DUSTING THE PHANTOM: GAUSSIAN ALLOCATION FOR FADING DYNAMIC CONTENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Dynamic Gaussian representations have made high-quality, real-time volumetric video increasingly practical, yet challenging motion can still break their reconstruction: a small fast-moving object may partially disappear, and fine textures in highly dynamic regions are often lost. Surprisingly, simply training longer or using more Gaussians does not reliably resolve these failures. We argue that the key bottleneck is not representational capacity itself, but *where and when that capacity is allocated*. We introduce two complementary strategies for improved spatiotemporal Gaussian allocation. First, *seed injection* introduces new Gaussians directly from multi-view observations, using temporal appearance changes and reconstruction errors to identify under-reconstructed dynamic regions and place new support at geometrically plausible 3D locations. This allows the model to recover content that may have little or no existing Gaussian support. Second, *temporally biased relocation* preserves time-resolved optimization signals and uses them to guide *when and where* relocated Gaussian capacity should be reassigned. Both strategies can be incorporated into existing dynamic Gaussian frameworks without modifying their underlying scene representation. Experiments across multiple dynamic-scene datasets show consistent improvements in reconstruction quality, with particularly pronounced gains, including better recovery of small, rapidly moving objects and fine-grained appearance details and textures.

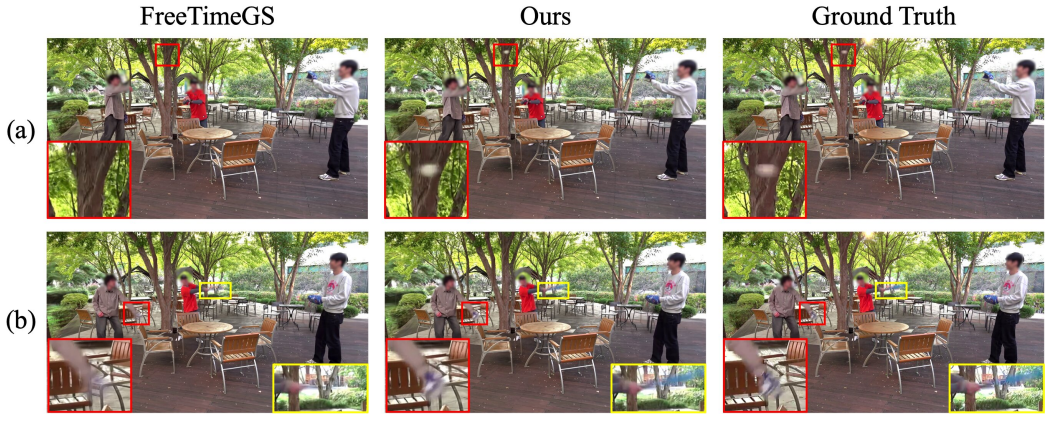

Figure 1: **Reconstruction under highly challenging dynamics.** Existing 4DGS methods can (a) completely miss rapidly moving objects and (b) lose fine details under fast motion. Our method addresses both failure cases, recovering missing objects while better preserving fine details.

1 INTRODUCTION

Recent advances in 4D volumetric video promise to transform how we experience visual content: rather than watching events from fixed, recorded viewpoints, viewers can freely explore dynamic scenes as they unfold in space and time. From free-viewpoint sports replays (Chen et al., 2019; Dai et al., 2024) and immersive performances (Collet et al., 2015) to lifelike telepresence (Lawrence

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.

et al., 2021), such experiences require faithful reconstruction of both appearance and motion, together with interactive rendering. Recent advances in Neural Radiance Fields (NeRFs) (Mildenhall et al., 2021) and 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) have brought this vision closer to reality, providing powerful representations for high-quality novel-view synthesis that have been successfully extended to dynamic scenes. In particular, 3DGS's high-quality, real-time rendering makes it especially attractive for next-generation immersive media, driving growing interest in Gaussian-based 4D scene modeling.

Many approaches have been developed to extend 3DGS to dynamic scenes, modeling temporal variations through deformation-based models (Yang et al., 2024b; Wu et al., 2024; Huang et al., 2024; Xu et al., 2024; Bae et al., 2024) or explicit spatiotemporal Gaussian representations (Yang et al., 2024a; Li et al., 2024; Wang et al., 2025; Liao et al., 2026; Lee et al., 2024; Wang et al., 2026; Yeom et al., 2026). Despite substantial progress, challenging dynamic content still remains difficult to reconstruct, with small, rapidly moving objects sometimes partially or entirely disappearing (Fig. 1a) and fine textures in highly dynamic regions often being poorly recovered (Fig. 1b). A natural remedy would be to simply train longer or increase the representational capacity by introducing more Gaussian primitives. Yet, despite our extensive efforts along these directions, neither strategy reliably resolves these failures, suggesting that the bottleneck is not insufficient representational capacity, but rather an algorithmic limitation in how Gaussian primitives are allocated during optimization. We therefore study these failures through the lens of Gaussian allocation: whether sufficient Gaussian support is assigned to challenging dynamic regions, and whether that support is placed at the appropriate spatial locations and timestamps.

Gaussian allocation is commonly adjusted through *densification* (Kerbl et al., 2023; Yang et al., 2024a; Li et al., 2024) and *relocation* (Kheradmand et al., 2024; Wang et al., 2025; 2026; Yeom et al., 2026; Lee et al., 2026). Densification increases representation capacity by cloning or splitting existing primitives, whereas relocation recycles low-contribution Gaussians by moving them to regions that require additional support, often while maintaining a fixed Gaussian budget. Both strategies rely on the current set of Gaussians to determine where Gaussians should be allocated. As a result, they struggle to place Gaussians in regions that are already poorly represented, which often correspond to challenging dynamic content such as small or rapidly moving objects. An insufficient number of Gaussians can hinder the reconstruction of fine appearance details and may even cause these objects to disappear altogether. To address this issue, we introduce the *seed injection* scheme, which injects Gaussian primitives directly from observations rather than deriving them from existing primitives. Specifically, *seed injection* identifies poorly reconstructed dynamic regions by combining temporal intensity changes with reconstruction error across synchronized multi-view images, and lifts selected observations into geometrically plausible 3D locations. By providing new Gaussian support independently of the current primitive distribution, it enables the model to refine under-represented dynamic regions and recover content that would otherwise remain missing.

Even when Gaussian capacity can be successfully redistributed spatially, a complementary limitation lies in determining *which timestamps should receive the relocated Gaussian capacity*. For example, FreeTimeGS (Wang et al., 2025) focuses on source Gaussian selection, while the temporal destination of the relocated capacity receives less consideration. In particular, relocation scores are typically aggregated across timestamps, discarding information about when strong optimization signals occur: a high-scoring Gaussian may indicate that additional capacity is needed, but provides little guidance about the particular timestamp at which that capacity is most useful. We therefore introduce *temporally biased relocation*, which preserves time-resolved optimization signals and uses them to guide the temporal destination of relocated Gaussians. By explicitly accounting for when reconstruction difficulty arises, our strategy allocates Gaussian capacity to timestamps where it is most needed, complementing the spatial allocation addressed by *seed injection* and improving reconstruction quality in both overall scenes and highly dynamic regions.

Together, *seed injection* and *temporally biased relocation* address two aspects of Gaussian allocation: allocating additional capacity to under-represented regions and determining appropriate spatial and temporal destinations for relocated capacity, respectively. We apply both strategies to FreeTimeGS, our baseline, without modifying its underlying representation. Our contributions are as follows: (i) We introduce *seed injection*, which allocates additional Gaussian primitives to poorly reconstructed dynamic regions and enables the recovery of content with insufficient or missing Gaussian support. (ii) We propose *temporally biased relocation*, which incorporates temporal optimization signals into the relocation destination, allowing Gaussian capacity to be placed at more

appropriate times and spatial locations. (iii) Experiments across multiple dynamic-scene datasets demonstrate improved reconstruction quality in both overall scenes and highly dynamic regions, including the recovery of small, rapidly moving objects.

2 Related Work

Dynamic Gaussian Splatting. Building upon 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023), dynamic approaches track or deform Gaussian primitives over time (Luiten et al., 2024; Yang et al., 2024b; Wu et al., 2024; Lu et al., 2024), with recent methods further modeling multi-frame motion dependencies (Huang et al., 2026). Deformation-based approaches, however, remain dependent on primitives established during canonicalization, making initially under-represented content difficult to recover through subsequent deformation. Explicit spatiotemporal approaches instead model temporal evolution through 4D primitives, interpolation, flexible lifespans, or continuous trajectories (Yang et al., 2024a; Li et al., 2024; Lee et al., 2024; Liao et al., 2026; Yeom et al., 2026; Wang et al., 2026). FreeTimeGS (Wang et al., 2025), which serves as our baseline, introduces flexible temporal lifespans to model Gaussian appearance and motion over time. FreeTimeGS++ (Lee et al., 2026) further improves initialization and optimization stability. SPIN-4DGS (Kim et al., 2026) addresses reconstruction under large inter-frame displacements by reducing cross-frame interference through attribute prediction from explicitly collected spatiotemporal positions. Our work instead targets Gaussian allocation as a source of dynamic reconstruction failures, without modifying the underlying dynamic Gaussian representation.

Gaussian Densification and Relocation. The original 3DGS adaptively clones or splits Gaussians based on positional gradients, while subsequent works improve density control through alternative densification criteria (Rota Bulò et al., 2024; Zhang et al., 2024; Ye et al., 2024; Zhou et al., 2026). Beyond densification, 3DGS-MCMC (Kheradmand et al., 2024) introduces relocation to redistribute underutilized Gaussian capacity. Dynamic methods additionally incorporate temporal properties such as lifespan, velocity, and visibility into density control (Liao et al., 2026; Sandu et al., 2026; Yeom et al., 2026; Wang et al., 2026). While these methods use temporal cues to guide primitive selection, the spatiotemporal destination of Gaussian capacity receives less consideration. Our method instead introduces new support in under-represented dynamic regions and uses timestamp-specific optimization signals to determine the temporal destination of relocated capacity.

Gaussian Allocation for Under-Reconstructed Regions. Error-based densification (Rota Bulò et al., 2024) directly targets under-reconstructed regions using reconstruction errors, but remains dependent on existing primitives as densification sources. LPM (Yang et al., 2025) alleviates this dependency by localizing error regions in 3D through cross-view feature correspondence, but relies on an external matching model, which may be less reliable in regions with weak visual correspondence. SpacetimeGS (Li et al., 2024) enables guided sampling even in regions with sparse or missing Gaussian support, but still relies on the current Gaussian representation to provide a coarse depth prior for localizing new primitives. In contrast, our approach directly lifts observations from under-reconstructed dynamic regions into 3D using synchronized multi-view geometry, without relying on existing Gaussian support or an external correspondence model.

3 Method

In this section, we present our approach for improving Gaussian allocation in dynamic scene reconstruction through *seed injection* and *temporal biased relocation*. We first review the FreeTimeGS representation and its Gaussian relocation strategy as preliminaries in Sec. 3.1. Then, Sec. 3.2 describes how seed injection technique introduces Gaussian primitives into under-reconstructed dynamic regions, and Sec. 3.3 details how *temporal biased relocation* redistributes Gaussian capacity across time based on temporal optimization signals.

3.1 Preliminaries

FreeTimeGS representation. FreeTimeGS (Wang et al., 2025) represents a dynamic scene using Gaussian primitives that can appear at arbitrary spatial locations and timestamps. Each Gaussian

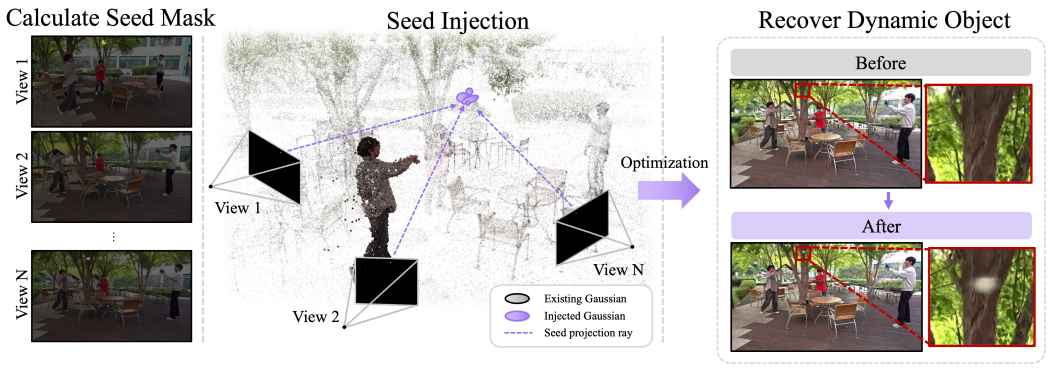

Figure 2: **Overview of Seed Injection.** We identify under-reconstructed dynamic regions from synchronized multi-view observations and sample candidate pixels from the resulting seed maps. The sampled observations are lifted into 3D using multi-view geometry to initialize new Gaussian primitives independently of the existing Gaussian distribution. After optimization, the injected Gaussians provide additional support for dynamic content that is poorly represented or missing in the original reconstruction.

includes a spatial position μ_x, temporal center μ_t, duration d, velocity v, and opacity σ, along with standard 3DGS attributes. To model local motion, FreeTimeGS assigns each Gaussian a linear motion function,

$$\mu_x(t) = \mu_x + v(t - \mu_t), \tag{1}$$

where $\mu_x(t)$ denotes the spatial position of the Gaussian at timestamp t. The contribution of each Gaussian over time is further controlled by a temporal opacity function,

$$\sigma(t) = \exp\left(-\frac{1}{2}\left(\frac{t - \mu_t}{d}\right)^2\right). \tag{2}$$

The final opacity is obtained by multiplying the base opacity by $\sigma(t)$, where μ_t and d control the temporal center and extent, respectively.

FreeTimeGS relocation. FreeTimeGS maintains a fixed Gaussian budget by periodically recycling low-opacity Gaussians through relocation. Specifically, it defines a sampling score s for each Gaussian primitive as

$$s = \lambda_g \nabla_g + \lambda_o \sigma, \tag{3}$$

where ∇_g and σ denote the spatial gradient and opacity of the Gaussian primitive, respectively, and λ_g and λ_o are their corresponding weights. At each relocation interval, Gaussian primitives whose opacity falls below a predefined threshold are reused and relocated toward primitives with high sampling scores.

3.2 Seed Injection

As described in Sec. 3.1, FreeTimeGS (Wang et al., 2025) reallocates a fixed Gaussian budget based on scores defined over existing Gaussian primitives. Consequently, highly dynamic regions, particularly those with insufficient Gaussian support, may have limited opportunities to receive additional capacity. To address this limitation, we introduce *seed injection*, which identifies under-reconstructed dynamic regions from image observations and introduces new Gaussian primitives independently of the existing Gaussian distribution. An overview of the seed injection scheme is illustrated in Fig. 2.

Dynamic Error-based Seed Map. To introduce new Gaussians into highly dynamic regions, we define a seed map using both a reconstruction error map and a dynamic mask. The reconstruction error map highlights regions that are poorly represented by the current Gaussian representation, but may also contain errors from static regions. We therefore use an additional dynamic mask to suppress static-region errors and focus the seed map on temporally varying regions. Specifically, for

each view v at time t, we construct a binary dynamic mask $D_{v,t} \in \{0,1\}^{H\times W}$ from the differences between adjacent frames. We define the per-pixel seed map as

$$M_{v,t}(p) = D_{v,t}(p)E_{v,t}(p), \qquad E_{v,t}(p) = \left\| I_{v,t}(p) - \hat{I}_{v,t}(p) \right\|_1 . \tag{4}$$

Here, $\hat{I}_{v,t}$ and $I_{v,t}$ denote the rendered and ground-truth images, respectively, and p denotes a pixel coordinate. The seed map assigns high values to pixels that are both dynamic and poorly reconstructed. The detailed procedure for constructing the seed map is provided in the supplementary material.

Multi-view Seed Lifting. For each time step t, we select a subset of C synchronized camera views, $\mathcal{S}_t \subseteq \mathcal{V}_t$. From each selected view $v \in \mathcal{S}_t$, we sample up to N_{seed} candidate seed pixels from the corresponding seed map $M_{v,t}$ and collect them as $\mathcal{P}_{v,t} = \{p^1_{v,t}, \dots, p^{N_{\text{seed}}}_{v,t}\}$, where $p^i_{v,t} \in \mathbb{R}^2$ denotes the 2D image coordinate of the i-th sampled seed candidate. Since each sampled seed provides only a 2D image location, its 3D location is ambiguous. To approximate the 3D location, we match seed candidates across views based on the proximity of their viewing rays, as candidates corresponding to the same or nearby 3D region are expected to produce rays that pass close to each other.

The viewing rays of the seed candidates are defined according to the corresponding camera parameters. Let $o_v \in \mathbb{R}^3$ denote the camera center of view v, and let $d^i_v \in \mathbb{R}^3$ denote the unit viewing direction corresponding to $p^i_{v,t}$. The corresponding viewing ray is defined as

$$r^i_v(\lambda) = o_v + \lambda d^i_v, \qquad v \in \mathcal{S}_t,\ i = 1, \dots, N_{\text{seed}}. \tag{5}$$

Here, λ is a scalar parameter that specifies a point along the viewing ray from the camera center o_v. This yields a total of CN_{seed} candidate rays, denoted by $\mathcal{R}_t = \{r^i_v \mid v \in \mathcal{S}_t,\ i = 1, \dots, N_{\text{seed}}\}$. To match seed candidates that are likely to correspond to similar 3D regions, we form all possible ray pairs from $\mathcal{R}_t$ and compute their minimum distances, considering only pairs from different views, i.e., $v \neq v'$. For a ray pair $(r^i_v, r^j_{v'})$ from views v and v', the minimum ray distance is obtained by finding the closest points on the two rays:

$$(\lambda^*_{ij}, \gamma^*_{ij}) = \arg\min_{\lambda,\gamma} \left\| r^i_v(\lambda) - r^j_{v'}(\gamma) \right\|^2_2 = \arg\min_{\lambda,\gamma} \left\| (o_v + \lambda d^i_v) - (o_{v'} + \gamma d^j_{v'}) \right\|^2_2 . \tag{6}$$

Here, λ^*_{ij} and γ^*_{ij} denote the distances from the corresponding camera centers to the points that minimize the ray-to-ray distance. The corresponding closest points in views v and v' are $q^{i,j}_v = r^i_v(\lambda^*_{ij})$ and $q^{i,j}_{v'} = r^j_{v'}(\gamma^*_{ij})$, respectively, and their distance is $\delta_{ij} = \|q^{i,j}_v - q^{i,j}_{v'}\|_2$.

We retain only geometrically consistent cross-view ray pairs and, for each ray, keep up to k nearest ray pairs based on the ray distance. For each retained pair, the corresponding 3D seed position is defined as the midpoint of the two closest points,

$$x_{\text{seed}} = \frac{1}{2}\left(q^{i,j}_v + q^{i,j}_{v'}\right). \tag{7}$$

The resulting 3D seed candidates are ranked by their ray distances δ_{ij}, and at most $N_{\max}$ seeds are retained for Gaussian initialization. New Gaussians are initialized at the retained seed positions and subsequently refined during optimization. Further implementation details are provided in the supplementary material.

Gaussian Injection. For each lifted seed x_{seed}, we initialize a new Gaussian centered at the seed position. Its scale is initialized isotropically based on the pixel footprint at the estimated depth, following prior works (Barron et al., 2021; Baranowski et al., 2026). Similar to the initialization strategy of FreeTimeGS (Wang et al., 2025), we initialize its velocity from the displacement to the nearest lifted seed in the subsequent frame. This nearest-neighbor displacement is used only as a coarse motion initialization and does not assume temporal correspondence between the seeds. The injected Gaussians thus provide new optimization starting points in under-represented regions and are subsequently optimized jointly with the existing representation.

Training Procedure. We perform seed injection during the second half of training, once the reconstruction error becomes sufficiently informative for identifying under-reconstructed regions. During this stage, relocation is temporarily disabled, and seed injection is applied sequentially over the training frames. For each frame, we sample N_{seed} candidate pixels from the corresponding seed map and inject up to $N_{\max}$ Gaussians from the resulting 3D seed locations. We then perform a small number of optimization iterations on the current frame before moving to the next one. This procedure is repeated until all frames have been processed. After completing seed injection over the full sequence, we re-enable relocation and resume the original training schedule.

3.3 Temporal Biased Relocation

As described in Sec. 3.1, FreeTimeGS reuses Gaussians whose opacity falls below a predefined threshold and relocates them toward primitives with high sampling scores. However, FreeTimeGS pays limited attention to *when* additional Gaussian capacity is needed, potentially leading to suboptimal temporal allocation. To address this limitation, we propose *temporal biased relocation*, which preserves timestamp-specific positional gradients and uses them to determine the relocation time.

We follow FreeTimeGS for selecting both the Gaussian to be relocated and the target Gaussian. However, once a target Gaussian g is selected, its trajectory spans multiple timestamps, and we must determine *when* along the trajectory the relocated Gaussian should be placed. A large screen-space positional gradient at timestamp t indicates that a stronger positional adjustment is required to explain the observations at that time. We interpret this as a signal that the current Gaussian allocation provides insufficient support at that timestamp. To estimate how strongly each timestamp requires additional Gaussian support, we use the magnitude of the target Gaussian's positional gradient at each $t \in \mathcal{T}_g$ and normalize these values over $\mathcal{T}_g$ to form a temporal relocation distribution:

$$P_g(t) = \frac{\|\nabla_{g,t}\|_2}{\sum_{u\in\mathcal{T}_g} \|\nabla_{g,u}\|_2}, \qquad t \in \mathcal{T}_g. \tag{8}$$

We select the timestamp with the highest relocation probability, $t^* = \arg\max_{t\in\mathcal{T}_g} P_g(t)$, as the time requiring the strongest additional Gaussian support. We set the temporal center of the relocated Gaussian, $\hat{\mu}_t$, to the selected timestamp t^*, and its spatial center, $\hat{\mu}_x$, to the position of the target Gaussian g at that time:

$$\hat{\mu}_t = t^*, \qquad \hat{\mu}_x = \mu_x(t^*) = \mu_x + v(t^* - \mu_t). \tag{9}$$

Here, μ_x, μ_t, and v denote the spatial center, temporal center, and velocity of the target Gaussian, respectively. We additionally adjust the temporal duration to account for the shifted temporal center, while the remaining parameters follow the original FreeTimeGS relocation scheme. Further details of the duration adjustment and relocation procedure are provided in the supplementary material.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate our method on three datasets with varying levels of scene dynamics. For N3DV (Li et al., 2022), we use all six scenes, each containing 300 frames captured at 30 FPS with 19–21 cameras at a resolution of 2704×2028. We downsample the images by $2\times$, use camera 0 for testing, and use the remaining cameras for training. For CMU Panoptic (Joo et al., 2015), we use six 150-frame sequences captured at 30 FPS and 640×360 resolution, with 27 cameras for training and cameras 0, 10, 15, and 30 for testing. Following the camera configuration, we set the near plane to 0.5 for test-view rendering to suppress artifacts from Gaussians located near the test cameras. Finally, we introduce the Custom dataset, consisting of the Baseball and Break sequences, to evaluate reconstruction under small and rapidly moving objects. Existing dynamic-scene datasets provide limited coverage of such challenging motions, despite small and fast-moving objects being particularly difficult to reconstruct with current 4DGS methods. We therefore captured two sequences specifically designed to include these cases using a synchronized multi-camera setup. The Baseball and Break sequences contain 250 and 300 frames, respectively, at a resolution of 1920×1080 and a frame rate of 30 FPS. For evaluation on dynamic regions, we generate test-view masks using

Table 1: Quantitative comparison on the N3DV dataset. † mark are sourced from the original paper. Best and second-best results are highlighted in **bold** and underlined, respectively.

Method	PSNR ↑	Masked PSNR ↑	DSSIM_1 ↓	DSSIM_2 ↓	LPIPS ↓
DyNeRF† (Li et al., 2022)	29.58	–	–	0.020	0.083
K-Planes† (Fridovich-Keil et al., 2023)	31.63	–	–	0.018	–
NeRFPlayer† (Song et al., 2023)	30.96	–	0.034	–	0.111
Deformable-3DGS† (Yang et al., 2024b)	31.15	–	0.030	–	0.049
SpaceTimeGS† (Li et al., 2024)	32.05	–	0.026	0.014	0.044
RealTime4DGS (Yang et al., 2024a)	31.97	31.90	0.027	0.015	0.051
Ex4DGS (Lee et al., 2024)	32.01	31.33	0.029	0.015	0.046
4DGaussian (Wu et al., 2024)	30.74	29.46	0.033	0.017	0.057
FreeTimeGS (Wang et al., 2025)	32.52	32.27	**0.022**	**0.011**	0.041
FreeTimeGS + Ours	**32.56**	**32.87**	**0.022**	**0.011**	**0.040**

Table 2: Quantitative comparison on the CMU Panoptic dataset.

Method	PSNR ↑	Masked PSNR ↑	DSSIM_1 ↓	DSSIM_2 ↓	LPIPS ↓
Dynamic 3DGS (Luiten et al., 2024)	28.72	27.13	0.051	0.024	0.095
RealTime4DGS (Yang et al., 2024a)	28.01	25.84	0.044	0.023	0.111
Ex4DGS (Lee et al., 2024)	24.86	23.64	0.054	0.029	0.120
4DGaussian (Wu et al., 2024)	26.92	23.68	0.047	0.024	0.116
FreeTimeGS (Wang et al., 2025)	28.34	26.68	0.039	0.019	0.110
FreeTimeGS + Ours	**29.05**	**27.28**	**0.032**	**0.016**	**0.063**

SAM3 (Carion et al., 2026), as ground-truth dynamic masks are unavailable. We use dataset-specific prompts to segment the moving objects and report masked metrics using the resulting masks. For extremely fast-moving objects that are not reliably captured by SAM3, we additionally incorporate the dynamic masks used for Seed Injection to ensure that these regions are included in the evaluation. Evaluation metrics include PSNR, dyanmic masked PSNR, DSSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018). All experiments are conducted on a single NVIDIA RTX 4090 GPU with 24 GB of memory.

Implementation Details. We build on the reproduced FreeTimeGS (Wang et al., 2025) implementation and follow its default training configuration unless otherwise specified. We use 500K Gaussians for N3DV (Li et al., 2022) and CMU Panoptic (Joo et al., 2015), and 1.5M Gaussians for the Custom dataset. We train N3DV, CMU Panoptic, and the Custom dataset for 30K, 50K, and 100K iterations, respectively, with longer training for the Custom dataset due to its more challenging outdoor scenes and high-frequency background details. For Seed Injection, we use a subset of eight synchronized camera views to construct the dynamic error-based seed map. We set $N_{\text{seed}} = 48$ and $N_{\max} = 256$, followed by 10 iterations of frame-specific optimization. The velocity and scale are initialized as described in Sec. 3.2, while opacity and temporal duration are initialized to 0.05 and 0.03, respectively. Temporal Biased Relocation is applied whenever relocation is performed throughout training.

4.2 Comparison Experiments

N3DV We first evaluate our method on N3DV (Li et al., 2022), which contains relatively mild scene dynamics compared with the other datasets considered in our experiments. Quantitative comparisons are presented in Table 1, with additional qualitative results provided in the supplementary material. Our method improves reconstruction quality over the FreeTimeGS baseline, with particularly noticeable improvements in dynamic regions, while preserving the well-reconstructed static content.

Table 3: Ablation study of our proposed components on the CMU Juggle, Football, and Basketball sequences.

Method	PSNR ↑	Masked PSNR ↑	$DSSIM_1$ ↓	$DSSIM_2$ ↓	LPIPS ↓	Training Time (m) ↓
FreeTimeGS (Wang et al., 2025)	28.22	27.58	0.045	0.023	0.161	**5.69**
FreeTimeGS w/ Seed Injection	28.53	27.82	0.045	0.022	0.160	5.85
FreeTimeGS w/ Temporal Biased Relocation	<u>28.57</u>	<u>28.02</u>	**0.034**	**0.017**	**0.080**	<u>5.74</u>
FreeTimeGS + Ours	**28.98**	**28.24**	**0.034**	**0.017**	<u>0.082</u>	5.91

Table 4: Quantitative comparison on the Custom dataset.

Method	PSNR ↑	Masked PSNR ↑	$DSSIM_1$ ↓	$DSSIM_2$ ↓	LPIPS ↓
RealTime4DGS (Yang et al., 2024a)	20.46	24.45	0.154	0.114	0.249
Ex4DGS (Lee et al., 2024)	19.74	22.00	0.179	0.132	0.335
4DGaussian (Wu et al., 2024)	20.80	22.03	0.163	0.114	0.308
FreeTimeGS (Wang et al., 2025)	<u>20.84</u>	<u>27.37</u>	<u>0.131</u>	<u>0.097</u>	<u>0.170</u>
FreeTimeGS + Ours	**20.98**	**27.47**	**0.128**	**0.095**	**0.160**

CMU Panoptic We further evaluate our method on CMU Panoptic (Joo et al., 2015), which contains diverse and highly dynamic human motions. In particular, the Juggle sequence includes small and rapidly moving objects, providing a challenging scenario where the juggling balls can be poorly reconstructed or even disappear due to an insufficient number of Gaussians. As shown in Fig. 3, our method successfully recovers the juggling balls while also better preserving the details of the rapidly moving hands. Beyond these challenging regions, our method improves reconstruction across a variety of dynamic content. It also better allocates Gaussians to under-reconstructed static regions, leading to improved overall reconstruction quality. These improvements are reflected in the quantitative results in Table 2. Additional qualitative results on the other CMU Panoptic sequences are provided in the supplementary material.

Custom Dataset We next evaluate our method on the Custom dataset, consisting of the Baseball and Break sequences. Quantitative results are reported in Table 4, with additional qualitative comparisons provided in the supplementary material. In the Baseball sequence, the rapidly moving ball and baseball bat disappear in the FreeTimeGS reconstruction, while fine details of the glove are poorly preserved. As shown in Fig. 1a, our method successfully recovers the missing baseball, demonstrating its effectiveness in reconstructing small and rapidly moving objects. Our method also better preserves the details of the bat and glove, as shown in Fig. 1b. Similarly, in the Break sequence, our method successfully reconstructs the small and rapidly moving pirate roulette that is missing in the baseline. Additional qualitative results for both sequences are provided in the supplementary material.

4.3 Ablation Studies and analysis

Component Ablation. We conduct an ablation study on three CMU Panoptic sequences, Basketball, Football, and Juggle, and report the average results across the three sequences to evaluate the contribution of each proposed component. As shown in Table 3, Temporal Biased Relocation improves reconstruction quality across both dynamic and under-reconstructed static regions, while Seed Injection complements it by recovering rapidly moving objects, such as the juggling balls, that are poorly represented by the baseline as shown in Fig. 5. Combining both components further improves reconstruction quality while introducing only a small increase in training time, with all CMU Panoptic experiments using 500K Gaussians and 50K training iterations.

Analysis of Seed Injection. We further analyze where new Gaussians are initialized by Seed Injection. Fig. 4 visualizes the accumulated injection locations by projecting the centers of newly injected Gaussians onto the ground-truth test view at the time of injection, where the red dots indicate the projected Gaussian centers. As shown in the figure, the injected Gaussians are concentrated around rapidly moving regions, such as the thrown baseball and the swinging bat. This demonstrates that

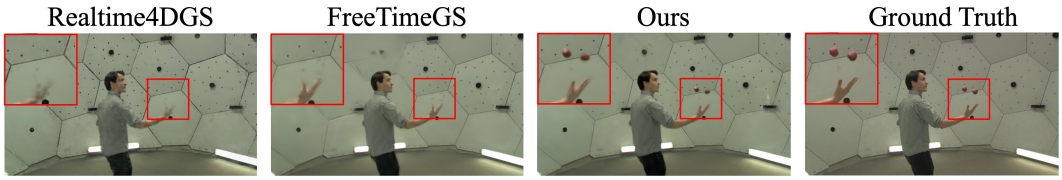

Figure 3: Qualitative comparison on the CMU Panoptic Juggle sequence. Our method better reconstructs the rapidly moving juggling balls while preserving fine details of the hands.



Figure 4: Visualization of Seed Injection on the Custom dataset. Red markers indicate the projected centers of newly injected Gaussians, which are concentrated around rapidly moving objects such as the baseball and bat.

Seed Injection effectively introduces new Gaussians into challenging dynamic regions where additional Gaussian support is needed.

5 Conclusions

We presented a Gaussian allocation framework for improving the reconstruction of challenging dynamic content in 4D Gaussian Splatting. Our approach addresses two complementary limitations in existing allocation strategies. Seed Injection introduces new Gaussian primitives directly from synchronized multi-view observations, providing additional support for under-represented dynamic regions even when suitable Gaussians are absent from the current representation. Temporally Biased Relocation preserves time-resolved optimization signals to guide relocated Gaussian capacity toward the timestamps where it is most needed. We integrate both strategies into FreeTimeGS while preserving its underlying representation. Experiments across multiple dynamic-scene datasets demonstrate consistent improvements in reconstruction quality, particularly in highly dynamic regions, while recovering small and rapidly moving objects that can disappear in the baseline reconstruction. These results highlight the importance of explicitly considering both spatial and temporal Gaussian allocation for robust dynamic-scene reconstruction.

Limitations. Our Seed Injection relies on a simple frame-difference-based criterion to identify dynamic regions by comparing adjacent frames. Consequently, rapidly moving objects may not be detected when their appearance is similar to the surrounding regions, resulting in weak temporal differences despite substantial motion. In such cases, Seed Injection may fail to introduce additional Gaussian primitives into these under-reconstructed dynamic regions. We leave addressing this limitation to future work. In addition, although we evaluate our method on multiple dynamic-scene datasets, further evaluation on more diverse and highly dynamic datasets would be valuable as such datasets become available.

AI use statement

In this work, we used a large language model (LLM) in a limited capacity for proofreading and improving the readability of the manuscript. All technical content, including the proposed method and experiments, was conceived and written by the authors. All LLM-assisted edits were reviewed and

verified by the authors. In accordance with the ICLR guidelines, we disclose our use of generative AI below.

Ethics statement

Our experiments primarily rely on publicly available datasets, together with a custom dataset used with appropriate permission. For the custom data, we blur most visible faces to protect individual privacy and prevent identification. We are not aware of any additional ethical concerns associated with the proposed method or its intended applications.

Reproducibility statement

The method, the implementation, training setup, and evaluation protocols are described in Sec. 4, and further details are provided in the supplementary materials. We will release the source code upon publication.

References

Jeongmin Bae, Seoha Kim, Youngsik Yun, Hahyun Lee, Gun Bang, and Youngjung Uh. Per-gaussian embedding-based deformation for deformable 3d gaussian splatting. In *European Conference on Computer Vision*, pp. 321–335. Springer, 2024.

Bartlomiej Baranowski, Stefano Esposito, Patricia Gschoßmann, Anpei Chen, and Andreas Geiger. Conegs: Error-guided densification using pixel cones for improved reconstruction with fewer primitives. In *2026 International Conference on 3D Vision (3DV)*, pp. 989–999. IEEE, 2026.

Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5835–5844. IEEE, 2021.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris Coll-Vinent, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. Sam 3: Segment anything with concepts. In *International conference on learning representations*, volume 2026, pp. 138846–138923, 2026.

Jun Chen, Ryosuke Watanabe, Keisuke Nonaka, Tomoaki Konno, Hiroshi Sankoh, and Sei Naito. Fast free-viewpoint video synthesis algorithm for sports scenes. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3209–3215. IEEE, 2019.

Alvaro Collet, Ming Chuang, Pat Sweeney, Don Gillett, Dennis Evseev, David Calabrese, Hugues Hoppe, Adam Kirk, and Steve Sullivan. High-quality streamable free-viewpoint video. *ACM Transactions on Graphics (ToG)*, 34(4):1–13, 2015.

Yanran Dai, Jing Li, Yuqi Jiang, Haidong Qin, Bang Liang, Shikuan Hong, Haozhe Pan, and Tao Yang. Real-time distance field acceleration based free-viewpoint video synthesis for large sports fields. *Computational Visual Media*, 10(2):331–353, 2024.

Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12479–12488. IEEE, 2023.

Tingxuan Huang, Haowei Zhu, Jun-hai Yong, Hao Pan, and Bin Wang. Mango-gs: Enhancing spatio-temporal consistency in dynamic scenes reconstruction using multi-frame node-guided 4d gaussian splatting. *arXiv preprint arXiv:2603.11543*, 2026.

Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4220–4230. IEEE, 2024.

Hanbyul Joo, Hao Liu, Lei Tan, Lin Gui, Bart Nabbe, Iain Matthews, Takeo Kanade, Shohei Nobuhara, and Yaser Sheikh. Panoptic studio: A massively multiview system for social motion capture. In *Proceedings of the IEEE international conference on computer vision*, pp. 3334–3342, 2015.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, George Drettakis, et al. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

Shakiba Kheradmand, Daniel Rebain, Gopal Sharma, Weiwei Sun, Yang-Che Tseng, Hossam Isack, Abhishek Kar, Andrea Tagliasacchi, and Kwang Moo Yi. 3d gaussian splatting as markov chain monte carlo. *Advances in Neural Information Processing Systems*, 37:80965–80986, 2024.

Seung-gyeom Kim, Areum Kim, Yongjae Yoo, and Sukmin Yun. Implicit 4d gaussian splatting for fast motion with large inter-frame displacements. In *International Conference on Learning Representations*, volume 2026, pp. 43331–43349, 2026.

Jason Lawrence, Dan B Goldman, Supreeth Achar, Gregory Major Blascovich, Joseph G. Desloge, Tommy Fortes, Eric M. Gomez, Sascha Häberling, Hugues Hoppe, Andy Huibers, Claude Knaus, Brian Kuschak, Ricardo Martin-Brualla, Harris Nover, Andrew Ian Russell, Steven M. Seitz, and Kevin Tong. Project starline: A high-fidelity telepresence system. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 40(6), 2021.

Junoh Lee, ChangYeon Won, Hyunjun Jung, Inhwan Bae, and Hae-Gon Jeon. Fully explicit dynamic gaussian splatting. *Advances in Neural Information Processing Systems*, 37:5384–5409, 2024.

Lucas Yunkyu Lee, Soonho Kim, Youngwook Kim, Sangmin Kim, and Jaesik Park. Freetimegs++: Secrets of dynamic gaussian splatting and their principles. *arXiv preprint arXiv:2605.03337*, 2026.

Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5511–5521. IEEE, 2022.

Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. Spacetime gaussian feature splatting for real-time dynamic view synthesis. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8508–8520. IEEE, 2024.

Zhanfeng Liao, Jiajun Zhang, Hanzhang Tu, Zhixi Wang, Yunqi Gao, Hongwen Zhang, and Yebin Liu. Sharptimegs: Sharp and stable dynamic gaussian splatting via lifespan modulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11798–11807, June 2026.

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 *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8900–8910. IEEE, 2024.

Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *2024 International Conference on 3D Vision (3DV)*, pp. 800–809. IEEE, 2024.

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. *Communications of the ACM*, 65(1):99–106, 2021.

Samuel Rota Bulò, Lorenzo Porzi, and Peter Kontschieder. Revising densification in gaussian splatting. In *European Conference on Computer Vision*, pp. 347–362. Springer, 2024.

Vikram Sandu, Mayurdeep Pathak, and Rajiv Soundararajan. Temporally aware densification for dynamic 3d gaussian splatting. *arXiv preprint arXiv:2606.23212*, 2026.

Liangchen Song, Anpei Chen, Zhong Li, Zhang Chen, Lele Chen, Junsong Yuan, Yi Xu, and Andreas Geiger. Nerfplayer: A streamable dynamic scene representation with decomposed neural radiance fields. *IEEE Transactions on Visualization and Computer Graphics*, 29(5):2732–2742, 2023.

Xuezhen Wang, Li Ma, Yulin Shen, Zeyu Wang, and Pedro V Sander. Retimegs: Continuous-time reconstruction of 4d gaussian splatting. *arXiv preprint arXiv:2603.13783*, 2026.

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21750–21760. IEEE, 2025.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

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 *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20310–20320. IEEE, 2024.

Jiawei Xu, Zexin Fan, Jian Yang, and Jin Xie. Grid4d: 4d decomposed hash encoding for high-fidelity dynamic gaussian splatting. *Advances in Neural Information Processing Systems*, 37: 123787–123811, 2024.

Haosen Yang, Chenhao Zhang, Wenqing Wang, Marco Volino, Adrian Hilton, Li Zhang, and Xiatian Zhu. Improving gaussian splatting with localized points management. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21696–21705. IEEE, 2025.

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*, volume 2024, pp. 9142–9159, 2024a.

Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20331–20341. IEEE, 2024b.

Zongxin Ye, Wenyu Li, Sidun Liu, Peng Qiao, and Yong Dou. Absgs: Recovering fine details in 3d gaussian splatting. In *Proceedings of the 32nd ACM international conference on multimedia*, pp. 1053–1061, 2024.

Suwoong Yeom, Joonsik Nam, Seunggyu Choi, Lucas Yunkyu Lee, Sangmin Kim, Jaesik Park, Joonsoo Kim, Kugjin Yun, Kyeongbo Kong, and Sukju Kang. Trigs: Temporal rigid-body motion for scalable 4d gaussian splatting. *arXiv preprint arXiv:2604.00538*, 2026.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 586–595. IEEE, 2018.

Zheng Zhang, Wenbo Hu, Yixing Lao, Tong He, and Hengshuang Zhao. Pixel-gs: Density control with pixel-aware gradient for 3d gaussian splatting. In *European conference on computer vision*, pp. 326–342. Springer, 2024.

Zheng Zhou, Yu-Jie Xiong, Jia-Chen Zhang, Chun-Ming Xia, Xihe Qiu, and Hongjian Zhan. Gradient-direction-aware density control for 3d gaussian splatting. In *International Conference on Learning Representations*, volume 2026, pp. 21415–21437, 2026.

A Appendix

A.1 More Ablation Studies

We provide additional qualitative ablation results on the CMU Panoptic dataset to further analyze the effect of each proposed component. As shown in Fig. 5, Seed Injection helps recover under-reconstructed fast-moving objects, while Temporal Biased Relocation improves the reconstruction of dynamic regions by reallocating Gaussian support to more appropriate timestamps. Combining both components yields the most complete reconstruction.

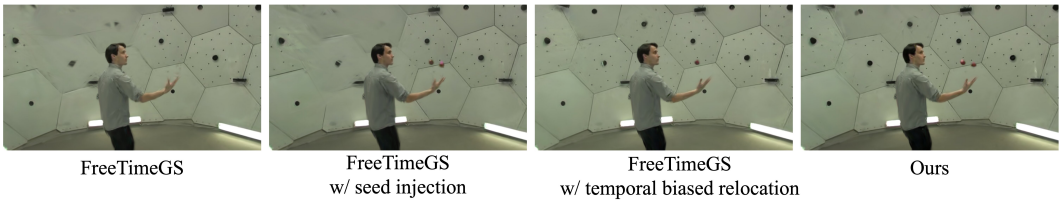

Figure 5: Qualitative ablation of Seed Injection and Temporal Biased Relocation on the CMU Panoptic Juggle sequence. Both components improve the reconstruction of dynamic content, while their combination provides the most complete reconstruction.

A.2 Analysis on Method

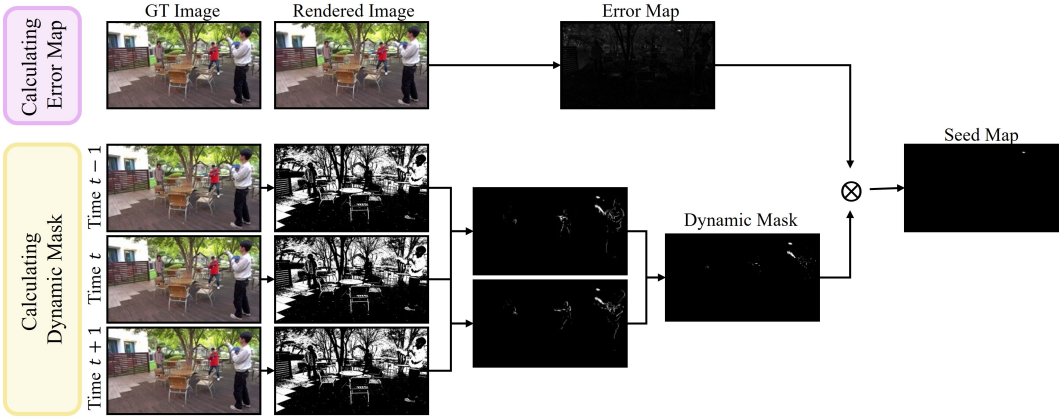

Figure 6: Construction of the dynamic error-based seed map. The reconstruction error map is computed from the difference between the ground-truth and rendered images, while the dynamic mask is obtained from the intersection of temporal difference masks from adjacent frames. The final seed map is obtained by element-wise multiplication of the reconstruction error map and the dynamic mask, highlighting under-reconstructed dynamic regions.

Dynamic Error-based Seed Map. In this section, we describe the detailed construction of the seed map. As introduced in the main paper, the seed map is defined by combining a reconstruction error map with a dynamic mask. As illustrated in Fig. 6, the construction consists of two main steps: computing the reconstruction error map and constructing the dynamic mask.

The reconstruction error map is designed to capture regions that are insufficiently represented by the current Gaussian representation. Specifically, for each view v at timestamp t, we compute the pixel-wise difference between the rendered image $\hat{I}_{v,t}$ and the ground-truth image $I_{v,t}$:

$$E_{v,t}(p) = \left\| I_{v,t}(p) - \hat{I}_{v,t}(p) \right\|_1, \tag{10}$$

where p denotes a pixel coordinate.

To restrict the reconstruction error to dynamic regions, we construct a binary dynamic mask using the adjacent frames at timestamps $t-1$ and $t+1$. We first convert $I_{v,t-1}$, $I_{v,t}$, and $I_{v,t+1}$ into grayscale images and compute two temporal difference masks:

$$\begin{aligned} M_{v,t}^{\text{rear}} &= \mathbf{1}\left(I_{v,t}^{\text{gray}} - I_{v,t-1}^{\text{gray}} > 0\right), \\ M_{v,t}^{\text{front}} &= \mathbf{1}\left(I_{v,t}^{\text{gray}} - I_{v,t+1}^{\text{gray}} > 0\right). \end{aligned} \tag{11}$$

The final dynamic mask is obtained by taking the intersection of the two difference masks:

$$D_{v,t} = M_{v,t}^{\text{rear}} \wedge M_{v,t}^{\text{front}}. \tag{12}$$

This intersection retains pixels that consistently exhibit temporal changes with respect to both neighboring frames, while suppressing transient differences that appear only in one direction.

Finally, we construct the seed map by multiplying the reconstruction error map with the dynamic mask:

$$M_{v,t}(p) = D_{v,t}(p) E_{v,t}(p). \tag{13}$$

This allows the seed map to reflect the degree of under-representation measured by the reconstruction error, while restricting the response to temporally dynamic regions. Consequently, regions with large reconstruction errors in static areas are suppressed, whereas poorly reconstructed dynamic regions receive high seed values.

Multi-view Seed Lifting. Alg. 1 describes the detailed process of Multi-view Seed Lifting. As mentioned earlier, we match seed candidates across views using the proximity of their viewing rays and use the resulting 3D locations to initialize new Gaussians.

Algorithm 1 Multi-view Seed Lifting

Require: Seed maps $\{M_{v,t}\}$, selected views $\mathcal{S}_t$, candidates per view N
Ensure: 3D seed positions $\mathcal{X}_{\text{seed}}$

```
Sample N seed pixels from each view and back-project them into viewing rays
Form all cross-view ray pairs
for each cross-view ray pair do
    Compute the closest points and ray distance
end for
Retain geometrically consistent k-nearest ray pairs
Set each 3D seed to the midpoint of the corresponding closest points
Keep at most N_max seeds with the smallest ray distances
return X_seed
```

Temporal Biased Relocation. Alg. 2 describes the detailed procedure of Temporal Biased Relocation. We follow the original FreeTimeGS strategy to select the Gaussian to be relocated and its target Gaussian, and determine the relocation timestamp using the target Gaussian's timestamp-specific positional gradients.

Algorithm 2 Temporal Biased Relocation

Require: Gaussian set $\mathcal{G}$, timestamp-wise positional gradients
Ensure: Relocated Gaussians

1: Select the Gaussian to be relocated and the target Gaussian g using the original FreeTimeGS relocation strategy
2: **for** each observed timestamp $t \in \mathcal{T}_g$ **do**
3: Retrieve the positional gradient associated with (g, t)
4: **end for**
5: Normalize the timestamp-specific gradient magnitudes over $\mathcal{T}_g$ to obtain $P_g(t)$
6: Sample the relocation timestamp $t^* = \arg\max_{t \in \mathcal{T}_g} P_g(t)$
7: Set the relocated temporal center $\hat{\mu}_t \leftarrow t^*$
8: Set the relocated spatial center $\hat{\mu}_x \leftarrow \mu_x + v(t^* - \mu_t)$
9: Compute the target's temporal support radius $r \leftarrow \sigma_t \sqrt{2 \ln 10}$
10: Set the relocated temporal duration $\hat{\sigma}_t \leftarrow \min\left(\sigma_t, \max\left(\epsilon, \frac{\min\{|t^* - (\mu_t - r)|, |t^* - (\mu_t + r)|\}}{\sqrt{2 \ln 10}}\right)\right)$
11: Apply the remaining FreeTimeGS relocation updates
12: **return** relocated Gaussians

A.3 Qualitative Comparisons

We provide additional qualitative comparisons on the N3DV, CMU Panoptic, and Custom datasets. Specifically, we present the Flame Steak, Sear Steak, and Cook Spinach sequences from N3DV, the Football and Baseball sequences from CMU Panoptic, and the Break sequence from the Custom dataset.



Figure 7: Qualitative results on the N3DV dataset.

A.4 Temporal Biased Relocation

We further analyze the behavior of Temporal Biased Relocation (TBR) by measuring the number of relocation operations during training on the CMU Panoptic dataset. Figure 10 compares the original FTGS with FTGS equipped with TBR. We observe that TBR generally requires fewer relocation operations during training. This suggests that incorporating temporal optimization signals guides

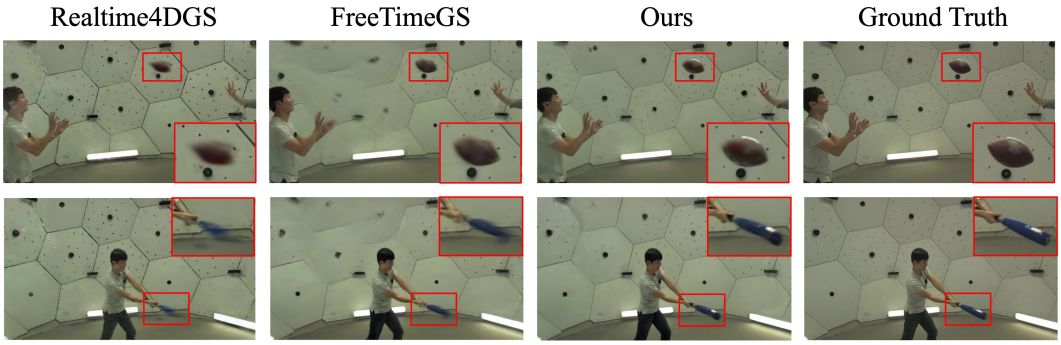

Figure 8: Qualitative results on the CMU Panoptic dataset.

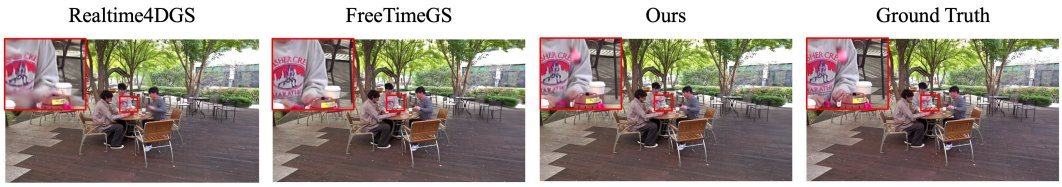

Figure 9: Qualitative results on the Custom dataset.

relocated Gaussians toward more effective temporal destinations, thereby reducing the need for repeated relocation.

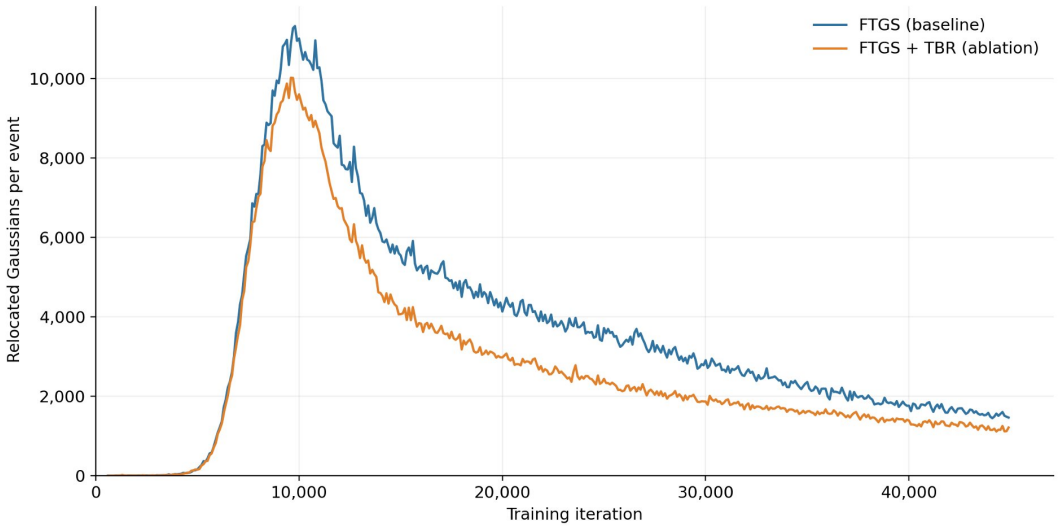

Figure 10: Number of relocation operations during training on the CMU Panoptic dataset. TBR denotes Temporal Biased Relocation.