GG4D: GEOMETRY-GUIDED DYNAMIC NOVEL-VIEW SYNTHESIS

Anonymous authors
Paper under double-blind review

ABSTRACT

Dynamic novel-view synthesis lets us freely explore a captured moving scene from any viewpoint, enabling free-viewpoint video and immersive VR and AR. To this end, feedforward methods synthesize novel views in a single forward pass without per-scene optimization. Among them, one line predicts explicit 3D representations such as 3D Gaussians, while another directly predicts target views from input images using a learned implicit renderer. Yet explicit methods are limited in fidelity by geometry errors and their fixed rendering process, and implicit renderers without shared 3D geometry struggle to maintain multi-view consistency. In this paper, we present **GG4D**, a feedforward framework for dynamic novel-view synthesis that combines explicit scene geometry with implicit image synthesis to achieve both multi-view consistency and high rendering fidelity. First, to aggregate geometry across frames without mixing dynamic regions from different timesteps, we propose an efficient segmentation module that separates static and dynamic regions without off-the-shelf optical flow or segmentation models. Using this separation, we construct a 4D point cloud for explicit 3D representation that guides novel-view synthesis. Second, to enable high-fidelity rendering in real time, we propose Large Chunk Attentive Test-Time Training (LaCAT), which effectively encodes input observations into a compact scene memory through attention. Our implicit renderer, built upon it, summarizes the input video once and reuses this information to synthesize high-quality novel views in real time. Extensive experiments demonstrate that GG4D achieves state-of-the-art quality and multi-view consistency on both monocular and multi-view dynamic scene benchmarks, taking a step toward immersive applications.

1 INTRODUCTION

Novel-view synthesis (NVS) of dynamic scenes takes monocular or multi-view videos of a time-varying scene and synthesizes photorealistic images from novel viewpoints. This technique enables a wide range of applications, such as free-viewpoint video, virtual and augmented reality, and interactive content creation. To this end, optimization-based approaches (Park et al., 2021a;b; Yang et al., 2023; Wu et al., 2024; Lei et al., 2025) extend NeRF (Mildenhall et al., 2020) or 3DGS (Kerbl et al., 2023) to dynamic scenes and achieve photorealistic results by fitting the representation to each scene. While promising, these methods typically rely on time-consuming iterative optimization, reducing their practical applicability.

To avoid this lengthy per-scene optimization, feedforward approaches have emerged that synthesize novel views of dynamic scenes in a single forward pass. One line of work predicts time-varying 3D Gaussian primitives (Xu et al., 2025b; Lin et al., 2026a; 2025; Liang et al., 2025; Wu et al., 2026a). Since novel views are rendered from an explicit 3D representation, these methods naturally preserve strong multi-view consistency. Yet, errors in the predicted geometry and a fixed rendering pipeline can constrain rendering fidelity (Jin et al., 2025a). Another line of work, including FSM (Ma et al., 2026) and REON (Kim et al., 2026a), follows LVSM (Jin et al., 2025a) and uses a learned implicit renderer to directly synthesize target images without an explicit 3D representation. Learning the rendering function from data provides considerable flexibility and can yield higher rendering fidelity. While effective, without shared 3D geometry to anchor their predictions, these methods struggle to maintain multi-view consistency under diverse camera transformations.

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.

In static novel-view synthesis, PVSM (Wu et al., 2026b) introduces explicit 3D geometry into an implicit rendering pipeline to improve multi-view consistency. Using geometry estimated by a geometric foundation model (Keetha et al., 2026), it unprojects the input images into a shared coordinate system to construct a single 3D point cloud. For each target view, an implicit renderer processes the point-cloud projection together with the input images to synthesize the final image. However, extending this design to dynamic scenes presents two challenges. First, merging observations from different timesteps into a single point cloud mixes different positions of moving objects, causing ghosting in the projection. Second, processing all input images for every output frame incurs repeated computation whose cost grows with the number of input images, hindering its application to videos with many frames.

In this paper, we propose **GG4D**, a feedforward NVS framework for dynamic scenes that combines the multi-view consistency of explicit 3D representations with the high rendering fidelity of implicit image synthesis. Unlike Gaussian-based methods, GG4D synthesizes each target image with a learned implicit renderer, while anchoring the synthesis in explicit scene geometry. Whereas PVSM is limited to static 3D scenes, GG4D constructs a 4D point cloud from monocular or multi-view video, extending geometry-guided implicit rendering to dynamic scenes.

Constructing this 4D point cloud requires separating static and dynamic regions, so that static geometry can be accumulated across frames while dynamic geometry is kept separate for each timestep. Existing motion segmentation approaches typically rely on external models or additional cross-view analysis (Huang et al., 2025; He et al., 2026; Hu et al., 2026), introducing extra computation while still producing inaccurate or incomplete masks. We therefore propose an efficient and accurate segmentation method that reuses features already computed by the geometric foundation model (Wang et al., 2026b) during geometry estimation. Yet, directly classifying individual patches using these features often identifies only parts of a moving object. We instead group patches into semantically coherent regions and predict a static or dynamic label for each region, enabling accurate and stable coverage of moving objects without additional off-the-shelf optical flow or segmentation models.

We use the projection of this 4D point cloud to guide implicit image synthesis. However, the projection can contain holes and distortions, so the renderer also uses appearance information from the input frames to synthesize high-quality images. To avoid processing all input frames for every output frame, we build on Large Chunk Test-Time Training (LaCT) (Zhang et al., 2026), which stores input information in a reusable scene memory. Yet, LaCT builds this memory with limited direct interaction between input views, making it harder to combine information about the scene from different viewpoints. To address this, we introduce Large Chunk Attentive Test-Time Training (LaCAT), which allows input views to exchange information directly through cross-view attention while building the scene memory. This improves synthesis quality without increasing the per-frame neural rendering cost, since the additional computation occurs only when the scene memory is built.

The resulting scene memory and the 4D point cloud together allow our framework to separate *scene construction* from *target-view rendering*. The construction stage prepares both representations from the input video, while the rendering stage reuses them to synthesize each output frame in real time. Users can therefore change viewpoints interactively during video playback without repeating scene construction, sharing the preparation cost across different viewing trajectories.

Extensive experiments demonstrate that GG4D achieves state-of-the-art visual quality and multi-view consistency under monocular and multi-view settings, improving over the best feedforward baseline by 3.6 dB mPSNR on DyCheck (Gao et al., 2022) and 2.6 dB PSNR on Neural3D (Li et al., 2022). After scene construction, GG4D renders at 58 FPS at 512 resolution, supporting interactive free-viewpoint video rendering. We summarize our contributions as follows:

- We introduce GG4D, a feedforward NVS framework for dynamic scenes that combines the multi-view consistency of explicit 3D geometry with the rendering fidelity of implicit image synthesis.
- We propose an efficient and accurate dynamic region segmentation method that reuses features from a frozen geometric foundation model, without relying on off-the-shelf optical flow or segmentation models.
- We present Large Chunk Attentive Test-Time Training (LaCAT), which enables direct cross-view interaction before storing input observations in scene memory, improving rendering quality without increasing per-view neural rendering cost.
- GG4D achieves state-of-the-art quality on monocular and multi-view dynamic scene benchmarks and supports interactive free-viewpoint video through real-time rendering after scene construction.

2 Related Work

Feedforward NVS for Static Scenes. Feedforward NVS methods synthesize novel views from sparse input in a single forward pass, avoiding costly per-scene optimization. Early approaches in this direction regress pixel-aligned 3D Gaussians and render them by rasterization (Charatan et al., 2024; Chen et al., 2024; Xu et al., 2025a; Zhang et al., 2024; Ye et al., 2025). While effective, LVSM (Jin et al., 2025a) pointed out that the inductive bias of a predefined 3D representation and rendering equation limits expressiveness, and showed that rendering target images implicitly with a learned renderer achieves higher quality. Motivated by this, subsequent methods have followed this implicit rendering (Jiang et al., 2025; Wang et al., 2026a; Kim et al., 2026b). Without shared 3D geometry, these models are left to infer scene structure implicitly, and their consistency tends to degrade under large viewpoint changes. To bring geometry back into this pipeline, PVSM (Wu et al., 2026b) unprojects all input images into a single point cloud with a geometric foundation model (Keetha et al., 2026) and conditions the learned renderer on its projection onto the target view, which improves fidelity, cross-view consistency, and robustness to diverse camera transformations. Yet because PVSM treats the input images as an unordered set without distinguishing static geometry from time-varying dynamic geometry, it remains limited to static scenes. Moreover, its renderer re-processes every input image for each target view, so rendering latency grows with the number of input views. In contrast, our framework separates static and dynamic geometry into a temporally structured 4D point cloud to extend feedforward, geometry-conditioned rendering to dynamic scenes. Furthermore, instead of re-processing every input image for each target view, it encodes input-view observations into a scene memory, so that rendering a new target view requires only this memory, keeping latency nearly constant in the number of input views.

NVS for Dynamic Scenes. Unlike static scenes, dynamic scenes require NVS methods to account for geometry that changes over time. Optimization-based methods extend static 3D representations over time by modeling motion with deformation fields or explicit motion trajectories (Wu et al., 2024; Wang et al., 2025b; Lei et al., 2025; Yan et al., 2025). Feedforward methods have also evolved from reconstructing static scenes to capturing scene dynamics, without requiring per-scene optimization. One line of work predicts 3D Gaussian primitives together with their temporal deformation (Xu et al., 2025b; Lin et al., 2026a; 2025; Liang et al., 2025; Quesado et al., 2026). Another line extends implicit image synthesis to dynamic scenes (Ma et al., 2026; Kim et al., 2026a), yielding higher rendering fidelity but struggling to remain multi-view consistent under diverse camera transformations. Among these, FSM (Ma et al., 2026) adopts a test-time training renderer (Zhang et al., 2026) that stores the input observations into a scene memory, keeping the rendering cost constant regardless of the number of input views. Yet, when constructing this memory, FSM combines observations across input views only indirectly through the memory, which can limit rendering fidelity. In contrast, our framework enables direct cross-view interaction during scene memory construction, improving rendering quality without increasing per-view rendering cost. Moreover, we leverage explicit 3D representations while synthesizing novel views implicitly to achieve both multi-view consistency and high rendering fidelity.

3 GG4D

Fig. 1 provides an overview of our framework. Given video frames $\{\mathbf{I}_t^k\}$ capturing a dynamic scene, our goal is to synthesize an image $\hat{\mathbf{I}}^{\mathrm{tgt}}$ at a target camera $\mathbf{P}^{\mathrm{tgt}}$ and timestep t^{tgt} in a feedforward manner. The superscript $k = 1, \ldots, K$ indexes input views, the subscript $t = 1, \ldots, T$ indexes timesteps, and monocular video corresponds to $K{=}1$. During scene construction, we build a 4D point cloud by separating static and dynamic geometry (Section 3.1) and construct a scene memory from the input frames (Section 3.2). During rendering, we project the point cloud at the requested timestep onto the target camera and synthesize the output image using this projection and the scene memory (Section 3.3).

3.1 4D Point Cloud Construction

To construct a 4D point cloud from input frames $\{\mathbf{I}_t^k\}$, we use the corresponding depth maps $\{\mathbf{D}_t^k\}$ and camera parameters $\{\mathbf{P}_t^k\}$, both readily obtained from a geometric foundation model, VGGT-Ω (Wang et al., 2026b).

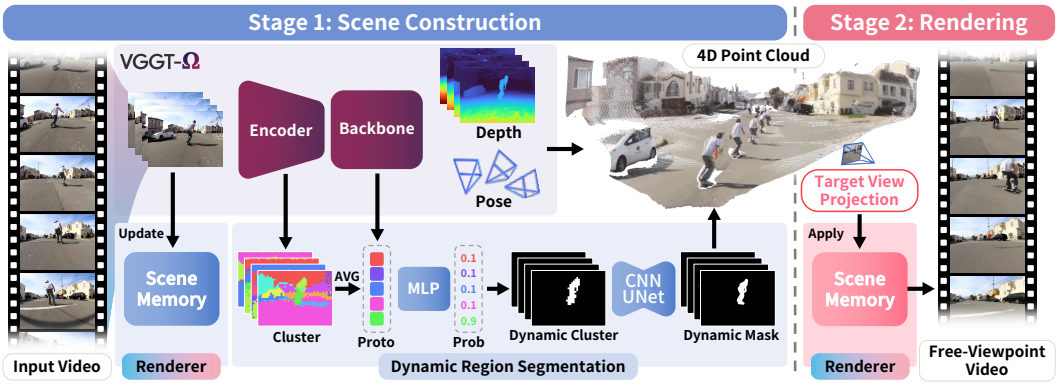


Figure 1: **Overview of GG4D.** Given an input video, GG4D operates in two stages. **Stage 1: Scene Construction.** A frozen VGGT-Ω (Wang et al., 2026b) estimates depth and pose, and its internal features are clustered and classified to separate dynamic regions, yielding a 4D point cloud. The input frames are also encoded into a scene memory. **Stage 2: Rendering.** For each target view, the point cloud is projected into the target view, and the renderer synthesizes the novel view from this projection by reading the stored scene memory.

Dynamic Region Segmentation. Wang et al. (2026b) observed that moving regions can be separated from static surroundings in the intermediate image features of VGGT-Ω, even without explicit motion supervision. Building on this finding, we reuse features from the global attention layers already used during geometry estimation. These layers incorporate cross-frame information into patch-level image tokens, forming a feature map $\mathbf{G}_t^k$ for each image $\mathbf{I}_t^k$. A natural approach is to train a lightweight binary classifier on these tokens to label each patch as static or dynamic. Unfortunately, this naive approach often misses parts of a moving object, so that the missed geometry is incorrectly accumulated into the static point cloud. To improve object coverage, we group tokens into semantically coherent regions and assign a shared static–dynamic label to each region, allowing patches with ambiguous motion cues to benefit from evidence elsewhere in the region.

To form semantically coherent regions, we use the semantic features of VGGT-Ω's image encoder, which is fine-tuned from DINOv3 (Siméoni et al., 2025). We apply agglomerative clustering (Müllner, 2011) to the encoder features of each image independently, partitioning its patch positions into clusters $\{\mathcal{C}_i\}$. The number of clusters is not predetermined, and a single merge threshold controls the granularity of the regions.

After clustering, we determine whether each cluster is dynamic in two steps. We first compute a prototype $\bar{\mathbf{g}}_i$ for each cluster $\mathcal{C}_i$ by averaging its global-attention features in $\mathbf{G}_t^k$. We then feed $\bar{\mathbf{g}}_i$ into a small MLP that predicts a dynamic probability ρ_i:

$$\bar{\mathbf{g}}_i = \frac{1}{|\mathcal{C}_i|} \sum_{p \in \mathcal{C}_i} \mathbf{G}_t^k[p], \qquad \rho_i = \mathrm{MLP}\big(\bar{\mathbf{g}}_i\big), \tag{1}$$

where $\mathbf{G}_t^k[p]$ denotes the feature vector at patch position p. The union of all clusters satisfying $\rho_i > \tau$ forms a coarse dynamic mask $\tilde{\mathbf{M}}_t^k$ at patch resolution. Finally, a U-Net (Ronneberger et al., 2015) refines this mask into a pixel-level mask $\mathbf{M}_t^k$ using the input image and estimated depth:

$$\mathbf{M}_t^k = \mathrm{UNet}\big([\mathbf{I}_t^k, \tilde{\mathbf{M}}_t^k, \mathbf{D}_t^k]\big), \tag{2}$$

where $[\cdot]$ denotes channel-wise concatenation.

Static-Dynamic Accumulation. Given the depth maps $\{\mathbf{D}_t^k\}$, camera parameters $\{\mathbf{P}_t^k\}$, and dynamic masks $\{\mathbf{M}_t^k\}$, we unproject each pixel into a shared world coordinate system. Points from static pixels are accumulated across all views and timesteps into a static point cloud $\mathbb{C}^{\mathrm{s}}$. Points from dynamic pixels are collected across views separately for each timestep into $\mathbb{C}_t^{\mathrm{d}}$. The point cloud at timestep t is then

$$\mathbb{C}_t = \mathbb{C}^{\mathrm{s}} \cup \mathbb{C}_t^{\mathrm{d}}. \tag{3}$$

Each point stores its observed color, the viewing direction from which it was captured, and a static–dynamic label, which together serve as conditioning information during target-view rendering.

3.2 Scene Memory Construction with LaCAT

We construct a scene memory from the input images and their corresponding camera rays to provide appearance information for target-view rendering. We first review the memory mechanism of LaCT, then describe LaCAT and its use in constructing the scene memory.

Preliminaries: TTT and LaCT. Test-time training (TTT) (Sun et al., 2025) provides a memory-based alternative to attention layers in Transformers. Standard attention retrieves information for each query by computing a weighted combination of the input values using their corresponding keys. This requires accessing the input keys and values for every query. TTT instead learns key–value associations in the weights W of a small neural network f_W, allowing subsequent queries to retrieve information by evaluating this network. These weights, referred to as *fast weights* (Schlag et al., 2021), store input information through an *update* operation and retrieve it through an *apply* operation. Recently, Large Chunk Test-Time Training (LaCT) (Zhang et al., 2026) has scaled this update to chunks of up to millions of tokens, achieving strong results across diverse tasks such as novel-view synthesis, language modeling, and autoregressive video generation.

Specifically, each input token x_j is projected into a key k_j, a value v_j, and a query q_j. The update trains the memory network to predict each token's value from its key:

$$f_W(k_j) \approx v_j. \tag{4}$$

LaCT learns these associations jointly over a large chunk of input tokens by adjusting W to minimize the prediction loss. The apply then retrieves information from the updated memory using q_j:

$$o_j = f_W(q_j). \tag{5}$$

The quality of this readout depends on how effectively input information is stored in the memory. In LaCT, each TTT layer is preceded by window attention and followed by a feedforward layer, together forming a LaCT block. Although window attention allows direct information exchange within each local window, the subsequent TTT update trains the memory to store information from each token without any objective for capturing relationships between tokens. Consequently, the memory may not fully capture relationships across input, limiting synthesis quality.

Large Chunk Attentive Test-Time Training. To build a memory that explicitly captures relationships between tokens, we propose Large Chunk Attentive Test-Time Training (LaCAT). For each token, LaCAT uses attention to gather information from all input tokens. We incorporate this information into the memory by adding the attention response a_j to the token's own value v_j and training the memory network to predict the combined target:

$$f_W(q_j) \approx v_j + \gamma a_j, \qquad a_j = \mathrm{Attn}\big(q_j,\, K,\, V\big), \tag{6}$$

where K and V contain the keys and values of all input tokens, and γ is a learnable scalar controlling the contribution of the attention response. We use q_j instead of k_j as the input to f_W so that it matches the query used to compute a_j. The target retains the token's own information through v_j, alongside the information gathered from other tokens through a_j. Our ablation shows that this combined target yields better rendering quality than using the attention response alone. Each LaCAT block retains the window-attention and feedforward layers of LaCT, but fits its memory network to the targets defined in Eq. (6). The fast-weight memory size and the apply operation of Eq. (5) remain unchanged.

Scene Memory Construction from Input Frames. Our renderer stacks LaCAT blocks, each with its own fast weights. For each input image $\mathbf{I}_t^k$, we concatenate the image with its pixel-wise Plücker ray map computed from $\mathbf{P}_t^k$. We divide the result into patches and map them to tokens using a learned linear projection. Tokens from all input frames pass through the stacked LaCAT blocks, fitting each block's fast weights to the targets defined in Eq. (6). The resulting fast weights collectively form the scene memory used during target-view rendering.

3.3 Geometry-Guided Target-View Rendering

Target-View Conditioning. To provide the renderer with explicit geometric guidance for the target view, we take the 4D point cloud $\mathbb{C}_t$ at the target timestep, denoted $\mathbb{C}_{\mathrm{tgt}}$, and project it onto the

target camera $\mathbf{P}^{\mathrm{tgt}}$. Rasterization produces a conditioning map $\mathbf{A}$ aligned with the target view and at the same spatial resolution as the output image:

$$\mathbf{A} = \big[\mathbf{A}_c,\, \mathbf{A}_\omega,\, \mathbf{A}_d,\, \mathbf{A}_m\big] = \texttt{Rast}\big(\mathbb{C}_{\mathrm{tgt}},\, \mathbf{P}^{\mathrm{tgt}}\big), \tag{7}$$

where $\mathbf{A}_c$ stores the observed color of the point visible at each pixel, $\mathbf{A}_\omega$ stores its observation direction rotated into the target camera frame, $\mathbf{A}_d$ stores its depth in the target view, and $\mathbf{A}_m$ indicates whether it is static or dynamic.

Image Synthesis. We concatenate $\mathbf{A}$ with the target Plücker ray map computed from $\mathbf{P}^{\mathrm{tgt}}$, divide the result into patches, and map them to tokens using a learned linear projection. These target tokens pass through the same LaCAT blocks used for scene memory construction, but each block now reads its stored memory through the apply operation of Eq. (5). The output tokens are decoded into RGB patches and reassembled into the target image $\hat{\mathbf{I}}^{\mathrm{tgt}}$. During rendering, the fast weights remain fixed and LaCAT's cross-view attention is not computed. As a result, the neural rendering cost is unchanged from LaCT, and since point-cloud rasterization is lightweight, the total rendering cost remains nearly constant regardless of the number of input frames.

3.4 Training

Our framework contains three trainable modules: the cluster classification MLP, the refinement U-Net, and the LaCAT renderer. The two segmentation modules are supervised with binary cross-entropy against ground-truth masks:

$$\mathcal{L}_{\mathrm{mask}}\big(\hat{y},\, y\big) = -y \log \hat{y} - (1-y)\log(1-\hat{y}), \tag{8}$$

where $\hat{y}$ is the predicted dynamic probability and y is the binary label. Following prior work (Zhang et al., 2024; Jin et al., 2025a), the renderer is trained with a photometric loss between the synthesized image and the ground-truth target image:

$$\mathcal{L}_{\mathrm{photo}} = \mathcal{L}_{\mathrm{MSE}}\big(\hat{\mathbf{I}}^{\mathrm{tgt}},\, \mathbf{I}^{\mathrm{tgt}}\big) + \lambda_{\mathrm{per}}\, \mathcal{L}_{\mathrm{per}}\big(\hat{\mathbf{I}}^{\mathrm{tgt}},\, \mathbf{I}^{\mathrm{tgt}}\big), \tag{9}$$

where $\mathcal{L}_{\mathrm{MSE}}$ and $\mathcal{L}_{\mathrm{per}}$ are an MSE loss and a perceptual loss (Johnson et al., 2016), respectively, and λ_{per} weights the perceptual loss. Additional training details are provided in the appendix.

4 Experiments

4.1 Experimental Setup

Datasets and Metrics. We evaluate the novel-view synthesis performance of our framework on two monocular benchmarks, DyCheck (Gao et al., 2022) and NVIDIA (Yoon et al., 2020), and two multi-view benchmarks, Neural3D (Li et al., 2022) and REON (Kim et al., 2026a), each with two input views. We report PSNR, SSIM, and LPIPS. We additionally report MEt3R (Asim et al., 2025), which measures multi-view consistency by registering an image pair with a stereo foundation model (Wang et al., 2024) and comparing their features in the overlapping region. Dynamic region segmentation is evaluated on DAVIS16-M (Perazzi et al., 2016) and DAVIS17-M (Pont-Tuset et al., 2017), where we report the mean and recall of the region similarity $\mathcal{J}$ and the boundary accuracy $\mathcal{F}$ at 512 resolution. For ablation, we use NVIDIA (Yoon et al., 2020), which provides ground-truth motion masks, and RealEstate10K (Zhou et al., 2018) to evaluate the renderer independently of segmentation. All evaluations are conducted on a single NVIDIA H200 GPU. Additional evaluation details are provided in the appendix.

Baselines. We compare GG4D against optimization-based and feedforward methods. MoSca (Lei et al., 2025) and IGS (Yan et al., 2025) are optimization-based methods for the monocular and multi-view settings, respectively. Among feedforward methods, 4DGT (Xu et al., 2025b) and MoVieS (Lin et al., 2026a) support monocular input, REON (Kim et al., 2026a) supports multi-view input, and FSM (Ma et al., 2026) runs in both settings. For dynamic region segmentation, we compare SegAnyMo (Huang et al., 2025), VGGT4D (Hu et al., 2026), and GeoMotion (He et al., 2026).

4.2 Dynamic Novel-View Synthesis

Table 1 and Fig. 2 summarize the comparison. Our single framework achieves the best image quality and multi-view consistency across all four benchmarks under both monocular and multi-view input settings. For monocular inputs, it surpasses the best feedforward baseline by 3.6 dB mPSNR on

Table 1: Quantitative comparison on dynamic NVS. "–" denotes settings a method does not support.

Method	Monocular								Multi-view							
	DyCheck (Gao et al., 2022)				NVIDIA (Yoon et al., 2020)				Neural3D (Li et al., 2022)				REON (Kim et al., 2026a)			
	mPSNR↑	mSSIM↑	mLPIPS↓	MEt3R↓	PSNR↑	SSIM↑	LPIPS↓	MEt3R↓	PSNR↑	SSIM↑	LPIPS↓	MEt3R↓	PSNR↑	SSIM↑	LPIPS↓	MEt3R↓
Optimization-based																
MoSca (Lei et al., 2025)	13.94	0.311	0.234	0.553	21.01	0.619	0.163	0.271	–	–	–	–	–	–	–	–
IGS (Yan et al., 2025)	–	–	–	–	–	–	–	–	15.03	0.412	0.609	0.626	19.25	0.509	0.459	0.412
Feedforward																
4DGT (Xu et al., 2025b)	12.10	0.287	0.250	0.430	14.67	0.319	0.640	0.509	–	–	–	–	–	–	–	–
MoVieS (Lin et al., 2026a)	13.79	0.345	0.205	0.415	18.11	0.536	0.269	0.257	–	–	–	–	–	–	–	–
FSM (Ma et al., 2026)	14.74	0.357	0.224	0.476	20.54	0.544	0.314	0.168	14.70	0.488	0.630	0.436	20.04	0.511	0.301	0.158
REON (Kim et al., 2026a)	–	–	–	–	–	–	–	–	20.63	0.625	0.396	0.251	23.37	0.615	0.311	0.219
Ours	18.38	0.397	0.138	0.371	22.23	0.661	0.144	0.164	23.27	0.770	0.147	0.226	26.67	0.762	0.143	0.128

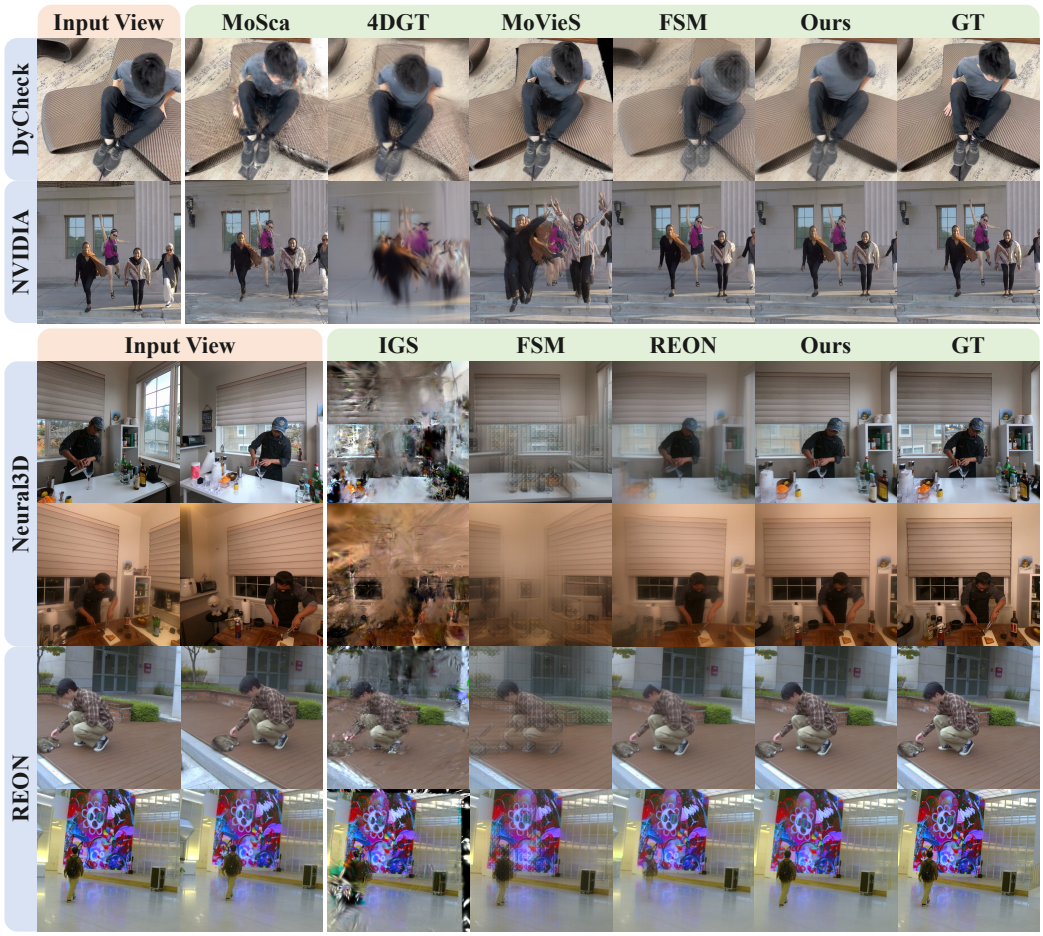


Figure 2: Qualitative comparison on dynamic novel-view synthesis.

DyCheck (Gao et al., 2022) and 1.7 dB PSNR on NVIDIA (Yoon et al., 2020). This advantage carries over to multi-view inputs, reaching +2.6 dB PSNR on Neural3D (Li et al., 2022) and +3.3 dB on REON (Kim et al., 2026a). Beyond image quality, GG4D also achieves the lowest MEt3R on every benchmark, indicating stronger multi-view consistency. Optimization-based methods (Lei et al., 2025; Yan et al., 2025) tend to overfit to sparse input views, degrading both multi-view consistency and novel-view rendering quality. Notably, FSM (Ma et al., 2026), which reconstructs the scene implicitly without explicit geometry, loses consistency under large viewpoint changes in DyCheck, yielding the worst MEt3R among feedforward methods on DyCheck despite its high PSNR. In contrast, GG4D combines an explicit 3D representation with implicit synthesis, preserving strong 3D consistency while leveraging implicit synthesis for high image quality.

Table 2: Quantitative results on dynamic region segmentation.

Method	DAVIS16-M (Perazzi et al., 2016)				DAVIS17-M (Pont-Tuset et al., 2017)				Time (s)↓
	$\mathcal{J}$-Mean↑	$\mathcal{J}$-Recall↑	$\mathcal{F}$-Mean↑	$\mathcal{F}$-Recall↑	$\mathcal{J}$-Mean↑	$\mathcal{J}$-Recall↑	$\mathcal{F}$-Mean↑	$\mathcal{F}$-Recall↑	
SegAnyMo (Huang et al., 2025)	0.7556	0.7859	0.8168	0.8354	0.7541	0.7984	0.7912	0.8227	226
VGGT4D (Hu et al., 2026)	0.5856	0.6048	0.4908	0.5218	0.5478	0.6165	0.4582	0.4937	95
GeoMotion (He et al., 2026)	0.7595	0.8775	0.7046	0.8138	0.7824	0.8985	0.7159	0.8151	16
Ours	0.8139	0.9266	0.7876	0.9021	0.8342	0.9353	0.8068	0.9099	3.4

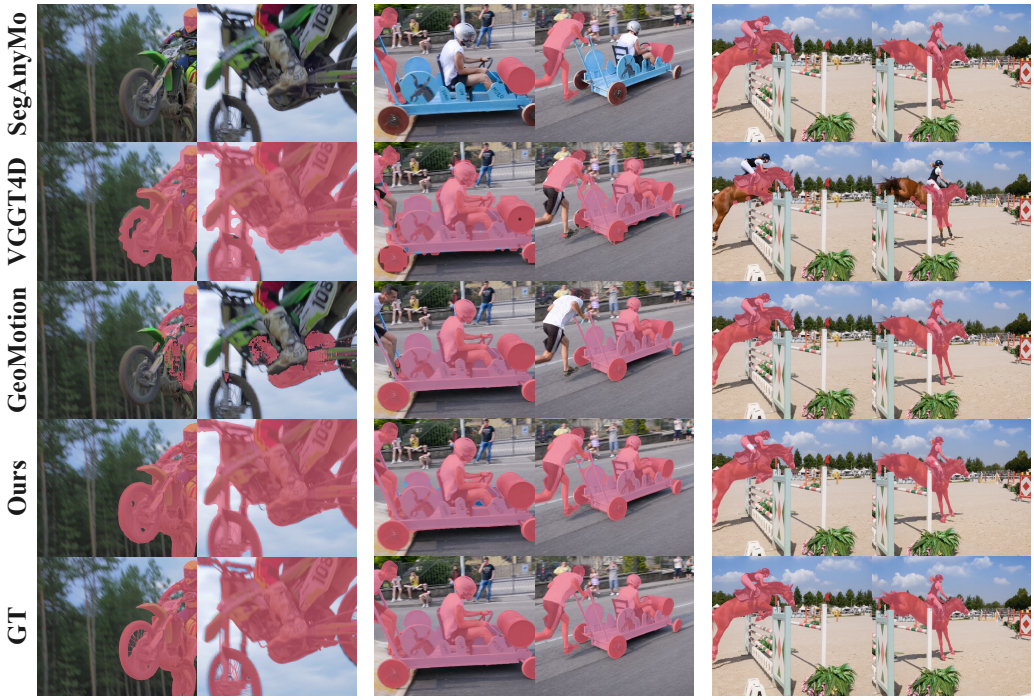


Figure 3: Qualitative comparison of dynamic region segmentation on DAVIS16-M(Perazzi et al., 2016).

4.3 Dynamic Region Segmentation

Table 2 compares our dynamic region segmentation with other pipelines on DAVIS16-M (Perazzi et al., 2016) and DAVIS17-M (Pont-Tuset et al., 2017). Reported time is the average per-sequence wall-clock time on DAVIS16-M, whose sequences have 68 frames on average, including the forward pass of all off-the-shelf models and geometric backbones. SegAnyMo (Huang et al., 2025) runs BootsTAP (Doersch et al., 2024) for long-range tracks, Depth-Anything (Yang et al., 2024a) for depth, DINOv2 (Oquab et al., 2024) for semantic features, and SAM2 (Ravi et al., 2024) for mask densification. SAM2 gives it strong boundaries, and its reliance on many external models makes it the slowest. VGGT4D (Hu et al., 2026) and GeoMotion (He et al., 2026) both read the internals of a geometric foundation model. VGGT4D mines Gram similarity statistics from the global attention layers of VGGT (Wang et al., 2025a) without any training, limiting its accuracy. GeoMotion learns a per-pixel decoder over π^3 (Wang et al., 2026c) features together with RAFT (Teed & Deng, 2020) optical flow. Yet, deciding each pixel independently leaves coarse boundaries and partial coverage of dynamic regions. In contrast, our segmentation classifies semantically coherent clusters with a learned small MLP and refines them with a CNN UNet on top of VGGT-Ω features that our pipeline already computes for geometry. This avoids dependence on off-the-shelf optical flow or segmentation models. Consequently, as shown in Fig. 3, our framework enables accurate segmentation while running faster.

4.4 Ablation and Analysis

Dynamic Region Segmentation. We ablate the dynamic region segmentation in terms of both mask quality and its effect on rendering. Table 3 compares tokens and clusters as the unit of classification. Token-level classification underperforms on all metrics, indicating less stable predictions,

Table 3: Classification unit ablation.

Unit	DAVIS16-M (Perazzi et al., 2016)		
	$\mathcal{J}\&\mathcal{F}\uparrow$	$\mathcal{J}\uparrow$	$\mathcal{F}\uparrow$
Token	0.7092	0.7207	0.6978
Cluster	0.8007	0.8139	0.7876

Table 4: Ablation on masking.

Mask	NVIDIA (Yoon et al., 2020)		
	PSNR↑	SSIM↑	LPIPS↓
None	19.89	0.626	0.237
Ours	23.50	0.670	0.135
GT mask	23.61	0.676	0.133

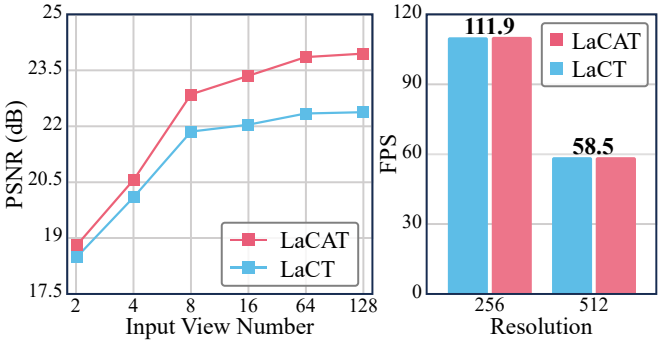


Figure 4: Rendering quality and speed of LaCT (Zhang et al., 2026) and our proposed LaCAT.

Table 5: LaCAT ablation.

Update Rule	RE10K (Zhou et al., 2018)			NVIDIA (Yoon et al., 2020)		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
LaCT (Zhang et al., 2026)	22.31	0.650	0.261	21.27	0.449	0.467
LaCAT ($\gamma = 0$)	22.76	0.689	0.217	21.44	0.473	0.453
LaCAT (a_j only)	23.31	0.706	0.208	21.67	0.457	0.464
LaCAT (intra)	23.79	0.724	0.193	21.79	0.486	0.446
LaCAT (full)	24.12	0.734	0.180	21.86	0.486	0.440

Table 6: Point attribute ablation.

Point Attributes	RE10K (Zhou et al., 2018)		
	PSNR↑	SSIM↑	LPIPS↓
RGB only	20.37	0.643	0.307
+ viewing direction	21.12	0.657	0.283
+ depth	21.23	0.660	0.279

whereas cluster-level classification provides more consistent coverage of dynamic regions. Table 4 evaluates how dynamic masking affects point cloud construction and the resulting rendering quality. Our mask prevents dynamic geometry from accumulating across time, improving rendering quality over using no mask and nearly matching the results obtained with the ground-truth mask.

LaCAT. We ablate the design of LaCAT in Table 5, using 128 input views on RealEstate10K and the same protocol as the main comparison on NVIDIA. As described in Eq. (6), LaCAT stores, at each input query q_j, the value v_j augmented with the attention output a_j that this query receives from the other input tokens. Compared to LaCAT (full), storing v_j alone without the attention output ($\gamma = 0$) or storing a_j alone without the value (a_j only) both degrade the results. Moreover, restricting the attention to tokens within the same view (intra) also falls short of LaCAT (full), showing that the cross-view exchange is beneficial. Fig. 4 further shows that this gain over LaCT holds consistently as the number of input views grows, while both run at the same FPS in the rendering stage, confirming that the attention in the update step adds no rendering cost. Finally, Table 6 with 8 input views shows that adding viewing direction and depth to the target-view conditioning map improves rendering quality over RGB alone.

5 Conclusion

We introduced GG4D, a feedforward dynamic NVS framework that combines the multi-view consistency of an explicit 4D point cloud with the rendering fidelity of implicit image synthesis. Given an input video, GG4D builds a 4D point cloud and reusable scene memory, then synthesizes target images from the point-cloud projection and scene memory. To construct the point cloud, we propose efficient and accurate dynamic region segmentation that reuses frozen geometric foundation model features without off-the-shelf optical flow or segmentation models. We further introduce LaCAT, enabling direct cross-view interaction during memory construction to improve rendering quality without increasing per-view neural rendering cost. Reusing both representations separates scene construction from rendering, sharing the preparation cost across viewing trajectories. Experiments demonstrate state-of-the-art rendering quality and multi-view consistency in monocular and multi-view settings, with 58 FPS rendering at 512 resolution after scene construction, marking a step toward interactive, immersive applications.

Limitation. Like prior deterministic dynamic NVS approaches (Xu et al., 2025b; Lin et al., 2026a; Ma et al., 2026; Kim et al., 2026a), our method produces blurry artifacts in unseen regions, and its quality depends on the accuracy of the geometric foundation model. We discuss these limitations and future directions further in the appendix.

AI USE STATEMENT

In this work, we used generative AI tools for translation assistance and for implementing parts of the experimental code. The research direction, the method, and the experimental design were developed by the authors without generative AI, and no such tools were used to derive or verify mathematical statements, to build the conceptual framework, to form or refine hypotheses, to prepare or clean datasets, or to analyze and interpret the results. Synthetic data generation does not apply to this work. Additionally, we used generative AI tools for drafting and revising portions of the manuscript, improving readability, and refining its structure. We have reviewed all AI-assisted work: AI-assisted code was verified and tested by the authors, and all AI-assisted text was checked for accuracy and consistency with our research. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

All datasets used in this study were used in accordance with their respective terms of use. We acknowledge the dataset creators and cite the original publications. We consider the potential for misuse to be limited, as our method focuses on synthesizing novel viewpoints of captured scenes.

REPRODUCIBILITY STATEMENT

Section 3 describes the framework and Section 3.4 the training objectives. Section A.1 provides the training datasets, architectural details, and training configurations, Section A.2 the full formulation of LaCAT, and Sections A.3 and 4.1 the benchmarks and evaluation protocols for our method and all baselines. We plan to release the code and trained models after publication.

REFERENCES

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

Jianhong Bai, Menghan Xia, Xiao Fu, Xintao Wang, Lianrui Mu, Jinwen Cao, Zuozhu Liu, Haoji Hu, Xiang Bai, Pengfei Wan, and Di Zhang. ReCamMaster: Camera-controlled generative rendering from a single video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

Ali Behrouz, Zeman Li, Praneeth Kacham, Majid Daliri, Yuan Deng, Peilin Zhong, Meisam Razaviyayn, and Vahab Mirrokni. Atlas: Learning to optimally memorize the context at test time. *arXiv preprint arXiv:2505.23735*, 2025a.

Ali Behrouz, Meisam Razaviyayn, Peilin Zhong, and Vahab Mirrokni. It's all connected: A journey through test-time memorization, attentional bias, retention, and online optimization. *arXiv preprint arXiv:2504.13173*, 2025b.

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

David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelSplat: 3D gaussian splats from image pairs for scalable generalizable 3D reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 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. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

Carl Doersch, Pauline Luc, Yi Yang, Dilara Gokay, Skanda Koppula, Ankush Gupta, Joseph Heyward, Ignacio Rocco, Ross Goroshin, João Carreira, and Andrew Zisserman. BootsTAP: Bootstrapped training for tracking-any-point. *arXiv preprint arXiv:2402.00847*, 2024.

FIXME Feng et al. In-place test-time training. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2604.06169.

Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *ICCV*, 2021.

Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. Monocular dynamic view synthesis: A reality check. In *NeurIPS*, 2022.

Xiankang He, Peile Lin, Ying Cui, Dongyan Guo, Chunhua Shen, and Xiaoqin Zhang. GeoMotion: Rethinking motion segmentation via latent 4D geometry. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Yu Hu, Chong Cheng, Sicheng Yu, Xiaoyang Guo, and Hao Wang. VGGT4D: Mining motion cues in visual geometry transformers for 4D scene reconstruction. In *Findings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Nan Huang, Wenzhao Zheng, Chenfeng Xu, Kurt Keutzer, Shanghang Zhang, Angjoo Kanazawa, and Qianqian Wang. Segment any motion in videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Hanwen Jiang, Hao Tan, Peng Wang, Haian Jin, Yue Zhao, Sai Bi, Kai Zhang, Fujun Luan, Kalyan Sunkavalli, Qixing Huang, and Georgios Pavlakos. Rayzer: A self-supervised large view synthesis model. *arXiv preprint arXiv:2505.00702*, 2025.

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 *International Conference on Learning Representations (ICLR)*, 2025a.

Linyi Jin, Richard Tucker, Zhengqi Li, David Fouhey, Noah Snavely, and Aleksander Holynski. Stereo4d: Learning how things move in 3d from internet stereo videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025b.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 2016.

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

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023.

Daeyeon Kim, Jinhyeok Kim, Gangmin Kwon, Seungjoo Shin, and Sunghyun Cho. Reon-nvs: Real-time online novel-view synthesis from sparse-view videos. In *European Conference on Computer Vision*, 2026a.

Evan Kim, Hyunwoo Ryu, Thomas W. Mitchel, and Vincent Sitzmann. Scaling view synthesis transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026b.

Mungyeom Kim, Minkyeong Jeon, Honggyu An, Jaewoo Jung, Hyunah Ko, Jisang Han, Hyeonseo Yu, Donghwan Shin, Sunghwan Hong, Takuya Narihira, Kazumi Fukuda, Yuki Mitsufuji, and Seungryong Kim. Learning global motion with compact gaussians for feed-forward 4d reconstruction. *arXiv preprint arXiv:2605.31595*, 2026c.

Jiahui Lei, Yijia Weng, Adam Harley, Leonidas Guibas, and Kostas Daniilidis. MoSca: Dynamic gaussian fusion from casual videos via 4D motion scaffolds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, and Zhaoyang Lv. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Hanxue Liang, Jiawei Ren, Ashkan Mirzaei, Antonio Torralba, Ziwei Liu, Igor Gilitschenski, Sanja Fidler, Cengiz Oztireli, Huan Ling, Zan Gojcic, and Jiahui Huang. Feed-forward bullet-time reconstruction of dynamic scenes from monocular videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Chenguo Lin, Yuchen Lin, Panwang Pan, Yifan Yu, Tao Hu, Honglei Yan, Katerina Fragkiadaki, and Yadong Mu. Movies: Motion-aware 4d dynamic view synthesis in one second. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, 2026a.

Chieh Hubert Lin, Zhaoyang Lv, Songyin Wu, Zhen Xu, Thu Nguyen-Phuoc, Hung-Yu Tseng, Julian Straub, Numair Khan, Lei Xiao, Ming-Hsuan Yang, Yuheng Ren, Richard Newcombe, Zhao Dong, and Zhengqin Li. DGS-LRM: Real-time deformable 3D gaussian reconstruction from monocular videos. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Kuan Heng Lin, Zhizheng Liu, Pablo Salamanca, Yash Kant, Ryan Burgert, Yuancheng Xu, Koichi Namekata, Yiwei Zhao, Bolei Zhou, Micah Goldblum, Paul Debevec, and Ning Yu. Vista4D: Video reshooting with 4d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 32671–32682, June 2026b.

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

Yu-Lun Liu, Chen Gao, Andreas Meuleman, Hung-Yu Tseng, Ayush Saraf, Changil Kim, Yung-Yu Chuang, Johannes Kopf, and Jia-Bin Huang. Robust dynamic radiance fields. In *CVPR*, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

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

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*, 2011.

Tsendsuren Munkhdalai, Manaal Faruqui, and Siddharth Gopal. Leave no context behind: Efficient infinite context transformers with infini-attention. *arXiv preprint arXiv:2404.07143*, 2024.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, et al. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.

FIXME Ouyang, FIXME Cai, and FIXME Hu. Fixme title. *arXiv preprint arXiv:2606.21803*, 2026.

Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. *ICCV*, 2021a.

Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *ACM Trans. Graph.*, 40(6), dec 2021b.

F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*, 2016.

Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017.

Pedro Quesado, Erkut Akdag, Yasaman Kashefbahrami, Willem Menu, and Egor Bondarev. LiveStre4m: Feed-forward live streaming of novel views from unposed multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024.

Xuanchi Ren, Tianchang Shen, Jiahui Huang, Huan Ling, Yifan Lu, Merlin Nimier-David, Thomas Müller, Alexander Keller, Sanja Fidler, and Jun Gao. GEN3C: 3D-informed world-consistent video generation with precise camera control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *International Conference on Machine Learning (ICML)*, 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. *arXiv preprint arXiv:2508.10104*, 2025.

Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, Tatsunori Hashimoto, and Carlos Guestrin. Learning to (learn at test time): RNNs with expressive hidden states. In *International Conference on Machine Learning (ICML)*, 2025.

Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision (ECCV)*, 2020.

Johannes von Oswald et al. Mesanet: Sequence modeling by locally optimal test-time training. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2506.05233.

Haoru Wang, Kai Ye, Minghan Qin, Yangyan Li, Wenzheng Chen, and Baoquan Chen. The less you depend, the more you learn: Synthesizing novel views from sparse, unposed images with minimal 3d knowledge. In *International Conference on Learning Representations (ICLR)*, 2026a. URL `https://openreview.net/forum?id=QXc2NBJFHr`.

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 (CVPR)*, 2025a.

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

Qianqian Wang, Vickie Ye, Hang Gao, Weijia Zeng, Jake Austin, Zhengqi Li, and Angjoo Kanazawa. Shape of motion: 4D reconstruction from a single video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025b.

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 (CVPR)*, 2024.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Permutation-equivariant visual geometry learning. In *International Conference on Learning Representations (ICLR)*, 2026c.

Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4D gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Rundi Wu, Ruiqi Gao, Ben Poole, Alex Trevithick, Changxi Zheng, Jonathan T. Barron, and Aleksander Holynski. CAT4D: Create anything in 4D with multi-view video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Zike Wu, Qi Yan, Xuanyu Yi, Lele Wang, and Renjie Liao. StreamSplat: Towards online dynamic 3D reconstruction from uncalibrated video streams. In *International Conference on Learning Representations (ICLR)*, 2026a.

Zirui Wu, Zeren Jiang, Martin R. Oswald, and Jie Song. From rays to projections: Better inputs for feed-forward view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026b.

Haofei Xu, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. Depthsplat: Connecting gaussian splatting and depth. In *CVPR*, 2025a.

Ning Xu, Linjie Yang, Yuchen Fan, Dingcheng Yue, Yuchen Liang, Jianchao Yang, and Thomas Huang. Youtube-vos: A large-scale video object segmentation benchmark. *arXiv preprint arXiv:1809.03327*, 2018.

Zhen Xu, Zhengqin Li, Zhao Dong, Xiaowei Zhou, Richard Newcombe, and Zhaoyang Lv. 4DGT: Learning a 4D gaussian transformer using real-world monocular videos. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025b.

Jinbo Yan, Rui Peng, Zhiyan Wang, Luyang Tang, Jiayu Yang, Jie Liang, Jiahao Wu, and Ronggang Wang. Instant gaussian stream: Fast and generalizable streaming of dynamic scene reconstruction via gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024a.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024b.

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

Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. *arXiv preprint arXiv:2309.13101*, 2023.

Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No pose, no problem: Surprisingly simple 3D gaussian splats from sparse unposed images. In *International Conference on Learning Representations (ICLR)*, 2025.

Jae Shin Yoon, Kihwan Kim, Orazio Gallo, Hyun Soo Park, and Jan Kautz. Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Mark Yu, Wenbo Hu, Jinbo Xing, and Ying Shan. TrajectoryCrafter: Redirecting camera trajectory for monocular videos via diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025a.

Wangbo Yu, Jinbo Xing, Li Yuan, Wenbo Hu, Xiaoyu Li, Zhipeng Huang, Xiangjun Gao, Tien-Tsin Wong, Ying Shan, and Yonghong Tian. ViewCrafter: Taming video diffusion models for high-fidelity novel view synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2025b.

Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. GS-LRM: Large reconstruction model for 3D gaussian splatting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

Tianyuan Zhang, Sai Bi, Yicong Hong, Kai Zhang, Fujun Luan, Songlin Yang, Kalyan Sunkavalli, William T. Freeman, and Hao Tan. Test-time training done right. In *International Conference on Learning Representations (ICLR)*, 2026.

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

A Appendix

A.1 Additional Implementation Details

Our renderer consists of 24 LaCAT blocks. Each block takes tokens of dimension $d = 768$, and images are tokenized with a patch size of $p = 8$. Except for the update rule, all architectural details follow LaCT (Zhang et al., 2026). The renderer is trained in two stages. In the first stage, we train at 128 resolution for 100K steps on the static datasets, RealEstate10K (Zhou et al., 2018) and DL3DV (Ling et al., 2024), with 8 input views and 8 target views per scene and a total batch size of 64. In the second stage, we fine-tune at 512 resolution for 20K steps on the mixture of the static datasets and Stereo4D (Jin et al., 2025b), with a total batch size of 16. Both stages use the AdamW optimizer (Loshchilov & Hutter, 2019) with $\beta_1 = 0.9$ and $\beta_2 = 0.95$, and a cosine learning rate schedule whose peak learning rate is 4e-4 in the first stage and 1e-4 in the second. We set the perceptual loss weight to $\lambda_{\text{per}} = 0.5$ in both training stages. All training runs on 4 NVIDIA H200 GPUs.

The cluster classifier is a three-layer MLP with a hidden dimension of 512, which takes as input the 1024-dimensional mean feature of each cluster from the global attention layer of VGGT-Ω. The refinement network is a four-scale CNN UNet (Ronneberger et al., 2015) with 32 base channels, which takes the image, the coarse mask, and the normalized disparity as a 5-channel input at 512 resolution. For clustering, we apply average-linkage agglomerative clustering (Müllner, 2011) on the cosine distance of L2-normalized encoder features with a merge threshold of 0.4. A cluster is marked dynamic when its predicted probability exceeds $\tau = 0.5$, and the refined mask is binarized at 0.5. Both modules are trained separately on YouTube-VOS 2019 (Xu et al., 2018) and SA-V (Ravi et al., 2024). We manually filter the training samples to retain only those whose annotated objects move within the sampled frames. A cluster is labeled dynamic when more than half of its area is covered by the ground-truth mask, and the refinement network is trained on the coarse mask derived from the ground truth. Each module is trained for 5K steps with a total batch size of 16 on 2 A100 GPUs, using AdamW (Loshchilov & Hutter, 2019) with a peak learning rate of 1e-3 and a cosine schedule.

A.2 Formulation of LaCAT

We give the update rules behind the simplified forms in Section 3.2. LaCT (Zhang et al., 2026) updates the fast weights W over a chunk of N input tokens as

$$W \leftarrow W - \nabla_W \sum_{j=1}^{N} \eta_j \, \mathcal{L}\big(f_W(k_j),\, v_j\big), \tag{10}$$

where $\mathcal{L}$ is the loss between the prediction and the value, and η_j is a per-token learning rate predicted from the token. LaCAT changes the fitted pair from (k_j, v_j) to $(q_j,\, v_j + \gamma a_j)$:

$$W \leftarrow W - \nabla_W \sum_{j=1}^{N} \eta_j \, \mathcal{L}\big(f_W(q_j),\, v_j + \gamma\, a_j\big), \qquad a_j = \mathrm{Attn}\big(q_j,\, K,\, V\big), \tag{11}$$

where K and V stack the keys and values of all input tokens, and γ is a learnable scalar per block initialized at zero, so that LaCAT starts from LaCT with query addressing. The loss, the optimizer, the per-token learning rates, and the apply operation $o_j = f_W(q_j)$ are those of LaCT, and the attention is computed once per scene during the update.

A.3 Additional Evaluation Details

For novel-view synthesis, all methods receive the same input frames, target views, and camera parameters. The input cameras are estimated by running VGGT-Ω (Wang et al., 2026b) on the input frames only, and the target cameras are obtained by aligning the ground-truth poses to this estimated coordinate frame with a Sim(3) transformation. DyCheck (Gao et al., 2022) is evaluated following prior work (Yang et al., 2026; Kim et al., 2026c), by sampling 32 frames with a stride of 2 as input and rendering the other two cameras at the same timesteps. For NVIDIA (Yoon et al., 2020), we follow prior work (Gao et al., 2021; Liu et al., 2023; Liang et al., 2025; Lin et al., 2026a) and sample

Table 7: Classification unit ablation.

Classification Unit	DAVIS17-M(Pont-Tuset et al., 2017)			
	$\mathcal{J}$-Mean↑	$\mathcal{J}$-Recall↑	$\mathcal{F}$-Mean↑	$\mathcal{F}$-Recall↑
Token	0.7720	0.8407	0.7410	0.8124
Cluster (Ours)	0.8342	0.9353	0.8068	0.9099

Table 8: Global attention layer ablation.

Global layer	DAVIS16-M(Perazzi et al., 2016)			
	$\mathcal{J}$-Mean↑	$\mathcal{J}$-Recall↑	$\mathcal{F}$-Mean↑	$\mathcal{F}$-Recall↑
4	0.7596	0.8636	0.7224	0.8050
11	0.8114	0.9201	0.7674	0.8606
17	0.7922	0.9101	0.7550	0.8457
23	0.8139	0.9266	0.7876	0.9021

12 input views in a round-robin style, where the i-th frame comes from the i-th camera at timestep i, and the remaining 11 cameras serve as targets. For the masking ablation in Table 4, we exclude the NVIDIA scenes whose ground-truth motion masks cover only part of the moving objects. For the multi-view benchmarks, Neural3D (Li et al., 2022) takes cam09 and cam02 as input and cam00 as the target, and REON (Kim et al., 2026a) uses the two outer cameras as input and the center camera as the target, both over 64 timesteps. On the REON benchmark, all methods are evaluated on the 19 scenes for which REON results are available, with the 256-resolution ground truth upsampled to 512. For MEt3R (Asim et al., 2025) evaluation, we compare the synthesized target view with the input view at the same timestep. For dynamic region segmentation, following prior work (Huang et al., 2025; Hu et al., 2026; He et al., 2026), DAVIS16-M and DAVIS17-M consist of the sequences with moving objects in the validation sets of DAVIS16 (Perazzi et al., 2016) and DAVIS17 (Pont-Tuset et al., 2017). All frames are 512 resolution, and reported time is the average per-sequence inference time on DAVIS16-M. Due to limited computational resources, the LaCAT and point attribute ablations use a smaller renderer with 12 LaCAT blocks and a patch size of 16, trained at 256 resolution for 60K steps with 8 input and 8 target views, and evaluated with 128 and 8 input views, respectively. The FPS measurements in Fig. 4 instead use the full-size model with 24 blocks and a patch size of 8.

A.4 Additional Experiments and Discussions

Ablation on the Classification Unit. Table 7 extends the classification unit ablation to DAVIS17-M (Pont-Tuset et al., 2017) and reports recall as well, where classifying clusters again outperforms classifying tokens on every metric. Both variants read identical features and differ only in what they assign a label to, so the gap isolates the unit of decision from the quality of the representation. Since motion belongs to objects rather than individual tokens, grouping the classification decision over semantically coherent regions aligns the prediction granularity with the level at which the target property is defined, yielding cleaner and more consistent mask assignments.

Table 9: Feature readout ablation.

Readout	DAVIS16-M(Perazzi et al., 2016)				#Param
	$\mathcal{J}$-Mean↑	$\mathcal{J}$-Recall↑	$\mathcal{F}$-Mean↑	$\mathcal{F}$-Recall↑	
Training-free statistics (Hu et al., 2026)	0.4909	0.5264	0.4352	0.4544	0 M
Per-pixel dense head	0.7694	0.9160	0.7100	0.8138	82.72 M
Cluster classification (Ours)	0.8139	0.9266	0.7876	0.9021	5.11 M

Ablation on the Feature Readout. Table 9 compares three different readouts that convert global attention features of the frozen VGGT-Ω (Wang et al., 2026b) into a dynamic mask. Specifically, we consider training-free Gram similarity statistics in the style of VGGT4D (Hu et al., 2026), a per-pixel dense head consisting of six attention layers, and our cluster classification, where the two learnable readouts are trained separately. Without any training, the statistics-based readout recovers only part of the moving objects and yields the lowest scores on every metric. The per-pixel dense head substantially narrows this gap and reaches a high $\mathcal{J}$-Recall of 0.9160, but its $\mathcal{F}$-Mean remains at 0.7100, as deciding each pixel independently leaves coarse boundaries. Moreover, it requires 82.72 M trainable parameters. In contrast, our cluster classification achieves the best results on every metric with only 5.11 M parameters, roughly 16× fewer than the dense head, and shows clear margins on the boundary metrics. Since all variants share the same backbone and ours uses far fewer parameters, the gain stems from the readout design itself rather than from the parameter budget, namely from classifying semantically coherent clusters instead of individual pixels.

We next fix the readout to our cluster classification and vary the global attention layer of VGGT-Ω from which the features are taken in Table 8. All variants are trained from scratch under the same setting. The shallow layer 4 performs worst, and accuracy rises sharply at layer 11. Although layer 17 slightly underperforms layer 11, the last layer 23 achieves the best results on every metric. We attribute this trend to deeper layers accumulating richer cross-frame relationships, which are informative for distinguishing dynamic regions. We therefore extract features from the last layer.

Table 10: Rendering quality in dynamic and static regions.

Method	NVIDIA (Yoon et al., 2020)						Neural3D (Li et al., 2022)					
	Dynamic region			Static region			Dynamic region			Static region		
	mPSNR↑	mSSIM↑	mLPIPS↓	mPSNR↑	mSSIM↑	mLPIPS↓	mPSNR↑	mSSIM↑	mLPIPS↓	mPSNR↑	mSSIM↑	mLPIPS↓
FSM (Ma et al., 2026)	17.07	0.526	0.266	23.44	0.642	0.274	9.71	0.314	0.608	15.50	0.558	0.569
REON (Kim et al., 2026a)	–	–	–	–	–	–	22.21	0.678	0.350	19.16	0.608	0.434
Ours	18.69	0.572	0.169	24.75	0.721	0.109	25.47	0.797	0.161	23.33	0.787	0.134

Rendering Quality in Dynamic and Static Regions. Table 10 splits the results on NVIDIA (Yoon et al., 2020) and Neural3D (Li et al., 2022) into dynamic and static regions. We split each target image into dynamic and static regions using our predicted dynamic masks, applying the same masks to all methods. GG4D outperforms the feedforward baselines in both regions on every metric, as the accumulated static point cloud provides dense geometric guidance for the background, while the per-timestep dynamic points anchor moving objects at the target time.

Table 11: Dynamic region segmentation with and without SAM2 refinement.

Method	SAM2	DAVIS16-M(Perazzi et al., 2016)				DAVIS17-M(Pont-Tuset et al., 2017)			
		$\mathcal{J}$-Mean↑	$\mathcal{J}$-Recall↑	$\mathcal{F}$-Mean↑	$\mathcal{F}$-Recall↑	$\mathcal{J}$-Mean↑	$\mathcal{J}$-Recall↑	$\mathcal{F}$-Mean↑	$\mathcal{F}$-Recall↑
VGGT4D (Hu et al., 2026)	×	0.5856	0.6048	0.4908	0.5218	0.5478	0.6165	0.4582	0.4937
GeoMotion (He et al., 2026)	×	0.7595	0.8775	0.7046	0.8138	0.7824	0.8985	0.7159	0.8151
Ours	×	0.8139	0.9266	0.7876	0.9021	0.8342	0.9353	0.8068	0.9099
SegAnyMo (Huang et al., 2025)	✓	0.7556	0.7859	0.8168	0.8354	0.7541	0.7984	0.7912	0.8227
GeoMotion (He et al., 2026)	✓	0.8431	0.9253	0.8234	0.8517	0.8169	0.8862	0.7943	0.8416
Ours	✓	0.8760	0.9699	0.8614	0.9241	0.8727	0.9492	0.8561	0.9265

Effect of SAM2 Refinement. Table 11 reports results with SAM2 (Ravi et al., 2024) refinement. In Table 2, we evaluate GeoMotion without SAM2 refinement, since its decoder already predicts dense motion masks and it applies SAM2 only as a test-time refinement to increase their resolution. We keep SAM2 for SegAnyMo, which relies on it to turn sparse track labels into masks. Applying the same refinement to our masks also improves accuracy, and our method remains the best under both settings.

Effect of Geometry Conditioning. To isolate the contribution of explicit geometry, we compare two renderers that differ only in whether the point-cloud conditioning map is provided. Both use the smaller renderer described in the evaluation details. Since a renderer without geometry conditioning receives no signal about the target timestep, we evaluate both per frame on DyCheck (Gao et al., 2022), using only the input frame at the target timestep. As shown in Table 12, point-cloud conditioning improves both rendering quality and multi-view consistency. This is consistent with the findings of PVSM (Wu et al., 2026b), which shows that point-cloud conditioning provides effective geometric guidance for implicit image synthesis.

Table 12: Effect of geometry conditioning.

Conditioning	DyCheck(Gao et al., 2022)			
	mPSNR↑	mSSIM↑	mLPIPS↓	MEt3R↓
Plücker Ray	14.36	0.396	0.249	0.568
+ Point cloud	17.89	0.428	0.211	0.465

Latency Breakdown. Table 13 breaks down the runtime of GG4D on a single NVIDIA H200 GPU. The pipeline consists of two parts. The first part depends only on the input video and runs once per scene. It includes geometry estimation with VGGT-Ω (Wang et al., 2026b), cluster classification, the CNN UNet, and the LaCAT update. The second part depends on the target camera and runs once for every rendered view. It includes rasterization of the 4D point cloud and the LaCAT apply and decoding step.

Table 13: Latency breakdown.

# Input Views	Once per scene (ms)				Per target view (ms)	
	Geometry estimation	Cluster classification	CNN UNet	LaCAT update	Rasterization	LaCAT apply & decoding
8	135.26	83.94	18.80	572.85	3.02	12.48
32	686.13	362.52	60.83	4998.67	4.61	12.48
64	1878.73	686.54	121.39	15142.62	5.13	12.48

Because the first part is computed only once per scene, its cost is amortized over the entire output. Each additional view therefore only adds rasterization and LaCAT apply & decoding. Notably, LaCAT apply & decoding remains constant regardless of the number of input views, keeping the per-view cost nearly constant as the input scales.

Effect of Camera Pose Source. In the main comparison, all methods receive input cameras estimated by VGGT-Ω (Wang et al., 2026b). Table 14 additionally evaluates the baselines with ground-truth camera poses. On NVIDIA (Yoon et al., 2020), MoSca degrades with GT poses because the injected VGGT-Ω depth is predicted jointly with VGGT-Ω poses and intrinsics and is consistent only with those cameras. We align the GT trajectory to the VGGT-Ω frame with one global similarity transform and use GT intrinsics, so relative poses and focal lengths differ from the cameras that produced the depth. MoSca keeps input cameras fixed and corrects only per-frame depth scale and shift, so this mismatch persists in the fused static geometry.

Table 14: Baselines evaluated with estimated (Est.) and ground-truth (GT) camera poses.

Method	Pose	DyCheck(Gao et al., 2022)				NVIDIA(Yoon et al., 2020)				Neural3D(Li et al., 2022)			
		mPSNR↑	mSSIM↑	mLPIPS↓	MEt3R↓	PSNR↑	SSIM↑	LPIPS↓	MEt3R↓	PSNR↑	SSIM↑	LPIPS↓	MEt3R↓
MoSca (Lei et al., 2025)	Est.	13.94	0.311	0.234	0.553	21.01	0.619	0.163	0.271	–	–	–	–
	GT	16.41	0.368	0.176	0.523	18.15	0.390	0.347	0.268	–	–	–	–
IGS (Yan et al., 2025)	Est.	–	–	–	–	–	–	–	–	15.03	0.412	0.609	0.626
	GT	–	–	–	–	–	–	–	–	15.46	0.416	0.596	0.568
4DGT (Xu et al., 2025b)	Est.	12.10	0.287	0.250	0.430	14.67	0.319	0.640	0.509	–	–	–	–
	GT	12.14	0.287	0.247	0.420	14.57	0.317	0.654	0.533	–	–	–	–
MoVieS (Lin et al., 2026a)	Est.	13.79	0.345	0.205	0.415	18.11	0.536	0.269	0.257	–	–	–	–
	GT	14.26	0.354	0.193	0.406	17.93	0.505	0.291	0.265	–	–	–	–
FSM (Ma et al., 2026)	Est.	14.74	0.357	0.224	0.476	20.54	0.544	0.314	0.168	14.70	0.488	0.630	0.436
	GT	14.70	0.352	0.225	0.477	20.90	0.565	0.310	0.167	14.58	0.477	0.632	0.433
REON (Kim et al., 2026a)	Est.	–	–	–	–	–	–	–	–	20.63	0.625	0.396	0.251
	GT	–	–	–	–	–	–	–	–	20.70	0.627	0.393	0.248

Ablation on the Number of Training Input Views. Fig. 5 compares two renderers that differ only in the number of input views during training, 2 and 8, as the number of input views at test time grows from 4 to 64. The renderer trained with 8 views yields higher PSNR at every input view count. Writing more views into the fast weights during training teaches the renderer to compress many observations into W, whereas the renderer trained with 2 views saturates as more input views are given. We therefore train our renderer with 8 input views to generalize better to a large number of input views.

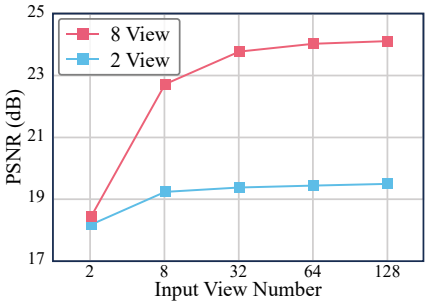


Figure 5: Training with 2 vs. 8 views.

Effect of Evaluation Resolution. FSM (Ma et al., 2026) and REON (Kim et al., 2026a) render at 256 resolution natively, and their outputs are upsampled to 512 for the main comparison. To rule out a bias from this upsampling, Table 15 rescores all methods at 256 resolution by downsampling both the ground truth and the rendered images. Scoring at 256 resolution removes the upsampling penalty, which mainly improves the LPIPS of FSM and REON, while GG4D remains the best on every metric.

Table 15: Evaluation at 512 and 256 resolution. FSM and REON render at 256 natively.

Resolution	Method	DyCheck(Gao et al., 2022)			NVIDIA(Yoon et al., 2020)			Neural3D(Li et al., 2022)			REON(Kim et al., 2026a)		
		mPSNR↑	mSSIM↑	mLPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
512	FSM (Ma et al., 2026)	14.74	0.357	0.224	20.54	0.544	0.314	14.70	0.488	0.630	20.04	0.511	0.301
	REON (Kim et al., 2026a)	–	–	–	–	–	–	20.63	0.625	0.396	23.37	0.615	0.311
	Ours	18.38	0.397	0.138	22.23	0.661	0.144	23.27	0.770	0.147	26.67	0.762	0.143
256	FSM (Ma et al., 2026)	14.92	0.340	0.163	21.30	0.625	0.111	14.77	0.374	0.569	19.96	0.435	0.208
	REON (Kim et al., 2026a)	–	–	–	–	–	–	20.92	0.584	0.247	23.23	0.577	0.186
	Ours	18.84	0.452	0.096	23.65	0.760	0.092	23.83	0.795	0.102	26.46	0.768	0.092

A.5 Discussion on LaCAT.

Test-time training methods update fast weights W so that f_W maps an input to a target, then read them out with a query. Prior work has almost exclusively varied how this memory is written, through the update rule (Schlag et al., 2021; Yang et al., 2024b; Behrouz et al., 2025c; Zhang et al., 2026; Ma et al., 2026) or the loss (Behrouz et al., 2025b;a; von Oswald et al., 2026), while keeping the premise that a token's key is fitted to its own value. The few that change what is written (Feng et al., 2026; Ouyang et al., 2026) build the target from neighboring tokens at preset offsets and use it in place of the value. LaCAT also changes what is written, but differs in two respects. Contributing tokens are chosen by content through attention rather than by position, and the original value is retained rather than replaced. The update rule and loss of LaCT are left untouched. Unlike prior work, LaCAT is therefore let the memory store a target assembled from the whole input while preserving the original association.

Prior hybrids of attention and test-time training place attention either before the fast-weight layer so that keys and values are contextualized before being written (Zhang et al., 2026), or after it so that the attention output is summed with the memory readout (Munkhdalai et al., 2024; Zhang et al., 2026). In contrast, LaCAT runs attention among the input tokens and places its output inside the target, so the memory still learns to associate each query with that target through the same gradient step as LaCT. Output-side mixing recomputes attention per query, whereas target-side mixing runs it once per scene and leaves the apply operation identical to LaCT. Attention thus enters the write operation itself rather than surrounding it.

A.6 Limitations

Our method belongs to deterministic feedforward NVS (Charatan et al., 2024; Chen et al., 2024; Xu et al., 2025a; Jin et al., 2025a; Jiang et al., 2025; Xu et al., 2025b; Lin et al., 2026a; Ma et al., 2026; Kim et al., 2026a), which predicts each target view from the input observations in a single forward pass. The output is therefore determined entirely by what the input views contain, and regions unseen from the input views come out blurry because the renderer has no information to reconstruct them. Generative approaches (Yu et al., 2025b; Wu et al., 2025; Ren et al., 2025; Yu et al., 2025a; Bai et al., 2025; Lin et al., 2026b) instead fill such regions with plausible details through multi-step sampling, at a rendering cost far above real time. Our method also depends on the accuracy of the geometric foundation model. While the implicit renderer can compensate for moderate errors, large depth or pose errors corrupt the conditioning map and degrade rendering quality. In addition, dynamic points exist only at the input timesteps, so our method renders novel views at observed timesteps and does not synthesize novel timesteps. Finally, the attention in the LaCAT update spans all input views, so the update requires the entire input at once and its cost grows with the number of input views, although it runs only once per scene. A promising future direction is to incorporate generative priors while preserving real-time rendering, to reconstruct complete scenes with plausible details in unseen regions.