For $z_0 \sim \mathcal{N}(0, I)$, target latent z_1, and $z_s = (1-s)z_0 + sz_1$, the joint codec–DiT stage minimizes

$$\mathcal{L}_{\text{flow}} = \mathbb{E}_{s,z_0,z_1}\,\|v_\psi(z_s, s \mid y_t, a_t, \widehat{z}_t^{\text{mem}}) - (z_1 - z_0)\|_2^2\,, \tag{D.10}$$

conditioned on recent context y_t, control a_t, and the predicted memory latents $\widehat{z}_t^{\text{mem}}$.

Adaptive retention rule (Sections 3.1 and 3.3). The allocation head predicts future clip-mean PSNR for each candidate prefix, conditioned on a requested PSNR. Equation F.1 gives the selection rule, and Appendix F.1 describes its causal rollout training.

E Implementation and Training Details

Table E.2 summarizes the training stages and branches. Allocation-head training is separate from latent refinement; pose-conditioned Wan is trained independently before memory integration. Bidirectional memory conditioning and autoregressive extension are separate generator branches. Appendix B specifies the checkpoint-dependent comparison protocol.

Encoder configuration and causal state. Table E.1 collects the documented configuration. Frame tokens attend within their chunk; current registers read historical K/V without updating those states, under the mask of Eq. D.1. Writer K/V and canonical retrieval entries are distinct states. Retention must restrict each subsequent write; pruning a dense-history encoding afterward does not reproduce its causal dependencies.

Component	Configuration
Input and entries	512×768 images; 8 frames/chunk; 4 entries of 16×32 tokens, width 768
Camera injection	Zero-indexed anchors $(3, 7, 1, 5)$; encoder layers $(1, 5, 11, 17, 23)$
Writer attention	Current chunk + 15 preceding chunks; surviving prefixes
Positional features	Head width 64: temporal/projective/horizontal/vertical $= 16/24/12/12$; rotary base 10,000
Landmarks and queries	$768 \to 64$ projection; $H_{\text{L}} = 4$ heads of width $d_{\text{L}} = 16$; $A = 5$ rays (center and corners)
Read budgets	Training: $K = 16/8/8$ for 128/64/32 frames; codec comparisons: $K = 12$ (6,144 tokens); video comparisons: $K = 16$ (8,192 tokens)
Reader and exploration	Layers 0–3: SLA; 4–11: ordinary attention; one-entry exploration swap with probability 0.1
Precision	Payloads/features: BF16; camera transforms, relative geometry, and landmark scores: FP32 until the feature boundary

Table E.1: Documented settings; read-budget sweeps are reported separately.

Context-decoder target-token initialization. The decoder starts each target patch from a shared learned query token, initialized from $\mathcal{N}(0, 0.02^2)$. A learned linear projection of the patch's camera-frame unit ray direction is added before target-token normalization; its weights and bias are initialized to zero. Target pose and patch geometry enter attention through PRoPE, and observation time through temporal RoPE.

Landmark construction and selection. The learned, slot-specific pooling probe constructs each landmark using Eq. D.2. The barred query and keys use their respective PRoPE transforms from the entry's camera geometry. Keys use inverse world-to-ray transforms; the query uses the transposed transform of a fixed reference ray at zero-based flattened index $\lfloor P/2 \rfloor$ in the anchor's grid. The entropy of Eq. D.3 has no positional transform. With $H = 4$ heads of width $d = 16$, each landmark contains 68 summary and entropy values, in addition to camera and time metadata.

Landmark construction is independent of the target query. The native reader rebuilds landmarks from the source archive on each forward pass; the optimized resident-archive benchmark caches summaries and entropies for an unchanged source archive, camera geometry, and indexing parameters. Selection precedes gathering payloads into the decoder; the benchmark's full source archive is already resident. Timestamps accompany entries for temporal RoPE in the geometry encoder and context decoder, but do not directly enter landmark pooling or content scoring.

Scores follow Eqs. D.4–D.6. Entropy is an untransformed landmark feature, distinct from $H(\pi)$, and feature normalization does not rescale the metric translations used by the encoder or decoder. Selection and the decoder mixture follow Eq. D.7. Any surviving entry is selectable independently of its siblings. During exploration, one selected entry is replaced by a sampled surviving unselected entry, if available. Payloads are gathered in stable chronological/entry order. Entry-batched and exact FA4 split-KV attention implement the same operator; 512-token entries align with 128-token blocks. Layers 4–11 and the reverse memory-to-target direction use ordinary attention. Updates affect only the target's temporary memory copy.

Generator interface and reuse. The latent head predicts $\hat{z}_t^{\mathrm{mem}}$ in the fixed VAE space for each target latent timestep, using its pose and time. The DiT's shared patch embedding converts these predictions to tokens, concatenated with video tokens along the sequence dimension. Memory tokens share the corresponding video's timestep embedding and temporal RoPE position, with spatial offset $(\Delta\tau, \Delta h, \Delta w) = (0, 64, 64)$. The bidirectional memory-conditioned stage uses a noncausal DiT; autoregressive extensions use causal attention, with the mask determined by the generator stage. In the memory-conditioned Wan stage (Table E.2), the flow loss of Eq. D.10 jointly trains the DiT and memory codec, including the writer, scorer, reader, and latent head, through the predicted context. The preceding RGB and VAE-latent stages initialize the codec; the VAE remains fixed.

Landmarks can be reused while entries and their producing parameters remain unchanged. Deterministic context outputs can be reused across denoising steps only with unchanged archive, pose/time, support, parameters, and other decoder inputs; temporary later-layer K/V remain target-dependent. These are inference reuse conditions, not measured speedups; rebuilding the index per query adds an archive-wide payload pass.

Pose-conditioned Wan. All generator stages use Wan2.2-TI2V-5B (Wan Team, 2025) with its VAE and text encoder frozen. Camera poses enter the DiT through Plücker-ray features combined with UCPE relative-ray encoding (Zhang et al., 2025). Before memory integration, the DiT is trained for camera control without external memory; the memory-conditioned stage continues from this model and trains the codec and DiT jointly.

Autoregressive training. We train the autoregressive generator using a method in the style of Self-Forcing++ (Cui et al., 2026).

Stage	Context → targets	Training description
Dense memory warmup	$32 \to 8$	Warm up memory writer and context decoder for scene reconstruction from memory with dense reads and RGB/perceptual losses.
Dense long-context training	$128 \to 32$	Train long-context scene reconstruction with dense memory reads.
Sparse RGB refinement	$32/64/128 \to 32$	Train long-context reconstruction with SLA, nested dropout and diverse revisits; four SLA layers, then eight ordinary-attention layers.
Allocation-head training	$32/64/128$ context	Predict future reconstruction quality and remaining storage for each prefix, conditioned on the requested PSNR.
VAE-latent refinement	$32/64/128 \to 32$	Predict Wan latents with latent/image losses and SLA/nested dropout; keep the target VAE frozen.
Pose-conditioned Wan	Variable length	Train camera-pose control on video clips without external memory.
Memory-conditioned Wan	$128 \to 65$	Jointly train the codec and DiT with flow matching, using retrieved latent tokens as context.
Autoregressive extension	$128 \to 128$	Generate causally, using stored observations as context throughout the continuation.

Table E.2: Training stages. Context and target counts are RGB frames at 512×768 (H×W). Calibration and generator branches are described above; rows do not imply a single linear initialization chain.

RGB codec objective and sampling. The RGB-output codec jointly trains its writer and reader with Eq. D.8, using unit weights for image MSE and frozen VGG19 perceptual loss. The mainline data mixture is listed in Table E.3. Real sources include SpatialVID, Sekai, and DL3DV (Wang et al., 2026a; Li et al., 2025c; Ling et al., 2024); synthetic data from a game engine provide revisit pairs, using one source video per example. Ordinary continuation and controls remain in the mix.

Table E.3: Documented codec sampling mixture. Hard-real and same-video synthetic examples retain the full archive before sparse retrieval; other examples use the sampled retention curriculum.

Sample type	Conditioning frames	Weight
Regular real	128 / 64 / 32	5% / 15% / 15%
Distant real	128 / 64 / 32	2.5% / 3.75% / 3.75%
Hard real	128	20%
Real controls	128	5%
Game-engine synthetic	128	30%

For ordinary, distant, and control examples, retention is dense with probability 0.25 and otherwise uses nested tail dropping. The sparse physical budgets are 30/15/8 entries for 128/64/32 conditioning frames. The hard-real and same-video synthetic examples keep all 64 entries in their 128-frame archive. Retrieval then uses $K = 16/8/8$ according to context length, independently of physical retention. All conditioning chunks are encoded; retained prefixes constrain what later encoding and reading may access. True observation times are preserved under irregular frame sampling.

VAE-latent refinement before memory-conditioned generation. The Wan-latent continuation keeps the four-entry writer and 4+8 reader and changes the output head and losses. Starting from the RGB-pretrained codec, we fine-tune its writer, reader, and latent head together. Each target is independently encoded with the frozen Wan2.2 framewise VAE using deterministic targets. The context decoder grid at $H_{\mathrm{img}}/8 \times W_{\mathrm{img}}/8 \times 768$ is grouped into 2×2 cells and projected to a frame latent at $H_{\mathrm{img}}/16 \times W_{\mathrm{img}}/16 \times 48$. Here H_{img} and W_{img} denote image height and width. This is framewise encoding, not temporal VAE packing. A linear skip and a small local residual branch predict the latent. Its training objective is Eq. D.9. Latent L1 uses all 32 targets; image losses use two sampled targets decoded differentiably through the frozen VAE in FP32 and resized

to 256×384. The Charbonnier penalty is $\sqrt{x^2+\epsilon^2}-\epsilon$ with $\epsilon = 10^{-3}$. The gradient term applies this penalty to differences between predicted and target horizontal and vertical image gradients, computed by adjacent-pixel finite differences, and averages the two directions. These decoded-image losses supervise detail that latent L1 alone may not preserve. The VAE weights remain fixed while gradients pass through its decoder to the codec. After refinement, the codec and DiT are trained jointly, end to end, with the predicted latents providing context (Eq. D.10).

F Adaptive Writes: Requested-Quality Control

Adaptive writes select how many entries to retain per chunk to meet a requested clip-mean PSNR with the fewest retained entries.

F.1 Causal allocation policy

For each chunk c, the controller commits a prefix of $L_c \in \{0, \dots, 4\}$ candidate entries; zero skips the chunk, and the first chunk always commits four. Given candidate and committed-history summaries, camera motion, clip position, and a request X in dB, the head predicts, for each count p, future clip-mean PSNR $\widehat{Q}_{c,p}(X)$ under continued policy rollout. The allocation is

$$L_c^X = \begin{cases} \min \mathcal{F}_c, & \mathcal{F}_c = \{p \in \{0,\dots,4\} : \widehat{Q}_{c,p}(X) \geq X\} \neq \emptyset, \\ \arg\max_{p\in\{0,\dots,4\}} \widehat{Q}_{c,p}(X), & \text{otherwise.} \end{cases} \tag{F.1}$$

Each candidate branch regenerates its causal continuation using only its committed entries.

Training. Only the prediction head is trained; the codec and memory representation are frozen. Episodes sample the pretraining mixture (Appendix E) with an additional 20% ordinary SpatialVID clips, requests uniformly in $[18.5, 22]$ dB, and histories from the current policy (75%) or a sparse random schedule (25%). The teacher rolls out all five candidate counts to the clip's end, measuring PSNR on 8 queries from the 32-frame future horizon. The head regresses PSNR with Huber loss, giving the branch selected before labeling the same weight as the five-branch mean. An additional penalty $3\,\mathrm{Huber}_{1\,\mathrm{dB}}\big(\max(\widehat{Q} - Q, 0)\big)$ penalizes quality overprediction fourfold. We warm-start from a head trained for 500 updates and train for 200 additional synchronous updates of 32 episodes using AdamW (learning rate 10^{-4}, seed 5507, 32 GPUs).

F.2 Evaluation protocol

We evaluate on 32 development clips used for checkpoint monitoring and selection: 16 DL3DV and 16 SpatialVID clips chosen deterministically by scene hash. Each has 16 chunks of four candidate entries (64 entries at full storage). We rerun the causal rollout for each of 25 requests from 17 to 23 dB in 0.25 dB steps and measure mean PSNR over the 32 future frames after the final chunk. Attainment is the fraction of clips meeting the request; shortfall is the mean deficit over all clips. Entries per chunk include the bootstrap, and compression is four divided by this count.

Results. At an 18.5 dB request, the controller retains 0.63 entries per chunk on SpatialVID ($6.4\times$ compression; 14/16 clips attain the target), compared with 2.9 on DL3DV (Figure F.1). DL3DV approaches full storage at 20 dB and reaches it at 21 dB, with achieved quality saturating at 21.1 dB. Across both datasets and the eight half-dB requests within the training range, attainment is 73.8%, with a mean shortfall of 0.57 dB at 3.05 entries per chunk. Below 19 dB, DL3DV retains 3–5 times as many entries as SpatialVID for the same request (Figure F.2).

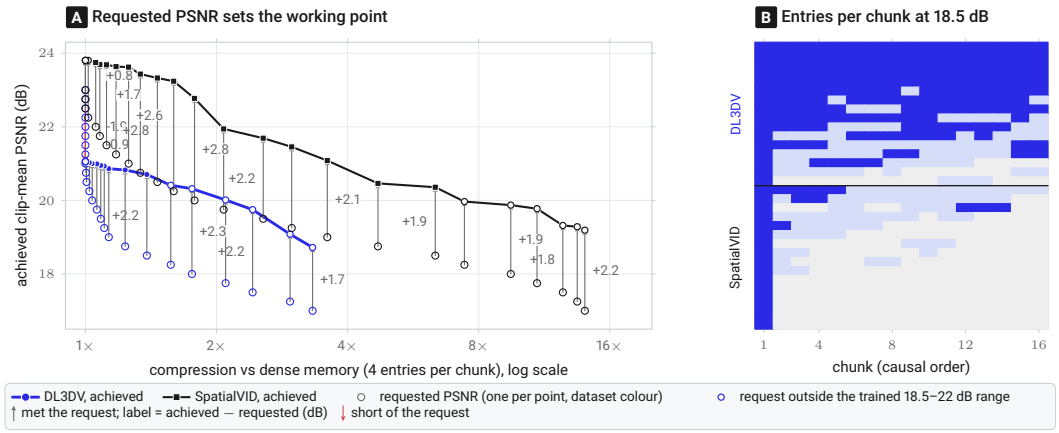

Figure F.1: Requested-quality control on the development panel: DL3DV (blue) and SpatialVID (black). **(A)** Requested PSNR (open circles) and achieved PSNR at the selected compression; arrows connect each pair, with gaps labeled at half-dB requests. Achieved-quality markers are hollow outside the trained range, 18.5–22 dB. **(B)** Entries committed per chunk at an 18.5 dB request, with clips sorted by average allocation within each dataset. Light gray cells indicate skipped chunks; the bootstrap retains four entries.

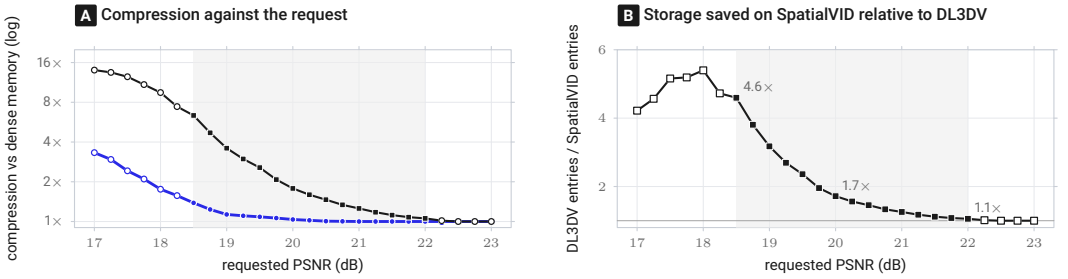

Figure F.2: **(A)** Compression versus requested PSNR for DL3DV (blue) and SpatialVID (black). Shading marks the training range; hollow markers indicate extrapolation. **(B)** Ratio of DL3DV to SpatialVID entries per chunk at the same request.

G Additional Retrieval Results

Figure G.1 complements Section 4.3. On whole-drive archives, reading $K = 8$ entries reaches 17.39 dB versus 14.58 dB for the dense read. Across six query drives and 384 strict revisits, adding the other eight drives keeps quality within 0.3 dB of relevant-only reads. All opened entries come from the query drive, and 92–100% come from relevant chunks. Figure G.2 shows retrieval across an 8.1-minute gap.

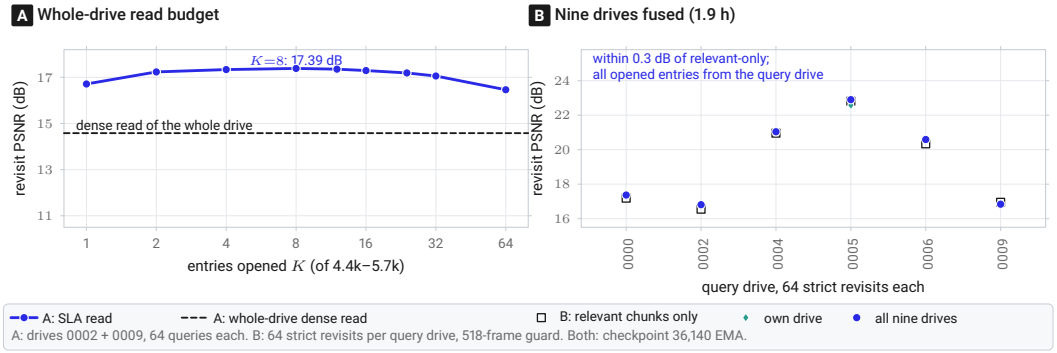

Figure G.1: Additional KITTI-360 retrieval results. **(A)** Whole-drive read-budget sweep on drives 0002 and 0009, 64 queries each; dashed line: the whole-drive dense read. **(B)** Relevant-only, own-drive, and nine-drive archives on six query drives, 64 strict revisits each, with a 518-frame guard. The fused archive spans 1.87–1.98 hours, summed over independent drives. Both evaluations use the final RGB codec.

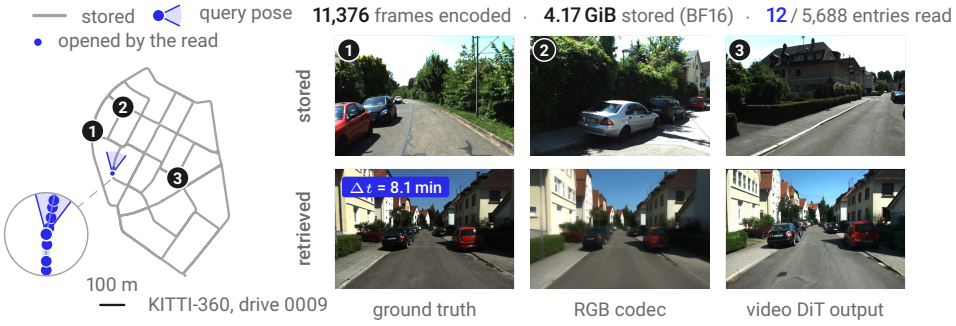

Figure G.2: Retrieval across an 8.1-minute revisit on KITTI-360 drive 0009, using 12 of 5,688 entries to recover the earlier street view. Left: the stored trajectory, the query pose and the entries opened by the read. Right: stored views, the ground truth, the RGB codec reconstruction and video DiT output.

H Full Qualitative Comparison

Figure H.1 complements Figure 6 with all nine methods on six examples from the evaluation cohort in Table 2. All six clips differ from those in Figure 6. The protocol qualifications in Appendix B also apply here.



Figure H.1: Full qualitative comparison from the final evaluated runs. Each column shares one target across all methods; method rows follow Table 2, with the closest stored view and ground truth for reference. Labels give the capture-time gap between the closest stored view and the target, not time since the last observation. All images use the shared evaluation crop for that clip, including corrected MIND calibration, with aspect ratios preserved. Memory-only uses the separately trained RGB codec.

I STATIC-OBJECT PERSISTENCE: SUCCESSES AND LIMITATIONS

Figure I.1 shows four static-target cases from our PersistBench (He et al., 2026) evaluation. The fountain and roadside sign remain recognizable after leaving the input view. The display illustrates content mismatch despite a stable physical surface; the statue illustrates positional drift despite retained identity. "Static" describes the target, not every object or pixel in the scene.

Both models generate at 768×512 using corrected metric camera estimates and source-only conditioning, with no reference RGB anchor. Their native window has 129 RGB frames; only real clip frames are evaluated.

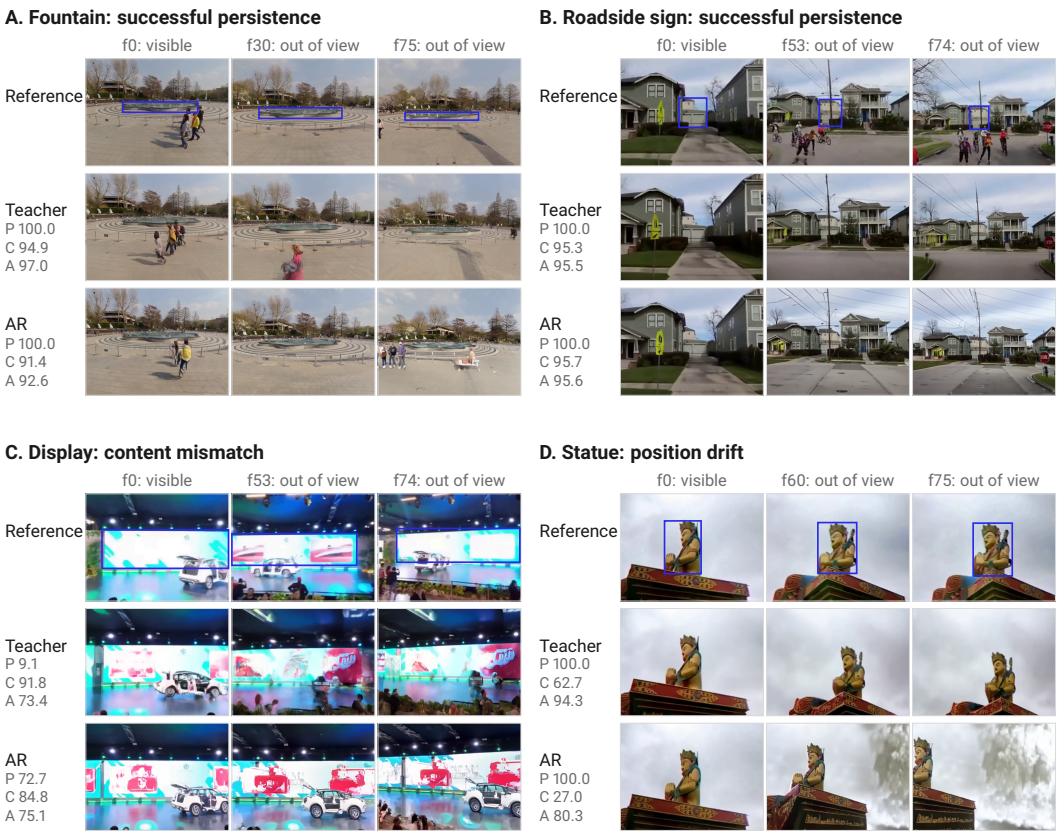

Figure I.1: Successful and weaker static-target examples. Rows show reference, teacher and AR; columns show the initial frame, the first frame with the target outside the input view, and the last such frame. Blue boxes identify the reference target. P/C/A denote invisible-interval permanence, continuity and appearance scores (0–100; higher is better). A–B preserve the evaluated static target, although moving bystanders may differ. C retains the display's location but changes its content: low permanence does not imply that the physical screen disappears. D retains the statue but misplaces it, especially in AR. Frames are shown at the shared evaluation field of view; no padded or interpolated frames are included.

These four source-distinct cases were selected from 947 common complete static cases using frozen scores before visual inspection: two with P$\geq$ 95, C$\geq$ 85 and A$\geq$ 85 for both models; one with teacher P$\leq$ 25; and one with teacher P$\geq$ 90 but C$\leq$ 65 or A$\leq$ 75. Deterministic selection also required the median reference target to occupy 2–50% of the image for legibility. The examples illustrate behavior rather than estimate its frequency. Continuity and appearance use tracked frames only, so they must be read alongside permanence.