# MEMORIZON: TRAINING WORLD MODELS BEYOND THEIR CONTEXT WINDOW

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Streaming world models should render a place consistently across repeated visits. Directly supervising such revisits requires training samples that capture both visits, often spanning minutes. Yet dense attention over the full span incurs quadratic cost, making long-span supervision expensive. Memorizon breaks this coupling: **long spans are needed for supervision, but not for attention** since the two visits can share a forward pass without including every intervening frame. A training sample covers a span of any length but is scored only on its last $k$ chunks. Instead of tokenizing the history before them, each scored chunk retrieves its own top-$K$ latents by camera co-visibility, and the union of these requests forms a shared bank. The bank is bounded by $kK$, so the sequence stays bounded however long the span; at the shortest span the recipe is exactly conventional training. Adding the bank raises the cost of a step once; beyond that, a longer span costs little, and going from 100 to 400 s adds 12% to the step time. Against a sliding-window baseline, retrieval raises revisit consistency on every split, and a span long enough to reach the first visit of each return adds a further 24% to 30%, at some cost in image quality; beyond that span, more length no longer helps. Filling the bank from another episode lowers revisit correlation by 83%, so the model uses what it retrieves. Project page: https://memorizon.github.io/.

## 1 INTRODUCTION

General video world models should stay consistent over a long rollout: a place the camera leaves and later returns to should look as it did before. The prevailing remedy is architectural — recurrent states, retrieval banks and learned summaries of history that outlive the attention window (Xiao et al., 2025; Yu et al., 2025; Zhang et al., 2025a; Wu et al., 2026a).

Whether a model learns to use a memory depends on what its training samples contain, not only on its architecture. When training is limited to a short video clip, that clip *is* the sequence the transformer attends to, so the span a sample covers equals its context. Attention cost often limits that span to about tens of seconds, while the model is deployed for minutes. If the camera takes longer than a training clip to leave a place and return, no sample contains both visits and no loss term relates them. On our corpus the shortest genuine return spans 13.25 s, and stricter definitions push it past 28 s (App. B.2). Memory-conditioned models (Yu et al., 2025; Xiao et al., 2025) already relax this by placing frames retrieved from outside the clip into the training sequence. Two questions remain open: how retrieval should be organized when the scored chunks of one sample look at different places, and how long a span must be for what they retrieve to contain the first visit at all.

The direct remedy, a longer training window, scales poorly. Attention cost is quadratic and activation memory linear in the context, so a 30 s window needs roughly three times the activation memory of a 10 s one, yet only 3.8% of such windows on our corpus contain a genuine return. Sparse attention reduces the cost of processing long sequences (Zhang et al., 2025b; Cai et al., 2025), while history compression reduces the number of tokens representing the past (Zhang et al., 2025a). Neither changes which pairs of visits a sample can relate. Retrieving once for the whole clip, as in Context-as-Memory (Yu et al., 2025), gives every chunk the same frames although each draws a different view; we show this loses most of what retrieval buys. We instead let each scored chunk retrieve its own top-$K$ and share their union as one bank, and treat the span as a sampled variable rather than a fixed clip length, so the sequence stays bounded however far back the first visit lies.



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.

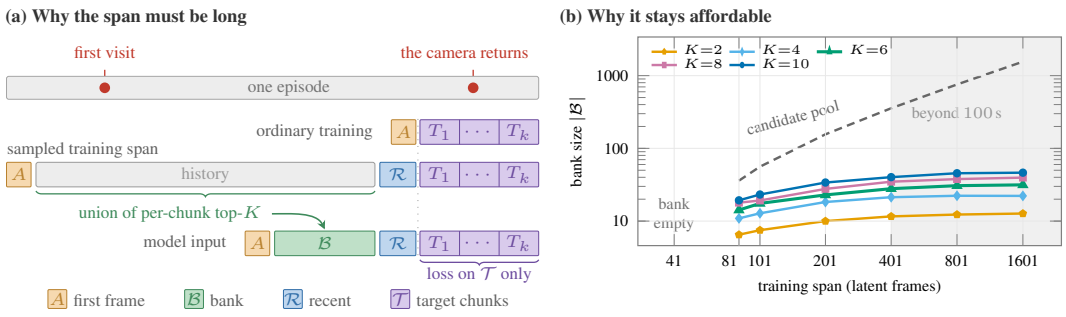


Figure 1: **Long-horizon training at bounded cost. (a)** Ordinary training covers only a short tail of the episode, so the first visit falls outside it; MEMORIZON samples a span that holds both visits, scores only its tail and supplies the history through a bank $\mathcal{B}$ of retrieved latents. **(b)** Bank size against training span: the bank levels off at every $K$ while the candidate pool grows; at 41 latents there is no history. Shading marks spans beyond 100 s.

**MEMORIZON.** Each training sample covers a span of $m$ chunks, drawn per sample, and scores only the last $k$; the unretrieved history before them is never tokenized. Instead, each scored chunk retrieves its own top-$K$ latents by camera co-visibility, ranking candidates by frustum overlap penalized by relative pose, and their union forms a shared memory bank. Within the sequence, attention is causal across chunks with a sliding window of $r{+}1$ chunks, the first frame serves as an attention sink, and a per-chunk mask lets each chunk read only its own retrieved entries, so every chunk directly attends to a bounded number of positions. A rollout reads the same layout one chunk at a time. MEMORIZON is thus neither a long-context method, which cheapens attention over more tokens, nor a method for longer rollouts, which are already routine; it is a recipe for training on long video at bounded cost.

**Results.** Retrieval raises revisit consistency on every split, already for a model trained on 10 s clips, and training on spans of up to 200 s raises it further (Tables 2 and 7). Widening the window without retrieval is no substitute. Longer spans cost some image quality. **The bank levels off**: for $K = 6$ it holds 28 entries at 100 s and 31.5 at 400 s, while the candidate pool grows tenfold from 50 to 400 s (Figure 1b). Against open world models, MEMORIZON and CaR are the only two with a clear memory, with MEMORIZON ahead on most memory metrics and on returns met mid-path, and replacing the bank's contents at inference shows that it reads what it retrieves (Sec. 4).

**Contributions.** (i) We propose MEMORIZON, a recipe that samples spans of any length and supplies their history through a shared bank formed from per-chunk retrieval, so the training sequence stays bounded however long the span and reduces exactly to ordinary training at the shortest one (Sec. 3). (ii) We show that its two parts do different jobs. Per-chunk retrieval under a shared positional index lets a model read frames far older than any it was trained on; a span that reaches the first visit turns returns into supervision and adds the gain on the returns that test memory hardest, those met mid-path, which the first frame cannot serve, and those more than 10 s apart, with no further gain once the span covers the first visits (Sec. 4.3, App. C.3–C.4). (iii) We show that the bank saturates and the step cost barely grows with the span, that the model reads what it retrieves rather than treating it as filler, and we report the cost in image quality this currently carries (Sec. 4).

## 2 RELATED WORK

**Long-Horizon Video World Models.** Recent interactive world models generate for a minute or more yet train on clips of a few seconds, a length set by attention cost, and rely on the model to generalize across the gap (Xiang et al., 2024; Bruce et al., 2024; Google DeepMind, 2025; He et al., 2025; Sun et al., 2025; Robbyant Team, 2026; Gao et al., 2026; Mao et al., 2025; DreamX Team, 2026; Xu et al., 2026b; Liu et al., 2025). Infinite-World (Wu et al., 2026a) observes that memory collapses beyond the temporal window seen in training and answers it with a pose-free hierarchical memory that compresses the history into a compact state, trained on revisit-dense data; we instead remove the need for a return to fit inside the window at all.

**Memory in Video World Models.** Memory mechanisms either compress history or select from it. Compression folds history into a compact state (Zhang et al., 2025a; Wu et al., 2026a; Mao et al., 2025; Hong et al., 2025), in the limit into a fixed summary of the opening chunk (Henschel et al., 2025). CaR (Peng et al., 2026) sits between the two: it compresses the history with a lightweight network and retrieves from it implicitly, through attention over viewpoints injected by positional encoding, so what it reads grows with the history it attends to; we instead select retrieved latents explicitly by camera co-visibility and keep the sequence bounded however long the span. Selection keeps a few past frames, retrieved by frustum overlap (Xiao et al., 2025; Yu et al., 2025; Oshima et al., 2026), by 3D structure such as point maps or image patches lifted to 3D (Li et al., 2025b; Huang et al., 2025a; Wu et al., 2025; Yu et al., 2026b), by camera-aware scores or gating (Sun et al., 2025; Wang et al., 2026; Guo et al., 2026), as retrieval-augmented context (Chen et al., 2025), or through a learned query (Yu et al., 2026a); training-free variants select inside the KV cache (Yi et al., 2026; Meng et al., 2026; Ma et al., 2026; Wu et al., 2026b), and so reach far back only at rollout. Context-as-Memory (Yu et al., 2025) retrieves clean frames for each predicted segment based on field-of-view overlap, enabling a bidirectional model to generate longer videos. WorldMem (Xiao et al., 2025), in contrast, trains a causal window with memory frames sampled from anywhere in the same video based on pose proximity. However, it is trained solely on Minecraft, leaving its memory mechanism specialized to a single environment without demonstrating the ability to generalize across environments.

**Efficient Attention and Long-Sequence Training.** Attention sinks (Xiao et al., 2024), sparse attention (Zhang et al., 2025b; Cai et al., 2025; Xu et al., 2026a) and history routing (Guo et al., 2025) make each token cheaper, and sequence parallelism shards one sequence across devices, as in LWM (Liu et al., 2024). Both still pay for every token in the sequence, whereas we keep almost none of a long span there, so the sequence does not grow with it. The closer precedent is retrieval-augmented language modelling, which trains on short subsequences while retrieving from a long document (Wu et al., 2022; Mohtashami & Jaggi, 2023; Tworkowski et al., 2023); we retrieve by camera co-visibility instead of learned similarity. Diffusion forcing (Chen et al., 2024) and self-forcing or distribution-matching distillation (Huang et al., 2025b; Yin et al., 2025; 2024), as used by RELIC (Hong et al., 2025), narrow the gap at rollout, but all leave the span of a sample equal to its length.

# 3 Method

## 3.1 Overview

Let $c$ be the chunk size in latents. A training sample covers the first frame $A$ and $m$ chunks behind it, with $m$ drawn per sample from $[m_{\min}, m_{\max}]$; the span is the only quantity that varies between samples. Two constants partition it: the last $k$ chunks are *scored*, the $r$ chunks before them form the *recent* block, and the remaining $\eta = m - k - r$ are *history*,

$$\underbrace{A}_{\text{first frame}} \;\Big|\; \underbrace{H_1 \dots H_\eta}_{\text{history}} \;\Big|\; \underbrace{R_1 \dots R_r}_{\mathcal{R},\ r \text{ chunks}} \;\Big|\; \underbrace{T_1 \dots T_k}_{\mathcal{T},\ k \text{ chunks}} .$$

A span shorter than $k + r$ chunks has fewer recent chunks and no history. Only the history grows with $m$, and it is never tokenized. It is a pool from which the scored chunks retrieve, so the transformer reads

$$[\underbrace{A}_{1} \mid \underbrace{\mathcal{B}}_{|\mathcal{B}|} \mid \underbrace{\mathcal{R}}_{r\,c} \mid \underbrace{\mathcal{T}}_{k\,c}], \qquad L \;=\; 1 + |\mathcal{B}| + rc + kc \text{ slots}, \tag{1}$$

where the bank $\mathcal{B}$ holds the retrieved history latents (Sec. 3.2) and never exceeds $k \cdot K$ entries. $L$ **is bounded independently of** $m$: the only term that answers to the span at all is $|\mathcal{B}|$, which is bounded by how many chunks ask rather than by how much history exists. The recent block is sized to the attention window of Sec. 3.2: a window of $r{+}1$ chunks reaches $r$ chunks back from $T_1$, which is exactly $\mathcal{R}$. We use $r = 1$ throughout.

**Ordinary Training as a Special Case.** At $m = k$ the recent block and the history are both empty, the span is $\mathcal{T}$ itself, and $L = 1 + kc$ is the standard training sample. With $k = 10$ and $r = 1$, a draw of $m = 10$ therefore carries no recent chunk and $T_1$ opens the window, while $m = 11$ is the shortest draw at which $\mathcal{R}$ is full and $m = 12$ the shortest with any history at all. Setting $m_{\min} = k$ makes

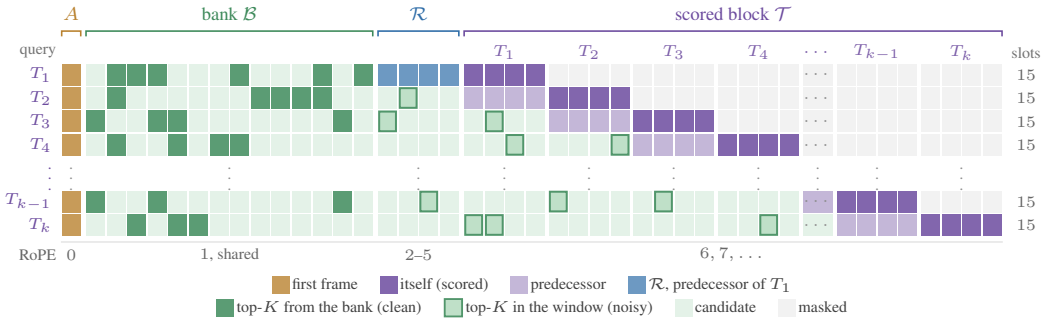


Figure 2: **The training sequence and what each scored chunk reads.** *Top:* the packed sequence and its RoPE index; bank entries share one index. *Grid:* the attention mask, one row per scored chunk. Each chunk attends to the first frame, its window of $r{+}1$ chunks and its own top-$K$: $1{+}K{+}(r{+}1)c = 15$ latents, whatever the span. Drawn for $K{=}6$, $r{=}1$, $c{=}4$.

ordinary training the shortest draw of our sampler, so every comparison against standard practice changes a single integer.

### 3.2 Attention and Memory Bank

**Attention Sink.** Every chunk and latent attend to the first frame $A$, which serves as an attention sink (Xiao et al., 2024). The first frame only attends to itself.

**Causal Sliding Window.** Attention is bidirectional within a chunk and causal across chunks. Each scored chunk $T_j$ sees itself and the $r$ chunks before it, a window of $r{+}1$ chunks. The window of $T_1$ covers $\mathcal{R}$, and it then slides right by one chunk for each subsequent $T_j$. The first frame, bank, and recent chunks are pure context and never attend to $\mathcal{T}$.

**Per-Chunk Retrieval.** Beyond its window, $T_j$ reads its own top-$K$ latents, ranked by Eq. 2 over everything completed before its window: the bank, which carries the history; the recent chunks outside the window; and the scored chunks $T_1 \dots T_{j-r-1}$. Candidates already in the sequence are read in place through the mask, so retrieving them costs no slots: those in $\mathcal{R}$ are read as the clean context they are, those in $T_1 \dots T_{j-r-1}$ as the noisy latents diffusion forcing has made them. No chunk ever sees a *scored* latent of its own window in clean form — the clean latents $T_j$ can reach are all pure context, which is never a target of the loss. $T_j$ therefore attends to at most $1 + K + (r{+}1)c$ positions, and to exactly that many once $K$ candidates have completed before its window, independent of the span and of $|\mathcal{B}|$ (Figure 2; App. A.2).

**Memory Bank.** The bank collects what the scored chunks retrieve from history: each $T_j$ contributes its own top-$K$ over the history, and the bank is their union. All $k$ cameras are known when a sample is drawn, so the union is formed before the forward pass and shared by every scored chunk, while the per-chunk mask leaves each chunk with only its own entries. Sharing costs 1.82 slots per scored latent against 3.75 for a private copy per chunk (App. A.2), and because the bank draws only from history, no latent the model is scored on ever enters the sequence in clean form. The union also beats curating a memory in advance: it contains each scored chunk's own top-$K$, so a chunk re-selecting inside it recovers exactly the set it would have chosen from the whole history (Proposition 1), which no rule that admits entries before knowing who will ask can guarantee.

**RoPE Index.** The first frame takes temporal index 0, every bank entry shares index 1, and the recent and scored chunks follow consecutively. A shared index reflects that a retrieved set has no order, and it keeps every position in the scored block fixed as $|\mathcal{B}|$ varies across samples and again at inference; methods that retrieve a fixed number of frames can enumerate them instead (Yu et al., 2025). It also withholds a frame's age: a bank entry carries its camera but not how long ago it was drawn. Positions therefore never depend on the length of the history, so they neither grow without bound nor leave the range seen in training, however long the rollout. What the model learns on entries of one age carries over to entries of any other: a model trained only on 10 s clips reads, at inference, bank entries up

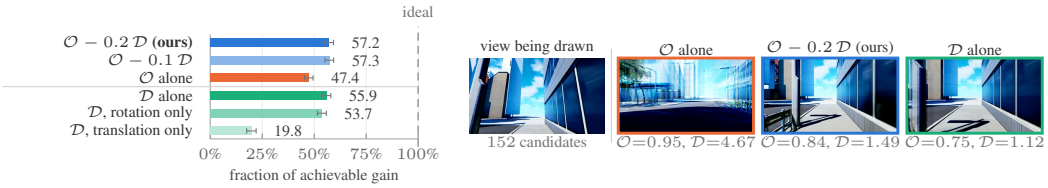


Figure 3: **Retrieval criterion graded against ground truth.** *Left:* fraction of achievable gain over 960 scored chunks, from random (0%) to the best candidates (100%); $\lambda = 0.1$ and 0.2 are level within the standard error; bars show $\pm 1$ standard error over chunks. *Right:* the frame each rule ranks first for one chunk.

to 59 s old and gains on returns 20 to 60 s apart (App. C.3). Numbering the bank in temporal order instead is not a clear win (Sec. 4.3).

### 3.3 Retrieval Criterion

A scored chunk should retrieve the frames that saw the place it is about to draw. We estimate this co-visibility from camera poses alone, with two cues. *Frustum overlap* $\mathcal{O}(q, c) \in [0, 1]$ is the fraction of points sampled in the query view $q$ that fall inside the frustum of candidate $c$ (Eq. 4); it measures how much of what is about to be rendered the candidate already holds, and requires intrinsics and a depth range (App. A.3 gives the grid and the range we use). *Pose distance* $\mathcal{D}(q, c)$ needs neither, though its translation term, like the depth range, is in the units of the corpus and so depends on the scene scale (App. A.4). Candidates are ranked by

$$\mathcal{S}(q, c) \;=\; \mathcal{O}(q, c) \;-\; \lambda\, \mathcal{D}(q, c), \tag{2}$$

$$\mathcal{D}(q, c) \;=\; \|t_q - t_c\| \;+\; w\, \angle(f_q, f_c)\,, \tag{3}$$

with camera centres $t$ in the units of the corpus, optical axes $f$, the angle $\angle(\cdot, \cdot)$ between them in radians, $\lambda = 0.2$ and $w = 4$, so that turning by 15° weighs about as much as moving one unit (4 m). Each chunk keeps its top $K$, ties in $\mathcal{S}$ broken in favour of the candidate nearer the query in time, so a candidate further back never displaces an equal-scoring incumbent. The form is not new: WorldMem (Xiao et al., 2025) and WorldPack (Oshima et al., 2026) subtract a time penalty from frustum overlap, and HY-WorldPlay 1.5 (Sun et al., 2025) combines overlap with camera distance. We penalize relative pose rather than elapsed time, since a camera can return to a place long after it left.

**Ground-Truth Evaluation.** The ground-truth latents of the view being drawn and of every candidate are available at training time, so a rule can be graded directly: by the cosine similarity between the frames it selects and that view, expressed as the fraction of achievable gain between random selection and the best candidates that exist (Figure 3). Three results follow. Overlap alone reaches 47.4%, because exact ties are common: any candidate that contains the whole query view scores $\mathcal{O} = 1$. The mixed score reaches 57.2% at $\lambda = 0.2$, level with the best value (57.3% at $\lambda = 0.1$). Pose distance reaches 55.9% on its own, mostly through its orientation term: ranking by camera centres alone gives 19.8%, since these cameras turn far more than they move. Neither cue accounts for occlusion or for motion along the viewing axis (App. A.4).

## 4 Experiments

We first describe training and evaluation (Sec. 4.1), then compare Memorizon with open world models (Sec. 4.2). An ablation adds retrieval, the bank and a longer span one at a time (Sec. 4.3).

### 4.1 Experimental Setup

**Training Details.** We initialize from Wan2.2-TI2V-5B (Wan Team, 2025), add a camera branch with PRoPE (Li et al., 2025a), and train the model as a chunked causal diffusion transformer using diffusion forcing (Chen et al., 2024). Each chunk contains $c = 4$ latents. Each sample scores $k = 10$

Table 1: **Comparison with open world models** on the web photographs, with returns to the starting pose and to mid-path poses scored apart. Mean over five seeds; Table 8 gives the spreads. MEMORIZON: the 100 s run of Table 2. DINO and D. Gain use DINO ViT-B/16 cosine in place of pixel correlation. †Tracks our walks only to within 1.1 m and 34°.

| | Starting pose | | | | Mid-path | | | | VBench | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Method | Revisit↑ | Gain↑ | DINO↑ | D. Gain↑ | Revisit↑ | Gain↑ | DINO↑ | D. Gain↑ | Subj.↑ | Backg.↑ | Imag.↑ |
| LingBot-World 2.0 | 0.143 | −0.006 | 0.526 | −0.001 | 0.147 | 0.011 | 0.546 | 0.017 | 0.765 | 0.852 | 0.741 |
| DreamX-World 1.0 | 0.146 | 0.033 | 0.547 | −0.045 | 0.109 | 0.009 | 0.591 | 0.007 | 0.781 | 0.868 | 0.684 |
| HY-WorldPlay 1.5† | 0.243 | 0.040 | 0.679 | 0.025 | 0.194 | 0.037 | 0.674 | 0.027 | **0.839** | **0.908** | 0.702 |
| Matrix-Game 3.0 | 0.254 | 0.075 | 0.717 | 0.042 | 0.283 | 0.089 | <u>0.760</u> | 0.100 | <u>0.813</u> | <u>0.895</u> | **0.761** |
| Infinite-World | 0.109 | 0.019 | 0.610 | 0.013 | 0.168 | 0.070 | 0.646 | 0.052 | 0.789 | 0.856 | <u>0.746</u> |
| CaR | <u>0.350</u> | <u>0.220</u> | **0.843** | **0.305** | **0.450** | <u>0.312</u> | 0.759 | <u>0.211</u> | 0.787 | 0.846 | 0.715 |
| **MEMORIZON (ours)** | **0.365** | **0.230** | <u>0.787</u> | <u>0.219</u> | <u>0.447</u> | **0.329** | **0.816** | **0.258** | 0.778 | 0.857 | 0.705 |

chunks behind $r = 1$ recent chunk, with the span sampled as $m \sim U[10, m_{\max}]$. Each scored chunk independently selects its top-$K$ entries, with $K = 6$, using Eq. 2. Entries preceding the window are accessed through the union bank, while entries within it are accessed through the mask. All runs use the same initialization, data, and training schedule, and train for 6,000 steps with a batch size of 32, using one sample per GPU across 32 H200 GPUs. App. A.6 details the optimizer, conditioning, and sampler; App. A.2 reports the resulting sequence sizes.

**Evaluation.** For each model, we average five rollouts with different seeds, using 20 denoising steps and classifier-free guidance at scale 4. The evaluation set comprises 20 60 s clips from ten training scenes, 16 clips from four held-out scenes, and 20 web photographs. Rollouts on the latter two splits last 400 s; all tables report the first 60 s.

**Metrics.** Two latents form a *return* when their cameras lie within 0.5 units and 15° of each other, at least 8 s apart, with the camera away in between. *Revisit* is the Pearson correlation between the frames a model generates at the two ends: it asks whether a model draws a place as it drew it before. Frames of one video correlate even without a return, so *Gain* subtracts the correlation of control pairs at the same time gaps whose cameras differ. *DINO* scores the same returns by the cosine similarity of DINO ViT-B/16 features (Caron et al., 2021) instead, which tolerates the small misalignments a pixel correlation penalizes. A return is *to the starting pose* when either end meets the first frame's camera and *mid-path* otherwise; the first frame is in every sequence, so only the second kind needs the bank. We also report PSNR and LPIPS (Zhang et al., 2018) against the rendered ground truth where it exists, and VBench (Huang et al., 2024) for consistency and quality. App. C.1 gives the rest.

### 4.2 COMPARISON WITH SOTA

We compare against open world models including: LingBot-World 2.0 (Gao et al., 2026), DreamX-World 1.0 (DreamX Team, 2026), HY-WorldPlay 1.5 (Sun et al., 2025), Matrix-Game 3.0 (Wang et al., 2026), Infinite-World (Wu et al., 2026a) and CaR (Peng et al., 2026). In Table 1, every method rolls out 60 s from the same photographs along the same path, using its released weights, default sampler, native resolution and frame rate, and its own intrinsics; a method driven by discrete actions receives the path converted to its action vocabulary. For Revisit, frames are matched to the path by timestamp and resized to $864 \times 480$. The action vocabulary of Matrix-Game 3.0 turns more slowly than the fastest paths, which it therefore follows only approximately.

MEMORIZON and CaR are the only systems with a clear memory. MEMORIZON is highest in five of the eight memory columns and CaR in the other three: 0.365 Revisit at the starting pose and 0.447 mid-path for MEMORIZON, 0.350 and 0.450 for CaR, against 0.254 and 0.283 for Matrix-Game 3.0, the strongest of the other baselines. In DINO features the two split by kind of return: CaR is ahead at the starting pose, which the first frame can serve, and MEMORIZON on mid-path returns, which need what was generated along the way. The Gain columns are the sharper statement, because Gain removes what any two frames of one video share at the same time gap. For the other five baselines it is at most 0.089, and for LingBot-World 2.0 at the starting pose it is −0.006: what these models draw at a return is about what an ordinary pair of frames already agrees on, so little of it can be credited




Figure 4: **Comparison with SOTA** on two web photographs. Each system is shown late in the walk, when the camera is back near the input view; the input is the reference.

to having been at that place before. Ours is 0.230 and 0.329, and CaR's 0.220 and 0.312. Figure 4 shows one return for every system, and App. C.10 (Figure 8) three more.

A natural objection is that this margin measures camera control rather than memory: the five other baselines follow the prescribed path with correlations of 0.70 to 0.88 against our 0.95 to 0.97 (App. C.7). It cannot explain the gap. Of the 173 returns in this window, 125 (72%) are fold-backs, where the path out is retraced on the way back, and on a fold-back any consistent error in the scale of a model's turns and steps largely cancels, so the camera returns close to the pose it held on the first visit (App. C.1). What differs at that pose is what the model draws there.

We do not lead the VBench columns and state it plainly. HY-WorldPlay 1.5 scores higher on subject and background consistency, 0.839 and 0.908 against our 0.778 and 0.857, and four of the six baselines score higher on imaging quality, up to 0.761 against our 0.705. Part of this is what the columns measure: both consistency scores compare neighbouring frames, so a rollout that drifts smoothly away from what it drew a minute earlier scores well on them and poorly on Revisit. The rest is a real cost of training on a longer span, which Sec. 4.3 reports and App. C.9 traces to conditioning on generated frames rather than to a weaker generator.

### 4.3 Ablation Study

Table 2 adds one ingredient per row (seen scenes in Table 7). The first three rows train on 10 s clips: a sliding window of the first frame, the predecessor and the chunk itself; a window widened to the whole clip, which measures what more attention buys without retrieval; and the sliding window plus per-chunk top-$K$, whose candidates lie inside the clip in training but span the whole generated history at inference, as for every row. The next three raise the span to 100 s and differ only in how the bank is formed: one retrieval for the whole scored block, as in Context-as-Memory; per-chunk retrieval with the bank numbered in temporal order; and ours, with one shared index. The last two raise $m_{\max}$ alone. All rows share the initialization, data, optimizer, step count and five rollout seeds of Sec. 4.1.

**Effect of Retrieval.** Letting each chunk retrieve raises Revisit on every split and multiplies Gain several times over, already for a model trained on 10 s clips (Figure 5; more returns from MEMORIZON in Figure 9). At inference, that model retrieves from the whole generated history like every other row, and since retrieval is by pose and the bank shares one index, nothing tells it how old a retrieved frame is: what it learns on frames less than 10 s old carries over to frames retrieved from far further back (App. C.3). The rise is smallest on seen scenes, whose places the model has already learned in training and can partly redraw without retrieval, and largest on web photographs, where only the retrieved frames tell the model what the place looked like. More attention is no substitute. Attending to every earlier chunk of the clip without retrieval helps on unseen scenes, is level on seen ones and lowers Revisit on web photographs, on 16 of the 20 photographs and in every one of the five rollout seeds. Apart from the first frame, such a window holds only frames the model generated itself. In the rendered domain those frames stay close to real; from a photograph outside it, each chunk drifts slightly, and a longer window keeps conditioning on that drift.

Who retrieves matters as much as whether. Sharing one retrieval across the whole scored block, the arrangement of Context-as-Memory, lowers Revisit and Gain on every split at the same span, slightly on rendered scenes and by about a third on web photographs: one retrieval cannot match the view of every chunk in the block, so most chunks read frames chosen for another pose. The model also reads what the bank holds, not only how many frames it has: varying only the contents of the bank's slots at inference, with checkpoint, trajectory and seed fixed, mid-path Revisit halves with random

Table 2: **Ablation on training span, bank and retrieval.** Rows add one ingredient at a time (Sec. 4.3); spans are drawn uniformly from 10 s to the length given. PSNR and LPIPS need rendered ground truth, which web photographs lack. All runs at step 6,000; mean over five rollout seeds, standard deviation of the per-seed means in small type. DINO is Revisit with the cosine of DINO ViT-B/16 features in place of pixel correlation. Best per split in bold, second underlined. Seen scenes are in Table 7 (App. C.2).

| | Fidelity to GT | | Memory | | | VBench | | |
|---|---|---|---|---|---|---|---|---|
| Training | PSNR↑ | LPIPS↓ | Revisit↑ | Gain↑ | DINO↑ | Subj.↑ | Backg.↑ | Imag.↑ |
| *Unseen scenes* | | | | | | | | |
| 10 s, no retrieval | 10.60 ±.06 | 0.646 ±.005 | 0.311 ±.018 | 0.049 ±.020 | 0.557 ±.020 | 0.751 ±.006 | 0.873 ±.001 | 0.666 ±.014 |
| 10 s, 10-chunk window | 10.47 ±.32 | 0.662 ±.006 | 0.363 ±.033 | 0.114 ±.033 | 0.572 ±.020 | 0.726 ±.008 | 0.863 ±.003 | 0.643 ±.007 |
| 10 s, top-$K$, no bank | 10.71 ±.20 | 0.630 ±.005 | 0.503 ±.022 | 0.255 ±.026 | 0.735 ±.011 | 0.757 ±.003 | 0.873 ±.003 | 0.675 ±.016 |
| 100 s, per-segment retrieval, bank | 11.27 ±.22 | 0.636 ±.013 | 0.511 ±.040 | 0.259 ±.023 | 0.743 ±.032 | 0.763 ±.003 | 0.873 ±.002 | 0.671 ±.022 |
| 100 s, top-$K$, ordered bank | **11.78** ±.15 | 0.618 ±.010 | 0.600 ±.022 | 0.299 ±.011 | 0.809 ±.016 | 0.775 ±.008 | 0.879 ±.004 | 0.630 ±.010 |
| 100 s, top-$K$, bank | 11.62 ±.21 | 0.643 ±.005 | 0.559 ±.037 | 0.300 ±.020 | 0.822 ±.022 | 0.772 ±.006 | 0.878 ±.002 | 0.668 ±.013 |
| 200 s, top-$K$, bank | 11.55 ±.30 | **0.616** ±.009 | **0.623** ±.026 | **0.343** ±.014 | **0.854** ±.003 | **0.776** ±.008 | **0.881** ±.003 | 0.627 ±.013 |
| 400 s, top-$K$, bank | 11.36 ±.19 | 0.642 ±.008 | 0.558 ±.012 | 0.293 ±.019 | 0.830 ±.011 | **0.776** ±.005 | 0.875 ±.003 | **0.690** ±.016 |
| *Web photographs* | | | | | | | | |
| 10 s, no retrieval | – | – | 0.183 ±.009 | 0.060 ±.011 | 0.571 ±.010 | 0.769 ±.004 | 0.849 ±.002 | **0.736** ±.003 |
| 10 s, 10-chunk window | – | – | 0.134 ±.025 | 0.032 ±.023 | 0.540 ±.018 | 0.742 ±.006 | 0.839 ±.006 | 0.702 ±.008 |
| 10 s, top-$K$, no bank | – | – | 0.359 ±.014 | 0.227 ±.010 | 0.725 ±.008 | 0.774 ±.003 | 0.855 ±.002 | 0.722 ±.003 |
| 100 s, per-segment retrieval, bank | – | – | 0.302 ±.026 | 0.193 ±.030 | 0.687 ±.025 | 0.767 ±.009 | 0.847 ±.002 | 0.715 ±.017 |
| 100 s, top-$K$, ordered bank | – | – | 0.443 ±.014 | 0.307 ±.025 | 0.779 ±.012 | 0.767 ±.002 | 0.854 ±.003 | 0.675 ±.012 |
| 100 s, top-$K$, bank | – | – | 0.434 ±.013 | 0.318 ±.016 | 0.811 ±.017 | 0.778 ±.007 | 0.857 ±.004 | 0.705 ±.012 |
| 200 s, top-$K$, bank | – | – | **0.467** ±.010 | **0.359** ±.016 | **0.831** ±.006 | 0.782 ±.002 | **0.863** ±.003 | 0.654 ±.013 |
| 400 s, top-$K$, bank | – | – | 0.396 ±.024 | 0.285 ±.015 | 0.806 ±.017 | **0.783** ±.006 | 0.857 ±.002 | 0.733 ±.009 |




Figure 5: **Ablation study** on five models of Table 2 on a web photograph. Top: back near the input view. Middle: a place a little further along. Bottom: the return to it.

frames from the same history and falls further with an empty bank or frames from another episode (App. C.8).

**Effect of Training Span.** Training on longer spans raises Revisit and Gain further on every split, most at 200 s, and the gain over 10 s top-$K$ lies mainly in returns more than 10 s apart and in mid-path returns (App. C.4). At 400 s both fall back on every split, to about the level of 100 s and below it on web photographs. Once the span covers the first visits of returns, more length only adds candidates no chunk chooses: a span of 100, 200 and 400 s reaches the first visit of 64%, 84% and 93% of returning latents in our corpus (App. B), and the bank barely grows past 200 s, since it stops changing once new history offers no candidate that outscores a chunk's current $K$-th (Figure 1b). The span is a coverage setting, not a quantity to maximize. Image quality moves the other way. Imaging quality falls as the span grows to 200 s, most on unseen scenes and web photographs, and recovers at 400 s, where Revisit falls back. The cause is not a weaker generator: given a ground-truth history every model we tested, from 10 to 200 s, reaches the quality of the rendered videos, and the loss comes from conditioning on its own output, most of it through the bank (App. C.9). The 400 s run reverses both trends at once, with Revisit falling back while imaging quality recovers, which is consistent with a model that relies less on its bank, the path through which most of the quality loss enters.

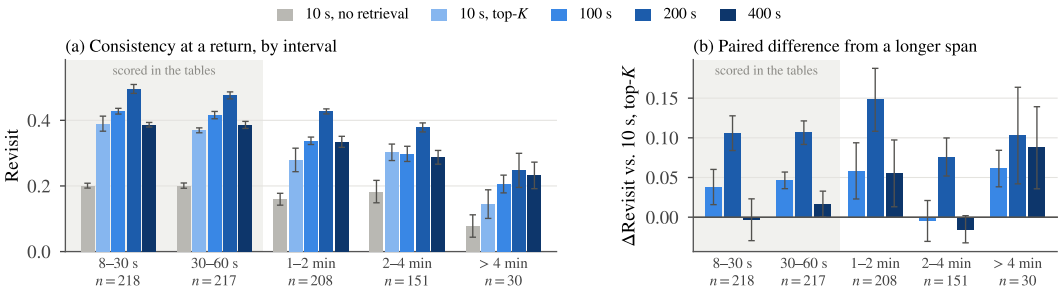


Figure 6: **Revisit against return interval** over 400 s rollouts on the web photographs; error bars are the standard deviation over five rollout seeds. **(a)** Revisit per model, grouped by the interval between the two visits; $n$ counts returns per seed. **(b)** Difference from 10 s top-$K$ on the same returns. Shading marks the first 60 s, which the tables score.

**Effect of the Bank Index.** Numbering the bank's entries in temporal order rather than giving them one shared index is not a clear win either way. At 100 s it raises Revisit on seen and unseen scenes, close to the 200 s run, and is level on web photographs. DINO Revisit, however, is higher only on seen scenes, and imaging quality is lower on every split: the ordered index reproduces the retrieved frames more closely in pixels without drawing the place better in features, and at a cost in image quality.

**Returns Beyond the Training Window.** Our splits score the first 60 s, where returns are at most a minute apart, so they cannot separate spans longer than that. We therefore roll the same web photographs out to 400 s and score every return by the interval between its two visits, over the same five rollout seeds (Figure 6). The 10 s top-$K$ model, whose retrieval also reaches the whole history, decays as the interval grows, from $0.390 \pm 0.023$ under half a minute to $0.303 \pm 0.025$ at two to four minutes and $0.145 \pm 0.044$ beyond four. On the same returns the 200 s model is ahead of it in every bin, by $0.106 \pm 0.022$ under half a minute, $0.148 \pm 0.040$ at one to two minutes and $0.103 \pm 0.061$ beyond four, and the 100 s model is ahead in four of the five bins and level at two to four minutes ($-0.005 \pm 0.026$). The 400 s model does not extend this. It is level with 10 s top-$K$ under a minute ($-0.003 \pm 0.026$ and $+0.017 \pm 0.016$) and at two to four minutes ($-0.015 \pm 0.017$), and ahead only at one to two minutes ($+0.055 \pm 0.042$) and beyond four ($+0.087 \pm 0.052$), as in Table 2: a span that already covers the first visits gains nothing from more length. Returns more than four minutes apart are few, 30 per seed, so the last bin is the least certain.

## 5 Discussion

MEMORIZON decouples the span a world model is trained on from the sequence it attends over: each scored chunk retrieves its own top-$K$ latents from the history by camera co-visibility, and the union of these requests forms a bank. Lengthening the span from 100 to 400 s adds only 12% to the step time. Retrieval raises revisit consistency over a sliding window on every split, and a span that reaches the first visits raises it further, here up to 200 s and not beyond, at a cost in image quality that we trace to conditioning on generated frames. Against open world models, MEMORIZON leads five of the eight memory columns of Table 1 and CaR the other three, with MEMORIZON ahead on the returns the first frame alone cannot serve.

**Limitations.** We mainly focus on static scenes: nothing moves but the camera, so a return is always to a place that should look the same, and Revisit asks only whether a model draws it the same way again. This leaves out scenes in which objects move or the place itself changes between visits, where drawing a place differently can be correct and consistency has to be judged against what should have changed. The cost in image quality comes from a mismatch between training, where the bank and the recent chunks are clean ground truth, and inference, where they hold the model's own output (App. C.9). This is the exposure bias that Self Forcing (Huang et al., 2025b) removes by training on the model's own rollouts; we will apply it to the bank as well as to the recent window.

## AI USE STATEMENT

Captions conditioning the model were generated by a vision-language model (Qwen3-VL-8B; App. B.1). We used large language models to help edit the text, write analysis and plotting scripts, and check that numbers quoted in the text match the tables. All experiments, results and claims were produced and verified by the authors, who take full responsibility for the content of the paper.

## REPRODUCIBILITY STATEMENT

App. A.1 gives pseudocode for a training step and for a rollout, and Table 5 lists every setting the runs share, from the chunk layout and retrieval constants to the optimizer, sampler and evaluation protocol. App. B describes how the training corpus is rendered, encoded and split, and App. C.1 defines a return, Revisit and Gain precisely enough to recompute every memory column. All baselines are run from their released weights with their default samplers, as Sec. 4.2 states, and every number in the tables is a mean over five rollout seeds with its spread reported. Our training and inference code will be released.

## REFERENCES

Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal Behbahani, Stephanie Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative interactive environments. In *International Conference on Machine Learning (ICML)*, 2024.

Shengqu Cai, Ceyuan Yang, Lvmin Zhang, Yuwei Guo, Junfei Xiao, Ziyan Yang, Yinghao Xu, Zhenheng Yang, Alan Yuille, Leonidas Guibas, Maneesh Agrawala, Lu Jiang, and Gordon Wetzstein. Mixture of contexts for long video generation. *arXiv preprint arXiv:2508.21058*, 2025.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Boyuan Chen, Diego Marti Monso, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Taiye Chen, Xun Hu, Zihan Ding, and Chi Jin. Learning world models for interactive video generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

DreamX Team. DreamX-World 1.0: A general-purpose interactive world model. *arXiv preprint arXiv:2606.16993*, 2026.

Zelin Gao, Qiuyu Wang, Jiapeng Zhu, Jingye Chen, Zichen Liu, Qingyan Bai, Jiahao Wang, Yufeng Yuan, Hanlin Wang, Yichong Lu, Ka Leong Cheng, Haojie Zhang, Jian Gao, Tianrui Feng, Yuzheng Liu, Yao Yao, Yinghao Xu, Xing Zhu, Yujun Shen, and Hao Ouyang. Infinite worlds with versatile interactions. *arXiv preprint arXiv:2607.07534*, 2026.

Google DeepMind. Genie 3: A new frontier for world models. Technical blog post, 5 August 2025. https://deepmind.google/blog/genie-3-a-new-frontier-for-world-models/, 2025. Accessed 26 September 2026.

Yanjun Guo, Zhengqiang Zhang, Pengfei Wang, Xinyue Liang, Zhiyuan Ma, and Lei Zhang. Memorize when needed: Decoupled memory control for spatially consistent long-horizon video generation. *arXiv preprint arXiv:2604.18215*, 2026.

Yuwei Guo, Ceyuan Yang, Hao He, Yang Zhao, Meng Wei, Zhenheng Yang, Weilin Huang, and Dahua Lin. End-to-end training for autoregressive video diffusion via self-resampling. *arXiv preprint arXiv:2512.15702*, 2025.

Xianglong He, Chunli Peng, Zexiang Liu, Boyang Wang, Yifan Zhang, Qi Cui, Fei Kang, Biao Jiang, Mengyin An, Yangyang Ren, Baixin Xu, Hao-Xiang Guo, Kaixiong Gong, Size Wu, Wei Li, Xuchen Song, Yang Liu, Yangguang Li, and Yahui Zhou. Matrix-Game 2.0: An open-source, real-time and streaming interactive world model. *arXiv preprint arXiv:2508.13009*, 2025.

Roberto Henschel, Levon Khachatryan, Hayk Poghosyan, Daniil Hayrapetyan, Vahram Tadevosyan, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. StreamingT2V: Consistent, dynamic, and extendable long video generation from text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Yicong Hong, Yiqun Mei, Chongjian Ge, Yiran Xu, Yang Zhou, Sai Bi, Yannick Hold-Geoffroy, Mike Roberts, Matthew Fisher, Eli Shechtman, Kalyan Sunkavalli, Feng Liu, Zhengqi Li, and Hao Tan. RELIC: Interactive video world model with long-horizon memory. *arXiv preprint arXiv:2512.04040*, 2025.

Junchao Huang, Xinting Hu, Boyao Han, Shaoshuai Shi, Zhuotao Tian, Tianyu He, and Li Jiang. Memory forcing: Spatio-temporal memory for consistent scene generation on Minecraft. *arXiv preprint arXiv:2510.03198*, 2025a.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025b.

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

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 *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025b.

Hao Liu, Wilson Yan, Matei Zaharia, and Pieter Abbeel. World model on million-length video and language with blockwise RingAttention. *arXiv preprint arXiv:2402.08268*, 2024.

Zihan Liu, Yi Gu, Mingkai Deng, Guangyi Liu, Zeyu Feng, Qiyue Gao, Yiyan Hu, Benhao Huang, Yichi Yang, Kun Zhou, Jiannan Xiang, Zhiting Hu, Zhengzhong Liu, and Eric P. Xing. PAN: A world model for general, actionable, and long-horizon world simulation. *arXiv preprint arXiv:2511.09057*, 2025.

Wenchao Ma, Changran Liu, Sharon X. Huang, and Haomiao Jiang. Closing the loop: Training-free revisit consistency for autoregressive generative rendering. *arXiv preprint arXiv:2607.21848*, 2026.

Xiaofeng Mao, Zhen Li, Chuanhao Li, Xiaojie Xu, Kaining Ying, Tong He, Jiangmiao Pang, Yu Qiao, and Kaipeng Zhang. Yume-1.5: A text-controlled interactive world generation model. *arXiv preprint arXiv:2512.22096*, 2025.

Yu Meng, Xiangyang Luo, Letian Li, Wenyuan Jiang, Chen Gao, Xinlei Chen, Yong Li, and Xiao-Ping Zhang. TetherCache: Stabilizing autoregressive long-form video generation with gated recall and trusted alignment. *arXiv preprint arXiv:2606.13035*, 2026.

Amirkeivan Mohtashami and Martin Jaggi. Landmark attention: 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 (TMLR)*, 2026.

Zhan Peng, Jie Ma, Huiqiang Sun, Chong Gao, Zhijie Xue, Zhiyu Pan, Zhiguo Cao, Jun Liang, and Jing Li. Compression and retrieval: Implicit memory retrieval for video world models. *arXiv preprint arXiv:2606.23105*, 2026.

Robbyant Team. Advancing open-source world models. *arXiv preprint arXiv:2601.20540*, 2026.

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

Szymon Tworkowski, Konrad Staniszewski, Mikołaj Pacek, Yuhuai Wu, Henryk Michalewski, and Piotr Miłoś. Focused transformer: Contrastive training for context scaling. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Wan Team. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Zile Wang, Zexiang Liu, Jiaxing Li, Kaichen Huang, Baixin Xu, Fei Kang, Mengyin An, Peiyu Wang, Biao Jiang, Yichen Wei, Yidan Xietian, Jiangbo Pei, Liang Hu, Boyi Jiang, Hua Xue, Zidong Wang, Haofeng Sun, Wei Li, Wanli Ouyang, Xianglong He, Yang Liu, Yangguang Li, and Yahui Zhou. Matrix-Game 3.0: Real-time and streaming interactive world model with long-horizon memory. *arXiv preprint arXiv:2604.08995*, 2026.

Ruiqi Wu, Xuanhua He, Meng Cheng, Tianyu Yang, Yong Zhang, Zhuoliang Kang, Xunliang Cai, Xiaoming Wei, Chunle Guo, Chongyi Li, and Ming-Ming Cheng. Infinite-World: Scaling interactive world models to 1000-frame horizons via pose-free hierarchical memory. *arXiv preprint arXiv:2602.02393*, 2026a.

Tong Wu, Shuai Yang, Ryan Po, Yinghao Xu, Ziwei Liu, Dahua Lin, and Gordon Wetzstein. Video world models with long-term spatial memory. *arXiv preprint arXiv:2506.05284*, 2025.

Xindi Wu, Sven Elflein, James Lucas, Olga Russakovsky, Laura Leal-Taixé, Despoina Paschalidou, Jonathan Lorraine, and Aljaša Ošep. Addressable memory for video world models. *arXiv preprint arXiv:2608.07408*, 2026b.

Yuhuai Wu, Markus N. Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. In *International Conference on Learning Representations (ICLR)*, 2022.

Jiannan Xiang, Guangyi Liu, Yi Gu, Qiyue Gao, Yuting Ning, Yuheng Zha, Zeyu Feng, Tianhua Tao, Shibo Hao, Yemin Shi, Zhengzhong Liu, Eric P. Xing, and Zhiting Hu. Pandora: Towards general world model with natural language actions and video states. *arXiv preprint arXiv:2406.09455*, 2024.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations (ICLR)*, 2024.

Zeqi Xiao, Yushi Lan, Yifan Zhou, Wenqi Ouyang, Shuai Yang, Yanhong Zeng, and Xingang Pan. WorldMem: Long-term consistent world simulation with memory. *arXiv preprint arXiv:2504.12369*, 2025.

Boxun Xu, Yuming Du, Zichang Liu, Siyu Yang, Ziyang Jiang, Siqi Yan, Rajasi Saha, Albert Pumarola, Wenchen Wang, and Peng Li. Sparse forcing: Native trainable sparse attention for real-time autoregressive diffusion video generation. *arXiv preprint arXiv:2604.21221*, 2026a.

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*, 2026b.

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

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T. Freeman. Improved distribution matching distillation for fast image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Tianwei Yin, Qiang Zhang, Richard Zhang, William T. Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast autoregressive video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Jiwen Yu, Jianhong Bai, Yiran Qin, Quande Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Xihui Liu. Context as memory: Scene-consistent interactive long video generation with memory retrieval. In *SIGGRAPH Asia 2025 Conference Papers*, 2025.

Jiwen Yu, Jianxiong Gao, Jianhong Bai, Yiran Qin, Kaiyi Huang, Quande Liu, Xintao Wang, Pengfei Wan, Kun Gai, and Xihui Liu. 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, Songheng Yin, Sri Siddarth Chakaravarthy P, Dennis Anthony, Yang Ye, Yidi Li, Weiwei Wan, and Animesh Garg. MosaicMem: Hybrid spatial memory for controllable video world models. *arXiv preprint arXiv:2603.17117*, 2026b.

Lvmin Zhang, Shengqu Cai, Muyang Li, Gordon Wetzstein, and Maneesh Agrawala. Frame context packing and drift prevention in next-frame-prediction video diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025a.

Peiyuan Zhang, Yongqi Chen, Haofeng Huang, Will Lin, Zhengzhong Liu, Ion Stoica, Eric Xing, and Hao Zhang. Faster video diffusion with trainable sparse attention. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025b.

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

# APPENDIX

The appendix is organized in three parts. App. A gives pseudocode for training and inference, shows that the shared bank is lossless and bounded, gives the overlap formula, grades the retrieval cues and chooses the retrieval width and bank capacity against ground truth with no trained model, and lists all settings. App. B describes the training and test data and counts how far apart revisits lie. App. C defines the revisit metrics, gives the seen-scene ablation, explains where the memory of a model trained on 10 s clips comes from and splits the gain by return interval, measures the cost of a training step, explains the baselines left out, checks how closely each system follows the camera path, tests what the bank must hold at inference, traces why image quality falls with span, and shows more returns from MEMORIZON and the open world models.

## A METHOD DETAILS

### A.1 TRAINING AND INFERENCE PROCEDURES

Algorithm 1 builds and scores one training sample, and Algorithm 2 generates a video chunk by chunk. $\mathrm{TopK}(q, \mathcal{C})$ returns the $\min(K, |\mathcal{C}|)$ latents of the candidate set $\mathcal{C}$ with the highest score $\mathcal{S}(q, \cdot)$ of Eq. 2, ties broken in favour of the candidate nearer $q$ in time, where $q$ is the camera of the last latent of the requesting chunk.

**Inference.** A rollout generates one chunk at a time, and the chunk being generated occupies the position of $T_1$: the model reads the first frame, a bank of $K$ entries, the $r$ most recent generated chunks and the chunk being denoised, $1 + K + rc + c$ slots in all. Retrieval follows the training rule with a single requesting chunk, so the bank is that chunk's own top-$K$ by Eq. 2 over every latent generated before its window, recomputed at each step from the cameras seen so far. Nothing in the rollout depends on $m$, which only sets how much history a training sample carries, so a rollout can run far beyond the longest training span: every generated chunk is attended over the same $1 + K + rc + c$ slots, whatever has been generated before it. The ranking is the one part that does grow. Eq. 2 is evaluated over every latent generated so far, so selecting the bank for chunk $n$ costs $O(n)$ score evaluations against attention that costs nothing extra; it is a comparison of poses outside the network, and we leave it unindexed.

**Algorithm 1** One training step

**Require:** episode latents $x_{0:N}$ with cameras $P_{0:N}$; constants $c, k, r, K$; span range $[m_{\min}, m_{\max}]$
1: $m \sim U[m_{\min}, m_{\max}]$; pick a start $s$ so the span of $1 + mc$ latents fits
2: $A \leftarrow x_s$; $|\mathcal{R}| \leftarrow \min(r, m - k)$, $|\mathcal{H}| \leftarrow \max(0, m - k - r)$ chunks ▷ empty when $m = k$
3: split the next $m$ chunks into history $\mathcal{H}$, recent $\mathcal{R}$ and scored $T_1, \ldots, T_k$
4: **for** $j = 1, \ldots, k$ **do**
5: $S_j^{\mathcal{H}} \leftarrow \mathrm{TopK}(q_j, \mathcal{H})$ ▷ $T_j$'s top-$K$ over the history; empty if $\mathcal{H} = \emptyset$
6: $\mathcal{B} \leftarrow \bigcup_j S_j^{\mathcal{H}}$ ▷ $|\mathcal{B}| \le kK$
7: sequence $z \leftarrow [\, A \mid \mathcal{B} \mid \mathcal{R} \mid T_1 \ldots T_k \,]$; temporal indices $0 \mid 1 \ldots 1 \mid 2, 3, \ldots$
8: express all cameras relative to the first latent of $T_1$
9: mask $M$: $A, \mathcal{B}, \mathcal{R}$ attend within the context and never into $\mathcal{T}$; each $T_j$ attends to
$A$, itself, the $r$ chunks before it, and $\mathrm{TopK}(q_j, \mathcal{C}_j)$ over the candidates
$\mathcal{C}_j = \mathcal{B} \cup \{\text{chunks of } \mathcal{R} \text{ outside its window}\} \cup \{T_1 \ldots T_{j-r-1}\}$ completed before its window
10: draw a noise level $\sigma_j$ per scored chunk and replace $T_j$ in $z$ by its noised $\tilde{T}_j$ ▷ diffusion forcing
11: $\mathcal{L} \leftarrow \sum_j \left\| f_\theta(z, M)_{T_j} - v_j \right\|^2$, with $v_j$ the flow-matching target of $T_j$ ▷ loss on $\mathcal{T}$ only
12: update $\theta$ with $\nabla_\theta \mathcal{L}$

### A.2 THE SHARED BANK IS LOSSLESS AND BOUNDED

Let $S_j^{\mathcal{H}} = \mathrm{TopK}(q_j, \mathcal{H})$ be the top-$K$ of scored chunk $T_j$ over the history, so that $\mathcal{B} = \bigcup_{j=1}^{k} S_j^{\mathcal{H}}$ (Algorithm 1), and let $\mathcal{W}_j$ be the candidates already in the sequence: the recent chunks outside the window of $T_j$ and $T_1 \ldots T_{j-r-1}$. The chunk reads $S_j = \mathrm{TopK}(q_j, \mathcal{B} \cup \mathcal{W}_j)$ (Sec. 3.2).

**Proposition 1** (The shared bank is lossless). *For every $j$, $S_j = \mathrm{TopK}(q_j, \mathcal{H} \cup \mathcal{W}_j)$: reading from the shared bank selects exactly what $T_j$ would select from the whole history.*

**Algorithm 2** Rollout

**Require:** first-frame latent $x_0$; cameras $P_{0:nc}$ for $n$ chunks; constants $c, r, K$; denoising steps $S$

$\mathcal{G} \leftarrow [\, x_0 \,]$ ▷ generated latents, in temporal order
**for** $i = 1, \ldots, n$ **do**
  $\mathcal{R} \leftarrow$ the last $r$ generated chunks (fewer at the start)
  $\mathcal{B} \leftarrow \mathrm{TopK}\big(q_i, \mathcal{G} \setminus (\{x_0\} \cup \mathcal{R})\big)$ ▷ recomputed every chunk; $O(|\mathcal{G}|)$ scores
  context $[\, x_0 \mid \mathcal{B} \mid \mathcal{R} \,]$ at the positions of training; cache its keys and values
  $y \sim \mathcal{N}(0, I)$ for the new chunk, placed where training places $T_1$
  **for** $t = 1, \ldots, S$ **do**
    $y \leftarrow$ one denoising step of $f_\theta$ on $y$, attending to the cached context
  $\mathcal{G} \leftarrow [\, \mathcal{G};\, y \,]$ ▷ append the $c$ latents of $y$
**return** decoded $\mathcal{G}$

*Proof.* TopK ranks by score and breaks ties by time, a strict order, so a candidate is in the top-$K$ of a set only if fewer than $K$ members of that set precede it. If $|\mathcal{H}| < K$ then $S_j^{\mathcal{H}} = \mathcal{H} \subseteq \mathcal{B}$ and there is nothing to show. Otherwise a latent of $\mathcal{H} \setminus \mathcal{B}$ lies outside $S_j^{\mathcal{H}}$, so the $K$ members of $S_j^{\mathcal{H}} \subseteq \mathcal{B}$ all precede it and it is not in $\mathrm{TopK}\big(q_j, \mathcal{H} \cup \mathcal{W}_j\big)$. That set therefore lies in $\mathcal{B} \cup \mathcal{W}_j \subseteq \mathcal{H} \cup \mathcal{W}_j$, and the top-$K$ of a subset that contains the top-$K$ of the whole is the same set. □

The bank is also bounded. As a union of $k$ sets of at most $K$ latents, $|\mathcal{B}| \le kK$ for every history and span, so the sequence of Eq. 1 holds $L \le 1{+}kK{+}rc{+}kc$ slots and a step costs $O\big((1{+}kK{+}rc{+}kc)^2\big)$ attention at any span; through the mask each $T_j$ attends to at most $1 + K + rc + c$ positions, whatever $|\mathcal{B}|$ and $m$.

With the constants of Sec. 4.1, packing the $k$ scored chunks of a sample into one sequence costs 1.82 slots per scored latent at the measured bank size, against $(1 + K + rc + c)/c = 3.75$ when each chunk is run as its own sequence (Table 5; bank sizes from the training sampler on 20 episodes, 30 draws per span).

### A.3 Frustum Overlap

Let $P_q, P_c \in \mathrm{SE}(3)$ be the camera-to-world poses of the query chunk and a candidate, and let $G$ be a grid of points in the query's camera frame, spread over the image and over a depth range $[z_{\mathrm{near}}, z_{\mathrm{far}}]$. Then

$$\mathcal{O}(q, c) \;=\; \frac{1}{|G|} \sum_{g \in G} \mathbb{1}\big[\, \pi\big(P_c^{-1} P_q\, g\big) \in \Omega \;\wedge\; z\big(P_c^{-1} P_q\, g\big) \in [z_{\mathrm{near}}, z_{\mathrm{far}}] \,\big]\,, \tag{4}$$

where $\pi$ projects through the shared intrinsics and $\Omega$ is the image rectangle. $G$ is an $8 \times 8$ grid of cell centres over the image rectangle at 4 depths, the centres of four log-uniform bins over $[z_{\mathrm{near}}, z_{\mathrm{far}}] = [0.5, 15]$ in the units of the corpus (2 to 60 m, one unit being 4 m), so the samples lie at 0.77, 1.79, 4.19 and 9.81 units; 256 points in all. Bin centres keep every sample strictly inside the range, so a camera scores $\mathcal{O} = 1$ against itself. The depth range is the one quantity in Eq. 4 that carries a scene scale; in Eq. 2 the translation term of $\mathcal{D}$ carries one too. In $\mathcal{O}$, a candidate is credited with a query point only if that point falls inside its frustum *and* within this range of it, so $z_{\mathrm{far}}$ sets how far behind the camera's own position a candidate may be and still be said to have seen the place. App. A.4 measures what a mis-set scale costs.

### A.4 Retrieval Cues

Rules are graded as in Sec. 3.3, over 960 scored chunks from 20 episodes with each chunk's candidates thinned by an even stride to between 120 and 240 (fewer when the history is shorter), each rule's six picks against the six most similar candidates of that chunk.

**Failure Modes.** Overlap cannot separate candidates along the viewing axis: moving the camera straight forward nests the new frustum inside the old one, and overlap still reads 1.000 after 1.80 units (7.2 m) of travel.

**Scale Sensitivity.** Under a global rescaling of a scene, the translation term of Eq. 3 scales while the rotation term does not, whereas Eq. 4 is invariant if the depth range scales with the scene. Table 3 rescales translations while holding each cue's constants fixed. Over a hundredfold scale error, pose distance varies by 10.2 points and overlap, with its depth range held fixed, by 13.5. Every scene of our corpus shares metric units scaled by one factor, which removes most of this sensitivity. Shrinking translations tenfold helps pose distance, as does raising $w$, which weighs the same trade from the other side: every value from 4 to 32 lies within 1.7 points, and the best, $w = 8$ to 16, is 1.7 points above the $w = 4$ we train with, chosen on an earlier version of the corpus.

Table 3: **Sensitivity to pose scale.** Achievable gain when translations are rescaled with each cue's constants held fixed.

| Translations rescaled by | Pose distance $\mathcal{D}$ ($w = 4$) | Frustum overlap $\mathcal{O}$ |
|---|---|---|
| $0.1\times$ | **57.1%** | 39.5% |
| $1\times$ (as posed) | 55.9% | 47.4% |
| $10\times$ | 46.9% | **53.0%** |

### A.5 Retrieval Width and Bank Capacity

Table 4: **Retrieval width and bank capacity.** *Top:* each width $K$ against its own ceiling, the $K$ most similar candidates. *Bottom:* six frames read from a bank that admits frames by coverage, dropping near-duplicates, and is capped at the given size, together with the latents already completed in the chunk's window, which supply the rest when the cap is below six; $\infty$ lifts the cap but not the de-duplication, so it differs slightly from $K = 6$ above.

| Per-chunk retrieval $K$ | 1 | 2 | 3 | 4 | 6 | 8 | 12 | 16 | 24 |
|---|---|---|---|---|---|---|---|---|---|
| Achievable gain (%)↑ | 59.6 | **60.0** | 58.9 | 57.4 | 55.4 | 54.2 | 51.1 | 49.1 | 45.9 |

| Bank capacity | 4 | 6 | 10 | 16 | 24 | 40 | 64 | $\infty$ |
|---|---|---|---|---|---|---|---|---|
| Achievable gain (%)↑ | 46.5 | 46.9 | 49.1 | 49.7 | 50.4 | 51.4 | 55.0 | **55.6** |

Table 4 varies the number of frames a chunk retrieves and, separately, a hard cap on the bank, both at $\lambda = 0.2$ over 240 query chunks from 12 episodes. Capping the bank costs quality: moving from ten to sixty-four entries gains six points, and sixty-four comes within a point of an unbounded bank. Achievable gain falls slowly with width because overlap saturates at 1 for any candidate containing the query view, leaving little to order beyond the first few picks. It falls from 60.0% at $K = 2$ to 55.4% at $K = 6$ and 45.9% at $K = 24$. These percentages are on a different scale from Figure 3: each column is graded against its own ceiling, the $K$ most similar candidates, over 240 query chunks from 12 episodes, while Figure 3 grades the six retrieved frames against the six best over 960 chunks from 20 episodes. The arms of each study are comparable with each other, not across the two, which is why $K = 6$ reads 55.4% here and 57.2% there. We use $K = 6$, which keeps most of that gain while giving each chunk several frames to read.

### A.6 Settings

Each chunk draws its own noise level, and the flow-matching loss is taken on the scored chunks only. The first frame enters as a single-frame latent, as a photograph does at inference. Table 5 lists all other settings.

## B Dataset

### B.1 Training Data

**Source.** The corpus is rendered in Unreal Engine 5.8 through UnrealCV along scripted coverage walks, with frozen trajectories, no physics, fixed exposure, and no motion blur, depth of field or on-screen overlays. A *scene* is one engine map, and a *walk* is one continuous capture of up to 6.7 h at $1280 \times 720$ and 24 fps through a pinhole camera with a 90° horizontal field of view, at an eye height

Table 5: **Settings.** Everything the runs of Table 2 share; the ablation rows differ only as Sec. 4.3 describes.

| Group | Setting | Value |
|---|---|---|
| Model and data | Initialization | Wan2.2-TI2V-5B (Wan Team, 2025) |
| | Camera conditioning | PRoPE (Li et al., 2025a) |
| | Video | $864 \times 480$ at 16 fps |
| Training sample | Latents per chunk $c$ | 4 |
| | Scored chunks $k$, recent chunks $r$ | 10, 1 |
| | Span $m$ (chunks) | $U[10, m_{\max}]$, $m_{\max} = 100$ for MEMORIZON (41–401 latents) |
| | First frame | single-frame latent |
| Retrieval | Entries per chunk $K$ | 6 |
| | Score weights $\lambda$, $w$ (Eq. 2) | 0.2, 4 |
| | Overlap grid, depth range | $8 \times 8 \times 4$ points, $[0.5, 15]$ units |
| Sequence (derived) | Bank $\lvert\mathcal{B}\rvert$, bound / measured at 100 s | $kK = 60$ / $\approx 28$ |
| | Slots $L$, bound / measured at 100 s | 105 / $\approx 73$ (run median 65, Table 10) |
| | Slots each scored chunk reads | $1 + K + (r+1)c = 15$ |
| | Slots per scored latent, packed / separate | 1.82 / 3.75 |
| Optimization | Optimizer, schedule | AdamW, constant, no warmup |
| | Learning rate (backbone, camera) | $2 \times 10^{-5}$, $10^{-4}$ |
| | Gradient clipping | 1.0 |
| | Steps, batch size | 6,000, 32 (32 H200, bfloat16, FSDP) |
| | Caption dropout | 10% |
| Evaluation | Denoising steps, guidance scale | 20, 4 |
| | Negative prompt | Wan2.2 default (English) |
| | Rollout seeds, scored window | 5, first 60 s |
| | Return | within 0.5 units and 15°, $\geq$ 8 s apart, away $>$ 1.5 units or 90° between |
| | Control pair | same time gap, cameras $>$ 1 unit or 30° apart |

of 1.7 m. Each walk is cut into non-overlapping 15 min segments, the *episodes* used throughout, and a shorter remainder is dropped.

**Camera Poses.** Camera poses are the engine's ground-truth camera-to-world transforms, read per frame in metres. Each trajectory is recentered on its first frame and divided by a fixed factor of 4, so one unit is 4 m in every scene. A latent takes the pose of the central frame of its four. When the camera translates, which it does on 47% of latents, it moves a constant 0.0625 units (0.25 m, so 1 m/s) per latent; it turns by at most 7.5° per latent, pitches by up to about 29° and never rolls.

**VAE Encoding.** Video is resampled from 24 to 16 fps by nearest frame, with the same map for images and poses, and resized to $864 \times 480$ with Lanczos filtering and no crop, giving intrinsics $f_x = 432$, $f_y = 426.7$, $c_x = 432$ and $c_y = 240$. The Wan2.2-TI2V-5B VAE (Wan Team, 2025) encodes it with $16\times$ spatial and $4\times$ temporal compression into 48 channels, so a 15 min episode holds 3,600 latents. Every fourth latent also carries a caption of its frame, written by Qwen3-VL-8B and embedded with umT5, the text encoder of Wan2.2, and a single-frame encoding of that frame, so that the first frame of a training sample enters exactly as a photograph does at inference. No episode is filtered by camera motion, and the 41-latent floor of the sampler removes none.

**Split.** Training uses 380 episodes from 51 walks over 50 scenes, 95.0 h in total, frozen as a list because the corpus keeps growing. The last episode of every training scene that has more than one is held out for the seen split below (44 episodes, 11.0 h; six scenes have a single episode and keep it), and the four scenes with the shortest captures are held out entirely for the unseen split.

### B.2 REVISIT STATISTICS

**Definition.** Two latents close in pose and far apart in time are not by themselves a revisit, since a camera that pauses satisfies both conditions without going anywhere. A latent $i$ is a return to an earlier latent $j$ when the camera centres are close, $\|t_i - t_j\| \leq \tau$, and the camera left in between, $\max_{j<l<i} \|t_l - t_j\| > \varepsilon$, with both thresholds in the units of the corpus. For every returning latent we record its shortest interval, $(i - j^\star)/4$ seconds with $j^\star$ the latest qualifying $j$, since a training

window must contain both ends to supervise the return. We count every latent of the 380 training episodes, so no interval exceeds 15 min. One unit is 4 m, so $\tau = 0.5$ is 2 m and $\varepsilon = 1.5$ is 6 m.

**Why This Differs From the Evaluation Metric.** Table 6 sets the definitions side by side: the corpus count leaves out the heading because it asks whether a training window could hold both visits, not whether two frames show the same view. Adding the heading can only drop returning latents and move each partner earlier, so the intervals below are a lower bound on the supervision gap; at $\varepsilon = 3.0$ the shortest one moves from 13.25 to 28.25 s.

Table 6: **Three definitions of a return.** Distances in the units of the corpus (one unit is 4 m).

| Definition | Position | Heading | Also required | Used for |
|---|---|---|---|---|
| Corpus return (App. B.2) | $\leq \tau = 0.5$ | — | camera away $> \varepsilon = 1.5$ in between | could one window hold both visits |
| Evaluation return (App. C.1) | $\leq 0.5$ | $\leq 15°$ | $\geq 8$ s apart; away $> 1.5$ or $90°$ | Revisit and Gain |
| Fold-back (App. C.1) | an evaluation return whose path back retraces the path out within 0.5 | | | camera-following check |

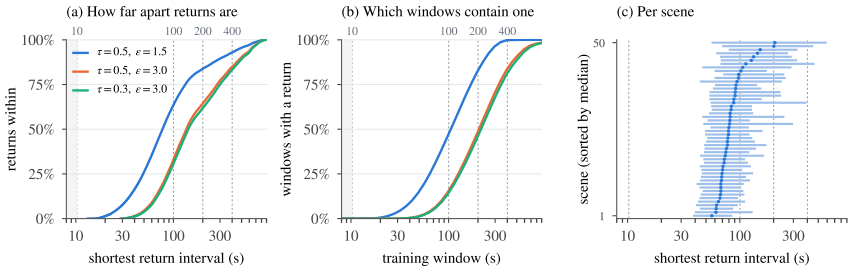


Figure 7: **Revisits in the training corpus. (a)** Shortest return interval for three definitions. **(b)** Share of training windows of each length that contain a return. **(c)** Per scene, sorted by median, at $\tau = 0.5$, $\varepsilon = 1.5$. Dashed lines mark spans of 10, 100, 200 and 400 s.

**Findings.** At $\tau = 0.5$ and $\varepsilon = 1.5$ the corpus holds 528,231 returning latents. None returns within 13.25 s, so no 10 s training window contains a return, and the median return takes 77.5 s (Figure 7a). Stricter definitions lengthen both: the shortest interval becomes 28.25 s and the median 136 s at $\varepsilon = 3.0$. A window must therefore reach well past ten seconds to contain a return: 3.8% of 30 s windows and 22% of 60 s windows contain one, against 47% of 100 s windows and nearly every 400 s window, and 64% of returns fit within 100 s, 84% within 200 s and 93% within 400 s (Figure 7b). The pattern holds across scenes, whose median intervals range from 56 to 206 s (Figure 7c).

## C Additional Experiments

We first give the remaining details of the revisit metrics of Sec. 4. The analyses that follow report the seen-scene ablation, where the memory of a model trained on 10 s clips comes from and how the gain splits by return interval, the cost of a training step, the baselines left out of Table 1, how closely each system follows the camera path, what the bank must hold at inference and why image quality falls with span; the last section is a qualitative comparison.

### C.1 Evaluation Metrics

Sec. 4.1 defines a return, Revisit and Gain; this section gives the rest of the rule. A pair $(i, j)$ counts as a return only if the camera has left in between by more than 1.5 units or 90°, and consecutive pairs from one pass are thinned to a single return, so a slow walk past a place contributes once rather than once per latent. The control pairs matched to a return at its own time gap are those whose cameras differ by more than 1 unit or 30°.

The correlation is taken over RGB values: each frame, resized to $864 \times 480$ where a system renders at another size, is flattened with its three channels into one vector, and the two vectors are centered

on their own means. This is a deliberately literal measure. It rewards a model for putting the same intensities back in the same places and is therefore sensitive to spatial misalignment, so a model that draws the right scene from a slightly wrong pose is penalized alongside one that draws the wrong scene. App. C.7 reports how closely each system follows the prescribed path, which is what separates those two failures.

The DINO column of Table 2 scores the same returns in feature space. Each frame is resized to $224 \times 224$, normalized with the ImageNet statistics and passed through DINO ViT-B/16 (Caron et al., 2021), the weights VBench uses for subject consistency; the feature is the CLS token of the final layer, L2-normalized, and the similarity of two frames is the dot product of their features. It is averaged over the returns of a video and then over videos, as for Revisit. The control pairs are the same as for Gain, so a DINO gain is the return similarity minus the control similarity; the table reports the return similarity. Because the CLS token pools over the whole frame, this measure forgives a small shift in pose that the pixel correlation penalizes.

After a few seconds of open-loop rollout a generated scene is no longer pixel-aligned with the rendered ground truth, so the PSNR and LPIPS we report against it measure how close a rollout stays to it rather than reconstruction.

We call a return $(i, j)$ a *fold-back* when the path back retraces the path out: with $\mu = (i + j)/2$, the positions over $[\mu, i]$, read backwards, follow those over $[j, \mu]$ within $0.5$ units in discrete Fréchet distance, the tolerance that already defines a return. Of the 173 returns on the web walks, 125 ($72\%$), in all 20 clips, are fold-backs, and 151 ($87\%$) leave the place by turning away by more than $90°$ rather than by walking off; at 1 unit the fold-backs are $98\%$.

### C.2 Ablation on Seen Scenes

Table 7 gives the seen split of Table 2: clips cut from held-out episodes of the training scenes with new trajectory.

Table 7: **Ablation of Table 2 on seen scenes**, held-out episodes of the training scenes along new trajectories. Same runs, protocol and columns; best in bold, second underlined.

| | Fidelity to GT | | Memory | | | VBench | | |
|---|---|---|---|---|---|---|---|---|
| Training | PSNR↑ | LPIPS↓ | Revisit↑ | Gain↑ | DINO↑ | Subj.↑ | Backg.↑ | Imag.↑ |
| *Seen scenes, new trajectories* | | | | | | | | |
| 10 s, no retrieval | 11.48 ±.18 | 0.610 ±.008 | 0.333 ±.017 | 0.074 ±.021 | 0.633 ±.013 | 0.801 ±.002 | 0.882 ±.003 | 0.648 ±.005 |
| 10 s, 10-chunk window | 11.08 ±.14 | 0.641 ±.008 | 0.321 ±.019 | 0.081 ±.017 | 0.599 ±.008 | 0.774 ±.004 | 0.875 ±.002 | 0.635 ±.005 |
| 10 s, top-$K$, no bank | 11.41 ±.28 | 0.609 ±.008 | 0.427 ±.026 | 0.186 ±.026 | 0.744 ±.011 | 0.799 ±.006 | 0.883 ±.004 | 0.654 ±.008 |
| 100 s, per-segment retrieval, bank | 11.92 ±.10 | 0.609 ±.006 | 0.454 ±.025 | 0.226 ±.031 | 0.761 ±.020 | 0.797 ±.005 | 0.879 ±.003 | 0.658 ±.008 |
| 100 s, top-$K$, ordered bank | 12.13 ±.18 | **0.603** ±.004 | 0.536 ±.017 | 0.280 ±.019 | 0.834 ±.007 | 0.800 ±.002 | 0.884 ±.002 | 0.639 ±.011 |
| 100 s, top-$K$, bank | 12.21 ±.15 | 0.619 ±.003 | 0.492 ±.021 | 0.258 ±.027 | 0.820 ±.010 | 0.804 ±.001 | 0.880 ±.002 | 0.646 ±.015 |
| 200 s, top-$K$, bank | **12.24** ±.13 | **0.603** ±.005 | **0.548** ±.006 | **0.308** ±.022 | **0.858** ±.009 | **0.806** ±.006 | **0.886** ±.003 | 0.637 ±.012 |
| 400 s, top-$K$, bank | 11.72 ±.09 | 0.621 ±.005 | 0.487 ±.015 | 0.250 ±.014 | 0.820 ±.010 | 0.801 ±.004 | 0.882 ±.004 | **0.684** ±.010 |

Table 8 repeats Table 1 with the standard deviation of every entry over the five rollout seeds.

Table 8: **Table 1 with its spreads.** Mean over five rollout seeds, with the standard deviation of the per-seed means in small type.

| | Starting pose | | | | Mid-path | | | | VBench | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Method | Revisit↑ | Gain↑ | DINO↑ | D. Gain↑ | Revisit↑ | Gain↑ | DINO↑ | D. Gain↑ | Subj.↑ | Backg.↑ | Imag.↑ |
| LingBot-World 2.0 | 0.143 ±.026 | −0.006 ±.032 | 0.526 ±.013 | −0.001 ±.029 | 0.147 ±.017 | 0.011 ±.015 | 0.546 ±.020 | 0.017 ±.019 | 0.765 ±.004 | 0.852 ±.004 | 0.741 ±.004 |
| DreamX-World 1.0 | 0.146 ±.009 | 0.033 ±.021 | 0.547 ±.016 | −0.045 ±.013 | 0.109 ±.006 | 0.009 ±.007 | 0.591 ±.013 | 0.007 ±.013 | 0.781 ±.001 | 0.868 ±.001 | 0.684 ±.006 |
| HY-WorldPlay 1.5† | 0.243 ±.014 | 0.040 ±.034 | 0.679 ±.022 | 0.025 ±.031 | 0.194 ±.011 | 0.037 ±.010 | 0.674 ±.004 | 0.027 ±.004 | **0.839** ±.002 | **0.908** ±.001 | 0.702 ±.008 |
| Matrix-Game 3.0 | 0.254 ±.018 | 0.075 ±.017 | 0.717 ±.013 | 0.042 ±.019 | 0.283 ±.012 | 0.089 ±.020 | 0.760 ±.016 | 0.100 ±.016 | 0.813 ±.004 | 0.895 ±.002 | **0.761** ±.003 |
| Infinite-World | 0.109 ±.022 | 0.019 ±.031 | 0.610 ±.026 | 0.013 ±.022 | 0.168 ±.017 | 0.070 ±.012 | 0.646 ±.013 | 0.052 ±.007 | 0.789 ±.005 | 0.856 ±.003 | 0.746 ±.006 |
| CaR | 0.350 ±.017 | 0.220 ±.048 | **0.843** ±.032 | **0.305** ±.028 | **0.450** ±.010 | 0.312 ±.035 | 0.759 ±.008 | 0.211 ±.011 | 0.787 ±.007 | 0.846 ±.003 | 0.715 ±.003 |
| **MEMORIZON (ours)** | **0.365** ±.013 | **0.230** ±.024 | 0.787 ±.017 | 0.219 ±.031 | 0.447 ±.011 | **0.329** ±.015 | **0.816** ±.020 | **0.258** ±.015 | 0.778 ±.007 | 0.857 ±.004 | 0.705 ±.012 |

### C.3 Where the 10 s Model's Memory Comes From

The 10 s top-$K$ run of Table 2 reaches most of the memory of the 100 s run: its DINO gain, the DINO similarity at a return minus that of the control pairs, which Table 2 does not list, is 0.144, 0.161 and

0.166 on seen scenes, unseen scenes and web photographs, against 0.227, 0.240 and 0.265 for the 100 s run (63% to 67%), and its pixel Gain in Tables 2 and 7 is 71% to 85% of the 100 s run's. Ten seconds is its training span, not its reach at inference. Every run retrieves from the whole generated history, with no limit on how old a candidate may be, so at inference the two runs draw on the same frames and differ only in how old the frames they were trained to read could be: on the seen clips both retrieve frames about 14.5 s old on average and up to 59 s old. A model trained on 10 s clips can use older frames because nothing in its input tells it a frame's age: retrieval is by pose, and every bank entry shares one temporal index (Sec. 3.2), so what it learns on frames less than 10 s old transfers to frames retrieved from much further back.

### C.4 Gain by Return Interval and Type

Table 9 splits the Gain by the interval between the two visits and by the kind of return. Three things follow. The 10 s top-$K$ run has a real gain at returns 20 to 60 s apart, far outside its training window, so it does use frames retrieved from long ago. What long-span training adds lies at returns more than 10 s apart: on web photographs the 100 s run leads by about 0.09 at both 10–20 s and 20–60 s, while at 8–10 s the 10 s run is slightly ahead on every split. And returns to the starting pose carry a gain even without retrieval, because the first frame is always attended, so they do not isolate long-range memory; the mid-path returns are the cleaner test.

Table 9: **Gain by return interval and type.** Pixel Gain with the five rollout seeds pooled; seen and unseen scenes as in Table 2, web photographs as in Table 1. *Start*: the earlier frame lies in the first second; *mid*: every other return. Pooling pairs rather than clips makes *all* differ slightly from Table 2. For the 100 s run the three interval bins hold 25, 140 and 470 returns on seen scenes, 30, 105 and 230 on unseen ones and 20, 100 and 605 on web photographs.

| | Interval between the visits | | | Type | | |
|---|---|---|---|---|---|---|
| Training | 8–10 s | 10–20 s | 20–60 s | Start | Mid | All |
| *Seen scenes* | | | | | | |
| 10 s, no retrieval | 0.230 | 0.064 | 0.073 | 0.160 | 0.067 | 0.078 |
| 10 s, top-$K$ | 0.397 | 0.174 | 0.185 | 0.130 | 0.199 | 0.191 |
| 100 s, top-$K$, bank | 0.337 | 0.216 | 0.253 | 0.213 | 0.252 | 0.248 |
| *Unseen scenes* | | | | | | |
| 10 s, no retrieval | 0.100 | 0.069 | 0.016 | 0.220 | 0.031 | 0.038 |
| 10 s, top-$K$ | 0.466 | 0.241 | 0.242 | 0.250 | 0.260 | 0.260 |
| 100 s, top-$K$, bank | 0.414 | 0.279 | 0.270 | 0.222 | 0.288 | 0.285 |
| *Web photographs* | | | | | | |
| 10 s, no retrieval | 0.062 | 0.015 | 0.065 | 0.135 | 0.046 | 0.058 |
| 10 s, top-$K$ | 0.515 | 0.284 | 0.221 | 0.182 | 0.247 | 0.238 |
| 100 s, top-$K$, bank | 0.458 | 0.374 | 0.307 | 0.249 | 0.332 | 0.321 |

### C.5 Training Cost

Table 10 reports the cost of the eight runs of Table 2, each trained for 6,000 steps on 32 H200 GPUs with one sample per GPU. Time per step includes data loading, retrieval, mask construction and checkpointing, and is pooled over the training segments between the supervisor's restarts; peak memory is the largest allocation on one GPU.

Table 10: **Training cost of the runs in Table 2**, from their logs. Slots: median and maximum over logged steps. Wall clock is the step time over the 6,000 steps every run trains for.

| Training | Sequence slots | Step time (s) | Wall clock (h) | Peak mem. (GB) |
|---|---|---|---|---|
| 10 s, no retrieval | 41 / 41 | 3.49 | 5.8 | 16.05 |
| 10 s, 10-chunk window | 41 / 41 | 4.02 | 6.7 | 16.05 |
| 10 s, top-$K$, no bank | 41 / 41 | 3.68 | 6.1 | 16.05 |
| 100 s, per-segment retrieval, bank | 51 / 51 | 4.17 | 6.9 | 17.99 |
| 100 s, top-$K$, ordered bank | 65 / 105 | 8.18 | 13.6 | 28.45 |
| 100 s, top-$K$, bank | 65 / 105 | 8.02 | 13.4 | 28.45 |
| 200 s, top-$K$, bank | 70 / 105 | 8.67 | 14.4 | 28.45 |
| 400 s, top-$K$, bank | 72 / 105 | 9.00 | 15.0 | 28.45 |

Adding the bank raises the cost once: the sequence grows from 41 slots to a median of 65 to 72, and time per step and peak memory rise by factors of 2.3 and 1.8 over the sliding window. Beyond

that, the span matters little. Quadrupling the maximum span from 100 to 400 s leaves the sequence at its bound of 105 slots and peak memory unchanged at 28.45 GB, and adds 12% to the time per step, spent on reading and ranking a longer history and on a median sequence seven slots longer. For comparison, attending to every earlier chunk of a 10 s window already costs 15% more per step than the sliding window. A 400 s window trained directly would hold 1,601 latents, 15× the largest sequence here. Sharing one bank across the scored block instead of retrieving per chunk is the one arrangement that is cheaper than ours, at 51 slots and 4.17 s a step, and it is the arm that loses most of the memory (Sec. 4.3).

### C.6 Baselines Not in Table 1

The two systems closest to ours are absent from Table 1 for different reasons. WorldMem (Xiao et al., 2025) releases its code and weights, but it is a Minecraft model driven by a discrete action vocabulary: it cannot be given a photograph and a continuous camera path, and a version retrained on our corpus would no longer be the system its numbers describe. Context-as-Memory (Yu et al., 2025) has released its dataset but neither its code nor its weights, so it cannot be run. What can be compared is its design: one row of Table 2 retrieves once for the whole scored block instead of once per chunk, so every chunk reads the same frames, as in Context-as-Memory, with everything else as in our runs.

### C.7 Camera Following

A model that ignored the camera path could score well on Revisit by drawing the same view all the time, so we check how closely each system turns when the path turns. We avoid pose reconstruction, which would fail on some clips, and measure the image instead: for every quarter second we estimate the horizontal shift between consecutive frames by phase correlation, and compare it with the yaw the path prescribes. *Correlation* is the Pearson correlation between the prescribed yaw change and the measured shift, and *Direction* the fraction of steps whose image shifts the way the path turns; neither depends on a system's resolution, frame rate or field of view. The implied field of view reads image shift per degree of yaw as a pinhole focal length. The top row applies the same measurement to the rendered frames of the seen split along their engine poses, which move the way the web walks do.

Table 11: **Camera following**: yaw only, web photographs, first 60 s, rollout seed 0. †Field of view implied by image shift per degree of prescribed yaw; the paths assume 90°. MEMORIZON: mean over six runs of Table 2, the three trained on 10 s clips and the 100, 200 and 400 s runs.

| System | Correlation↑ | Direction↑ | Implied hfov† |
|---|---|---|---|
| *Calibration* | | | |
| Rendered frames, engine poses | 0.906 | 0.974 | 89.1° |
| *Open world models* | | | |
| LingBot-World 2.0 | 0.855 | 0.966 | 73.3° |
| DreamX-World 1.0 | 0.825 | 0.955 | 132.5° |
| HY-WorldPlay 1.5 | 0.697 | 0.909 | 123.6° |
| Matrix-Game 3.0 | 0.849 | 0.968 | 98.4° |
| Infinite-World | 0.877 | 0.958 | 107.5° |
| **MEMORIZON** (mean of 6 runs) | 0.961 | 0.986 | 90.6° |

Every system turns the right way almost always, so direction separates little. Our runs track the prescribed yaw with a mean correlation of 0.961, 0.95 to 0.97 across the six runs, and realize the turns at about the scale the paths assume, with a mean implied field of view of 90.6° against 89° for rendered frames. The open world models follow the path less closely, with correlations of 0.70 to 0.88, and read the prescribed yaw at their own scale: LingBot-World 2.0 turns as though its camera were narrower than the paths assume, 73°, and DreamX-World 1.0 and HY-WorldPlay 1.5 as though it were far wider, 133° and 124°. HY-WorldPlay 1.5 follows least closely of all, so part of its low Revisit may be path following rather than memory; the others turn the right way on 95% of steps or more. Systems can therefore differ in the scale at which they realize the prescribed motion, which the last column records. That difference matters little for most returns, which retrace the path out (App. C.1): a model that applies a roughly consistent gain to its turns and steps comes back close to the pose of its first visit, and a lower Revisit there reflects what it draws rather than where it stands.

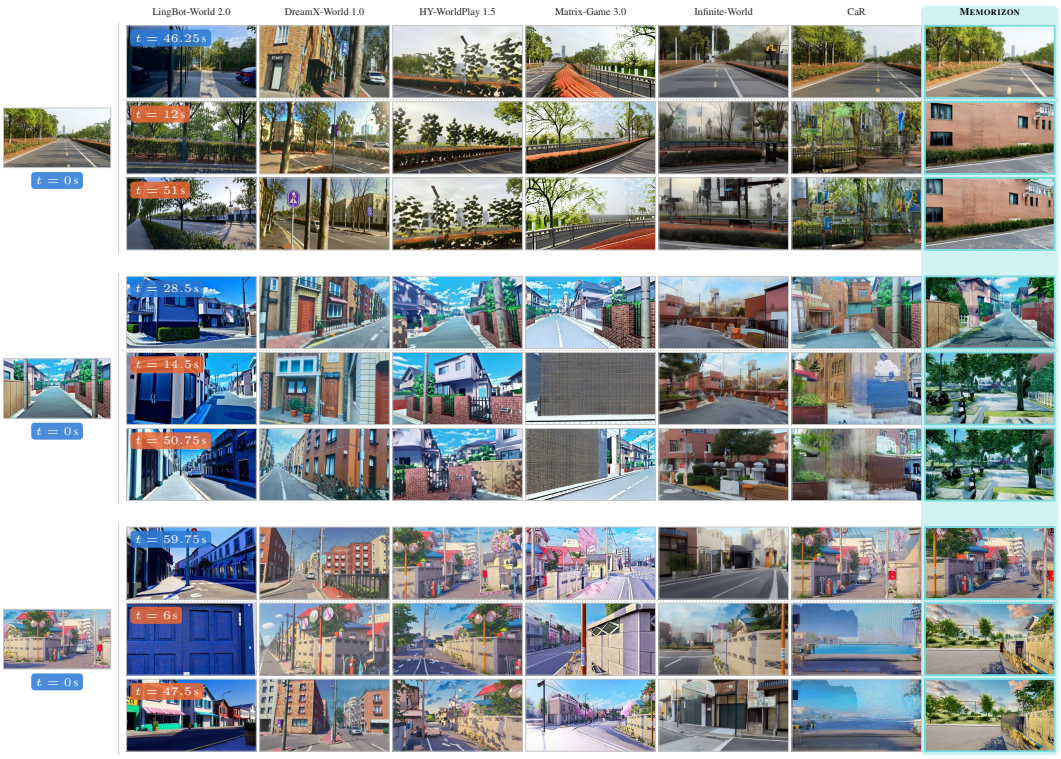


Figure 8: **More returns from open world models**, extending Figure 4 to three more web photographs. Each block follows one photograph's walk. Blue tags mark returns to the starting pose; orange tags the two ends of a return to a pose met mid-path. Rollout seed 0.

### C.8 BANK CONTENTS

An unused retrieval mechanism is the standing failure mode in this area, so we test use directly. Holding the checkpoint, trajectory and seed fixed, we vary *only the contents of the bank's slots* at inference and keep everything else in the sequence as trained. Each arm answers a different question. If the bank were ignored, filling it from *another episode* would change nothing; if it were read but unnecessary, *emptying* it would cost nothing; and if what the criterion selects mattered only as extra frames, $K$ *random* frames from the same history would do as well as the retrieved ones.

Table 12: **Bank contents at inference.** The 100 s run with only the bank's frames replaced: its top-$K$ (ours), nothing, $K$ random frames from the same history, or $K$ frames from another episode. Web photographs, first 60 s, returns split as in Table 1, DINO columns as there; rollout seed 0.

| | Starting pose | | | | Mid-path | | | |
|---|---|---|---|---|---|---|---|---|
| Bank contents | Revisit↑ | Gain↑ | DINO↑ | D. Gain↑ | Revisit↑ | Gain↑ | DINO↑ | D. Gain↑ |
| Top-$K$ (ours) | **0.377** | **0.213** | **0.794** | **0.218** | **0.440** | **0.319** | **0.826** | **0.256** |
| Empty | 0.176 | 0.053 | 0.553 | 0.085 | 0.105 | 0.019 | 0.504 | 0.035 |
| $K$ random frames | 0.153 | 0.055 | 0.583 | 0.044 | 0.207 | 0.109 | 0.634 | 0.087 |
| $K$ frames, another episode | 0.071 | 0.012 | 0.445 | −0.020 | 0.073 | 0.016 | 0.459 | −0.014 |

Table 12 answers all three, on the returns that need the bank. Mid-path Revisit falls from 0.440 with the retrieved bank to 0.207 with random frames from the same history, 0.105 with an empty bank and 0.073 with frames from another episode, and Gain from 0.319 to 0.109, 0.019 and 0.016. A bank from the wrong episode leaves the model no better off than an empty one and far below the retrieved bank, so the model reads what the bank holds rather than treating it as filler. Retrieved frames also beat random ones from the same history by more than a factor of two on both columns, so what the criterion selects matters beyond the number of frames it supplies.




Figure 9: **More returns from MEMORIZON** on six web photographs. Blue: a pose close to the first frame's. Orange and green: two returns, each shown at its first visit and on coming back; they meet the Revisit rule of Sec. 4.1, or, where a path has only one such return, the same rule at twice its tolerance.

### C.9 WHY IMAGE QUALITY FALLS WITH SPAN

Table 13 replaces what a rollout conditions on, all else as in Sec. 4.1. At inference every slot but the first frame is generated; in training all are clean. Given a clean history, every model from 10 to 200 s comes within about a point of the ground-truth videos' imaging quality, so the loss comes from conditioning on generated frames. Every model pays it, two to three times more on unseen scenes, and only the 200 s run loses clearly more. With a bank, most of it comes through the bank: a ground-truth bank alone recovers three quarters to four fifths of the gap on unseen scenes. Guidance is not the cause: without it, at seed 0, the 200 s run still scores below 10 s top-$K$.

Table 13: **Quality against what fills the context** on seen and unseen scenes, first 60 s, five seeds. *Generated*: ordinary rollout. *GT bank*: the bank holds ground-truth latents. *GT context*: bank and recent window both ground truth. The ground-truth videos themselves score 0.678 and 0.803 on seen scenes and 0.719 and 0.774 on unseen ones. With a clean history each chunk generates only four latents, so the seed spread rounds to zero.

| | Imaging↑ | | | Subject↑ | | |
|---|---|---|---|---|---|---|
| Training | Generated | GT bank | GT context | Generated | GT bank | GT context |
| *Seen scenes* | | | | | | |
| 10 s, no retrieval | 0.648 ±.005 | – | 0.671 ±.001 | 0.801 ±.002 | – | 0.820 ±.000 |
| 10 s, top-$K$, no bank in training | 0.654 ±.008 | 0.663 ±.004 | 0.673 ±.000 | 0.799 ±.006 | 0.811 ±.002 | 0.824 ±.000 |
| 100 s, top-$K$, bank | 0.646 ±.015 | 0.666 ±.004 | 0.668 ±.000 | 0.804 ±.001 | 0.818 ±.002 | 0.820 ±.000 |
| 200 s, top-$K$, bank | 0.637 ±.012 | 0.656 ±.003 | 0.666 ±.000 | 0.806 ±.006 | 0.820 ±.001 | 0.821 ±.000 |
| *Unseen scenes* | | | | | | |
| 10 s, no retrieval | 0.666 ±.014 | – | 0.720 ±.001 | 0.751 ±.006 | – | 0.803 ±.000 |
| 10 s, top-$K$, no bank in training | 0.675 ±.016 | 0.704 ±.003 | 0.721 ±.001 | 0.757 ±.003 | 0.780 ±.005 | 0.809 ±.000 |
| 100 s, top-$K$, bank | 0.668 ±.013 | 0.706 ±.000 | 0.714 ±.001 | 0.772 ±.006 | 0.801 ±.004 | 0.805 ±.000 |
| 200 s, top-$K$, bank | 0.627 ±.013 | 0.689 ±.007 | 0.710 ±.001 | 0.776 ±.008 | 0.801 ±.002 | 0.806 ±.000 |

### C.10 ADDITIONAL VISUAL COMPARISONS

Figure 8 extends Figure 4 to three more web photographs; Figure 9 shows more returns from MEMORIZON alone.