GAMMA: A Feed-Forward 4D Gaussian Reconstruction Model for Novel View Synthesis of Monocular Videos

Weiqi Zhang[1*] **Junsheng Zhou**[1*] **Xuancheng Zhang**[1,2]
Juntong Fang[1] **Zequn Chen**[2] **Donglin Di**[2] **Yu-Shen Liu**[1†]
[1]School of Software, Tsinghua University, Beijing, China [2]Li Auto
{zwq23, zhou-js24, fangjt21}@mails.tsinghua.edu.cn
{zhangxuancheng,chenzequn,didonglin}@lixiang.com liuyushen@tsinghua.edu.cn

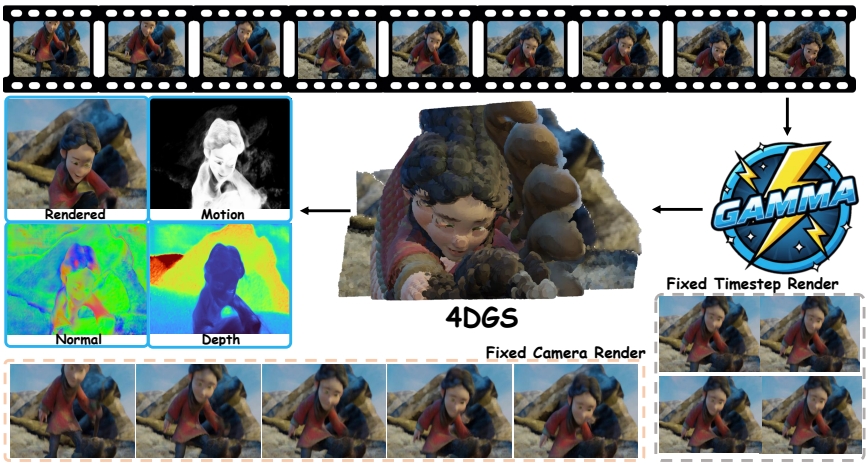


Figure 1: GAMMA reconstructs 4D Gaussian Splatting from monocular videos, enabling real-time novel view synthesis across different timesteps.

Abstract

We present **GAMMA**, a feed-forward model that reconstructs 4D Gaussian Splatting from monocular videos, enabling dynamic novel view synthesis in real-time. Most existing approaches for dynamic scene modeling require multi-view videos as input or costly per-scene optimization. In contrast, GAMMA learns a unified 4D scene representation from monocular videos and directly predicts 4D Gaussian primitives within seconds. The key insight of GAMMA lies in its unified Gaussian reconstruction model, which jointly estimates temporally consistent geometry, appearance, and motion from the input video. We train GAMMA on a mixture of synthetic and real-world videos, and it reconstructs an input clip in a single forward pass without per-scene optimization. The reconstructed GAMMA representation supports interactive scene exploration via real-time rendering across timesteps and local viewpoints. Extensive experiments demonstrate that GAMMA outperforms existing methods in both reconstruction fidelity and efficiency. *Project Page: https://weiqi-zhang.github.io/GAMMA.*

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

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

1 Introduction

This paper aims to address the challenging task of reconstructing 4D scenes from monocular videos, enabling real-time dynamic novel view synthesis across timesteps and local viewpoints. This task is fundamental to various computer vision and graphics applications, including VR/AR, film production, and robotics. Most existing approaches [52, 42, 44, 50] rely on optimization-based frameworks that fit monocular videos to 4D Gaussian Splatting (4DGS) representations using carefully designed priors from pretrained models and geometric constraints. However, these methods typically require hours of per-scene optimization to capture dynamic geometry and appearance.

Recently, feed-forward reconstruction models [66, 37, 43, 15, 26, 20, 5] have shown great promise for achieving efficient and high-quality 3D reconstruction. Starting from DUSt3R [39] and extended by follow-up works such as MASt3R [17] and VGGT [35], these methods predict dense pointmaps from image pairs, enabling 3D reconstruction within seconds. Extending this paradigm to dynamic scenes, emerging 4D reconstruction models aim to capture time-varying geometry and appearance directly from monocular inputs. Nonetheless, they still struggle to produce temporally consistent scene representations, which are essential for high-quality novel view synthesis across timesteps.

The key challenges in 4D scene reconstruction arise from two fundamental aspects: (1) designing a representation that efficiently captures both spatial and temporal variations, and (2) establishing an effective data setup that provides robust supervision. A naive representation like 4D point cloud fails to support seamless dynamic novel view synthesis, as it lacks the ability to handle occlusions and maintain temporal consistency. Similarly, a straightforward supervision scheme that relies on multi-view video datasets suffers from limited scalability, as such datasets are difficult to capture and often insufficient in diversity and quantity for large-scale training.

In this paper, we introduce GAMMA, a feed-forward model that reconstructs 4D Gaussian primitives from monocular videos. Built upon the 4D Gaussian Splatting representation, GAMMA naturally supports real-time dynamic novel view synthesis and achieves superior temporal modeling through fundamental temporal attributes such as the Gaussian timestep and Gaussian duration. GAMMA is a unified 4D reconstruction model that jointly estimates temporally consistent geometry, appearance, and motion directly from monocular inputs. To this end, we design a dual-head architecture that disentangles the dynamic modeling of scene geometry and appearance. Both heads share a common backbone based on a large-scale Multi-View Transformer pretrained on extensive 3D reconstruction data. This backbone provides strong spatial reasoning capabilities and significantly benefits 4D reconstruction.

GAMMA is trained only on posed monocular videos and does not require multi-view video capture. We adopt a two-stage training pipeline that progressively learns geometry, appearance, and dynamics. In the first stage, GAMMA focuses on learning a foundational geometric scaffold and static appearance components that remain stable over time. Building on this foundation, the second stage keeps the lifespan constraint on the longest-lived Gaussians while releasing the rest to represent motion discontinuities, enabling accurate and robust modeling of detailed temporal variations in geometry, appearance, and motion. Through comprehensive learning of temporal attributes, GAMMA exhibits strong capability in representing both static backgrounds and highly dynamic instances, achieving high-fidelity and temporally consistent 4D scene reconstruction.

We train GAMMA on a large-scale dataset collected from both synthetic and real-world sources. Comprehensive evaluations across diverse benchmarks demonstrate that GAMMA consistently outperforms existing feed-forward reconstruction models in both quality and efficiency. Moreover, GAMMA exhibits strong generalization capability, achieving promising results on downstream tasks such as depth estimation without any task-specific training. Our main contributions can be summarized as follows.

- We present GAMMA, a feed-forward approach for 4D Gaussian Splatting reconstruction from monocular videos within seconds. The predicted 4D Gaussians enable real-time novel view synthesis across timesteps and local viewpoints.
- We design a dual-head architecture that disentangles the dynamics of scene geometry and appearance. Both branches share a foundation transformer backbone pretrained at large scale on 3D reconstruction tasks, providing strong spatial reasoning capabilities that significantly enhance 4D reconstruction performance.

- We introduce a two-stage training pipeline that progressively learns 4D scene representations in an easy-to-hard manner. A selective lifespan constraint keeps most Gaussians persistent while allowing the rest to end early at motion discontinuities, leading to accurate and robust modeling of temporal dynamics.

- We evaluate GAMMA on 4D scene reconstruction under both synthetic and real-world datasets, and show our superiority over the state-of-the-art methods.

2 Related Work

2.1 Feed-Forward 3DGS Reconstruction

Feed-forward architectures predict 3D Gaussians directly from images, removing the need for iterative optimization. Stereo-based methods [1] leverage epipolar constraints between image pairs, while volumetric fusion approaches [4] aggregate multi-view information for geometric coherence; sparse-view variants [46] further exploit monocular depth priors. Splatter Image [33] pushes this paradigm to a single RGB input, and follow-ups extend it to uncalibrated [53] and online [41] settings. Scaling this idea, GS-LRM [55] and GRM [47] adopt large reconstruction model (LRM) [9] backbones, demonstrating that transformer-based feed-forward 3DGS scales favorably with data and parameters. Beyond feed-forward reconstruction, Gaussian Splatting has also been used to generate 3D content [62, 60, 59] and to recover complete scenes from sparse views with video diffusion priors [34]. Neural distance fields remain a common representation for reconstructing [28, 29] and generating [63, 64] 3D shapes. Despite these advances, all of the above methods are confined to static scenes and do not support 4D reconstruction.

2.2 4D Gaussian Splatting Reconstruction

Building upon the real-time rendering capabilities of 3D Gaussian Splatting (3DGS) [16], the community has explored several paradigms to extend this representation into the temporal domain. Early efforts establish frame-to-frame correspondences by tracking individual primitives [25], while canonical-space formulations [49, 8, 21, 24, 50, 18] model dynamics through learned deformations of static Gaussians. More recent works treat time as an intrinsic component of the scene representation. 4DGS [51] couples spatial parameters with motion descriptors via angular-space parameterization, and FreeTimeGS [42] allows primitives to appear and move freely across space and time for complex dynamics. To eliminate multi-view dependencies, monocular pipelines such as ProDyG [2] and Shape of Motion [36] cascade pretrained modules for depth, optical flow, point tracking, and segmentation; however, this sequential design incurs heavy computation and suffers from compounding errors that degrade quality on challenging real-world scenes. All these methods rely on costly per-scene optimization, motivating a feed-forward alternative.

Extending the feed-forward paradigm to dynamic scenes remains an open problem, and prior efforts pursue distinct routes. L4GM [32] regresses 4D Gaussians from multi-view video using an LRM-style backbone, but its reliance on dense multi-view inputs limits applicability to in-the-wild monocular footage. Some methods [48, 22] learn a transformer that maps monocular videos to 4D Gaussians directly, yet they degrade on long or fast-motion sequences. A parallel line leverages video generative priors: CAT4D [45] synthesizes multi-view video via diffusion to enable 4D reconstruction, and Geo4D [12] repurposes video generators to recover 4D geometry; both are bounded by the fidelity and dynamics coverage of the underlying generators. Generative models can also produce 4D Gaussians directly, by decoding them from video diffusion latents [57] or by generating animated objects natively [58]. A third group reconstructs dynamic scenes through SfM/bundle-adjustment style estimation, e.g., MegaSaM [19], producing camera trajectories and dynamic point geometry rather than a renderable 4D representation. Feed-forward models likewise predict correspondence, 3D positions, and scene flow for queried points [3], or segment the regions that actually move under camera motion [13]. In contrast, GAMMA learns a unified, temporally consistent 4D Gaussian representation directly from monocular videos in a single forward pass, supporting real-time novel view synthesis across timesteps without per-scene optimization or auxiliary cascaded modules.

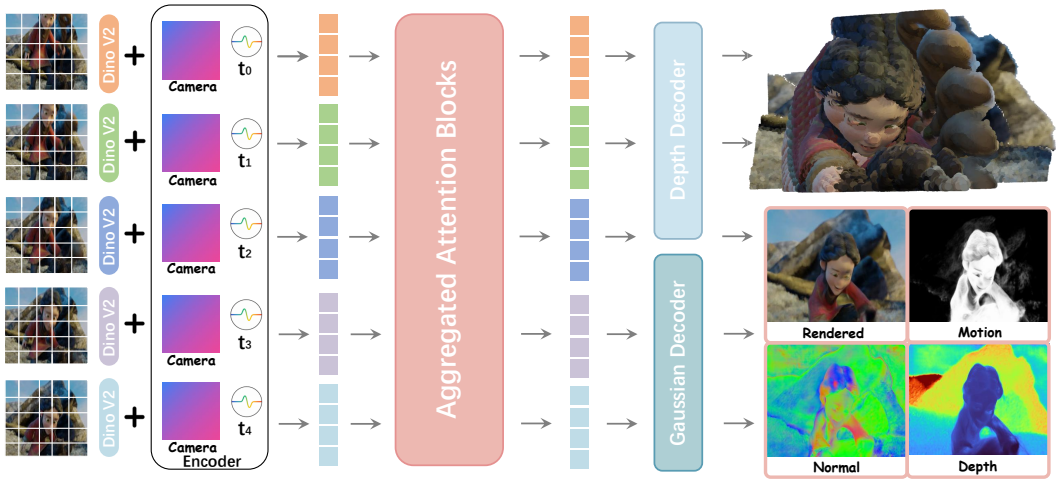


Figure 2: Given a monocular video sequence with T frames (t_0, t_1, ..., $t_{(T-1)}$), each frame is processed through a DINOv2 encoder to extract visual features, while a camera encoder embeds its camera parameters and timestep. The encoded features from all frames are passed through aggregated attention blocks that alternate between frame-wise and global attention to capture spatio-temporal information. The shared backbone features are then decoded through two parallel heads: a depth decoder that predicts per-frame depth maps, and a Gaussian decoder that predicts complete 4D Gaussian parameters including geometry, appearance, motion, and temporal attributes. The reconstructed 4D Gaussians enable real-time rendering of novel views across timesteps, producing rendered RGB images, depth maps, motion fields, and surface normals.

3 Method

We introduce **GAMMA**, a feed-forward framework that maps monocular videos to 4D Gaussian representations through spatio-temporal modeling. Given an input video $V = \{I_t\}_{t=1}^{T}$ with T frames, GAMMA reconstructs a set of dynamic 4D Gaussians $G = \{g_i\}_{i=1}^{N}$ that jointly capture spatial structure and temporal dynamics. The model predicts the complete set of spatio-temporal Gaussian parameters in a single forward pass, enabling efficient and generalizable 4D reconstruction without iterative optimization. We begin by introducing the backbone architecture that extracts spatio-temporal features from monocular videos in Sec. 3.1. In Sec. 3.2, we present the dual-head decoding scheme that disentangles geometry and appearance modeling to predict the full 4D Gaussian parameters. We further introduce a two-stage training strategy and learning objectives in Sec. 3.3.

3.1 Backbone Architecture

Encoding. GAMMA encodes both visual appearance and camera pose for each input frame. Following MapAnything [15], we extract dense visual features $\mathbf{F}_t \in \mathbb{R}^{1024\times\frac{H}{14}\times\frac{W}{14}}$ from each frame I_t using a pretrained DINOv2 [30] encoder, which provides robust semantic representations across diverse scenes.

To incorporate geometric information, we further encode the camera parameters associated with each frame. The camera rotation $\mathbf{R}_t$ is represented as a unit quaternion $\mathbf{q}_t \in \mathbb{R}^4$, and the camera translation $\mathbf{t}_t$ is normalized to obtain the translation direction $\hat{\mathbf{t}}_t$. We construct a ray direction map $\mathbf{U}_t \in \mathbb{R}^{H\times W\times 3}$ from camera intrinsics $\mathbf{K}_t$, mapping each pixel to its corresponding 3D ray direction. These ray directions are encoded by a convolutional network to produce spatial features $\mathbf{F}_t^U \in \mathbb{R}^{1024\times\frac{H}{14}\times\frac{W}{14}}$, while the quaternion and translation direction are encoded into compact embeddings $\mathbf{F}_t^Q, \mathbf{F}_t^T \in \mathbb{R}^{1024}$ via MLPs.

We apply sinusoidal positional encoding to the timestep t to capture temporal information, producing $\mathbf{E}_t^{time}$. All encoded components are aggregated as:

$$\mathbf{F}_t^E = \text{Flatten}\left(\mathbf{F}_t + \mathbf{F}_t^U + \mathbf{F}_t^Q + \mathbf{F}_t^T + \mathbf{E}_t^{time}\right), \tag{1}$$

where the result is flattened into tokens $\mathbf{F}_t^E \in \mathbb{R}^{1024\times\frac{HW}{196}}$ for subsequent processing.

Information Sharing. The encoded token features from all frames are passed through a shared transformer backbone to aggregate spatio-temporal information. Following recent practices in multi-view reconstruction [35], we adopt an alternating attention scheme that processes features through two complementary modes. At each layer l, we alternate between frame-wise attention and global attention:

$$\mathbf{F}^l = \begin{cases} \text{SelfAttn}_{\text{frame}}(\mathbf{F}^{l-1}), & \text{if } l \text{ is odd,} \\ \text{SelfAttn}_{\text{global}}(\mathbf{F}^{l-1}), & \text{if } l \text{ is even.} \end{cases} \tag{2}$$

where $\text{SelfAttn}_{\text{frame}}$ operates on tokens within each individual frame separately, while $\text{SelfAttn}_{\text{global}}$ attends across all frames jointly. This alternating design allows the model to efficiently process long video sequences while maintaining both spatial precision and temporal consistency.

The backbone consists of $L = 24$ transformer layers, all implemented with self-attention without additional cross-attention modules. Through this architecture, features from different timesteps interact bidirectionally, enabling the model to learn consistent geometric and appearance representations across the video. Since temporal information has been explicitly encoded through positional embeddings at the input stage, we do not introduce additional temporal position encodings such as 3D RoPE within the backbone, keeping the architecture simple while maintaining strong temporal awareness. The output features $\mathbf{F}^S \in \mathbb{R}^{1024 \times \frac{THW}{196}}$ encode integrated spatio-temporal information, serving as input to the subsequent dual-head decoder for 4D Gaussian parameter prediction.

3.2 Decoding for 4D Gaussian

Modeling of 4DGS. We model dynamic scenes through a collection of 4D Gaussian primitives [42], where each primitive unifies geometric attributes with temporal dynamics. Each 4D Gaussian g_i is parameterized by:

$$g_i = \{\mathbf{x}_i, \mathbf{s}_i, \mathbf{r}_i, o_i, \mathbf{c}_i, t_i, l_i, \mathbf{v}_i, \boldsymbol{\omega}_i\}. \tag{3}$$

Specifically, $\mathbf{x}_i \in \mathbb{R}^3$ indicates the center position, $\mathbf{s}_i \in \mathbb{R}^2$ defines anisotropic scaling along principal axes, and $\mathbf{r}_i \in \mathbb{R}^4$ encodes orientation as a unit quaternion. Base opacity $o_i \in [0, 1]$ and color $\mathbf{c}_i \in \mathbb{R}^3$ determine the primitive's appearance. Temporally, four attributes govern primitive behavior over time. The Gaussian timestep $t_i \in [0, T]$ marks peak activity, while duration $l_i \in \mathbb{R}^+$ controls lifespan. Linear velocity $\mathbf{v}_i \in \mathbb{R}^3$ and angular velocity $\boldsymbol{\omega}_i \in \mathbb{R}^3$ (in angle-axis form) encode motion characteristics.

2D Gaussian Primitives. Following 2DGS [11], each primitive is a flat disk whose two tangential axes are scaled by $\mathbf{s}_i$ and whose normal is given by $\mathbf{r}_i$. A disk has no thickness along its normal, so its center lies in the surface plane, and placing centers by unprojecting the supervised depth (Eq. 10) introduces no bias, unlike for volumetric 3D Gaussians, whose centers need not lie on the surface. Each disk also carries an explicit normal, which the rendered normal maps read out directly (Sec. 4.1). 2D Gaussian disks have also been used to reconstruct reflective scenes [56].

At query time t, opacity evolves via a temporal Gaussian centered at t_i:

$$\sigma_i = \sqrt{\frac{(l_i/2)^2}{-2\log(\epsilon)}}, \qquad o_i(t) = o_i \cdot \exp\left(-\frac{(t - t_i)^2}{2\sigma_i^2}\right), \tag{4}$$

where $\epsilon = 0.05$ sets the threshold at duration boundaries, ensuring smooth fade-in and fade-out while maintaining full opacity at the temporal center. Spatial parameters follow a velocity-based motion model:

$$\mathbf{x}_i(t) = \mathbf{x}_i + \mathbf{v}_i \cdot (t - t_i), \tag{5}$$

$$\mathbf{r}_i(t) = \mathbf{r}_i \otimes \text{AxisAngle2Quat}(\boldsymbol{\omega}_i \cdot (t - t_i)), \tag{6}$$

where $\otimes$ denotes quaternion multiplication. The scale $\mathbf{s}_i$ remains constant, preserving intrinsic shape.

While this linear motion formulation offers computational efficiency, it inherently assumes uniform velocity throughout the primitive's lifespan. Consequently, primitives cannot naturally model complex trajectories involving acceleration, deceleration, or directional changes. To address this limitation, we introduce a two-stage training strategy in Sec. 3.3 that progressively refines the reconstruction.

Dual-Head Decoding. Building upon the spatio-temporal features $\mathbf{F}^S$ extracted by the backbone, we adopt a dual-head architecture to predict the complete set of 4D Gaussian parameters. This design

leverages the strong geometric priors learned by pretrained 3D reconstruction models while extending them to handle dynamic scene modeling.

We inherit the depth prediction head from MapAnything [15], which has been pretrained on large-scale 3D reconstruction datasets and demonstrates robust depth estimation capabilities across diverse scenes. For each input frame I_t, the depth head predicts a dense depth map $\mathbf{D}_t \in \mathbb{R}^{H\times W}$:

$$\mathbf{D}_t = \mathrm{DepthHead}(\mathbf{F}_t^S), \tag{7}$$

where $\mathbf{F}_t^S$ denotes the features corresponding to frame t. Given the camera intrinsics $\mathbf{K}_t$ and extrinsics $[\mathbf{R}_t|\mathbf{t}_t]$, we unproject each pixel (u, v) with predicted depth d_{uv} to obtain geometric centers.

The Gaussian head predicts all remaining 4D Gaussian attributes $\{\mathbf{s}_i, \mathbf{r}_i, o_i, \mathbf{c}_i, t_i, l_i, \mathbf{v}_i, \boldsymbol{\omega}_i\}$ for each 4D primitive with the geometric center initialized by the predicted depth. The dual-head design allows us to fully leverage pretrained geometric knowledge from the depth head while learning dynamic-specific attributes through the Gaussian head in an end-to-end manner.

3.3 Training Objective and Strategy

Training Strategy. The flexibility of 4D Gaussian representations introduces a fundamental ambiguity: models can reconstruct dynamic scenes either through per-frame independent 3D Gaussians or through genuinely mobile 4D Gaussians with coherent motion. While both approaches achieve comparable rendering quality, only the latter captures meaningful motion patterns that are crucial for generalization and for supporting local novel view synthesis. We propose a two-stage training strategy that explicitly encourages coherent 4D motion while addressing the limitations of linear velocity formulation in 4DGS.

Without proper constraints, networks naturally degenerate toward per-frame reconstruction, as independent frame modeling provides a simpler optimization path than learning coherent trajectories. To address this issue, a straightforward solution is to regularize Gaussian lifespan l_i, penalizing brief appearances to encourage temporal persistence. However, this conflicts with the expressiveness of the linear velocity formulation in 4DGS, which cannot capture acceleration, deceleration, or directional changes. Primitives with long lifespans cannot simply disappear to model sudden stops without violating lifespan constraints, creating inherent tension between preventing degeneration and modeling motion complexity.

We resolve this through a progressive two-stage training strategy. Stage 1 enforces a uniform lifespan regularization across all Gaussians ($\mathcal{L}_{\text{life}}^{(1)}$ in Eq. 12), which encourages primitives to explain dynamics through consistent motion rather than temporal flickering. Stage 2 introduces selective regularization: the top 90% of Gaussians ranked by their predicted lifespan in each forward pass remain regularized to maintain persistent motion, while the bottom 10% are freed from constraints to model motion discontinuities ($\mathcal{L}_{\text{life}}^{(2)}$ in Eq. 12). This strategy ensures temporal coherence while providing necessary modeling flexibility.

Training Objective. We optimize GAMMA with a composite loss balancing rendering quality, geometric accuracy, and temporal coherence:

$$\mathcal{L} = \mathcal{L}_{\text{render}} + \lambda_{\text{depth}}\mathcal{L}_{\text{depth}} + \lambda_{\text{reg}}\mathcal{L}_{\text{reg}}. \tag{8}$$

Rendering Loss. We combine L1, SSIM, and perceptual losses for photometric supervision:

$$\mathcal{L}_{\text{render}} = \lambda_{L1}\mathcal{L}_{L1} + \lambda_{\text{SSIM}}\mathcal{L}_{\text{SSIM}} + \lambda_{\text{LPIPS}}\mathcal{L}_{\text{LPIPS}}. \tag{9}$$

Depth Supervision. We supervise predicted depth maps with ground-truth depth, or with MoGe-2 [38] pseudo-depth for datasets without it:

$$\mathcal{L}_{\text{depth}} = \frac{1}{THW}\sum_{t=1}^{T}\sum_{u,v}\left\|D_t(u,v) - \hat{D}_t(u,v)\right\|_1. \tag{10}$$

Motion Regularization. We regularize Gaussian lifespan and velocities to encourage physically plausible dynamics:

$$\mathcal{L}_{\text{reg}} = \mathcal{L}_{\text{life}} + \lambda_v\mathcal{L}_v + \lambda_\omega\mathcal{L}_\omega. \tag{11}$$

Table 1: Quantitative evaluation on NVIDIA Dynamic Scenes, DyCheck, and DynamicReplica datasets.

Method	NVIDIA Dynamic Scenes		DyCheck		DynamicReplica		
	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	RMSE↓
L4GM [32]	13.68	0.595	11.34	0.628	14.02	0.490	2.075
4DGT [48]	14.99	0.627	15.66	0.438	18.08	0.410	0.996
Ours	**17.70**	**0.324**	**16.39**	**0.380**	**20.10**	**0.268**	**0.152**

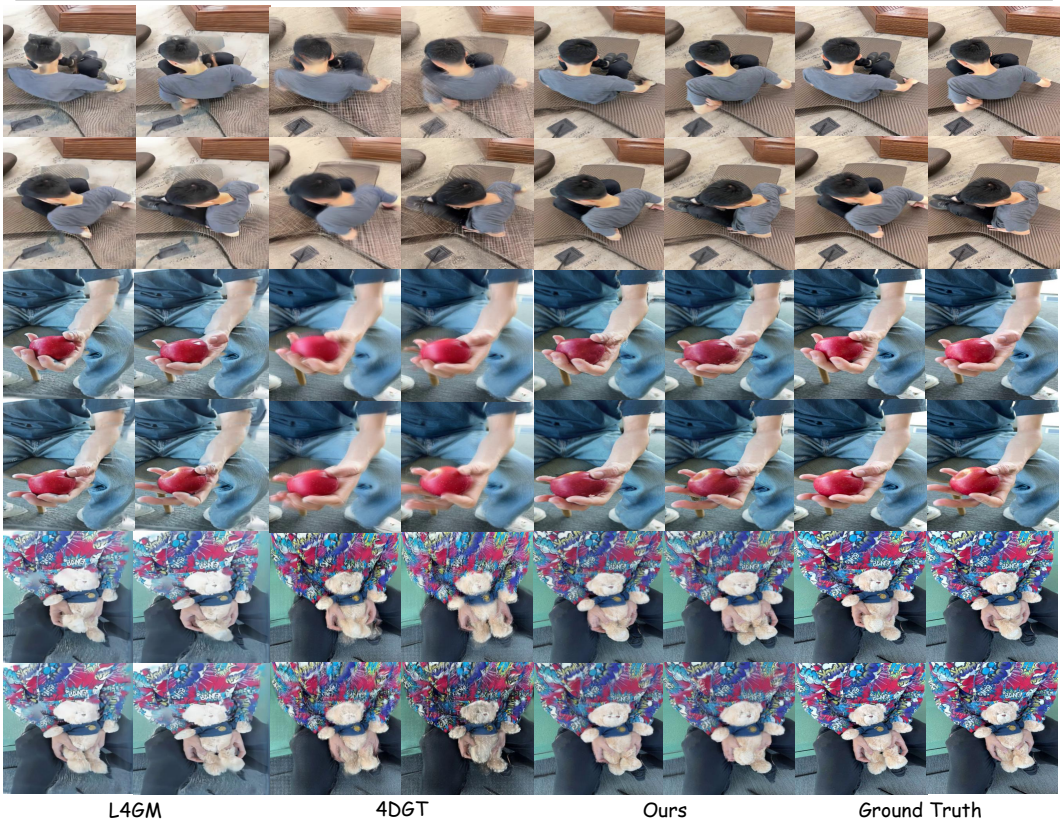


Figure 3: Qualitative results on the DyCheck dataset.

The lifespan regularization adapts across training stages. In Stage 1, we uniformly penalize all Gaussians to establish coherent trajectories, while in Stage 2, we selectively regularize only the top 90% longest-lived Gaussians, freeing the remaining 10% to model motion discontinuities:

$$\mathcal{L}_{\text{life}}^{(1)} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{l_i}, \quad \mathcal{L}_{\text{life}}^{(2)} = \frac{1}{|\mathcal{T}_{90}|}\sum_{i\in\mathcal{T}_{90}}\frac{1}{l_i}. \tag{12}$$

Motivated by 4DGT [48], we adopt velocity regularization to constrain linear and angular motion magnitudes:

$$\mathcal{L}_v = \frac{1}{N}\sum_{i=1}^{N}\|\mathbf{v}_i\|_2^2, \quad \mathcal{L}_\omega = \frac{1}{N}\sum_{i=1}^{N}\|\boldsymbol{\omega}_i\|_2^2. \tag{13}$$

Both velocity terms average over all N primitives in both stages, and only the scope of the lifespan term changes between stages, since ending a lifespan early never requires moving fast.

4 Experiments

Metrics and Baselines. We evaluate reconstruction quality using two standard metrics. PSNR measures pixel-level accuracy as $\text{PSNR} = 10\log_{10}(\text{MAX}^2/\text{MSE})$, where higher values indicate

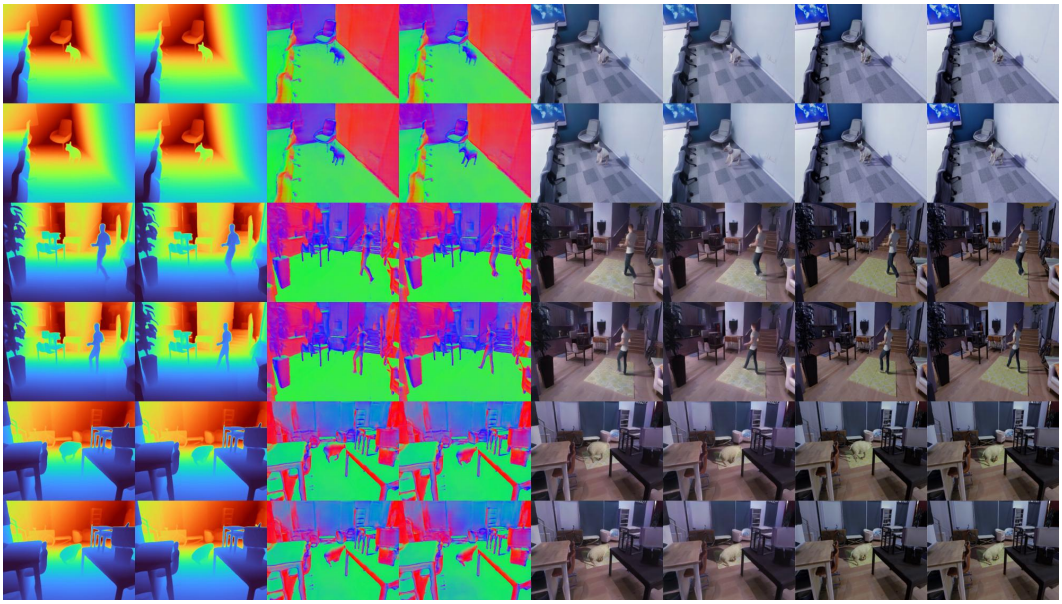


Figure 4: Visualization of depth and normal maps on DynamicReplica.

better fidelity. LPIPS computes perceptual distance in deep feature space, with lower values reflecting better perceptual quality. On DynamicReplica, we additionally report the RMSE of depth rendered from the reconstructed Gaussians. We compare against two state-of-the-art feed-forward methods: L4GM [32], which leverages large-scale pretraining for 4D Gaussian reconstruction, and 4DGT [48], which learns 4D Gaussians from monocular videos using transformers.

4.1 Dynamic Scene View Synthesis

DyCheck Dataset. The DyCheck dataset [6] provides a challenging benchmark for dynamic scene reconstruction, featuring real-world sequences captured with iPhone cameras. These sequences exhibit diverse dynamic content including object deformations, camera movements, and complex scene interactions, along with realistic lighting variations and motion blur inherent in casual video capture. Following the evaluation protocol of 4DGT [48], Tab. 1 compares GAMMA with L4GM [32] and 4DGT on this dataset. GAMMA outperforms both baselines in PSNR and LPIPS, demonstrating consistent reconstruction quality on challenging real-world scenarios. The visual comparison in Fig. 3 shows that GAMMA handles rapid motion and complex lighting conditions without the artifacts and inconsistencies that affect the baselines.

NVIDIA Dynamic Scenes Dataset. We evaluate GAMMA on the NVIDIA Dynamic Scenes dataset [54] following standard protocols. As shown in Tab. 1, GAMMA consistently outperforms L4GM [32] and 4DGT [48] in PSNR and LPIPS. Qualitatively (see Fig. 7 in Appendix), while baselines exhibit blur and ghosting, GAMMA delivers sharper reconstructions and superior temporal consistency. This is driven by our dual-head architecture and two-stage training, which effectively disentangles geometry from appearance to handle both persistent and discontinuous motions.

DynamicReplica Dataset. To further assess robustness in synthetic indoor environments, we evaluate on the DynamicReplica dataset [14]. Tab. 1 shows that GAMMA achieves state-of-the-art performance in both rendering quality and depth accuracy. Visually, GAMMA yields sharper results and better temporal alignment compared to the oversmoothed reconstructions from baselines (see Fig. 8, Appendix). Beyond appearance, Fig. 4 highlights our model's geometric fidelity. GAMMA recovers geometrically coherent depth maps with sharp boundaries, even for challenging regions like articulated limbs and rapid body movements. The normal maps further confirm that our approach faithfully captures fine-grained local geometry and smooth surface transitions, underscoring the efficacy of our 4D representation in complex dynamic settings.

Qualitative Results on DAVIS. To further evaluate the generalization capability of GAMMA, we apply our model directly to videos from the DAVIS dataset [31], a widely-used benchmark featuring diverse real-world dynamic scenes. Since DAVIS does not provide multi-view ground truth, we

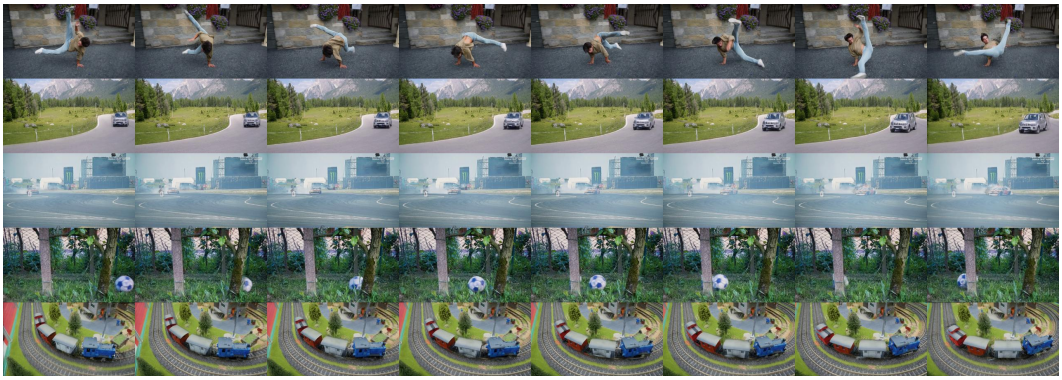


Figure 5: Qualitative results on the DAVIS dataset.

present qualitative results only. Importantly, none of the DAVIS sequences are included in our training set, making this a pure zero-shot evaluation.

As shown in Fig. 5, we visualize the rendered RGB images from different scenes. The results confirm that GAMMA generalizes well to unseen real-world videos with diverse dynamic content, producing temporally consistent renderings without any scene-specific optimization or fine-tuning.

4.2 Ablation Study

We validate our core components on the DyCheck Dataset (Tabs. 2 and 3), using all seven sequences with 25 input frames from each 50-frame clip, which gives slightly higher numbers than the six-sequence protocol of Tab. 1. Without $\mathcal{L}_{\text{life}}$, the model collapses into per-frame 3DGS as independent optimization is inherently easier than learning coherent 4D motion, resulting in severe temporal flickering and ghosting (Fig. 6, top).



Figure 6: Qualitative ablation of lifespan regularization and two-stage training.

Table 2: Ablations on key designs.

Configuration	PSNR↑	LPIPS↓
No lifespan reg. (λ_{life}=0)	13.70	0.605
+ Uniform lifespan reg. (Stage 1 only)	9.93	0.640
+ Two-stage selective reg. (full)	**16.68**	**0.330**

Table 3: Different input resolutions on DyCheck; 378×518 is our default.

	PSNR↑	LPIPS↓
518×518	**16.72**	0.339
378×784	16.33	0.373
378×518	16.68	**0.330**

While introducing $\mathcal{L}_{\text{life}}$ enforces temporal persistence, uniform regularization struggles to represent motion discontinuities (e.g., stopping or transient events) due to the constraints of linear motion modeling. It is even worse than no regularization: under linear velocity, a primitive can represent a sudden stop only by ending its lifespan, so a persistent primitive moves through the discontinuity instead. Our two-stage strategy addresses this by selectively relaxing constraints in Stage 2, enabling sharp reconstruction of both persistent structures and sudden motion changes (Fig. 6, bottom). These results confirm that the synergy between lifespan regularization and two-stage training is essential for high-fidelity 4D modeling.

We evaluate GAMMA across various input resolutions to assess its stability (Tab. 3). Our method maintains consistently strong performance in PSNR and LPIPS with only marginal fluctuations across all settings. This robustness suggests that GAMMA is insensitive to resolution choices, eliminating the need for meticulous per-scene tuning—a significant advantage for diverse real-world applications where input quality often varies.

5 Conclusion

We present GAMMA, a feed-forward model that reconstructs 4D Gaussian Splatting from monocular videos within seconds, enabling real-time dynamic novel view synthesis across timesteps and viewpoints. Through a dual-head architecture that disentangles geometry and appearance modeling, a pretrained transformer backbone with strong spatial reasoning capabilities, and a two-stage training strategy with selective lifespan regularization, GAMMA achieves accurate modeling of both persistent motion and sudden discontinuities. Extensive experiments demonstrate that GAMMA substantially outperforms existing methods in reconstruction quality and efficiency, while exhibiting strong generalization to unseen real-world videos. GAMMA represents a significant step toward practical 4D scene reconstruction, opening new possibilities for applications in VR/AR, film production, and robotics.

Acknowledgments and Disclosure of Funding

This work was partially supported by Deep Earth Probe and Mineral Resources Exploration – National Science and Technology Major Project (2024ZD1003405) and the National Natural Science Foundation of China (62272263).

References

[1] David Charatan, Sizhe 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.

[2] Shi Chen, Erik Sandström, Sandro Lombardi, Siyuan Li, and Martin R. Oswald. ProDyG: Progressive dynamic scene reconstruction via Gaussian splatting from monocular videos. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[3] Tiancheng Chen, Sheng Tang, Wenhua Jin, Weiqi Zhang, Juntong Fang, Junsheng Zhou, and Zesong Li. UniQuery4R: Unified 4D scene reconstruction from a single query. *arXiv preprint arXiv:2608.17283*, 2026.

[4] 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.

[5] Juntong Fang, Zequn Chen, Weiqi Zhang, Donglin Di, Xuancheng Zhang, Chengmin Yang, and Yu-Shen Liu. MoRe: Motion-aware feed-forward 4D reconstruction transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

[6] Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. Monocular dynamic view synthesis: A reality check. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[7] Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, et al. Ego-Exo4D: Understanding skilled human activity from first- and third-person perspectives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[8] Zhiyang Guo, Wengang Zhou, Li Li, Min Wang, and Houqiang Li. Motion-aware 3D Gaussian splatting for efficient dynamic scene reconstruction. *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, 35(4):3119–3133, 2025.

[9] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. LRM: Large reconstruction model for single image to 3D. In *International Conference on Learning Representations (ICLR)*, 2024.

[10] Yuan-Ting Hu, Jiahong Wang, Raymond A. Yeh, and Alexander G. Schwing. SAIL-VOS 3D: A synthetic dataset and baselines for object detection and 3D mesh reconstruction from video data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[11] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2D Gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH Conference Papers*, 2024.

[12] Zeren Jiang, Chuanxia Zheng, Iro Laina, Diane Larlus, and Andrea Vedaldi. Geo4D: Leveraging video generators for geometric 4D scene reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

[13] Wenhua Jin, Tiancheng Chen, Sheng Tang, Jiahao Li, Yupeng Liu, Weiqi Zhang, Juntong Fang, and Zesong Li. Moving regions, not moving objects: Segmenting what actually moves under camera motion. *arXiv preprint arXiv:2610.31209*, 2026.

[14] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. DynamicStereo: Consistent dynamic depth from stereo videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[15] 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)*, 2026.

[16] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (TOG)*, 42(4):139:1–139:14, 2023.

[17] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3D with MASt3R. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

[18] Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. Spacetime Gaussian feature splatting for real-time dynamic view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[19] Zhengqi Li, Richard Tucker, Forrester Cole, Qianqian Wang, Linyi Jin, Vickie Ye, Angjoo Kanazawa, Aleksander Holynski, and Noah Snavely. MegaSaM: Accurate, fast and robust structure and motion from casual dynamic videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[20] 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 *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[21] Yiqing Liang, Numair Khan, Zhengqin Li, Thu Nguyen-Phuoc, Douglas Lanman, James Tompkin, and Lei Xiao. GauFRe: Gaussian deformation fields for real-time dynamic novel view synthesis. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2025.

[22] 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 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

[23] 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 *International Conference on Learning Representations (ICLR)*, 2026.

[24] Zhicheng Lu, Xiang Guo, Le Hui, Tianrui Chen, Min Yang, Xiao Tang, Feng Zhu, and Yuchao Dai. 3D geometry-aware deformable Gaussian splatting for dynamic view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[25] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3D Gaussians: Tracking by persistent dynamic view synthesis. In *International Conference on 3D Vision (3DV)*, 2024.

[26] Ziqiao Ma, Xuweiyi Chen, Shoubin Yu, Sai Bi, Kai Zhang, Ziwen Chen, Sihan Xu, Jianing Yang, Zexiang Xu, Kalyan Sunkavalli, Mohit Bansal, Joyce Chai, and Hao Tan. 4D-LRM: Large space-time reconstruction model from and to any view at any time. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[27] Lukas Mehl, Jenny Schmalfuss, Azin Jahedi, Yaroslava Nalivayko, and Andrés Bruhn. Spring: A high-resolution high-detail dataset and benchmark for scene flow, optical flow and stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*,

2023.

[28] Takeshi Noda, Chao Chen, Weiqi Zhang, Xinhai Liu, Yu-Shen Liu, and Zhizhong Han. MultiP-ull: Detailing signed distance functions by pulling multi-level queries at multi-step. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[29] Takeshi Noda, Chao Chen, Junsheng Zhou, Weiqi Zhang, Yu-Shen Liu, and Zhizhong Han. Learning bijective surface parameterization for inferring signed distance functions from sparse point clouds with grid deformation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[30] 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. *Transactions on Machine Learning Research (TMLR)*, 2024.

[31] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[32] Jiawei Ren, Kevin Xie, Ashkan Mirzaei, Hanxue Liang, Xiaohui Zeng, Karsten Kreis, Ziwei Liu, Antonio Torralba, Sanja Fidler, Seung Wook Kim, and Huan Ling. L4GM: Large 4D Gaussian reconstruction model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[33] Stanislaw Szymanowicz, Christian Rupprecht, and Andrea Vedaldi. Splatter Image: Ultra-fast single-view 3D reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[34] Jimin Tang, Wenyuan Zhang, Junsheng Zhou, Zian Huang, Kanle Shi, Shenkun Xu, Yu-Shen Liu, and Zhizhong Han. VidSplat: Gaussian splatting reconstruction with geometry-guided video diffusion priors. In *ACM SIGGRAPH Conference Papers*, 2026.

[35] 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)*, 2025.

[36] 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)*, 2025.

[37] 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 (CVPR)*, 2025.

[38] Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. MoGe-2: Accurate monocular geometry with metric scale and sharp details. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[39] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jérôme Revaud. DUSt3R: Geometric 3D vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[40] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. TartanAir: A dataset to push the limits of visual SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[41] Yunsong Wang, Tianxin Huang, Hanlin Chen, and Gim Hee Lee. FreeSplat: Generalizable 3D Gaussian splatting towards free view synthesis of indoor scenes. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[42] Yifan Wang, Peishan Yang, Zhen Xu, Jiaming Sun, Zhanhua Zhang, Yong Chen, Hujun Bao, Sida Peng, and Xiaowei Zhou. FreeTimeGS: Free Gaussian primitives at anytime anywhere for dynamic scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[43] 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)*, 2026.

[44] 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.

[45] 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.

[46] Haofei Xu, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. DepthSplat: Connecting Gaussian splatting and depth. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[47] Yinghao Xu, Zifan Shi, Yifan Wang, Hansheng Chen, Ceyuan Yang, Sida Peng, Yujun Shen, and Gordon Wetzstein. GRM: Large Gaussian reconstruction model for efficient 3D reconstruction and generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

[48] 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)*, 2025.

[49] Zhen Xu, Yinghao Xu, Zhiyuan Yu, Sida Peng, Jiaming Sun, Hujun Bao, and Xiaowei Zhou. Representing long volumetric video with temporal Gaussian hierarchy. *ACM Transactions on Graphics (TOG)*, 43(6):171:1–171:18, 2024.

[50] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3D Gaussians for high-fidelity monocular dynamic scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[51] Zeyu Yang, Zijie Pan, Xiatian Zhu, Li Zhang, Jianfeng Feng, Yu-Gang Jiang, and Philip H. S. Torr. 4D Gaussian splatting: Modeling dynamic scenes with native 4D primitives. *arXiv preprint arXiv:2412.20720*, 2024.

[52] Zeyu Yang, Hongye Yang, Zijie Pan, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4D Gaussian splatting. In *International Conference on Learning Representations (ICLR)*, 2024.

[53] 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.

[54] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[55] 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.

[56] Wenyuan Zhang, Jimin Tang, Weiqi Zhang, Yi Fang, Yu-Shen Liu, and Zhizhong Han. MaterialRefGS: Reflective Gaussian splatting with multi-view consistent material inference. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[57] Weiqi Zhang, Wenyuan Zhang, Yu-Shen Liu, and Junsheng Zhou. AEON: Unifying video and 3D world models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2026.

[58] Weiqi Zhang, Junsheng Zhou, Juntong Fang, Kanle Shi, Shenkun Xu, and Yu-Shen Liu. TESLA: Native 4D Gaussian splatting generation with temporally structured latents. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2026.

[59] Weiqi Zhang, Junsheng Zhou, Haotian Geng, Kanle Shi, Shenkun Xu, Yi Fang, and Yu-Shen Liu. GaussianGrow: Geometry-aware Gaussian growing from 3D point clouds with text guidance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

[60] Weiqi Zhang, Junsheng Zhou, Haotian Geng, Wenyuan Zhang, and Yu-Shen Liu. GAP: Gaussianize any point clouds with text guidance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

[61] Yang Zheng, Adam W. Harley, Bokui Shen, Gordon Wetzstein, and Leonidas J. Guibas. PointOdyssey: A large-scale synthetic dataset for long-term point tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[62] Junsheng Zhou, Weiqi Zhang, and Yu-Shen Liu. DiffGS: Functional Gaussian splatting diffusion. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[63] Junsheng Zhou, Weiqi Zhang, Baorui Ma, Kanle Shi, Yu-Shen Liu, and Zhizhong Han. UDiFF: Generating conditional unsigned distance fields with optimal wavelet diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[64] Junsheng Zhou, Weiqi Zhang, Baorui Ma, Kanle Shi, Yu-Shen Liu, and Zhizhong Han. UDFStudio: A unified framework of datasets, benchmarks and generative models for unsigned distance functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2026.

[65] Yang Zhou, Yifan Wang, Jianjun Zhou, Wenzheng Chang, Haoyu Guo, Zizun Li, Kaijing Ma, Xinyue Li, Yating Wang, Haoyi Zhu, Mingyu Liu, Dingning Liu, Jiange Yang, Zhoujie Fu, Junyi Chen, Chunhua Shen, Jiangmiao Pang, Kaipeng Zhang, and Tong He. OmniWorld: A

multi-domain and multi-modal dataset for 4D world modeling. In *International Conference on Learning Representations (ICLR)*, 2026.

[66] Dong Zhuo, Wenzhao Zheng, Jiahe Guo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Streaming visual geometry transformer. In *International Conference on Learning Representations (ICLR)*, 2026.

A More Implementation Details

Training Datasets and Details. We train GAMMA on a diverse collection of monocular video datasets including PointOdyssey [61], DynamicReplica [14], OmniWorld-Game [65], TartanAir [40], SAIL-VOS 3D [10], Spring [27] and Ego-Exo4D [7]. To ensure balanced training and enhance the model's ability to learn Gaussian motion, we apply dataset-specific resampling strategies. For datasets without ground-truth depth, we use MoGe-2 [38] to estimate depth as supervision.

We trained GAMMA on 64 NVIDIA A800 GPUs. The transformer backbone is initialized with pretrained weights from MapAnything [15], inheriting strong spatial reasoning capabilities from large-scale 3D reconstruction pretraining. Our two-stage training employs distinct optimization schedules and loss configurations. In Stage 1, we set loss weights as: $\lambda_{depth} = 0.1, \lambda_{L1} = 1.0, \lambda_{SSIM} = 0.2, \lambda_{LPIPS} = 2.0, \lambda_v = \lambda_\omega = 0.5, \lambda_{life} = 1.0$. This stage focuses on learning geometric structures and motion awareness. Stage 2 introduces selective lifespan regularization on the top 90% longest-lived Gaussians while keeping other loss weights unchanged. The complete training takes approximately 7 days. By utilizing bfloat16 precision along with gradient checkpointing, we are able to significantly reduce memory requirements and support efficient training.

A.1 Training Data Preparation.

We train GAMMA on a diverse collection of monocular video datasets including PointOdyssey [61], TartanAir [40], DynamicReplica [14], OmniWorld-Game [65], SAIL-VOS 3D [10], Spring [27], and Ego-Exo4D [7], which cover both synthetic and real-world scenarios with varying scene complexities and motion patterns. Tab. 4 summarizes the detailed characteristics of each dataset, including their properties such as dynamic content, pose availability, depth annotations, real or synthetic nature, as well as their number of scenes and the resampling factors applied during training. For datasets without ground-truth depth annotations such as Ego-Exo4D, we employ MoGe-2 [38] to generate monocular depth estimates as pseudo-supervision, with a reduced depth loss weight of $\lambda_{\text{depth}} = 0.05$ compared to $\lambda_{\text{depth}} = 0.1$ for ground-truth depth. To ensure balanced training and enhance the model's ability to learn motion patterns, we apply dataset-specific resampling strategies where datasets with fewer scenes or richer motion dynamics receive higher resampling weights. During training, we randomly sample continuous sequences of $T = 24$ frames from each video and apply standard color jittering for augmentation while deliberately avoiding spatial transformations that would invalidate the camera pose annotations.

Table 4: More Implementation Details about Training Dataset.

Dataset	Dynamic?	Pose?	Depth?	Real?	#Scenes	#Resampled
PointOdyssey [61]	✓	✓	✓	×	0.1K	100×
TartanAir [40]	✓	✓	✓	×	0.5k	50×
DynamicReplica [14]	✓	✓	✓	×	0.5K	20×
OmniWorld-Game [65]	✓	✓	✓	×	0.06k	100×
SAIL-VOS 3D [10]	✓	✓	✓	×	0.16K	50×
Spring [27]	✓	✓	✓	×	0.03K	1000×
Ego-Exo4D [7]	✓	✓	×	✓	4.2K	1×

A.2 Training Details

We train GAMMA on 64 NVIDIA A800 GPUs for approximately 7 days using a two-stage training strategy, where the transformer backbone is initialized with pretrained weights from MapAnything [15] to leverage strong spatial reasoning capabilities learned from large-scale 3D reconstruction tasks. To enable efficient large-scale training, we employ bfloat16 mixed precision along with gradient checkpointing, which significantly reduces memory consumption while maintaining numerical stability.

The first training stage focuses on establishing geometric structures and static appearance components that remain stable over time, running for 100K iterations with loss weights configured as $\lambda_{\text{depth}} = 0.1$, $\lambda_{L1} = 1.0$, $\lambda_{\text{SSIM}} = 0.2$, $\lambda_{\text{LPIPS}} = 2.0$, $\lambda_v = \lambda_\omega = 0.5$, and $\lambda_{\text{life}} = 1.0$. During this stage, we apply uniform lifespan regularization $\mathcal{L}_{\text{life}}^{(1)} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{l_i}$ across all Gaussians to encourage temporal persistence and prevent the model from degenerating into per-frame 3D reconstruction. We optimize

the model using AdamW with a learning rate of 1×10^{-4} and weight decay of 0.01, following a cosine learning rate schedule that includes 5K warmup steps for stable initialization.

Building upon this foundation, the second stage introduces selective lifespan regularization to enable accurate modeling of motion discontinuities while preserving temporal coherence for persistent structures. Specifically, we rank the Gaussians by their predicted lifespan in each forward pass and apply the regularization $\mathcal{L}_{\text{life}}^{(2)} = \frac{1}{|\mathcal{T}_{90}|} \sum_{i \in \mathcal{T}_{90}} \frac{1}{l_i}$ only to the top 90% longest-lived Gaussians, thereby freeing the remaining 10% from lifespan constraints to capture sudden stops and transient events. All other loss weights remain unchanged, while this stage runs for an additional 50K iterations with a reduced learning rate of 5×10^{-5} to ensure stable convergence. Throughout both training stages, we apply gradient clipping with a maximum norm of 1.0 to prevent gradient explosion.

A.3 Model Architecture

We adopt a hierarchical encoder-decoder model based on a DINOv2 [30] Vision Transformer that processes multi-frame posed inputs from a monocular video. These inputs are encoded into a unified feature space, fused through a aggregated attention transformer, and decoded using a DPT-style architecture with multi-scale fusion, enabling end-to-end learning from images to dynamic 4D Gaussian representations, as illustrated in Tab. 5.

Table 5: Model Architecture Overview.

Module	Component	Input Dim	Output Dim	Key Parameters
Image Encoder	DINOv2 ViT	$3 \times H \times W$	$1024 \times \frac{H}{14} \times \frac{W}{14}$	24 blocks, patch size 14×14
Dense Encoders	Ray Directions	$3 \times H \times W$	1024	Unshuffle + 2 ResBlocks
Global Encoders	Camera	4/3/1	1024	4-layer MLP (quat/trans)
Fusion	Multi-View Attention	1024	768	24 self-attention blocks
DPT Decoder	Feature Pyramid + Refinement	768	256	4-scale hierarchical fusion
Dense Head	Regression	256	6	Conv $256 \to 128 \to 6$
GS Head	Base Features Attribute MLPs	256 41	41 1/3/4/16/8	Conv $256 \to 128 \to 41$ 12-layer MLPs (opacity/scale/rot/SH/etc.)

A.4 Evaluation Details

We employ a comprehensive set of quantitative metrics to assess 4D reconstruction and novel view synthesis quality from complementary perspectives. PSNR measures pixel-level reconstruction accuracy as $\text{PSNR} = 10 \log_{10}(\text{MAX}^2/\text{MSE})$, where higher values indicate better fidelity in capturing fine-grained details. LPIPS evaluates perceptual similarity by computing distances in deep feature space extracted from pretrained networks, with lower scores reflecting stronger alignment with human perception of image quality. While PSNR quantifies low-level pixel accuracy, LPIPS provides crucial insights into perceptual realism by capturing high-level semantic and structural similarities that better correlate with visual quality as perceived by human observers. This combination ensures robust evaluation across diverse dynamic scenes and viewpoint variations.

We provide the exact evaluation settings used in all benchmarks for reproducibility. For each dataset, we follow the official train/test split. Unless otherwise noted, all images are resized to 378×518 resolution before inference, and quantitative metrics are computed at the native benchmark resolution after rendering. We report PSNR and LPIPS for novel-view synthesis, and additionally report depth RMSE on DynamicReplica. LPIPS is computed using the VGG backbone from the official implementation. No test-time optimization is used.

For NVIDIA Dynamic Scenes [54], we evaluate on all views per sequence and average over all timesteps. For DyCheck [6], we follow 4DGT [48] and compute metrics on masked foreground regions. For DynamicReplica [14], we report both rendering and geometric metrics using the official camera trajectories and evaluation split.

On DyCheck, Tab. 1 follows the protocol of 4DGT, which evaluates six sequences, excluding *wheel*, with 64 input frames from each 128-frame clip. Since 4DGT does not report results on NVIDIA Dynamic Scenes, we evaluate its official checkpoint under our protocol. Although GAMMA is trained on clips of $T = 24$ frames, every result in this paper is obtained with a single forward pass over the full input clip, without sliding windows, keyframe sampling, or autoregressive buffers. Nothing in

the model is tied to a particular T: attention is agnostic to the sequence length, both heads act per token, and the sinusoidal timestep embedding remains defined at indices never seen during training, unlike a learned embedding table.

B More Results

B.1 More Visualization

We show the qualitative comparisons against baselines under the NVIDIA Dynamic Scenes dataset and the DynamicReplica test dataset in Fig. 7 and Fig. 8. GAMMA consistently outperforms state-of-the-art methods like L4GM [32] and 4DGT [48], delivering significantly sharper high-frequency details and superior temporal coherence. These improvements are especially pronounced in regions characterized by rapid motion, where baselines frequently suffer from severe blur and ghosting artifacts. These results collectively underscore GAMMA's robustness and generalizability when handling diverse real-world scenarios with complex motion patterns.

B.2 Additional Ablations

Backbone Initialization. To test whether the training strategy depends on the MapAnything weights, we retrain GAMMA from a Depth Anything 3 [23] initialization under a reduced-scale schedule (2 datasets, 30K iterations). The ranking of Tab. 2 is preserved (Tab. 6a): the full schedule performs best (12.55 PSNR), ahead of uniform regularization (10.85) and no regularization (9.76).

Regularization Threshold. Tab. 6b varies the percentage k of longest-lived Gaussians kept under the lifespan prior in Stage 2. Any k between 80% and 95% performs within 0.23 dB, and only the extremes fail, for opposite reasons: with no primitive regularized, temporal coherence is lost, and with all regularized, no primitive can end early. Since Gaussians are ranked by predicted lifespan in each forward pass, k only bounds how many primitives may escape the prior; in practice, 2.8% become short-lived on static sequences and 9.4% under rapid motion.

Table 6: Additional ablations on DyCheck (PSNR↑). (a) Lifespan regularization with a Depth Anything 3 initialization, reduced-scale run. (b) Percentage k of longest-lived Gaussians kept under the lifespan prior (MapAnything initialization, full schedule). Same setting as Tab. 2.

(a) Configuration	DA3
No lifespan reg. (λ_{life}=0)	9.76
+ Uniform lifespan reg. (Stage 1 only)	10.85
+ Two-stage selective reg. (full)	**12.55**

(b) k (%)	0	80	90	95	100
PSNR↑	13.70	16.45	**16.68**	16.51	9.93

B.3 Inference Efficiency Analysis

We profile the inference cost of GAMMA on a single NVIDIA A800 GPU with input frames at 378×518 resolution and clips of $T = 48$ frames. The complete 4D Gaussian reconstruction is obtained in a single forward pass within approximately 3.5 seconds, bypassing the hours of per-scene optimization required by methods such as Shape of Motion [36] and FreeTimeGS [42]. Once reconstructed, novel views at arbitrary timesteps can be rendered at 100+ FPS via standard Gaussian rasterization, as the temporal opacity modulation and linear motion model introduce negligible overhead over static 3DGS rendering. As shown in Tab. 7, GAMMA reduces the reconstruction time by several orders of magnitude compared to optimization-based approaches while maintaining competitive speed among feed-forward methods.

C Limitations

GAMMA is still limited by the linear-motion assumption of 4D Gaussian primitives. While our two-stage training improves robustness to motion discontinuities, modeling highly non-linear motion remains challenging. Each primitive moves linearly within its lifespan, so a complex trajectory is approximated piecewise by successive short-lived primitives, and its accuracy depends on how



Figure 7: We compare our method against L4GM [32] and 4DGT [48] on view synthesis for dynamic scenes under the NVIDIA Dynamic Scenes dataset. Our method produces significantly sharper reconstructions with better temporal consistency and fewer artifacts compared to baseline methods.

Table 7: Inference efficiency comparison. Wall-clock time of one forward pass on a 48-frame clip at 378×518, averaged over 40 sequences, with all methods timed on the same A800 GPU.

Method	Type	Recon. Time ↓
L4GM [32]	FF	93 s
4DGT [48]	FF	5.5 s
Ours	FF	**3.5 s**

finely the motion is segmented in time. Second, the time complexity of global attention is quadratic in the number of tokens, which limits the clip length that can be processed in a single pass; the training context of $T = 24$ frames is a throughput decision rather than an architectural bound. Third, GAMMA requires posed input, which limits its direct use on unposed in-the-wild video.



Figure 8: Qualitative results on DynamicReplica test dataset.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state GAMMA's feed-forward 4D Gaussian Splatting reconstruction from monocular videos, the dual-head architecture that disentangles geometry and appearance modeling, and the two-stage training strategy with lifespan regularization. These contributions are detailed in Sec. 3 and supported by experiments in Sec. 4.1–4.2.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We analyze the limitations of our method in the supplementary materials.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: This paper does not include theoretical results. All formulas describe the 4D Gaussian primitive parameterization, the temporal opacity and motion models, and the loss functions, rather than theorems requiring proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Sec. 3 fully describes the dual-head architecture and 4D Gaussian parameterization, and Sec. 4.1 together with the supplementary material provide the training datasets, two-stage training schedule, optimizer (AdamW), learning rate, loss weights, hardware (64 NVIDIA A800 GPUs), input resolution, sequence length ($T = 24$), and per-module architecture specifications used for the main experiments.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The complete codebase and pretrained model weights are planned for public release upon acceptance to facilitate reproducibility, and the supplementary material provides the implementation specifications needed to reproduce the main experiments. All training datasets used (PointOdyssey, TartanAir, DynamicReplica, OmniWorld-Game, SAIL-VOS 3D, Spring, Ego-Exo4D) are existing public datasets accessible from their original sources.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Sec. 4.1 and the supplementary material specify the AdamW optimizer with learning rate 1×10^{-4} (Stage 1) and 5×10^{-5} (Stage 2), weight decay 0.01, cosine schedule with 5K warmup steps, loss weights (λ_{depth}=0.1, λ_{L1}=1.0, λ_{SSIM}=0.2, λ_{LPIPS}=2.0, λ_v=λ_ω=0.5, λ_{life}=1.0), the two-stage training (100K + 50K iterations), the resampling weights for each training dataset, sequence length ($T = 24$), evaluation protocols, baselines (L4GM, 4DGT), and metrics (PSNR, LPIPS, depth RMSE).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Following the convention of prior feed-forward 4D reconstruction works (L4GM, 4DGT), we report averaged PSNR, LPIPS, and depth RMSE across all evaluation views and timesteps for each benchmark. Error bars are not reported because each method produces a deterministic output for a given input video.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Sec. 4.1 and the supplementary material report training on 64 NVIDIA A800 GPUs for approximately 7 days using bfloat16 mixed precision and gradient checkpointing for memory efficiency. The supplementary material additionally reports inference cost: a complete 4D Gaussian reconstruction is produced in about 3.5 seconds in a single forward pass on a single NVIDIA A800 GPU at 378×518 resolution, and rendering reaches 100+ FPS via standard Gaussian rasterization.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: We have reviewed the NeurIPS Code of Ethics and confirm that this research conforms with it in every respect.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: Sec. 1 and Sec. 5 discuss positive applications in VR/AR, film production, and robotics. Because GAMMA reconstructs scenes faithfully from existing monocular videos rather than generating new synthetic content, the direct path to malicious applications is limited; nonetheless, photorealistic novel-view rendering of recorded individuals and scenes raises potential privacy concerns, and we encourage downstream users to obtain consent and respect data-source licenses when applying the model to recorded human subjects.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: GAMMA is a reconstruction model that recovers a 4D Gaussian representation of scenes already present in an input monocular video; it is not a generative content-creation model and does not synthesize identities or scenes that did not exist in the input. We do not redistribute any of the underlying training datasets, and the model output is bounded by the content of the user-provided video, so we judge the misuse risk to be low and do not require dedicated safeguards for release.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper cites all existing assets used in the work, including the training datasets (PointOdyssey, TartanAir, DynamicReplica, OmniWorld-Game, SAIL-VOS 3D, Spring, Ego-Exo4D), the evaluation datasets (NVIDIA Dynamic Scenes, DyCheck, DAVIS), the pretrained backbones (DINOv2, MapAnything), the depth-estimation prior (MoGe-2), and the baselines (L4GM, 4DGT). Each asset is used in accordance with its original license terms; no proprietary or restricted-license data is redistributed.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release any new dataset. The only new asset is the GAMMA model itself, whose architecture, training data composition, training procedure, and inference behavior are fully documented in Sec. 3, Sec. 4.1, and the supplementary material; release of code and pretrained weights upon acceptance is described in the supplementary.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.

- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve any crowdsourcing or research with human subjects. All evaluations are conducted on existing public benchmarks using automatic metrics (PSNR, LPIPS, depth RMSE).

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not involve any research with human subjects, so IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: GAMMA's core methodology does not use any large language models. The only pretrained components are vision models (DINOv2 for visual features, MapAnything for backbone initialization, and MoGe-2 for depth pseudo-supervision on datasets without ground-truth depth), all of which are standard, properly cited, and not LLMs.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.

- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.