WORLD TURING MACHINE: LEARNING READ–WRITE MEMORY FOR 3D WORLDS

Anonymous authors
Paper under double-blind review

ABSTRACT

Spatial world modeling requires collecting evidence across viewpoints and maintaining it in a persistent state for predictions at unobserved views. However, organizing and accessing such accumulated information over time remains a central challenge. In this study, we introduce the **World Turing Machine (WTM)**, which addresses this through learned read–write computation over an Address–Content memory. This memory evolves with observations while the learned read–write operations remain fixed at inference. Before each write, the Reader predicts incoming visual features from the current memory and the condition. The Writer reuses the Reader's access weights to correct Content with prediction residuals and update Address with observation-derived geometric evidence. Local and global predictive objectives train acquisition, retention, and inference across viewpoints without directly supervising memory. The same Reader supports forward prediction at unvisited poses and inverse pose matching against target-frame features, without additional localization head. On ScanNet, **WTM** outperforms evaluated novel-view synthesis and feed-forward 3D reconstruction baselines in held-out feature prediction. With a simple search over candidate poses, WTM achieves **65.7%** success on ViewSuite tasks without task-specific policy training, surpassing ViewAgent by **+13.1** percentage points. Analyses reveal emergent spatial organization during online memory formation and show that the learned read–write mechanism generalizes across 3D environments.

1 INTRODUCTION

Spatial world modeling aims to maintain an internal state of an environment from visual observations and use it to predict representations under arbitrary viewpoints. As each observation reveals only part of the environment, the state must retain earlier evidence and incorporate new observations as they arrive. The central challenge lies in learning to organize, access, and revise the evolving state so that it remains useful for future queries. Neural Turing Machines (NTMs) (Graves et al., 2014; Gulcehre et al., 2017) provide a natural abstraction to address this challenge: an explicit Address–Content memory manipulated through learned read and write operations.

We adapt this abstraction for spatial world modeling in two ways. First, we make the memory's Address an evolving scene-specific state, rather than a fixed indexing space. The Address is revised by geometric evidence accumulated across observations at inference, allowing memory organization to adapt to the underlying spatial structure of the scene. Second, we tie memory access and revision through latent view prediction, so that the same learned access mechanism governs both querying the current world state and incorporating newly observed evidence.

Building on these principles, we introduce the World Turing Machine (WTM), illustrated in Fig. 1, a neural architecture with jointly trained Reader and Writer to construct, revise, and query an evolving world memory. WTM maintains a fixed-capacity *Address–Content memory* composed of slots, each containing a pair of vectors: Content stores latent evidence, while Address organizes how that evidence is accessed. The learned Reader and Writer operate on this scene-specific memory during inference, revising its state without parameter updates. Rather than simply storing observations under their viewpoints for retrieval, WTM integrates complementary evidence across viewpoints into a shared scene state that supports queries beyond visited viewpoints.

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.

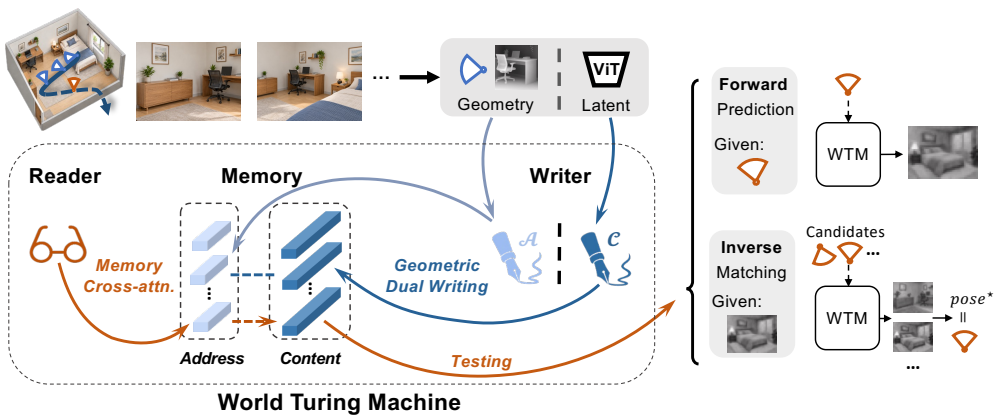

Figure 1: **World Turing Machine (WTM).** The Writer incorporates each observation into memory by updating Address and Content with geometric and visual evidence, respectively. The Reader queries the memory to predict visual features at arbitrary viewpoints. The same interface naturally supports inverse matching by evaluating predictions over candidate poses.

Before each write, the Reader first predicts features corresponding to the observation's camera pose from the memory. The Writer then follows a prediction–correction mechanism, reusing the Reader's access weights to route prediction residuals into Content corrections and geometric evidence into Address adaptations. The same Reader supports both forward feature prediction at arbitrary poses and inverse viewpoint matching through search over candidate poses, without a localization head.

An effective write must make new evidence useful across viewpoints while preserving earlier knowledge. Yet the memory state has no direct supervision. We therefore jointly train the Reader and Writer through **local** and **global** predictions from updated memory. Local queries at neighboring poses encourage new evidence to support nearby viewpoint changes; global queries at previously visited and unvisited poses encourage retention and prediction from accumulated observations.

On ScanNet (Dai et al., 2017), the pose-conditioned **WTM** predicts held-out visual features given sparse observations, outperforming the evaluated novel-view synthesis, structure-from-motion, and feed-forward 3D reconstruction baselines. For spatial decision-making, we evaluate WTM on ViewSuite IVP, where an agent seeks to localize a target view in 6-DoF space (Wang et al., 2026b). WTM achieves **48.5%** success via inverse view matching over candidate poses without task-specific policy training, compared with 47.8% for the task-trained ViewAgent. Across all ViewSuite tasks, WTM reaches an average score of **65.7%**, **13.1** percentage points above ViewAgent. Further analyses show that learned memory routing develops a clear spatial organization and that WTM transfers to other datasets. The same read–write architecture also extends to pose–time queries in dynamic scenes.

In a nutshell, the main contributions of this study include:

(i) World Turing Machine. We introduce a neural read–write architecture that maintains a persistent world memory, where predictive reads guide residual Content correction and geometric Address adaptation during inference.

(ii) Learning persistent world memory. We introduce local and global predictive objectives that train memory acquisition, retention, and prediction without direct supervision of its internal state.

(iii) World memory for spatial decision making. We show that the same Reader supports both forward view prediction and inverse viewpoint matching, enabling spatial decision making through search over candidate poses without task-specific policy training.

2 Related Work

3D Reconstruction and View Prediction. This encompasses geometric reconstruction and novel-view synthesis. SfM and SLAM methods (Duisterhof et al., 2025; Campos et al., 2021) recover camera poses and geometric scene models, while learned systems, such as DUSt3R and VGGT(Wang et al., 2024; 2025a; 2026a), infer scene structure and cameras from input images. Novel view synthesis has similarly moved toward feed-forward prediction from sparse observations, spanning gener-

alizable NeRFs, Gaussian-splatting methods, and Transformer-based formulations (Yu et al., 2021; Charatan et al., 2024; Chen et al., 2024; Jin et al., 2025). In a different way, WTM predicts visual embeddings, providing a shared feature memory for scene prediction and view matching rather than directly targeting dense geometry or RGB appearance.

World Models and Persistent Scene Memory. These models learn predictive representations for inference and control. The JEPA family (Assran et al., 2023; 2025; Mur-Labadia et al., 2026) realizes this through target prediction directly in latent space. For 3D scenes, GQN (Eslami et al., 2018) conditions novel-view generation on an aggregated scene representation, while CUT3R (Wang et al., 2025b) maintains a recurrent state for geometry predictions. Recently, Spa3R (Jiang et al., 2026) learns a compact spatial representation by predicting feature fields at unseen viewpoints, and neural map methods (Fraccaro et al., 2018; Parisotto & Salakhutdinov, 2018; Zhang et al., 2020) combine prediction with spatial memory. More recent approaches, such as LaCT (Zhang et al., 2026c) and FSM (Ma et al., 2026), explore online state update for scene representation. Theory of Space (Zhang et al., 2026b) and ViewSuite (Wang et al., 2026b) further examine the use of acquired evidence for cross-view reasoning. WTM differs in the prediction–correction mechanism: the memory access predicting an observation also guides how its evidence is incorporated.

Neural Memory and Learned Updates. Neural Turing Machines (Graves et al., 2014) learn differentiable read and write operations over explicit memory. D-NTM (Gulcehre et al., 2017) further separates learned address vectors from writable content. Token Turing Machines (Ryoo et al., 2023) extend bounded-memory read–write computation to sequential visual processing, while ViTTM (Jajal et al., 2025) adapts token-based memory computation to Vision Transformers. Fast-weight associative models offer a complementary formulation, using delta-rule updates to correct stored key–value associations (Schlag et al., 2021; Yang et al., 2024). Test-time training (Sun et al., 2024) updates a model using a self-supervised objective on test inputs. These methods typically update key–value or Address–Content associations derived from the current sequential token input. WTM instead revises its Address with geometric evidence, organizing persistent visual Content for spatial queries, thus coupling such learned memory operations to the 3D world.

3 METHODOLOGY

We introduce the *World Turing Machine* (WTM) to maintain a compact world representation from partial observations (Fig. 2). We first outline its components and interface, then describe the read–write mechanisms and training objectives.

3.1 PRELIMINARIES: MODEL AND OBSERVATIONS

Model overview. Building on the read–write memory abstraction of Neural Turing Machines (NTMs), WTM combines a persistent memory state with two learnable neural modules:

- **Persistent World Memory** maintains a compact state $\mathcal{M}_t = (\boldsymbol{A}_t, \boldsymbol{C}_t)$ before incorporating observation t, where Address $\boldsymbol{A}_t$ describes how stored information is accessed and Content $\boldsymbol{C}_t$ retains accumulated visual information.
- **Reader** learns to query $\mathcal{M}_t$ under a condition ξ_q and predict visual features $\widehat{\boldsymbol{Y}}_q$.
- **Writer** learns to incorporate incoming features $\boldsymbol{Y}_t$ and condition ξ_t by updating $\boldsymbol{A}_t$ and $\boldsymbol{C}_t$, producing $\mathcal{M}_{t+1} = (\boldsymbol{A}_{t+1}, \boldsymbol{C}_{t+1})$ while retaining useful past information.

Here, t and q index steps and queries, respectively. Together, these components form a continual prediction–correction process, with each write updating the memory available to subsequent reads.

Shared feature space. To connect memory updates with view prediction, the Writer's visual evidence $\boldsymbol{Y}_t$ and the Reader's output $\widehat{\boldsymbol{Y}}_q$ use the same feature space. For an image $\boldsymbol{I}_t$ acquired under condition ξ_t, a frozen teacher encoder Φ, DINOv2 (Oquab et al., 2024), produces

$$\boldsymbol{Y}_t = \Phi(\boldsymbol{I}_t) \in \mathbb{R}^{N \times d_y}, \tag{1}$$

where N is the number of patch tokens in an image and d_y is their feature dimension. Our primary setting uses camera pose as the condition, $\xi_t = P_t$.

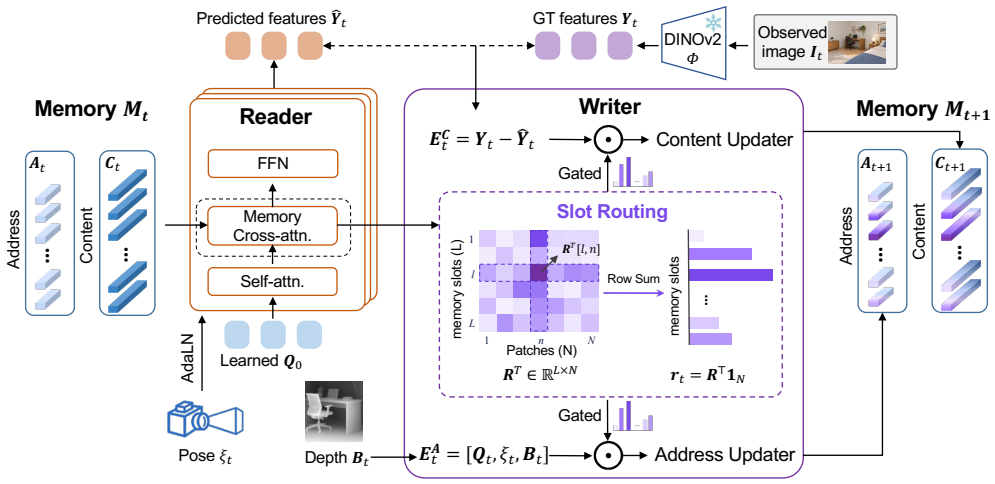

Figure 2: **WTM diagram**. At step t, when the observation I_t arrives, the Reader first queries $\mathcal{M}_t$ under condition ξ_t and predicts target features $\hat{\boldsymbol{Y}}_t$. The Writer then uses the observation at time t and routing weights from the Reader to revise the memory into $\mathcal{M}_{t+1}$.

3.2 Persistent World Memory

To implement the memory state $\mathcal{M}_t = (\boldsymbol{A}_t, \boldsymbol{C}_t)$ within a fixed memory budget, we organize Address and Content into L persistent *Address–Content slots*, as

$$\mathcal{M}_t = (\boldsymbol{A}_t, \boldsymbol{C}_t) = \{(\boldsymbol{A}_t[l], \boldsymbol{C}_t[l])\}_{l=1}^{L}, \tag{2}$$

where $\boldsymbol{A}_t \in \mathbb{R}^{L \times d_A}$ contains descriptors for accessing the stored information, and $\boldsymbol{C}_t \in \mathbb{R}^{L \times d_C}$ contains the accumulated visual content. We use $\boldsymbol{A}_t[l]$ and $\boldsymbol{C}_t[l]$ to denote the l-th slot. This separation allows WTM to adapt both how information is accessed and what information is retained. At the start of each episode, Address is initialized from fixed orthogonal codes and Content to zero. The Writer then updates both states as observations arrive.

3.3 Reader

To retrieve visual information under a desired condition, the Reader takes memory $\mathcal{M}_t$ and a query condition ξ_q as input and predicts

$$\hat{\boldsymbol{Y}}_q = \mathcal{R}_{\theta_R}(\mathcal{M}_t, \xi_q), \tag{3}$$

where θ_R denotes learned Reader parameters and $\hat{\boldsymbol{Y}}_q \in \mathbb{R}^{N \times d_y}$. Reading requires no image at the query condition and leaves memory unchanged. The Reader starts from a learned grid $\boldsymbol{Q}_0 \in \mathbb{R}^{N \times d}$, with one token per output patch. A Transformer decoder (Vaswani et al., 2017) transforms these tokens through self-attention, memory cross-attention, and an FFN in each block.

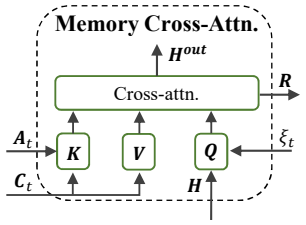

Figure 3: Memory Cross-Attention in the Reader.

To query the memory at the target view, a condition encoder produces $\boldsymbol{e}_q = \psi_{\text{cond}}(\xi_q)$, which modulates each block through AdaLN (Peebles & Xie, 2023).

Memory cross-attention (Fig. 3) then retrieves information using both access descriptors and stored features. For query states $\boldsymbol{H} \in \mathbb{R}^{N \times d}$ entering a cross-attention layer, we form

$$\begin{aligned} \boldsymbol{Q}^A &= \boldsymbol{H}\boldsymbol{W}_Q^A, & \boldsymbol{K}^A &= \boldsymbol{A}_t\boldsymbol{W}_K^A, \\ \boldsymbol{Q}^C &= \boldsymbol{H}\boldsymbol{W}_Q^C, & \boldsymbol{K}^C &= \boldsymbol{C}_t\boldsymbol{W}_K^C, & \boldsymbol{V}^C &= \boldsymbol{C}_t\boldsymbol{W}_V^C, \end{aligned} \tag{4}$$

where the $\boldsymbol{W}$ matrices are learned projections; block and head indices are omitted. Address and Content similarity scores are added before normalization:

$$\boldsymbol{R} = \text{softmax}_{\text{slot}}\left(\frac{\boldsymbol{Q}^A(\boldsymbol{K}^A)^\top + \boldsymbol{Q}^C(\boldsymbol{K}^C)^\top}{\sqrt{d_h}}\right), \tag{5}$$

where d_h is the per-head dimension and $\boldsymbol{R} \in \mathbb{R}^{N\times L}$ assigns each query token weights over the L slots. The readout uses only Content values, as

$$\boldsymbol{H}^{\text{out}} = (\boldsymbol{R}\boldsymbol{V}^{C})\boldsymbol{W}_{O}, \tag{6}$$

with learned output projection $\boldsymbol{W}_O$. Address and Content jointly determine *where* to read, while Content supplies *what* is retrieved. The outputs are projected to the feature space to produce $\widehat{\boldsymbol{Y}}_q$.

3.4 WRITER

To incorporate new evidence, the Writer uses the observation $O_t = (\boldsymbol{Y}_t, \xi_t, \boldsymbol{B}_t)$ to update memory:

$$\mathcal{M}_{t+1} = \mathcal{W}_{\theta_W}(\mathcal{M}_t, O_t), \tag{7}$$

where θ_W denotes learned parameters and $\boldsymbol{B}_t$ is optional evidence. ($\boldsymbol{B}_t = \varnothing$ when unused). In our implementation, $\boldsymbol{B}_t$ can be metric depth estimated by a monocular depth model.

WTM first obtains the pre-write prediction $\widehat{\boldsymbol{Y}}_t^{-} = \mathcal{R}_{\boldsymbol{\theta}_R}(\mathcal{M}_t, \xi_t)$ and its final-layer routing $\boldsymbol{R}_t$. To update the memory, both Content and Address aggregate their patch evidence $\boldsymbol{E} \in \mathbb{R}^{N\times D}$ through

$$\text{Route}_t(\boldsymbol{E}) = \text{Diag}(\boldsymbol{r}_t + \boldsymbol{\epsilon})^{-1}\boldsymbol{R}_t^{\top}\boldsymbol{E}, \tag{8}$$

where $\boldsymbol{r}_t = \boldsymbol{R}_t^{\top}\mathbf{1}_N \in \mathbb{R}^L$ is the per-slot routing mass used for normalization, and $\boldsymbol{\epsilon} \in \mathbb{R}^L_{>0}$ is a small positive vector for numerical stability; D denotes the feature dimension.

Content Update. To update Content based on visual prediction errors, the Writer routes the residual and applies a gated update:

$$\boldsymbol{Z}_t^{C} = \text{Route}_t(\boldsymbol{Y}_t - \widehat{\boldsymbol{Y}}_t^{-}), \tag{9}$$

$$\boldsymbol{d}_t^{C} = f_{\text{dir}}(\boldsymbol{Z}_t^{C}), \qquad \boldsymbol{\alpha}_t^{C} = f_{\text{gate}}(\boldsymbol{r}_t, \boldsymbol{C}_t, \boldsymbol{Z}_t^{C}), \tag{10}$$

$$\Delta\boldsymbol{C}_t = \boldsymbol{\alpha}_t^{C} \odot \boldsymbol{d}_t^{C}, \qquad \boldsymbol{C}_{t+1} = \boldsymbol{C}_t + \Delta\boldsymbol{C}_t. \tag{11}$$

The learned MLPs f_{dir} and f_{gate} produce the update direction and its strength, both in $\mathbb{R}^{L\times d_C}$.

Address update. To adapt Address, the Writer combines the pre-write Address-side queries $\boldsymbol{Q}_t^{A}$, which provide the current access signal, with the observation condition and auxiliary evidence

$$\boldsymbol{Z}_t^{A} = \text{Route}_t\big(\psi_{\text{evid}}(\boldsymbol{Q}_t^{A}, \xi_t, \boldsymbol{B}_t)\big), \tag{12}$$

where ψ_{evid} is a learned encoder. In the pose-conditioned setting, $\boldsymbol{B}_t$ may include pseudo-depth to provide geometric cues. A learned projection proposes $\boldsymbol{\Delta}_t^{A} = f_A(\boldsymbol{Z}_t^{A})$ for adapting Address. To change the Address direction while preserving its norm, we project this displacement to the tangent space and apply a spherical update. With $\widehat{\boldsymbol{A}}_t = \boldsymbol{A}_t/\|\boldsymbol{A}_t\|_2$ and all subsequent operations slot-wise,

$$\boldsymbol{\Delta}_t^{A,\perp} = \boldsymbol{\Delta}_t^{A} - \langle\boldsymbol{\Delta}_t^{A}, \widehat{\boldsymbol{A}}_t\rangle\widehat{\boldsymbol{A}}_t, \tag{13}$$

$$\boldsymbol{A}_{t+1} = \text{SphericalInterp}\Big(\boldsymbol{A}_t, \boldsymbol{A}_t + \boldsymbol{\Delta}_t^{A,\perp}; \beta_t^{A}\Big), \tag{14}$$

where β_t^{A} is a mass-adaptive update rate that increases with routing mass $\boldsymbol{r}_t$, and SphericalInterp interpolates toward the proposed direction while preserving the fixed radius.

3.5 TRAINING OBJECTIVES

The memory state has no direct ground-truth target. To learn how to organize, access, and update it, we train the Reader and Writer through predictions from the post-write state $\mathcal{M}_{t+1}$. For each supervised query q, the target is $\boldsymbol{Y}_q = \Phi(\boldsymbol{I}_q)$. In the primary pose-conditioned setting, where $\xi_q = P_q$, we use two groups of objectives.

Local supervision. To make newly acquired evidence useful across nearby viewpoints, we query a neighboring-view set $\mathcal{Q}_t^{\text{nbr}}$ around the current observation. This *neighboring-view prediction* objective encourages WTM to incorporate the observation and learn local viewpoint transformations.

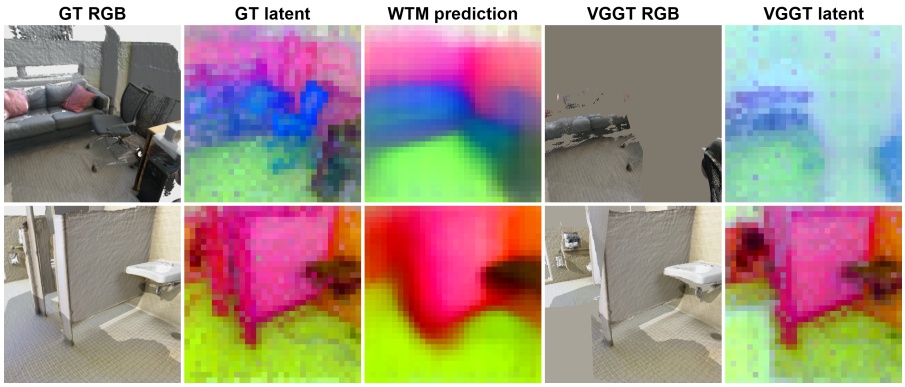

Figure 4: **Qualitative view prediction results.** With limited information about the queried viewpoint, VGGT produces incomplete or misaligned renderings. In contrast, WTM directly predicts latent features which preserve the spatial structure. Latent features are visualized using PCA.

Global supervision. To keep memory useful beyond the latest observation, we supervise *visited reconstruction* and *unvisited-view extrapolation*. The replay set $\mathcal{Q}_t^{\text{rep}}$ contains previously visited views and encourages retention of earlier information. The unvisited set $\mathcal{Q}_t^{\text{unv}}$ requires prediction from accumulated evidence at views not yet written to memory.

Using teacher-feature MSE ℓ_{MSE}, we define, for $X \in \{nbr, rep, unv\}$,

$$\mathcal{L}_X = \mathbb{E}_{q \in \mathcal{Q}_t^X} \left[\ell_{\text{MSE}}(\mathcal{R}_{\theta_R}(\mathcal{W}_{\theta_W}(\mathcal{M}_t, O_t), P_q), \boldsymbol{Y}_q) \right]. \tag{15}$$

To discourage unnecessarily large Content corrections, we also regularize the applied update with $\mathcal{L}_{\Delta C} = \|\Delta \boldsymbol{C}_t\|_F^2/(Ld_C)$. Together, the complete objective of WTM is defined as

$$\mathcal{L} = \underbrace{\mathcal{L}_{\text{nbr}}}_{\text{local}} + \underbrace{\lambda_r \mathcal{L}_{\text{rep}} + \lambda_u \mathcal{L}_{\text{unv}}}_{\text{global}} + \lambda_d \mathcal{L}_{\Delta C}, \tag{16}$$

where λ_r, λ_u, and λ_d are loss weights to balance the contributions of the loss terms.

4 Experiment

4.1 Training Pose-Conditioned WTM

Training setup. As the primary instantiation of WTM, WTM-Pose is trained on camera trajectories rendered from ScanNet meshes, detailed in Sec. A.2.

View prediction and retrieval. To evaluate viewpoint feature prediction given sparse observations, we construct a benchmark on ScanNet test scenes, detailed in Sec. A.3. The WTM memory is built upon $K = 4$ observations, while query viewpoints are sampled to be separated from the inputs, reducing near-view interpolation and emphasizing prediction across different viewpoints. In Table 1, we report *Cos. Sim.* in the teacher embedding space and *top-1 retrieval accuracy*. Retrieval is performed in all held-out views of the scene using Cos. Sim. It measures relative ranking and reveals representation collapse: scene-average predictions may attain similarity but fail to identify the correct target view.

Table 1: View prediction and Retrieval.

Method	**Cos. Sim.** ↑	**R@1** ↑
Nearest Observed†	0.291	1.35
Novel View Synthesis		
LVSM	0.271	20.95
FSM-LVSM-3D(TTT)‡	0.276	22.62
SfM		
MASt3R-SfM	0.228	10.04
Feed-Forward 3D		
VGGT	0.262	19.28
VGGT-Ω	0.273	24.36
CUT3R	0.205	10.91
WTM-Pose (Ours)	**0.528**	**26.91**

† Using the nearest observed input view as prediction.
‡ Test-time training.

WTM outperforms NVS, SfM and feed-forward 3D reconstruction baselines on both metrics. For a fair comparison, all methods are given the same number of posed observations. For methods

Table 2: **Effect of training objectives.** Local Acq. Δ measures the cosine similarity gain on nearby views after a single write from initialized memory, reflecting instantaneous memory acquisition. Generalization Cos. Sim. is evaluated by splitting each scene into two spatially separated view sets, building memory from one set and predicting the other to measure generalization to unvisited views. This table uses a different evaluation setting from Table 1; see Sec. A.3 for details.

Training objective	Cos. Sim. ↑	R@1 ↑	Local Acq. Δ ↑	Generalization Cos. Sim. ↑
Local + Global (Full)	**0.662**	**86.1**	0.199	**0.578**
Global only	0.531	16.3	0.071	0.508
Local only	0.611	53.4	0.221	0.522
Local + Replay	0.649	77.3	**0.233**	0.554
Local + Unvisited	0.658	83.3	0.176	**0.578**

that produce RGB outputs, we encode their predictions with the same teacher model before evaluation. Further results and discussions on this feature-space evaluation are provided in Sec. B.1 and Sec. C.2. As shown in Fig. 4, under this setting, reconstruction-based methods produce incomplete views in unobserved regions (row 1) and misaligned views when multi-view fusion fails (row 2). In contrast, WTM preserves the scene's spatial structure and predicts features for unobserved query poses. These results suggest that WTM learns a persistent scene memory rather than reconstructing views solely from observed points, which enables it to infer representations for unexplored regions.

Training objectives. We ablate the two groups of objectives introduced in Sec. 3.5: **Local** supervision from neighboring-view prediction and **Global** supervision from replayed and unvisited views. We renormalize the loss coefficients to keep the overall supervision strength fixed when individual terms are removed. As shown in Table 2, removing Local supervision severely degrades both local acquisition and overall prediction, while a model trained with Local supervision alone remains substantially below the full model. This suggests that Local supervision provides the foundation for incorporating new observations and transferring their evidence across viewpoints.

Replay and unvisited supervision impose different prediction targets, but share a higher-level role. Replay requires previously acquired evidence to remain accessible after subsequent writes, whereas unvisited-view prediction requires the accumulated memory to support inference beyond observed viewpoints. Both therefore constrain each write to *remain useful beyond the current local observation*, shaping the predictive memory state. Consistently, adding either objective atop Local supervision improves generalization, whereas global supervision alone is weak, indicating that the broader constraints rely on the acquisition capability learned from local supervision.

Together, these results show that Local supervision establishes effective acquisition, while Global supervision improves the persistence and predictive use of accumulated evidence. Combining both yields the strongest overall performance across the main evaluation metrics. We also provide additional ablations on WTM's design choices in Sec. B.3 and discussion on its retention in Sec. B.5.

4.2 Utilizing WTM for view planning

View Planning. View planning requires an agent to move through a 3D scene and gather visual evidence in order to localize a given target view. This naturally matches WTM's ability to query candidate viewpoints from accumulated memory. In the ViewSuite benchmark, P2V and V2P test the correspondence between camera motions and views, while its sequential IVP task requires an agent to move step by step toward a target viewpoint, receiving a new observation after each action.

A key challenge in IVP is that the target region may not yet have been sufficiently observed during exploration. In such cases, reconstruction-based scene representations are limited by the visual evidence already acquired, whereas WTM is explicitly trained to infer views beyond those directly written into memory. IVP is also intrinsically sequential: each action reveals new evidence that should refine subsequent decisions, naturally matching WTM's evolving-memory formulation. We therefore use ViewSuite to test whether WTM provides useful signals for view planning.

Testing View Planning with WTM. WTM supports two complementary uses. Given a pose p, it runs **forward prediction** by querying the expected representation $\hat{\boldsymbol{Y}}(p)$. Given a target representa-

Table 3: **Performance on ViewSuite:** accuracy (%) for P2V/V2P and success rate (%) for IVP.

Method	Parameters	P2V	V2P	IVP	All
General VLMs					
GPT-5.4 Pro (OpenAI, 2026)	–	53.1	50.7	18.5	40.8
Qwen3.5 (Qwen Team, 2026)	397B	39.7	35.5	4.3	26.5
Qwen2.5-VL (Bai et al., 2025)	7B	29.5	24.2	2.5	18.7
Task-specifically Trained VLM					
ViewAgent (Wang et al., 2026b)	7B	57.3	52.8	47.8	52.6
Iterative Inverse Matching					
VGGT-Ω (Wang et al., 2026a)	1.1B	-	-	19.6	-
FSM-LVSM-3D (Ma et al., 2026)	0.31B	-	-	15.8	-
WTM-pose (Ours)	0.12B	**66.4**	**82.3**	**48.5**	**65.7**

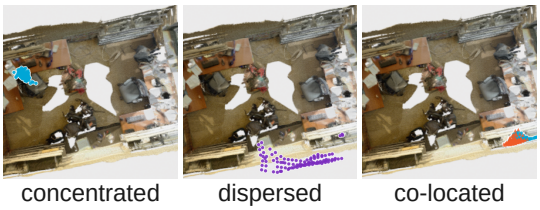

Figure 5: **Emergent spatial organization of memory slots.** Examples illustrate spatial locality, variation in coverage extent, and overlap across slots. Colors denote slots; points show query-patch locations with the strongest routing responses.

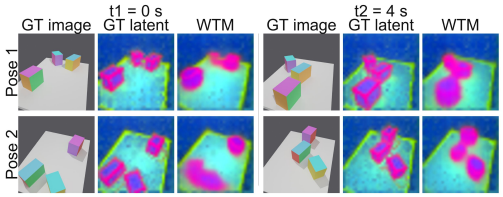

Figure 6: **Illustration of extending WTM to dynamic scenes.** WTM-4D learns to predict latent visual features queried at different poses and times. We visualize its predictions and the corresponding ground-truth features using PCA.

tion $Y^\star$, it performs **inverse viewpoint matching** by searching over candidate poses P, as

$$p^\star = \arg\min_{p \in \mathcal{P}} D\big(\hat{\boldsymbol{Y}}(p), \boldsymbol{Y}^\star\big), \tag{17}$$

where D is the Chamfer distance. These two operations instantiate P2V and V2P in one-shot setting.

IVP applies Eq. 17 repeatedly as memory evolves. At each step, we query two sets of candidate poses: a **global** set densely sampled over the scene, providing coarse long-range hypotheses, and a **local** set generated by short action sequences from the current pose, providing directly reachable candidates. Candidates are ranked by the Chamfer distance to the target. The agent moves toward a sufficiently matching candidate, or explores otherwise; the resulting observation is then written into memory before the next matching step, as detailed in Sec. A.4. We do not use rollout-based planning adopted by action-conditioned world models (Maes et al., 2026), as discussed in Sec. C.1.

Main Results. As shown in Table 3, without any task-specific training, WTM performs well across all ViewSuite tasks. It substantially outperforms the VLMs and on P2V and V2P, and achieves **48.5%** success on multi-turn IVP. With only 120M trainable parameters, WTM outperforms ViewAgent (47.8%) with task-specific training on ViewSuite.

To separate the model from the search procedure, we replace WTM with VGGT-Ω and FSM-LVSM-3D under the same search rules. Their IVP success rates drop to 19.6% and 15.8%, respectively. We provide additional results under different search budgets for IVP in Sec. B.6.

4.3 Analysis

Spatial structure in memory routing. We analyze what organizes the routing R. Across pairs of queried patches, we measure how closely routing similarity tracks similarity in several attributes in Table 4. Among these attributes, which are not available to the Reader, 3D location shows the strongest correlation, exceeding ray, visual feature, and image-grid location.

Since the Reader receives only memory and query pose at inference, this correspondence suggests that WTM learns to route query patches by their inferred spatial regions, inducing a geometric organization of memory.

Table 4: **Routing similarity vs. query attributes.** Spearman's ρ is computed over queried patch pairs within each scene.

Query attribute	ρ **with** R
3D Location	**0.307**
Viewing Ray	0.264
Visual feature	0.194
Image grid	0.168

Fig. 5 supports this interpretation: slots' responses *concentrate* within a spatial region, whose size can vary from a broad surface, such as a wall, to a smaller area with fine detail. Multiple slots can cover the same region and jointly contribute to its feature predictions through the Reader's weighted aggregation. Together, these observations suggest an emergent, distributed spatial organization of the memory.

Cross-Dataset Generalization. We evaluate whether WTM's memory mechanism transfers beyond ScanNet using ARKitScenes and ScanNet++ (Baruch et al., 2021; Yeshwanth et al., 2023). As shown in Table 5, updating memory consistently improves ScanNet-only WTM on all datasets. It outperforms VGGT-Ω despite the broader data mixture covering these domains.

Table 5: **Cross-dataset generalization of WTM.** Mean cosine similarity of held-out-view predictions before (*init. mem.*) and after (*written mem.*) incorporating scene observations.

Dataset	VGGT-Ω	WTM (ScanNet-only)		WTM (Mixed-data)	
		init. mem.	written mem.	init. mem.	written mem.
ScanNet rendered	0.649	0.501	0.662 **(+0.161)**	0.492	0.647 **(+0.155)**
ARKitScenes	0.341	0.379	0.478 **(+0.099)**	0.482	0.582 **(+0.100)**
ScanNet++	0.516	0.431	0.581 **(+0.150)**	0.482	0.632 **(+0.150)**
Average	0.502	0.437	0.574 **(+0.137)**	0.485	0.620 **(+0.135)**

To distinguish improved domain coverage from stronger memory use, we compare mixed-data and ScanNet-only training under the same budget. The former increases the average prediction similarity, both with initialized and written memory. However, the relative gain from memory writing over the initialized memory $\mathcal{M}_0$ remains nearly unchanged across training and evaluation domains (the parenthesized gains in Table 5). This suggests that mixed-data training improves the underlying visual representations, while the benefit of WTM's read–write memory mechanism remains.

Beyond Pose Conditioning. WTM can be extended beyond static 3D scenes by incorporating additional variables to the query condition. We explore this in a dynamic setting by jointly conditioning WTM on both camera pose and time, while retaining the same underlying read–write memory architecture. As shown in Fig. 6, we generate tabletop scenes where multiple cubes move over time while cameras orbit the scene. WTM-4D receives observations from two camera trajectories with their poses and timestamps, and is queried at held-out pose–time coordinates. The predictions follow both viewpoint changes and the evolving object states, showing that the memory captures the spatiotemporal evolution of the scene. Quantitatively, WTM-4D achieves a dynamic-region cosine similarity of **0.826**, compared with **0.563** for FSM-LVSM-4D (Ma et al., 2026).

5 Conclusion

We proposed the **World Turing Machine (WTM)** to address a central challenge in spatial world modeling: how to accumulate and revise visual evidence from partial observations in a persistent memory that supports the prediction of unobserved representations. WTM maintains a persistent Address–Content memory operated on by learned Reader and Writer. Experiments show that scene-specific updates contribute beyond knowledge stored in the parameters, while adaptive addressing helps make accumulated evidence accessible across viewpoints. With a fixed candidate-pose search procedure, the same memory supports target-view localization and sequential decisions without task-specific policy training. These findings suggest that spatial inference can emerge as a downstream capability of learned read–write computation, extending scene modeling beyond reconstruction to evaluating spatial hypotheses as observations accumulate.

AI DISCLOSURE

In this work, we have not used generative AI tools for any of the required disclosure tasks. Additionally, we only used generative AI tools for some of the recommended disclosure tasks, specifically summarizing and analyzing existing literature and assisting with sourcing and searching for information. We have reviewed all AI-assisted outputs and verified their relevance and correctness through manual inspection. We take responsibility for the final content of this work, including all text, claims, and artifacts.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we describe the implementation details in Sec. A. We also include the core implementation of WTM in the supplementary material. We further commit to publicly releasing the complete codebase, including the additional data-generation and evaluation pipelines, after the review process has concluded.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. *arXiv preprint arXiv:2301.08243*, 2023.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba, Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-jepa 2: Self-supervised video models enable understanding, prediction and planning, 2025. URL https://arxiv.org/abs/2506.09985.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. URL https://arxiv.org/abs/2502.13923.

Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15791–15801, June 2025.

Gilad Baruch, Zhuoyuan Chen, Afshin Dehghan, Tal Dimry, Yuri Feigin, Peter Fu, Thomas Gebauer, Brandon Joffe, Daniel Kurz, Arik Schwartz, and Elad Shulman. ARKitscenes - a diverse real-world dataset for 3d indoor scene understanding using mobile RGB-d data. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

Carlos Campos, Richard Elvira, Juan J. Gómez, José M. M. Montiel, and Juan D. Tardós. ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021.

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

Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. *MVSplat: Efficient 3D Gaussian Splatting from Sparse Multi-view Images*, pp. 370–386. Springer Nature Switzerland, October 2024. ISBN 9783031726644. doi: 10.1007/978-3-031-72664-4_21. URL http://dx.doi.org/10.1007/978-3-031-72664-4_21.

Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5828–5839, 2017.

Bardienus Pieter Duisterhof, Lojze Zust, Philippe Weinzaepfel, Vincent Leroy, Yohann Cabon, and Jerome Revaud. Mast3r-sfm: a fully-integrated solution for unconstrained structure-from-motion. In *2025 International Conference on 3D Vision (3DV)*, pp. 1–10. IEEE, 2025.

S. M. Ali Eslami et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018. doi: 10.1126/science.aar6170. URL https://doi.org/10.1126/science.aar6170.

Marco Fraccaro, Danilo Jimenez Rezende, Yori Zwols, Alexander Pritzel, S. M. Ali Eslami, and Fabio Viola. Generative temporal models with spatial memory for partially observed environments. In *Proceedings of the International Conference on Machine Learning*, 2018.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines, 2014. URL https://arxiv.org/abs/1410.5401.

Caglar Gulcehre, Sarath Chandar, Kyunghyun Cho, and Yoshua Bengio. Dynamic neural turing machine with soft and hard addressing schemes, 2017. URL https://arxiv.org/abs/1607.00036.

Purvish Jajal, Nick Eliopoulous, Benjamin Shiue-Hal Chou, George K. Thiravathukal, James C. Davis, and Yung-Hsiang Lu. Token turing machines are efficient vision models. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, pp. 7887–7896, February 2025.

Haoyi Jiang, Liu Liu, Xinjie Wang, Yonghao He, Wei Sui, Zhizhong Su, Wenyu Liu, and Xinggang Wang. Spa3r: Predictive spatial field modeling for 3d visual reasoning, 2026. URL https://arxiv.org/abs/2602.21186.

Haian Jin, Hanwen Jiang, Hao Tan, Kai Zhang, Sai Bi, Tianyuan Zhang, Fujun Luan, Noah Snavely, and Zexiang Xu. LVSM: A large view synthesis model with minimal 3d inductive bias. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=QQBPWtvtcn.

Haotong Lin, Sili Chen, Jun Hao Liew, Donny Y. Chen, Zhenyu Li, Yang Zhao, Sida Peng, Hengkai Guo, Xiaowei Zhou, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth anything 3: Recovering the visual space from any views. In *Proceedings of the International Conference on Learning Representations*, volume 2026, pp. 141261–141285, 2026.

Ziqiao Ma, Xueyang Yu, Haoyu Zhen, Yuncong Yang, Joyce Chai, and Chuang Gan. Fast spatial memory with elastic test-time training. In *Proceedings of the European Conference on Computer Vision*, 2026.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworld-model: Stable end-to-end joint-embedding predictive architecture from pixels, 2026. URL https://arxiv.org/abs/2603.19312.

Lorenzo Mur-Labadia, Matthew Muckley, Amir Bar, Mido Assran, Koustuv Sinha, Mike Rabbat, Yann LeCun, Nicolas Ballas, and Adrien Bardes. V-jepa 2.1: Unlocking dense features in video self-supervised learning, 2026. URL https://arxiv.org/abs/2603.14482.

Riku Murai, Eric Dexheimer, and Andrew J. Davison. MASt3R-SLAM: Real-time dense SLAM with 3D reconstruction priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

OpenAI. Introducing GPT-5.4, March 2026. URL https://openai.com/index/introducing-gpt-5-4/.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. URL https://arxiv.org/abs/2304.07193.

Emilio Parisotto and Ruslan Salakhutdinov. Neural map: Structured memory for deep reinforcement learning. In *Proceedings of the International Conference on Learning Representations*, 2018.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4172–4182, 2023.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

Michael S Ryoo, Keerthana Gopalakrishnan, Kumara Kahatapitiya, Ted Xiao, Kanishka Rao, Austin Stone, Yao Lu, Julian Ibarz, and Anurag Arnab. Token turing machines. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19070–19081, 2023.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *Proceedings of the International Conference on Machine Learning*, pp. 9355–9366, 2021.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. Dinov3, 2025. URL https://arxiv.org/abs/2508.10104.

Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, et al. Learning to (learn at test time): Rnns with expressive hidden states. *arXiv preprint arXiv:2407.04620*, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Proceedings of Advances in Neural Information Processing Systems*, 30, 2017.

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

Jianyuan Wang, Minghao Chen, Shangzhan Zhang, Nikita Karaev, Johannes Schönberger, Patrick Labatut, Piotr Bojanowski, David Novotny, Andrea Vedaldi, and Christian Rupprecht. VGGT-Ω. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026a.

Kangrui Wang, Linjie Li, Zhengyuan Yang, Shiqi Chen, Zihan Wang, Li Fei-Fei, Jiajun Wu, Leonidas Guibas, Lijuan Wang, and Manling Li. Planning with the views, 2026b. URL https://arxiv.org/abs/2605.29563.

Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025b. URL https://arxiv.org/abs/2501.12387.

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

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. *Proceedings of Advances in Neural Information Processing Systems*, 37:115491–115522, 2024.

Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12–22, 2023.

Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. URL https://arxiv.org/abs/2012.02190.

Jingwei Zhang, Lei Tai, Ming Liu, Joschka Boedecker, and Wolfram Burgard. Neural slam: Learning to explore with external memory, 2020. URL https://arxiv.org/abs/1706.09520.

Mingkun Zhang, Wangtian Shen, Fan Zhang, Haijian Qin, Zihao Pei, and Ziyang Meng. Rae-nwm: Navigation world model in dense visual representation space. *arXiv preprint arXiv:2603.09241*, 2026a.

Pingyue Zhang, Zihan Huang, Yue Wang, Jieyu Zhang, Letian Xue, Zihan Wang, Qineng Wang, Keshigeyan Chandrasegaran, Ruohan Zhang, Yejin Choi, Ranjay Krishna, Jiajun Wu, Li Fei-Fei, and Manling Li. Theory of space: Can foundation models construct spatial beliefs through active exploration? In *Proceedings of the International Conference on Learning Representations*, 2026b.

Tianyuan Zhang, Sai Bi, Yicong Hong, Kai Zhang, Fujun Luan, Songlin Yang, Kalyan Sunkavalli, William Freeman, and Hao Tan. Test-time training done right. In *Proceedings of the International Conference on Learning Representations*, volume 2026, pp. 157604–157638, 2026c.

Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. In *Proceedings of the International Conference on Learning Representations*, volume 2026, pp. 35791–35820, 2026.

A Implementation Details

A.1 Implementing WTM

Feature space and memory. We use a frozen DINOv2 ViT-L/14 as the visual teacher. Images are resized to 448×448, producing a 32×32 grid of $N = 1024$ patch tokens with feature dimension $d_y = 1024$. We apply token-wise LayerNorm to the teacher features and use the normalized tokens as both prediction targets and observation features for writing. WTM-Pose maintains $L = 512$ persistent Address–Content slots, with $d_A = 512$ and $d_C = 1024$. At the beginning of each scene, Content is initialized to zero, while Address is initialized from a fixed deterministic orthogonal code shared across scenes. Memory is scene-specific and is reset when entering a new scene.

Reader. The Reader contains four Transformer decoder blocks with hidden width 1024 and 8 attention heads. Each block consists of self-attention, the memory cross-attention described in Sec. 3, and an FFN. The initial query $\boldsymbol{Q}_0$ is a learned 32×32 grid of position-only tokens.

Camera poses are represented relative to the first observation of each scene. Translation and rotation are encoded using two MLPs and then concatenated into a pose embedding with 1024 dimensions. Then this pose embedding modulates every Reader block through AdaLN. Following LeWM (Maes et al., 2026), AdaLN is applied to the inputs of the self-attention, cross-attention, and FFN sublayers, respectively, in each block.

Writer implementation. The Writer reuses the routing $\boldsymbol{R}_t$ from the final Reader block for both memory updates. We average the routing weights across attention heads before applying $\mathrm{Route}_t(\cdot)$, and set the numerical-stability term to $\epsilon = 10^{-6}$ for all slots.

Content update. For the Content update, f_{dir} is implemented as LayerNorm followed by a linear projection. We implement the gate f_{gate} as the product of a routing-mass-dependent scale and a channel-wise learned gate. Specifically,

$$\boldsymbol{a}_t^C = \mathrm{clip}(0.03\,\boldsymbol{r}_t, 0, 1) \in \mathbb{R}^{L\times 1}, \qquad \mathbf{g}_t^C = \sigma\big(\mathrm{MLP}\big([\boldsymbol{C}_t, \boldsymbol{Z}_t^C]\big)\big) \in \mathbb{R}^{L\times d_C},$$

and use

$$\alpha_t^C = \boldsymbol{a}_t^C \odot \mathbf{g}_t^C.$$

The notation follows Sec. 3. Thus, routing mass controls the overall update magnitude of each slot, while the learned gate determines its per-channel update strength.

Address update. For WTM-Pose, the auxiliary evidence $\boldsymbol{B}_t$ in Eq. 12 is pseudo-depth produced by DepthAnythingV3 (Lin et al., 2026) aligned with the DINO patch grid. In each patch, we calculate the mean depth of all points, after applying a validity mask, so that $\boldsymbol{B}_t$ has dimension 32×32 in our DINOv2 setting. Together with the observation pose ξ_t, it provides geometric evidence to ψ_{evid}, while the pre-write Address-side queries $\boldsymbol{Q}_t^A$ provide the current access signal.

The resulting routed evidence $\boldsymbol{Z}_t^A$ is mapped by f_A to the proposed Address displacement $\boldsymbol{\Delta}_t^A$. The update strength β_t^A uses the same mass-adaptive form,

$$\beta_t^A = \mathrm{clip}(0.03\,\boldsymbol{r}_t, 0, 1),$$

with the routing mass computed over valid geometric evidence. We then apply the tangent-space projection and spherical interpolation, in order to keep all address vectors a fixed magnitude. The configured spherical radius is $\sqrt{512}$.

Extension to WTM-4D. We extend the query condition from pose to joint pose–time conditioning in WTM-4D,

$$\xi_q^{4D} = (P_q, \tau_q), \qquad \widehat{Y}_q = \mathcal{R}_{\theta_R}(M_t, \xi_q^{4D}),$$

where τ_q denotes the physical timestamp of the queried view, rather than its frame index or observation order.

Given the episode start time τ_0 and episode length τ_a, we first normalize time as

$$u_\tau = \frac{\tau - \tau_0}{\tau_a},$$

and encode it using a fourier encoder followed by an MLP. The original pose encoder and position-only image queries remain unchanged. We combine pose and time conditioning additively at each block,

$$\xi_l^{4D} = \xi^{\text{pose}} + \xi^{\text{time}}.$$

We also augment the geometric evidence used by the Address writer. In WTM-Pose, pseudo-depth and camera geometry are converted into a reference-frame patch coordinate $\boldsymbol{x}_{t,i}^{\text{ref}} \in \mathbb{R}^3$ with validity indicator $\boldsymbol{v}_{t,i}$, and encoded as

$$\boldsymbol{h}_{t,i}^{3D} = \phi_x\left([\boldsymbol{x}_{t,i}^{\text{ref}}/5,\ \boldsymbol{v}_{t,i}]\right).$$

WTM-4D augments this evidence with the timestamp of the observation,

$$\boldsymbol{h}_{t,i}^{4D} = \phi_{\boldsymbol{x}\tau}\left([\boldsymbol{x}_{t,i}^{\text{ref}}/5,\ \gamma(\boldsymbol{u}_{\tau_t}),\ \boldsymbol{v}_{t,i}]\right).$$

We introduce no separate temporal attention term, keeping our WTM architecture fixed, except for changing the condition and geometric evidence from 3D to 4D in reader and writer.

WTM contains approximately 120M trainable parameters, excluding the frozen DINOv2 teacher encoder and the depth estimator.

A.2 Training Setups

We construct training trajectories from the ScanNet meshes and their recorded camera trajectories. Camera poses are sampled from the original trajectories every 30 frames to form the observation keyframes. For each keyframe, we additionally sample two nearby held-out views at offsets of -10 and $+10$ frames. All views are rendered from the ScanNet meshes using fixed camera intrinsics. We pre-compute their DINOv2 teacher features and cache them for training, so the frozen teacher encoder does not need to be loaded online. We use rendered rather than raw ScanNet RGB primarily because it also allows us to obtain ground-truth views at arbitrary camera poses during evaluation. We only construct training episodes from ScanNet train set and keep the test set for evaluations.

Each training episode corresponds to one scene trajectory. At the beginning of an episode, memory is initialized as $M_0 = (\boldsymbol{A}_0, \boldsymbol{C}_0)$, where $\boldsymbol{A}_0$ is initialized from the fixed orthogonal Address code and $\boldsymbol{C}_0$ is zero. We then process the observation keyframes sequentially, with each newly observed keyframe producing one memory write and one optimization step.

For an observation at step t, the local neighboring-view objective uses its two associated held-out views. The replay objective samples $N_{\text{rep}} = 2$ previously written keyframes, while the unvisited-view objective samples $N_{\text{unv}} = 2$ keyframes from the same trajectory that have not yet been written into memory. If fewer valid targets are available for an objective, the unavailable targets are masked rather than replaced by additional samples. After every optimization step, the updated memory state is detached before the next observation is processed. Therefore, gradients never propagate across observation steps and training does not use truncated backpropagation through time.

We train WTM on 8 NVIDIA A100 GPUs for a total of 30K steps, with a global batch size of 32. The first 20K updates use a constant learning rate of $\mathbf{1} \times \mathbf{10^{-4}}$, followed by cosine decay over the final 10K updates to $\mathbf{1} \times \mathbf{10^{-6}}$. We use `AdamW` with weight decay $\mathbf{0.01}$ and gradient clipping at $\mathbf{1.0}$. Unless otherwise specified, we set the loss weights to ($\lambda_{\text{nbr}} = 1.0$), ($\lambda_r = 0.5$), ($\lambda_u = 0.2$), and ($\lambda_d = 0.1$) for neighbor prediction, replay, unvisited-view prediction, and Content-delta regularization, respectively.

A.3 Unified Evaluation Protocols

We use two evaluation protocols for different purposes. For ScanNet, we follow the official dataset split when constructing the training and evaluation sets.

WTM within-trajectory evaluation. For experiments within the WTM setting, we use all keyframes along a ScanNet trajectory as sequential observations to construct the memory, and query the resulting memory at all associated held-out views. For retrieval, the candidate gallery consists of all held-out views along the trajectory, and candidates are ranked by cosine similarity to the predicted representation. This protocol is used for evaluating WTM variants, training-objective ablations, and analyses of memory behavior. For this within-trajectory evaluation, we use no tolerance: a retrieval

is counted as correct only if it matches the exact target viewpoint. We also construct corresponding evaluation sets for ARKitScenes and ScanNet++ using the same protocol for results reported in Table 5 of the main paper.

Cross-trajectory view prediction benchmark. For comparison with other methods, we construct a shared sparse-observation benchmark using ScanNet test scenes with multiple available trajectories. For each pair of trajectories (τ_a, τ_b) from the same scene, we evaluate both directions: $\tau_a \to \tau_b$, where observations are sampled from τ_a and targets from τ_b, and $\tau_b \to \tau_a$ vice versa.

For each input trajectory, we select 4 spatially compact keyframes as observations shared by all methods, so that they cover only a limited region of the scene. Query views are taken from the paired trajectory, which typically follows a different spatial path through the room. We further remove query views that are spatially close to any input observation under a fixed threshold, and use the remaining views as held-out targets. This reduces near-view interpolation and emphasizes prediction in regions not directly covered by the sparse inputs. The resulting benchmark is intentionally challenging, as each method must predict distant cross-trajectory views from only four observations with limited scene coverage. For retrieval evaluation, we use a threshold of 0.1 m and 10°, treating alternative candidates within this threshold as correct to mitigate the effect of duplicate frames caused by little or no camera motion.

Each method is evaluated through its native inference interface while using the same input images and query views whenever applicable. Pose-conditioned methods receive the input observations and their camera poses and are queried directly at the target pose. For methods that do not take camera poses as input, for example, VGGT, which jointly predicts depth maps for the 4 input views in a single forward pass, we use the ground-truth input camera poses to back-project and fuse the RGB-D observations into a scene point cloud, which is then rendered from the queried pose.

A.4 Applying WTM to ViewSuite

We use bidirectional nearest-neighbor cosine similarity between predicted and target DINO features for view matching. This view-matching formulation naturally enables WTM to address the ViewSuite tasks. Given the predicted and target patch-token sets $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1}^{N}$ and $\boldsymbol{Y} = \{\boldsymbol{y}_j\}_{j=1}^{M}$, respectively, we define their bidirectional Chamfer similarity as the average of the mean nearest-neighbor cosine similarities in both directions:

$$\mathcal{S}_{\text{Chamfer}}(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{2}\left[\frac{1}{N}\sum_{i=1}^{N}\max_{j}\frac{\boldsymbol{x}_i^\top \boldsymbol{y}_j}{\|\boldsymbol{x}_i\|_2\|\boldsymbol{y}_j\|_2} + \frac{1}{M}\sum_{j=1}^{M}\max_{i}\frac{\boldsymbol{x}_i^\top \boldsymbol{y}_j}{\|\boldsymbol{x}_i\|_2\|\boldsymbol{y}_j\|_2}\right].$$

P2V and V2P. For both one-shot tasks, we initialize WTM using the first provided observation. In P2V, the Reader predicts the representation at the queried pose and selects the closest among the four candidate views. In V2P, we query the Reader at the four candidate poses and select the prediction that best matches the target view.

IVP candidate generation. At each decision step, we construct separate local and global candidate sets. Local candidates are the deduplicated endpoint poses of all valid action sequences with horizon of 1–4 in the 10-action space. Global candidates are sampled on a 0.5 m grid over the scene XY plane, retaining locations within 1.5 m of observed poses. At each location, we instantiate 12 yaw directions at $30°$ intervals and three pitch choices: base, $30°$ upward, and $30°$ downward.

Selection and execution. We use a three-level decision rule with local search horizon H and interaction budget B. At each decision step, local candidates consist of action sequences of up to H primitive actions. We further exclude the two roll actions, reducing the action space from 12 to 10 actions. If the best local candidate exceeds a stricter similarity threshold, we execute the entire corresponding action sequence as a single macro step. Otherwise, we evaluate the global candidates and execute the best global hypothesis if it exceeds a lower threshold, reflecting the lower confidence of global predictions. If neither criterion is met, we perform deterministic frontier exploration toward a nearby under-observed grid cell. Following the ViewSuite evaluation protocol, intermediate observations along a macro action sequence are not written to memory; only the resulting

observation after execution is incorporated into WTM before the next decision. We allow at most B such macro steps per episode, and use $H = 4$ and $B = 10$ for our main results, matching the interaction budget used by ViewAgent. We submit an answer only after the agent physically reaches the matched-view location to ensure accurate localization. If the budget is exhausted before this occurs, we instead submit the highest-scoring candidate available at the final step. The thresholds are empirically determined using the validation set.

B More Experiment Results

B.1 More Results for View Prediction and Retrieval

Table 6 provides additional results on the cross-trajectory view prediction benchmark used in Sec. 4.1. We additionally include MASt3R-SLAM (Murai et al., 2025) as a reference for a full SLAM system. In our setting, however, MASt3R-SLAM requires dense sequential inputs and is therefore provided with approximately $150\times$ more frames than the other methods. We include this result only as a SLAM reference rather than a directly comparable baseline.

Our primary evaluation encodes predicted views into the shared DINOv2 space, which provides a common representation for comparing methods with different output interfaces. However, this space is native to WTM but not to RGB reconstruction and novel-view synthesis methods. We therefore additionally evaluate RGB-producing methods directly in their output space. Specifically, for each predicted view we rank the candidate ground-truth views using either PSNR or LPIPS, using the standard AlexNet-based LPIPS model, and report the resulting R@1.

As shown in Table 6, the comparable RGB-based methods remain substantially below WTM in view retrieval even when evaluated directly in RGB space. For example, VGGT-Ω reaches 8.02% R@1 with PSNR and 14.17% with LPIPS, compared with 26.91% for WTM in its latent prediction space. This shows that WTM's retrieval performance remains competitive even when RGB-producing baselines are evaluated in their native output space. The MASt3R-SLAM retrieval result is higher, but uses the substantially denser input described above and is not directly comparable.

We further investigate how retrieval performance scales with the number of input frames, as shown in Fig. 7. Both VGGT and WTM improve steadily as more frames are provided. WTM consistently outperforms VGGT with 2–6 input frames, while the two methods perform comparably with 8 frames. These results demonstrate that WTM maintains stronger view-prediction performance than the reconstruction-based baseline in the sparse-input regime, where only limited information is available, while also scaling favorably with additional observations. We do not report results beyond 8 input frames because many rooms in the ScanNet test scenes are relatively small; denser input sampling often leaves no held-out target frames that satisfy our spatial-separation criterion.

Table 6: Full results on view prediction and retreival.

Method	Cos. Sim. ↑	Cos. Sim. R@1 ↑	PSNR R@1 ↑	LPIPS R@1 ↑
Nearest Observed[†]	0.291	1.35	2.71	2.53
Novel View Synthesis				
LVSM	0.271	20.95	10.24	14.47
FSM-LVSM-3D	0.276	22.62	11.52	15.70
SfM / SLAM				
MASt3R-SfM	0.228	10.04	5.43	5.85
MASt3R-SLAM[‡]	0.309	35.81	6.41	20.72
Feed-Forward 3D				
VGGT	0.262	19.28	6.58	12.02
VGGT-Ω	0.273	24.36	8.02	14.17
CUT3R	0.205	10.91	11.30	6.84
WTM-Pose (Ours)	**0.528**	**26.91**	–	–

[†] This baseline simply uses the nearest observed input view as prediction.
[‡] Uses approximately $150\times$ denser input; included for reference only.

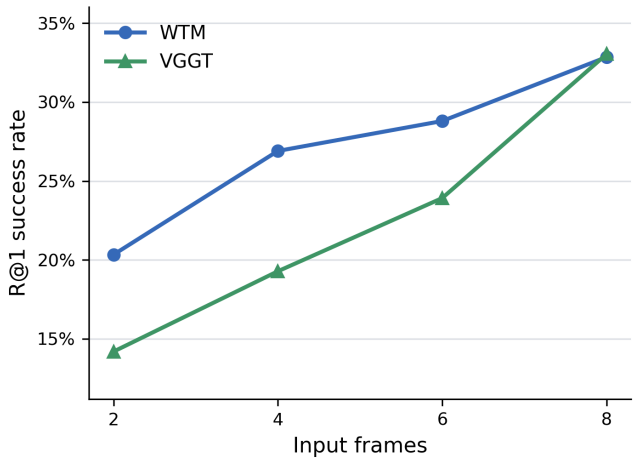

Figure 7: **Retrieval Success Rate with Different Input Frames.** We change the number of input frames while keep the targets fixed, so that the 4-frame-input scenario is our main benchmark reported before.



Figure 8: **Examples of decoding WTM predictions to images.** Ground-truth images are shown in the first row, and the corresponding RAE-decoded WTM predictions are shown in the second row.

B.2 Using WTM to Produce RGB Outputs

WTM is designed to predict view-conditioned latent representations rather than RGB pixels. We further decode WTM outputs into RGB images using a pretrained RAE decoder (Zheng et al., 2026). Since RAE and WTM operate in different feature spaces, we first apply a lightweight linear adapter to project WTM predictions into the RAE input space. We jointly train the adapter and the RAE decoder on our rendered data for 3K steps with batch size 64.

Qualitative results are shown in Fig. 8. The decoded images preserve the overall scene layout and major structures, while appearing blurry and lacking fine textures. This behavior is expected: WTM is trained to predict high-level view-dependent representations for spatial reasoning and retrieval, rather than optimizing photorealistic appearance reconstruction.

We further evaluate RGB reconstruction quality on the same held-out view prediction benchmark using PSNR and LPIPS, as shown in Table 7. Although WTM is not designed for fine-grained pixel reconstruction, equipping it with an RAE decoder yields competitive RGB performance, including better results than VGGT. We do not attribute this to highly detailed image synthesis. Rather, on this

challenging extrapolative prediction benchmark, reconstruction-based methods often fail to recover the target-view structure, whereas WTM preserves the major scene layout and semantic content despite lacking fine appearance details, as illustrated in Fig. 8.

B.3 Ablations on Design Choices

Unless otherwise specified, all ablations follow the WTM within-trajectory evaluation described in Sec. A.3. We report prediction cosine similarity and retrieval after the full observation sequence.

Address–Content memory and routing. We first ablate the main components of the Address–Content memory in Table 8. Removing the Content contribution from memory routing causes a substantial drop in retrieval, showing that Address alone is insufficient for selecting memory slots. Conversely, removing the independent Address state and using Content-only memory also degrades performance considerably. Keeping Address but disabling its scene-dependent update performs similarly poorly, whether the static Address is initialized from the fixed orthogonal code or learned during training. Together, these results indicate that both the Address–Content separation and the continual adaptation of Address are important.

Writer and geometric evidence. We next examine the Writer in Table 9. Removing depth from the Address evidence reduces retrieval performance, showing that explicit geometric evidence benefits memory organization. Nevertheless, the model retains substantial performance using camera pose alone, suggesting that pose conditioning already provides a strong spatial signal, while depth serves as complementary geometric evidence.

We also replace the prediction residual $Y_t - \widehat{Y}_t^-$ with the target feature Y_t in the Content update. Replacing it in both the gating and Content evidence reduces retrieval performance, while retaining the residual for gating recovers most of the performance. Removing the slot-wise r_t-adaptive write strength causes a substantial degradation, showing the importance of adapting the write magnitude to the routed evidence.

Architecture and optimization sensitivity. Finally, we vary optimization schedules and model capacity in Table 10. Our default training uses a constant learning rate for the first 20K steps followed by a 10K-step cosine decay, with batch size 32. Since Address slots are initialized using orthogonal codes, increasing the number of slots to 1024 requires increasing the Address dimension to 1024 accordingly.

Increasing the Reader depth from four to six blocks provides limited benefit, while further increasing it to eight blocks does not improve retrieval. Reducing either the Content width or the number of memory slots degrades performance, whereas increasing the Content width to 2048 does not provide additional gains. Scaling the memory capacity to 1024 slots yields substantially stronger performance, outperforming all other variants. Nevertheless, we retain 512 slots as our default configuration.

Table 7: **View prediction quality.** We evaluate predictions in the RGB space.

Method	PSNR ↑	LPIPS ↓
Nearest Observed†	10.28	0.663
Novel View Synthesis		
LVSM	12.21	0.707
FSM-LVSM-3D	12.44	**0.656**
SfM / SLAM		
MASt3R-SfM	11.97	0.782
MASt3R-SLAM	10.95	0.779
Feed-Forward 3D		
VGGT	12.00	0.748
VGGT-Ω	12.11	0.735
CUT3R	**12.94**	0.791
WTM + RAE (Ours)	12.10	0.675

† This baseline simply uses the nearest observed input view as prediction.

Table 8: **Ablations on memory and routing.**

Variant	Cos. Sim. ↑	R@1 ↑
Full WTM	**0.662**	**86.07**
w/o Content score S^C	0.615	66.34
Content-only memory	0.597	52.23
Fixed orthogonal Address	0.588	44.38
Fixed learned Address	0.599	50.00

Table 9: **Ablations on Writer design and geometric evidence.**

Variant	Cos. Sim. ↑	R@1 ↑
Full WTM	0.662	**86.07**
w/o depth evidence	0.651	77.42
Replace residual $Y_t - \widehat{Y}_t^-$ with Y_t in both gating and Content evidence	0.660	80.81
Replace residual $Y_t - \widehat{Y}_t^-$ with Y_t in Content evidence only	**0.668**	84.85
w/o r_t-adaptive write strength	0.580	52.49

Different teacher encoders. We conduct a ablation of the frozen teacher encoder, keeping all other model configurations and the model size unchanged. The results, as shown in Table 11, show that DINOv2 achieves the highest retrieval success rate among the evaluated encoders. To understand why higher feature-prediction similarity does not necessarily translate into better retrieval performance, we analyze the geometry of the corresponding teacher feature spaces. WTM variants trained with DINOv3 (Siméoni et al., 2025) and V-JEPA 2.1 (Mur-Labadia et al., 2026) achieve higher prediction cosine similarity, suggesting that their latent targets are easier to predict. However, their target representations are less well separated from those of competing viewpoints. In contrast, DINOv2 yields a larger retrieval margin despite its lower prediction cosine similarity. This result suggests that view discriminability, rather than prediction accuracy alone, is critical for this task, and that encoders producing more separable feature spaces in our domain are better suited to WTM. CLIP (Radford et al., 2021) exhibits a different failure mode: although its target feature space is not intrinsically more ambiguous, WTM predictions fail to preserve a positive retrieval margin. This behavior is consistent with CLIP's relatively weak dense spatial correspondence, which may arise from its global image–text contrastive training objective.

B.4 Sensitivity to Depth and Pose Inputs

Our default WTM-Pose uses pseudo-depth together with ground-truth camera poses. We assume access to ground-truth pose because our primary goal is to study the read–write mechanism under pose-conditioned observations, a common setting in embodied tasks where agents can obtain reliable pose estimates from onboard sensors. We further test sensitivity to these geometric inputs by replacing pseudo-depth with ground-truth depth, or replacing the ground-truth input poses with camera poses estimated by VGGT from the input views. As shown in Table 12, using ground-truth depth yields essentially unchanged performance, while pseudo poses cause only a small degradation. This suggests that WTM is relatively robust to estimation errors in the geometric inputs used by our primary setting.

B.5 Analyzing Retention under Memory Capacities

We further investigate WTM's long-term retention and its dependence on memory capacity. As shown in Fig. 9, we evaluate three WTM variants with 256, 512, and 1024 memory slots on a long tour spanning three rooms. Throughout the tour, we track the cosine similarity between predicted and target features for a fixed set of held-out views sampled exclusively from the first room (A). After leaving A and traversing rooms B and C, all three variants exhibit a gradual decline in performance on these views as new observations accumulate. The 1024-slot variant, however, degrades most slowly before re-entering A, whereas the 256-slot variant suffers a pronounced drop, indicating that greater memory capacity improves robustness to interference from intervening observations.

Table 10: **Sensitivity to optimization and model capacity.**

Variant	Cos. Sim. ↑	R@1 ↑
Default (cosine decay)	0.662	86.07
Constant LR	0.647	76.74
Linear decay	0.661	84.34
Reader: 6 blocks	0.664	86.46
Reader: 8 blocks	0.659	83.47
Content width: 512	0.641	77.25
Content width: 2048	0.662	84.74
Memory slots: 256	0.652	81.46
Memory slots: 1024, $d_A = 1024$	**0.668**	**88.64**

Table 11: **Teacher-encoder ablations.** We additionally report two measures of encoder feature-space geometry that may help explain the observed performance differences. **Feature discrim.** is the maximum cosine similarity between a ground-truth target feature and any non-matching gallery feature, with lower values indicating better feature separability. **Retrieval margin** is the difference between the best-positive and hardest-negative similarities, with higher values indicating better separation.

Variant	Cos. Sim. ↑	R@1 ↑	Feature Discrim. ↓	Retrieval margin ↑
DINOv2 ViT-L/14(default)	0.662	**86.07**	0.511	**0.024**
DINOv3 ViT-L/16	0.783†	66.34	0.709	0.023
V-JEPA 2.1 ViT-L/16	0.769†	52.23	0.649	0.018
CLIP ViT-L/14	0.679†	44.38	**0.507**	-0.008

† compute cosine similarities in each's own feature space, thus not directly comparable.

Although the 512- and 1024-slot variants perform comparably before returning to A, the 1024-slot variant achieves substantially better predictions after re-entering and revisiting the room (0.670 vs. 0.650). These results suggest that increasing memory capacity facilitates information retention and integration over longer horizons, making larger-capacity WTM variants promising for deployment in larger-scale environments.

B.6 Ablations on Search Horizon and Interaction Budget

We further ablate the interaction step budget B and local search horizon H used by WTM inverse matching in ViewSuite IVP. As shown in Table 13, increasing the interaction budget consistently improves success for both horizon settings. Increasing the local horizon from $H = 3$ to $H = 4$ provides a smaller but consistent improvement, while increasing the number of candidate queries per movement step by approximately $5.4\times$. We use $H = 4$ and $B = 10$ as the default setting in our main experiments.

B.7 Analyzing Routing through Cross-View Correspondence

To further examine whether the learned routing reflects physical scene structure, we construct a cross-view patch correspondence task. Patch pairs from different images are labeled as positive when they observe the same physical region and negative otherwise. We then measure how well different similarity signals separate the two groups using AUROC. Unlike the correlation analysis in the main paper, this directly tests whether the learned access signals are predictive of physical correspondence across viewpoints.

As shown in Table 14, image-grid similarity is close to chance, while DINO features already provide a strong correspondence signal. This is expected, since the same physical region observed from different viewpoints can still produce similar semantic features. The learned Address score is substantially more predictive, reaching 0.945 AUROC, indicating that the Address space becomes strongly aligned with cross-view physical correspondence. Content score also captures correspon-

Table 12: Sensitivity to depth and pose inputs.

Condition	Cosine Sim. ↑	R@1 ↑
Pseudo depth + GT pose (ours default)	0.528	26.9
GT depth + GT pose	0.527	27.1
Pseudo depth + estimated pose	0.525	24.7

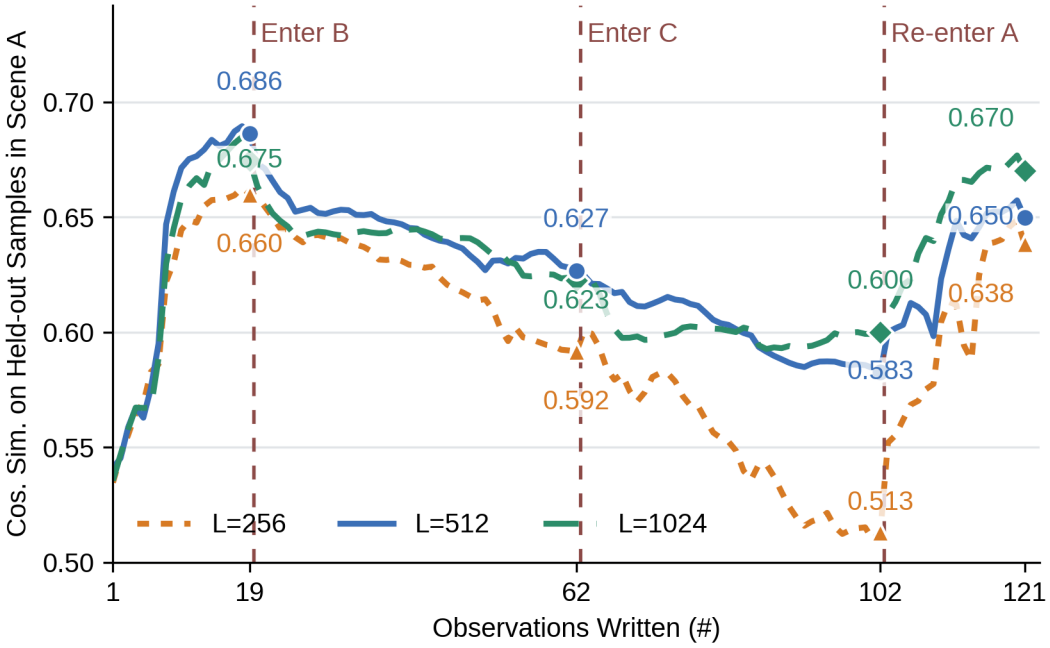

Figure 9: **Retention over long tours with varying memory capacities.** We construct a long tour by connecting three rooms in sequence. Throughout the tour, we continually evaluate WTM's predictive performance on a fixed set of held-out views sampled exclusively from the first room.

dence structure, but remains weaker than Address. The final joint routing achieves 0.947 AUROC, preserving the strong geometric structure encoded by Address while incorporating Content similarity. This further supports the spatial organization of learned memory addressing observed in the main paper.

B.8 Comparison with RAE-NWM

Navigation World Models (NWMs) provide a natural comparison for predicting unobserved views, typically modeling transitions from a short, locally continuous observation context to future views (Bar et al., 2025). WTM instead integrates incoming observations into a persistent scene memory and predicts views by querying this memory from arbitrary viewpoints. For a strong matched baseline, we reproduce RAE-NWM (Zhang et al., 2026a) on our rendered ScanNet data in the same DINOv2 feature space, extending it to support variable-length observation contexts and adapting its action conditioning to our 6-DoF viewpoint setting. We then evaluate both models under the two settings illustrated in Fig. 10.

In the nearby-transition setting (Fig. 10a), the observations come from a short, locally continuous segment and the query is a nearby future view, closely matching the standard NWM setting. RAE-NWM performs slightly better with a single context view, but WTM surpasses it once multiple observations are available and remains consistently stronger as the context grows. Thus, although WTM consolidates observations into a compact memory rather than directly modeling transitions over the input context, it remains effective in this local regime.

Table 13: **Results of different local search horizon and interaction budget on ViewSuite IVP.** We report success rate (%) under different local planning horizons H and interaction budgets B. Query/step denotes the average number of candidate-view queries per movement step, affected by horizon H.

	Interaction budget B			**Query/step**
Local horizon H	**10**	**15**	**20**	
$H = 3$	46.6	52.5	56.0	420
$H = 4$	<u>48.5</u>	53.6	**57.0**	2,250

Table 14: **Cross-View Correspondence Measured by AUROC.** A higher AUROC indicates that the similarity measure more reliably ranks physically corresponding patches above non-corresponding ones, while 0.5 denotes chance-level discrimination.

Grid	**DINO**	**Address Score S^A**	**Content Score S^C**	**Routing R**
0.490	0.856	**0.945**	0.886	**0.947**

The scene-tour setting (fig. 10b) tests a different regime: observations are sparse and distributed along the scene trajectory, while queries span the scene. Here, WTM improves steadily as the tour progresses, reaching over 85% retrieval accuracy, whereas RAE-NWM remains around 15–20%. The gap highlights the benefit of continually integrating spatially distributed observations into a persistent scene memory.

C Discussions

C.1 Why We Do Not Use Rollout-Based Planning

World action models (WAMs) are often used for planning by recursively rolling out imagined future states under planned action sequences. This setting differs from our view-planning problem. In typical world-model planning, actions change the underlying world state, and rollout predicts the consequences of these state transitions. In ViewSuite IVP, the scene itself is static: taking an action only changes the agent's viewpoint and reveals new evidence, thereby updating its belief about the same underlying world.

This distinction makes recursive imagined-view rollout less natural for WTM. An imagined observation does not provide new scene evidence, and feeding WTM's own prediction back through the Writer would not constitute the prediction–correction update for which the memory is trained. Instead, memory is updated only after a real observation is acquired. Moreover, WTM can directly query its current memory at an arbitrary candidate pose in a single Reader call, without sequentially imagining the intermediate views required to reach that pose. We therefore use direct spatial hypothesis testing to score candidate viewpoints, and reserve memory updates for newly acquired real observations.

C.2 Discussion on Evaluating in the Latent Space

Our view-prediction benchmark focuses on extrapolation from very sparse observations. Under this setting, accurately recovering fine-grained pixel appearance can be inherently difficult because much of the queried content is not directly observed. Our primary interest is therefore whether a model can infer sufficiently viewpoint-specific scene structure to support downstream operations such as view matching, localization, and planning, rather than whether it reconstructs every low-level visual detail.

This motivates our use of retrieval as the primary metric. Absolute cosine similarity depends on the feature space in which predictions are represented, and is therefore most meaningful as an auxiliary measure of prediction quality. Retrieval instead evaluates whether the predicted representation can distinguish the correct queried view from other views in the same scene, which directly matches

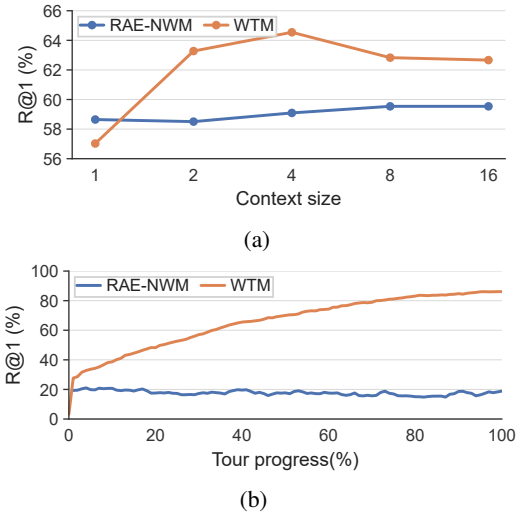

Figure 10: **Comparison with RAE-NWM.** (a) Nearby-transition setting: a sequence of context views is used to predict nearby future views. (b) Scene-tour setting: observations are sampled along a scene trajectory, while queries span the scene; tour progress denotes the fraction of the trajectory observed.

Table 15: Comparison with Spa3R in DINOv3 feature space on the cross trajectory view prediction benchmark.

Method	**Cos. Sim.** ↑	**R@1** ↑
Spa3R	0.490	13.38
WTM-DINOv3	0.632	**22.27**

the inverse-view-matching operation used by WTM. A prediction may achieve high average feature similarity while still lacking sufficient viewpoint discrimination.

Retrieval in native prediction spaces. Our main evaluation encodes RGB predictions into the common DINOv2 space so that methods with different output interfaces can be compared under a unified metric. Since DINOv2 is also the native prediction space of WTM, we additionally evaluate RGB-producing methods directly in RGB space to remove this dependence. Specifically, we rank candidate views using PSNR or AlexNet-based LPIPS and report the resulting R@1, as shown in Table 6. Under the shared sparse-input setting, WTM reaches 26.91% R@1 in its native latent space, remaining above the retrieval accuracy obtained by RGB baselines using either PSNR or LPIPS. This indicates that WTM's retrieval advantage is not solely a consequence of projecting RGB predictions into DINOv2 space. MASt3R-SLAM obtains a higher retrieval score under one metric, but uses approximately $150\times$ denser input and is included only as a reference.

Matched latent-space comparison. For direct view prediction in feature space, the most closely related work to our knowledge is Spa3R (Jiang et al., 2026), which learns a compact latent spatial representation and predicts feature fields at arbitrary unseen viewpoints. However, Spa3R is trained using a DINOv3 (Siméoni et al., 2025) teacher. To enable a matched comparison, we therefore train a WTM variant directly in DINOv3 space. Both methods are evaluated on the same cross-trajectory benchmark using identical DINOv3 targets and the same retrieval protocol. WTM-DINOv3 achieves an R@1 of **22.27%**, compared with 13.38% for Spa3R. This matched comparison further shows that WTM's stronger viewpoint discrimination is not specific to its original DINOv2 teacher space.