# GauFRe: Gaussian Deformation Fields for Real-time Dynamic Novel View Synthesis

Yiqing Liang‡, Numair Khan, Zhengqin Li, Thu Nguyen-Phuoc,
Douglas Lanman, James Tompkin‡, Lei Xiao
Meta ‡Brown University

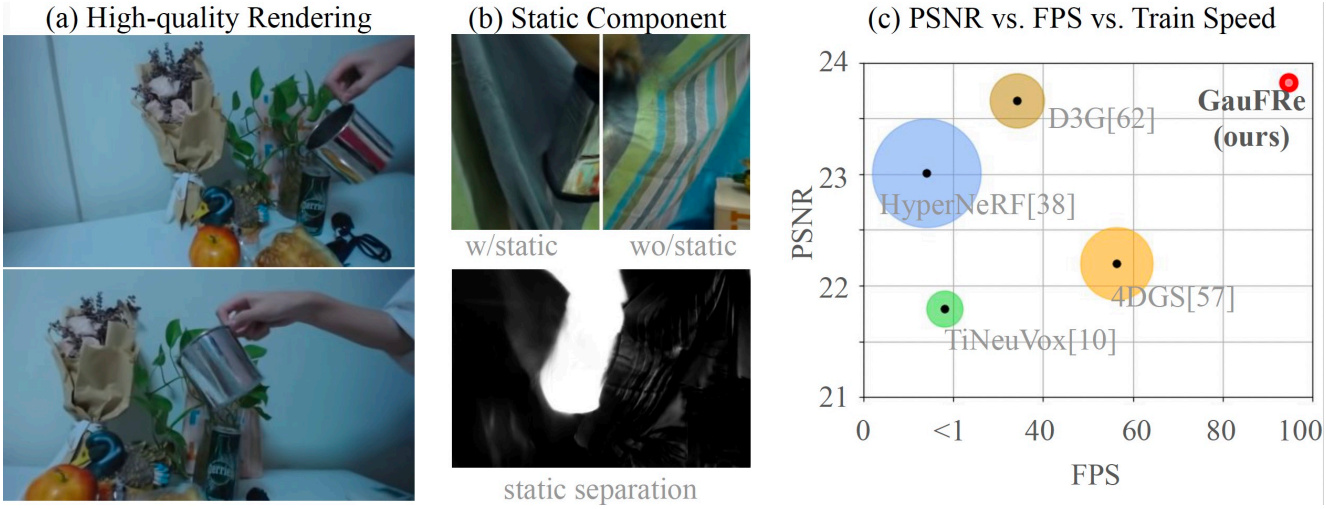


Figure 1. **(a)**: GauFRe's dynamic scene reconstruction results on the NeRF-DS [61] real-world dataset. **(b)**: Our static component improves dynamic object rendering. **(c)**: PSNR, real-time rendering performance, and optimization time (circle size) of state-of-the-art NeRF-based methods [10, 38] and Gaussian-Splatting-based methods [57, 62] on NeRF-DS at 480×270 resolution.

## Abstract

*We propose a method that achieves state-of-the-art rendering quality and efficiency on monocular dynamic scene reconstruction using deformable 3D Gaussians. Implicit deformable representations commonly model motion with a canonical space and time-dependent backward-warping deformation field. Our method, GauFRe, uses a forward-warping deformation to explicitly model non-rigid transformations of scene geometry. Specifically, we propose a template set of 3D Gaussians residing in a canonical space, and a time-dependent forward-warping deformation field to model dynamic objects. Additionally, we tailor a 3D Gaussian-specific static component supported by an inductive bias-aware initialization approach which allows the deformation field to focus on moving scene regions, improving the rendering of complex real-world motion. The differentiable pipeline is optimized end-to-end with a self-supervised rendering loss. Experiments show our method achieves competitive results and higher efficiency than both previous state-of-the-art NeRF and Gaussian-based methods. For real-world scenes, GauFRe can train in ≈20 mins and offer 96 FPS real-time rendering on an RTX 3090 GPU.*

## 1. Introduction

High-quality 3D reconstruction of dynamic scenes from RGB images is a persistent challenge in computer vision. The challenge is especially great from monocular camera video: the setting is ill-posed as constraints on the surface geometry must be formed by simultaneously solving for an estimate of the scene's motion over time. Structure from motion [43, 44] provides an estimate of rigid motion for static scenes, but real-world scenes have motions that extend beyond rigid or piecewise rigid to continual deformation, such as on human subjects. Given this challenge, one relaxation of the problem is to consider novel view synthesis instead, where we reconstruct the appearance of the scene to allow applications in editing to re-pose or re-time the scene.

For scenes with many cameras, optimization-based inverse graphics approaches can use image reconstruction losses to achieve high quality static or dynamic view synthesis. These often use neural networks as a function to predict the values of physical properties in a field, such as the density and radiance volumes within the influential neural radiance field (NeRF) technique [35]. Faster optimiza-

tion and subsequent rendering can be achieved with the help of spatial indexing data structures, such as voxel grids [12], octrees [67], and multi-scale hash tables [36, 48], or with proxy geometries such as planes [3, 11]. Despite the spatial structures, these rely on computationally-expensive volume rendering to create an image, which requires many samples along each ray to render a pixel.

Following point-based graphics [65, 73], differentiable primitive-based counterparts can be rasterized for faster speed. Representing a scene by a Gaussian primitive set is convenient as they are differentiable everywhere [41], can be splatted in closed form [34, 47], and can be z-sorted efficiently under small-Gaussian assumptions without ray marching [21]. Careful efficient implementation [5, 20] leads to real-time static scene rendering at high resolutions, and overall produces compelling results.

Extending a primitive system to dynamic scenes is natural, with the idea that each Gaussian primitive represents a moving and deforming particle or blob/area of space tracked through time—the Lagrangian interpretation in the analogy to fluid flow. Directly optimizing each primitive's trajectory along time is easy in settings with sufficient constraints upon the motion of the flow, e.g., in 360°multi-camera settings [33]. For the under-constrained monocular video setting where constraints are sparse, it is challenging as accurate prediction of both geometry and motion are required, leading to failed reconstruction or low-quality output.

Backward-warping deformation fields (DFs) are well-studied for volume-rendering-based neural fields as a solution to the monocular challenge [51]. This approach samples multiple ray points at specific times and passes them into a backward-warping DF to query a canonical space of reconstruction [10, 37, 38, 40, 51]. For primitive-based systems like Gaussian Splatting (GS), to render a ray, the status of all primitives at a timestep must be known at once, causing a backward-warping DF approach to be inefficient.

Instead, our method uses a forward-warping DF design, where the DF predicts the deformed primitive system at a certain time. More specifically, GauFRe uses a Gaussian primitive set residing in a canonical space, and a forward-warping DF conditioned on time to estimate the temporal set. We relate the forward-warping DF to how a physical 3D space point should change state along time [33], and parameterize the deformation for Gaussian attributes accordingly. Thanks to efficient GS optimization and real-time rendering with a CUDA rasterizer [20], GauFRe optimization takes $\approx$20 mins. instead of hours for a NeRF, with real-time rendering close to 100 FPS.

Beyond that, many regions of real-world dynamic scenes are static, and handling these separately can increase quality of both static and dynamic regions. Previous volume-rendering-based works have kept a static component parallel to the dynamic component, and combined the prediction from the two components for each queried ray sample. There is no ray sampling for primitive-based systems and so this design is no longer valid. As such, we design a GS-specific static component with a separate non-deforming set of Gaussians that are initialized around SfM-derived 3D points [43], and combine this set with the deformable set to serve as the final primitive system. Additionally, we show that an inductive-bias-aware Gaussian initialization helps to incorporate the static component. The method, including the canonical GS primitive set, the non-deforming primitive set, and the forward-warping DF, is optimized end-to-end with self-supervised rendering loss.

In summary, we contribute:

1. GauFRe, a forward-deformation-based dynamic scene representation of Gaussian primitives for real-time monocular-input dynamic view synthesis, enabling 96 FPS rendering on one RTX 3090.
2. A GS-specific static component that represents static regions and further enhances reconstruction when complex motion is present.
3. Experiments on both synthetic and real-world datasets show that GauFRe achieves competitive qualitative/quantitative results and efficiency compared with previous state-of-the-art methods.

## 2. Related Work

Following the success of neural radiance fields (NeRFs) [35] at static reconstruction, early methods [26, 37, 40] proposed extending it to dynamic scenes by using an additional network to model motion. However, such methods must implicitly model both the 3D scene and its motion as a continuous representation using large MLPs. Thus they tend to be even more expensive to train and evaluate than the original, static radiance field representations.

To address these shortcomings, recent approaches incorporate explicit constraints on the 3D space to train with simpler MLPs. Back to static reconstruction, variants of grid structure including planes are the most commonly used constraint [4, 12, 18, 48, 67]. Optimization-based point graphics [1, 56, 60, 69] are also popular. These include spherical proxy geometries [24], splatting-based approaches [20, 21, 65], methods for computing derivatives of points rendered to single pixels [42], and view-varying optimization of points [22]. Such explicit representations succeed in accelerating the optimization and rendering.

Given above success, an intuitive approach is to extend them to dynamic scenes. 3D voxel grids can be extended into a fourth dimension for time. Unfortunately, the memory requirements of such a 4D grid quickly become prohibitive even for short sequences. As a result, a number of methods propose structures and techniques that reduce the memory complexity while still fundamentally being four-

dimensional grids. Park *et al*. [39] extend Muller *et al*.'s multi-level spatial hash grid [36] to 4D, and additionally allow for the separate learning of static and dynamic features. This latter capability allows the model to focus the representational power of the 4D hash grid on dynamic regions. Another approach factorizes the spatio-temporal grid into low-dimensional components. Jang and Kim [17] propose rank-one vectors or low-rank matrices, providing a 4D counterpart of the 3D tensorial radiance fields of Chen *et al*. [4]. Shao *et al*. [45] hierarchically decompose the scene into three time-conditioned volumes, each represented by three orthogonal planes of feature vectors. An even more compact *HexPlanes* solution by Cao and Johnson [3] uses six planes, each spanning two of the four spatio-temporal axes. A similar decomposition is presented by Fridovich-Keil *et al*. [11] as part of a general representation that uses $\binom{d}{2}$ planes to factorize any arbitrary $d$-dimensional space. GauFRe models dynamic scene using deformable 3D-GS, achieving superior rendering quality, training speed and real-time rendering on real-world data in comparison.

Some NeRF-based works also attempt to split static and dynamic parts [2, 26, 28, 31, 46, 52, 53, 58]. They either make heuristic-based assumptions [46, 52, 53, 59], or needs extra supervision than monocular RGB video [2, 14, 26–28, 31, 53, 66, 71]. In comparison, GauFRe proposes a static component compatible with GS-based representation, does not make such assumption and only requires color image as input. [58] has closest setting, but we enjoy quick training (mins vs. days) and real-time rendering.

**Dynamic Gaussian Splatting.** A number of papers propose using dynamic Gaussian representations [6–9, 16, 19, 23, 25, 29, 30, 32, 49, 63, 68]. Liuten *et al*. [33] consider the 360° multi-camera case that is constrained in spacetime and take a Lagrangian tracking approach. Yang et al. [64] directly optimize 4D Gaussian position and color over time. Zielonka *et al*. [72] use Gaussians to approach the problem of tracking driveable human avatars from multi-camera capture. Similar to our approach, Wu *et al*. [57] and Yang *et al*. [62] both use a deformation field to model motion. Unlike our approach, neither considers static modeling. Further, we perform more detailed analysis of the design choices in a forward-warping DF approach, such that GauFRe sits within a sweet spot of competitive rendering quality with faster training and rendering.

# 3. Method

## 3.1. Preliminaries: 3D Gaussian Splatting

We use Kerbl *et al*.'s [20] 3D Gaussians as our underlying scene representation. We recapitulate the primary aspects of their method to establish context for our discussion.

A 3D scene is represented as a set of $n$ points $\{\mathbf{x}_i \in \mathbb{R}^3, i = 1, ..., n\}$. Each point is associated with features $(\mathbf{\Sigma}_i, \sigma_i, \mathbf{c}_i)$. These define the local radiance field as an anisotropic Gaussian distribution centered at $\mathbf{x}_i$ with covariance $\mathbf{\Sigma}_i \in \mathbb{R}^{3\times3}$, scalar density $\sigma_i$, and view-dependent color $\mathbf{c}_i$ represented by $m$-order spherical harmonics. Given a set of multi-view images of the scene and a suitable volumetric renderer, we can optimize a reconstruction objective over the set of Gaussians $\{\mathcal{G}_i = (\mathbf{x}_i, \mathbf{\Sigma}_i, \sigma_i, \mathbf{c}_i)\}$ to represent the scene's global radiance field. To constrain $\mathbf{\Sigma}_i$ to a valid positive semi-definite covariance matrix during optimization, it is factored into a rotation matrix $\mathbf{R}_i \in \mathbb{R}^{3\times3}$ and scaling vector $\mathbf{s}_i \in \mathbb{R}^3$:

$$\mathbf{\Sigma}_i = \mathbf{R}_i \mathrm{Exp}(\mathbf{s}_i) \mathrm{Exp}({\mathbf{s}_i}^T) {\mathbf{R}_i}^T \tag{1}$$

with the exponential activation preventing negative values while retaining differentiability. In practice, $\mathbf{R}_i$ is inferred from a unit-length quaternion $\mathbf{q}_i \in \mathbb{R}^4$ that provides better convergence behavior

The initial position $\mathbf{x}_i$ of the Gaussians is provided by a 3D point cloud obtained with a SfM (structure-from-motion) algorithm or randomly initialized. As the optimization proceeds, the Gaussians are periodically cloned, split, and pruned to achieve a suitable trade-off between rendering quality and computational resources.

Additionally, Kerbl *et al*. demonstrate how the many continuous Gaussian radiance distributions can be efficiently rendered on graphics hardware. Given a target camera view transformation $\mathbf{V}$ and projection matrix $\mathbf{K}$, each $\mathcal{G}_i$ is reduced to a Gaussian distribution in screen space with projected mean $\mathbf{u}_i = \mathbf{KV}\mathbf{x}_i \in \mathbb{R}^2$ and 2D covariance defined by the Jacobian $\mathbf{J}$ of $\mathbf{K}$ as

$$\mathbf{\Sigma}'_i = \mathbf{JV}\mathbf{\Sigma}_i\mathbf{V}^T\mathbf{J}^T \tag{2}$$

The 2D Gaussians are rasterized using elliptical weighted average splatting [73].

## 3.2. Forward-warping Deformation Field

We model dynamics by augmenting Kerbl *et al*.'s representation with a time-conditioned deformation field. For volume-rendered representations such as NeRFs, deformation is commonly modeled as a backward warp from 4D spatio-temporal coordinates into a canonical 3D space [35, 37, 40, 51]. This is similar to a query search based on 3D position $\mathbf{x}$ and time $t$. However, since 3D Gaussians represent the scene as an explicit set of primitives, we model the deformation as a forward warp instead. That is, the set of Gaussians $\{\mathcal{G}_i\}$ resides in the canonical space and a forward-warping field $\mathbf{\Phi}(\cdot)$ outputs the deformed set $\{\mathcal{G}^t_i\}$ with corresponding features $\{(\mathbf{x}^t_i, \mathbf{\Sigma}^t_i, \sigma^t_i, \mathbf{c}^t_i)\}$, for each time step $t$. Thus, instead of a query search, the field explicitly represents changes in scene attributes over time.

Among these attributes, we model rigid motion by deforming the position $\mathbf{x}_i$ and rotation $\mathbf{q}_i$. To allow the

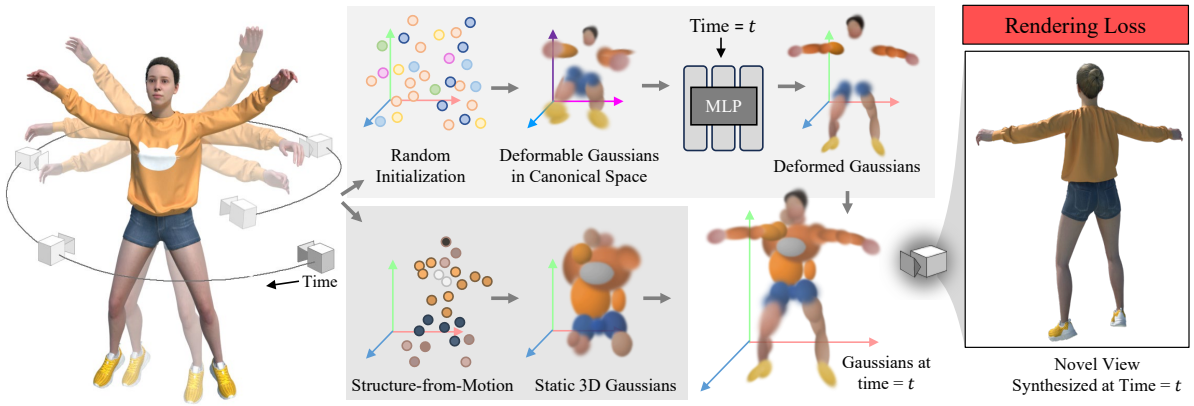


Figure 2. **An overview of our dynamic scene representation.** At each time frame $t$, our method reconstructs the scene as a combination of static and deformable anisotropic 3D Gaussians. The features of the deformable Gaussians are optimized in a canonical space and warped into frame $t$ using a deformation field. The static Gaussians are optimized in world space.

Gaussians to stretch and squeeze for capturing non-rigid transformations, we additionally deform $\mathbf{s}_i$. As a result, $\mathbf{\Phi} : \mathbb{R}^3 \times \mathbb{R}^4 \times \mathbb{R}^3 \to \mathbb{R}^3 \times \mathbb{R}^4 \times \mathbb{R}^3$. In comparison, the work of Luiten *et al*. [33] only models rigid deformations. This relies on a dense multiview capture setting to allow accurate initialization of the Gaussian points from SfM, which makes their reconstruction problem easier than the monocular setting. While Katsuma *et al*. [19] extend Luiten *et al*.'s approach to monocular settings, they freeze scaling during deformation which hurts reconstruction quality (Fig. 8).

Our deformation field models the $\delta$-change in the canonical space configuration of each attribute at time $t$:

$$\mathbf{\Phi}(\mathbf{x}_i, \mathbf{q}_i, \mathbf{s}_i, t) = (\delta\mathbf{x}_i^t, \delta\mathbf{q}_i^t, \delta\mathbf{s}_i^t) \tag{3}$$

Then, the deformed attributes at time $t$ are given as,

$$\mathbf{x}_i^t = \mathbf{x}_i + \delta\mathbf{x}_i^t \tag{4}$$

$$\mathrm{Exp}(\mathbf{s}_i^t) = \mathrm{Exp}(\mathbf{s}_i + \delta\mathbf{s}_i^t) \tag{5}$$

$$\mathbf{q}_i^t = \mathbf{q}_i \cdot \delta\mathbf{q}_i^t \tag{6}$$

Note that the shape deformation $\delta\mathbf{s}_i^t$ can alternatively be a post-exponentiation delta: $\mathrm{Exp}(\mathbf{s}_i^t) = \mathrm{Exp}(\mathbf{s}_i) + \delta\mathbf{s}_i^t$. However, pre-exponentiation is a log-linear estimation problem which is simpler than an exponential one, and allows the optimization to handle negative changes. Practically, pre-exponentiation improves reconstruction quality (Table 2).

We multiply the rotation deformation $\delta\mathbf{q}_i^t$ with the canonical space value as combining two rotation operations mathematically corresponds to quaternion multiplication, rather than addition. To stabilize training, we normalize the quaternion after deformation. While addition is faster and works via the half-angle rotation, the reconstruction had more artifacts at novel viewpoints and timesteps as the resulting quaternion is not bound geometrically.

**Should we deform opacity and color?** In the presence of atmospheric effects, or when a scene entity changes color across time, the opacity $\sigma_i$ and appearance $\mathbf{c}_i$ of the canonical Gaussians should also deform. However, we found that deforming these attributes caused severe overfitting on training sets, and introduced more artifacts than were fixed in novel views. Thus, we choose not to deform $\sigma_i$ and $\mathbf{c}_i$ over time and optimize them in the canonical space only.

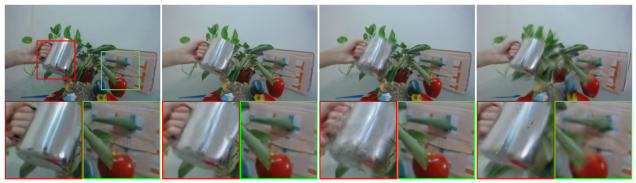


Figure 3. **Separate deformable and static regions improves quality in dynamic regions.** *Left to right:* Ground truth, our method, deformable Gaussians with no static component, and the results of Wu et al.'s 4D Gaussians method [57].

### 3.3. Static Component for Gaussian Splatting

In scenes with many static regions, we observe that the forward-warping deformation field sometimes struggles to represent motion adequately. Even though static scene regions do not need to change with time, a fully-deformable model will still spend capacity to describe tiny deformations in irrelevant regions due to noise in the camera pose or on the sensor. This issue is compounded by the fact that deformable Gaussians can easily densify to represent noise, dragging down efficiency. Therefore, if these regions can be ignored, then the network can focus its representational power and increase overall reconstruction quality.

To achieve this, we introduce a set of $k$ static points $\{\mathbf{x}_j \in \mathbb{R}^3, j = 1, ..., k\}$ along with Gaussian features $\{\mathcal{G}_j\}$ which resides alongside the deformable set $\{\mathcal{G}_i\}$ (in a slight abuse of notation, we use the indices $i$ and $j$ to distinguish elements of the two sets). To render the scene at time $t$, we compute the deformed set $\{\mathcal{G}_i^t\}$ and concatenate it to $\{\mathcal{G}_j\}$.

GT Full Static Deformable

Figure 4. Visualizing the static and deformable 3D Gaussians optimized by our method.

The combined set is treated as a static world representation and rendered together. During optimization, the two sets are densified and pruned separately.

**Inductive Bias-Aware Initialization:** Kerbl *et al*. initialize their Gaussian primitives either with a pre-computed 3D point cloud from an SfM algorithm, or by uniformly sampling points within the scene bounding box. In our case, when initializing the 3D points $\{\mathbf{x}_i\}$ and $\{\mathbf{x}_j\}$ in the separate static and deformable sets, we found that if both are initialized with the sparse SfM point cloud, or both with uniform samples, the two sets fight each other over scene occupancy, leading to worse reconstruction than the all-deformable case. To address this issue, we benefit from knowing that the feature-matching nature of SfM means the pre-computed point cloud only captures static scene regions. If the static set $\{\mathcal{G}_j\}$ is initialized with SfM, it can quickly converge to its optimal state. On the other hand, initializing $\{\mathbf{G}_i\}$ with SfM points will take a long time, if ever, for the deformable set to converge. As such, we randomly initialize $\{\mathcal{G}_i\}$ with uniform samples, and initialize $\{\mathcal{G}_j\}$ with the SfM point cloud. As a result, the static Gaussian set is able to lighten the reconstruction burden by handling unchanging regions, so the deformable set focuses on dynamic parts (Fig. 4).

### 3.4. Implementation Details

**Positional & Temporal Encoding:** We facilitate high-frequency deformation fields through PE encoding both the position $\mathbf{x}$ and time $t$ inputs by $\gamma$, where, for example, $L_{\mathbf{x}}$ is the respective encoding base for $\mathbf{x}$:

$$\begin{aligned}\gamma(\mathbf{x}) = (&\sin(2^0\mathbf{x}), \cos(2^0\mathbf{x}), \sin(2^1\mathbf{x}),\\ &\cos(2^1\mathbf{x}), \ldots, \sin(2^{L_{\mathbf{x}}-1}\mathbf{x}), \cos(2^{L_{\mathbf{x}}-1}\mathbf{x}))\ .\end{aligned} \tag{7}$$

$L_{\mathbf{x}} = 10, L_t = 10$ for both synthetic and real-world scenes.

**Network Architecture:** Our forward-warping DF architecture is inspired by Fang *et al*. [10]'s MLP, with 6 depth and 256 width. We use both a time $t$ embedding vector space and a Gaussian position $\mathbf{x}$ embedding vector space.

**Optimization:** We use Adam with $\beta = (0.9, 0.999)$ and eps $= 1e^{-15}$. The learning rate for the DF is 0.001 for all datasets, with exponential scheduling that shrinks to $0.001\times$ the original learning rate until 30 K iterations. We densify both static and deformable Gaussian sets until 20 K iterations, and keep optimizing both the Gaussians and the network until 40 K iterations.

**Objective:** We optimize using self-supervised image-based reconstruction loss only given prediction $I$ and groundtruth $I_{gt}$. In early optimization until 20 K iterations, we use an L2 loss; then, we switch to an L1 loss. This helps to increase reconstruction sharpness late in the optimization while allowing gross errors to be minimized quickly. We weighted sum L1/L2 loss with a SSIM loss and weight $\lambda_{ssim}$ to get the final objective at training step $itr$:

$$L(I, I_{gt}) = \begin{cases} (1. - \lambda_{ssim})L2(I, I_{gt}) & \\ +\lambda_{ssim}(1. - SSIM(I, I_{gt})) & \text{if } itr \leq \text{2e+4} \\ (1. - \lambda_{ssim})L1(I, I_{gt}) & \\ +\lambda_{ssim}(1. - SSIM(I, I_{gt})) & \text{else} \end{cases} \tag{8}$$

## 4. Experiments

**Metrics:** We measure novel view synthesis performance using PSNR, SSIM [54], MS-SSIM [55] and LPIPS [70]. Metric subsets are reported following each dataset's convention. All running times are on one NVIDIA 3090 GPU.

**Datasets:** We evaluate all methods on the synthetic **D-NeRF** [40] and real-world **NeRF-DS** [61], **HyperNeRF** [38] datasets. The former consists of $800\times800$ exocentric 360° views of 8 dynamic objects with large motion and realistic materials. To accommodate all baselines, we train and render all methods at half resolution ($400\times400$) with a white background. The real-world NeRF-DS dataset has monocular dynamic sequences with specular objects captured with a handheld camera. We also evaluate on the HyperNeRF dataset that contains general dynamic scenes captured by monocular cameras. Some prior methods do not report LPIPS on HyperNeRF; as such, we exclude it.

### 4.1. Results

**Deformation Modeling:** We compare with NDVG [15] and TiNeuVox [10] on both synthetic and real-world datasets, and with Nerfies [37], HyperNeRF [38] and NeRF-DS [61] on real-world datasets (Tab. 1, Fig. 5, Fig. 6). Our method achieves high reconstruction quality on both synthetic and real-world datasets, while utilizing the 3D Gaussian's rasterization pipeline to achieve real-time rendering speeds. At the same time, optimizing our method only takes 10–20 minutes on D-NeRF and NeRF-DS, which is considerably faster than the MLP-based NeRF representations.

**Efficiency:** Many recent methods have proposed auxiliary structures to accelerate the training and rendering of

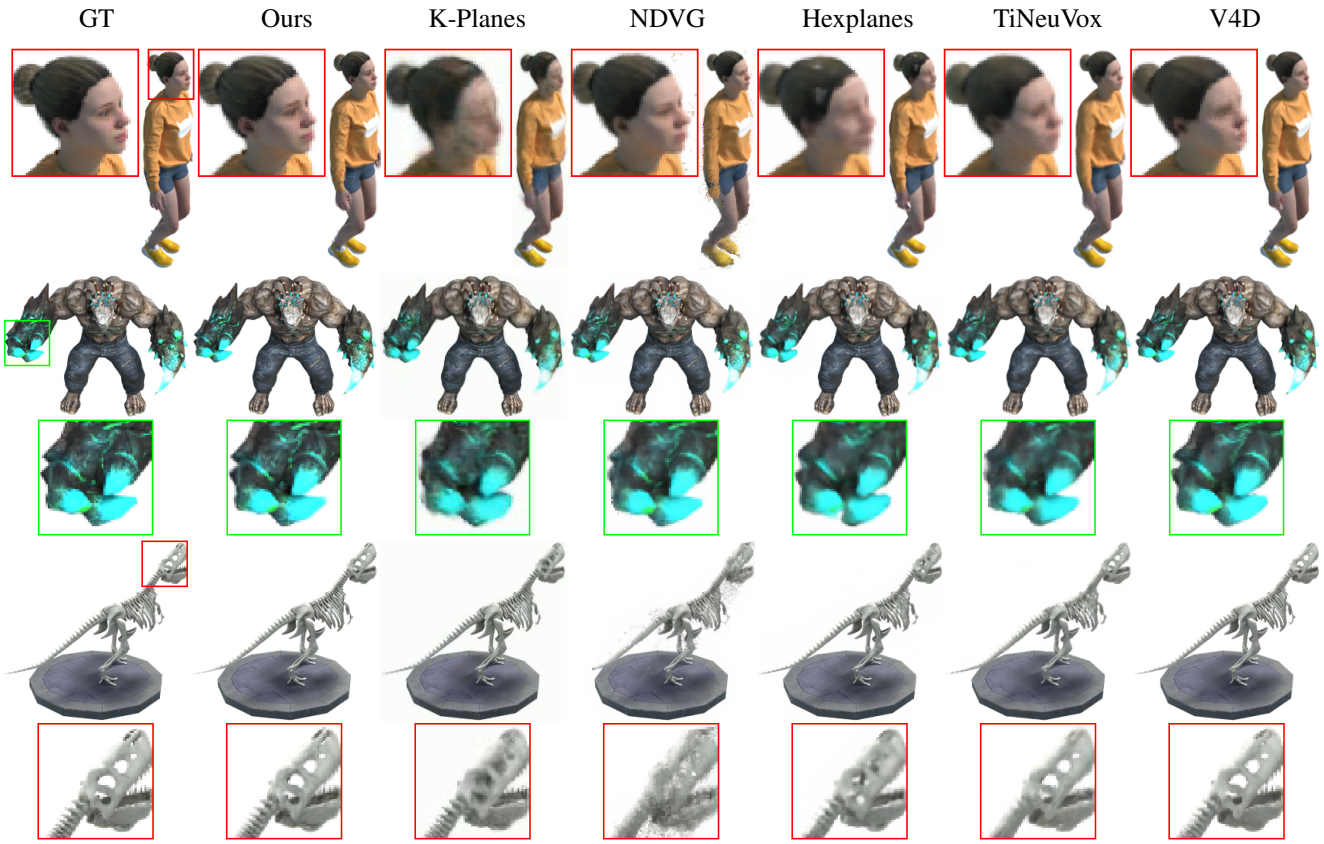


Figure 5. **Quantitative comparison of GauFRe and the baseline methods for test views from DNeRF [40] (400×400) dataset.** All methods reproduce the rough geometry, but sufficient sampling is necessary to reproduce the fine detail. Our approach can efficiently spread Gaussians to both static and dynamic regions to maximize quality, producing the sharpest image of all compared methods.

volume-rendered dynamic NeRFs. Among these, we compare with the voxel-grid variants of NDVG [15] and TiNeuVox [10], and with the plane-based representations of K-Planes [11] and HexPlane [3]. Compared to the MLP-based Nerfies [37] and HyperNeRF [38], these representations trade image quality for efficient rendering. Gan *et al.*'s V4D [13] seeks to maintain both image quality and rendering speed at the cost of slow training (Fig. 5, Fig. 6). Our method, on the other hand, achieves high reconstruction quality alongside efficient training and rendering (Tab. 1).

**Ablations:** We ablate the alternatives discussed in Sec. 3.2 for parameterizing the deformations in (Tab. 2). We observe that the "Fix Scale", "Deform Opacity", and "Deform SH" ranked lowest. "Fix Scale" corresponds to fixing $\mathbf{s}_i$, that is, not morphing $\mathcal{G}_i$ with time. Understandably, as most real-world scenes are not strictly rigid this limits the Gaussian's representational power. "Deform Opacity" and "Deform SH" corresponds to allowing the transparency and appearance to change. Despite increased representational power, these variants strongly overfit the training views leading to lower reconstruction quality. "Quaternion Addition" models the deformation on $\mathbf{q}_i$ by addition instead of the geometrically-correct multiplication. Finally, "Scale Post-Exponentiate" confirms that pre-activation is marginally better than post-activation.

The variants "No Static Gaussians" and "No IB Init" evaluate the effect of the static separation. The former removes the static primitive set $\{\mathcal{G}_j\}$, allowing all Gaussians to be deformable; the latter replaces our proposed inductive bias-aware initialization with SfM initialization (the default setting for real-world scenes). Both lower performance. Moreover, "No IB Init" has worse performance than "No Static Gaussians" indicating the necessity of a sound initialization strategy for the static component. In addition, removing L2/L1 loss transit marginally hurt GauFRe performance as shown in "No LR Transit".

**Comparison with Gaussian-based Works:** We compare with D3G [62], 4DGS [57] and EffG [19] on real-world datasets in Tab. 3 and Fig. 8, Fig. 7. Yang *et al.*'s D3G [62] is most similar to our approach as it used a network-based deformation field to represent motion. Despite using a smaller network, however, our method achieves comparable results thanks to the additional static component. In addition, our method trains almost 4×/2× faster and renders

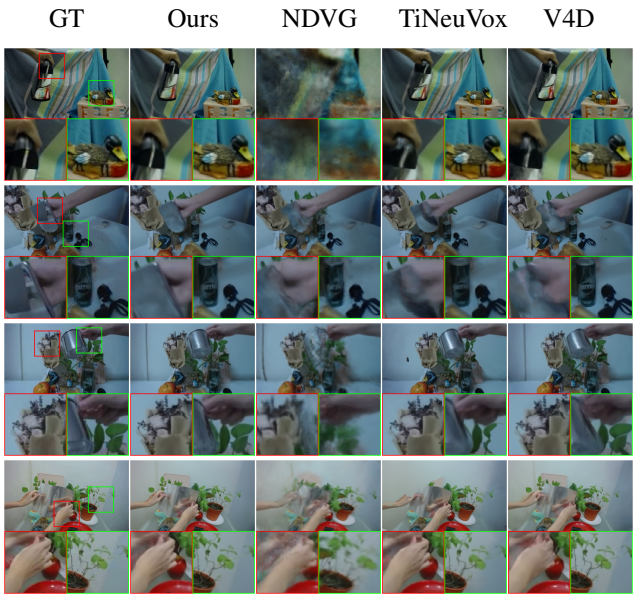


Figure 6. **Qualitative results on the NeRF-DS [61] monocular video dataset.** Compared to the baseline methods, our approach reproduces finer details on dynamic objects such as hands, and shows overall high quality in static regions.

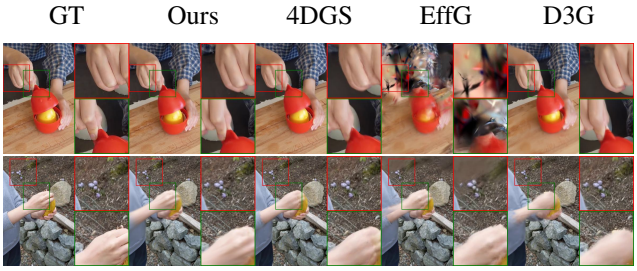


Figure 7. **Qualitative results on the HyperNeRF dataset for dynamic Gaussian approaches.** Our method achieves comparable if not better results to 4DGS [57], D3G [62] and EffG [19].

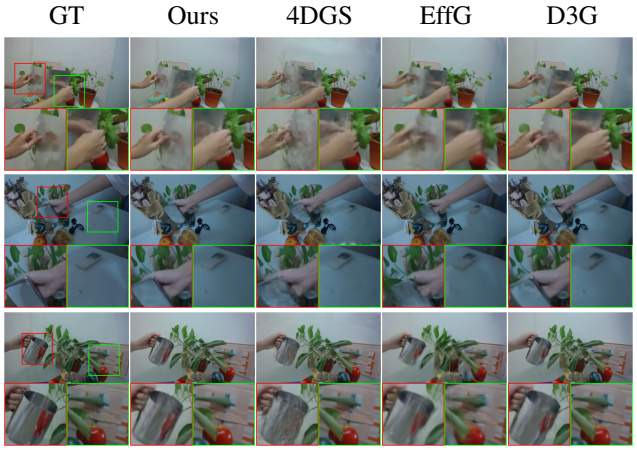


Figure 8. **Qualitative results on the NeRF-DS dataset for dynamic Gaussian approaches.** Our method achieves comparable results to D3G [62], while training and rendering much faster.

| D-NeRF [40] (synthetic) | | | | | |
|---|---|---|---|---|---|
| | PSNR↑ | SSIM↑ | LPIPS↓ | Optim. ↓ | Render↓ |
| NDVG [15] | 30.5 | 0.97 | 0.054 | 25mins | $>$ 1s |
| TiNeuVox [10] | 32.9 | 0.97 | 0.041 | 20mins | $>$ 1s |
| K-planes [11] | 29.2 | 0.96 | 0.060 | 60mins | $>$ 1s |
| Hexplane [3] | 31.0 | 0.97 | 0.039 | 15mins | 0.5s |
| V4D [13] | 33.4 | 0.98 | 0.027 | 6hours | 0.5s |
| Ours | 34.5 | 0.98 | 0.023 | 13mins | 0.01s |
| NeRF-DS [61] (real) | | | | | |
| | PSNR↑ | MS-SSIM↑ | LPIPS↓ | Optim. ↓ | Render↓ |
| Nerfies [37] | 20.1 | 0.71 | 0.349 | ∼hours | $>$ 1s |
| HyperNeRF [38] | 23.0 | 0.85 | 0.181 | ∼hours | $>$ 1s |
| NeRF-DS [61] | 23.7 | 0.89 | 0.143 | ∼hours | $>$ 1s |
| NDVG [15] | 19.1 | 0.58 | 0.417 | 1hours | $>$ 1s |
| TiNeuVox [10] | 21.7 | 0.82 | 0.219 | 30mins | $>$ 1s |
| V4D [13] | 23.5 | 0.88 | 0.142 | 7hours | $>$ 1s |
| Ours | 23.9 | 0.89 | 0.148 | 20mins | 0.01s |
| HyperNeRF [38] (real) | | | | | |
| | PSNR↑ | MS-SSIM↑ | Optim. ↓ | Render↓ | |
| Nerfies [37] | 22.2 | 0.80 | ∼hours | $>$ 1s | |
| HyperNeRF [38] | 22.3 | 0.81 | ∼hours | $>$ 1s | |
| NDVG [15] | 23.3 | 0.82 | 35mins | $>$ 1s | |
| TiNeuVox [10] | 24.3 | 0.84 | 30mins | $>$ 1s | |
| V4D [13] | 24.8 | 0.83 | ∼hours | $>$ 1s | |
| Ours | 24.1 | 0.83 | 1.5hours | 0.06s | |

Table 1. **Comparison with Deformation-based methods on the dataset from D-NeRF [40] (400×400), NeRF-DS [61] and HyperNeRF [38].** GauFRe demonstrates top image rendering quality, is the fastest to train, and enables real-time rendering.

| NeRF-DS [61] | | | |
|---|---|---|---|
| | PSNR↑ | MS-SSIM↑ | LPIPS↓ |
| Fix Scale | 20.0 | 0.69 | 0.300 |
| Deform Opacity | 21.9 | 0.79 | 0.230 |
| Deform SH | 22.1 | 0.80 | 0.210 |
| Quaternion Addition | 23.1 | 0.86 | 0.165 |
| No IB Init | 23.2 | 0.86 | 0.166 |
| No Static Gaussians | 23.5 | 0.88 | 0.154 |
| Scale Post-exponentiate | 23.8 | 0.89 | 0.155 |
| No LR Transit | 23.8 | 0.88 | 0.148 |
| Full | 23.9 | 0.89 | 0.148 |

Table 2. **Model ablations using the NeRF-DS dataset**, ordered by increasing PSNR. Our full model maximizes all three metrics.

3×/2× faster. Wu *et al.*'s 4DGS [57] attempts to accelerate reconstruction by using HexPlane [3] to model motion. Compared to plain MLP, HexPlane is more prune to overfitting. Thus, their method lags in reconstruction quality and trains slower on challenging scenario as in NeRF-DS. While more efficient to render than D3G, it is 2× slower

| D-NeRF [40] (synthetic) | | | | | |
|---|---|---|---|---|---|
| | PSNR↑ | MS-SSIM↑ | LPIPS↓ | Optim. ↓ | FPS↑ |
| RTGS [63] | 34.1 | 0.98 | 0.02 | - | - |
| 4DGS [57] | 33.3 | 0.98 | 0.03 | - | - |
| Ours | 34.5 | 0.98 | 0.02 | 13mins | 112 |
| NeRF-DS [61] (real) | | | | | |
| | PSNR↑ | MS-SSIM↑ | LPIPS↓ | Optim. ↓ | FPS↑ |
| D3G [62] | 23.7 | 0.89 | 0.132 | 72mins | 32 |
| 4DGS [57] | 22.2 | 0.82 | 0.202 | 98mins | 54 |
| EffG [19] | 21.7 | 0.81 | 0.214 | 8mins | 250 |
| Ours | 23.9 | 0.89 | 0.148 | 20mins | 96 |
| HyperNeRF [38] (real) | | | | | |
| | PSNR↑ | MS-SSIM↑ | LPIPS↓ | Optim. ↓ | FPS↑ |
| D3G [62] | 21.4 | 0.68 | - | 2hours | 8 |
| 4DGS [57] | 24.3 | 0.82 | - | 3hours | 20 |
| EffG [19] | 21.2 | 0.74 | - | 30mins | 96 |
| Ours | 24.1 | 0.83 | - | 1.5hours | 15 |

Table 3. **Comparison with GS-based methods on D-NeRF [40] (400×400), NeRF-DS [61] and HyperNeRF [38].** GauFRe achieves compelling rendering quality and state-of-the-art training and efficiency among network-based Dynamic GS representations.

| NeRF-DS [61] (real) | | | | | |
|---|---|---|---|---|---|
| | PSNR↑ | MS-SSIM↑ | LPIPS↓ | Optim. ↓ | FPS↑ |
| $D^2$NeRF [58] | 22.7 | 0.85 | 0.189 | ∼hours | <1 |
| Ours | 23.9 | 0.89 | 0.148 | 20mins | 96 |
| HyperNeRF [38] (real) | | | | | |
| | PSNR↑ | MS-SSIM↑ | LPIPS↓ | Optim. ↓ | FPS↑ |
| NeuralDiff [50] | 19.5 | 0.68 | - | ∼hours | <1 |
| $D^2$NeRF [58] | 22.1 | 0.80 | - | ∼hours | <1 |
| Ours | 24.1 | 0.83 | - | 1.5hours | 15 |

Table 4. **Comparison with Decomposition-aware methods on NeRF-DS [61] and HyperNeRF [38].** GauFRe shows better quality while being faster than previous decomposition-aware NeRFs.

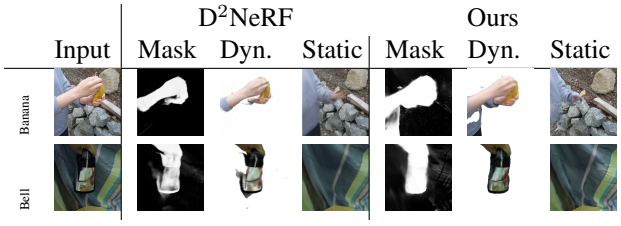


Table 5. **Scene decoupling (qualitative) on (top) HyperNeRF [38], and on (bottom) NeRF-DS [61] dataset.** GauFRe's static/dynamic decomposition is comparable to previous state-of-the-art decomposition-aware method.

than our method. Finally, Network-Free Gaussian Splatting (EffG) [19] represents a group of works that assume that the motion of Gaussian primitives can be approximated using pre-defined functions like polynomials. This method optimizes function parameters to fit each scene, obviating the use of neural networks and, thus, maximizing efficiency. However, this approach suffers on reconstruction quality in the underconstrained monocular data setting.

We also compare with GS methods that report quality metrics on D-NeRF [40] at 400×400: RTGS [63] and 4DGS [57], in Tab. 3. We achieve higher quality than both.

**Comparing to Static/Dynamic Decomposition Methods:** There have been attempts to decompose scenes into static and dynamic parts during reconstruction with NeRFs. We compare with NeuralDiff [50] and $D^2$NeRF [58] on real-world datasets quantitatively (Tab. 4) and qualitatively visualizing dynamic masks and static scene renderings following $D^2$NeRF's convention (Tab. 5) to validate GauFRe's static component design. Both NeuralDiff and $D^2$NeRF produce worse novel view rendering while being significantly slower in both training and rendering; and GauFRe learns separation aligning with underlying scenes.

## 5. Conclusion

We propose GauFRe, an efficient monocular dynamic scene reconstruction algorithm that achieves state-of-the-art rendering quality and efficiency using deformable 3D Gaussians. GauFRe keeps a Gaussian primitive set residing in a canonical space, and a carefully parameterized per-attribute DF to estimate the temporal primitive conditioning on a timestep. Additionally, we introduce a GS-specific static component supported by an inductive-bias-aware initialization paradigm, which guides the DF to focus on moving regions of the scene thus improves GauFRe rendering with complex real-world motion. The whole pipeline is optimized end-to-end with self-supervised image-based rendering loss without extra supervision. Experiments show that GauFRe achieves *sweet spot* performance: competitive qualitative/quantitative results and efficiency compared to previous state-of-the-art methods, enabling 96 FPS real-time rendering of real-world scenes on one RTX 3090 GPU.

**Limitations.** GauFRe makes progress on representing dynamic scenes for fast rendering, but it still suffers from issues shared by Gaussian-based methods. Gaussian primitive systems are prone to overfitting: they are powerful enough to recover training views while totally failing to reconstruct the scene, especially when observation is sparse and motion is complex. Dynamic GS may struggles with large motions, for which the iterative optimization struggles to correspond primitives; or for thin structures, which are hard to represent accurately with Gaussians. Concerning densification, one way for dynamic GS methods is to use the original densification policy proposed in 3D-GS [20], which sometimes does not cope well with motion.

## References


[1] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. 2020. 2

[2] Benjamin Attal, Jia-Bin Huang, Christian Richardt, Michael Zollhoefer, Johannes Kopf, Matthew O'Toole, and Changil Kim. HyperReel: High-fidelity 6-DoF video with ray-conditioned sampling. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3

[3] Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 130–141, 2023. 2, 3, 6, 7

[4] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision*, pages 333–350. Springer, 2022. 2, 3

[5] Guikun Chen and Wenguan Wang. A survey on 3d gaussian splatting, 2024. 2

[6] Devikalyan Das, Christopher Wewer, Raza Yunus, Eddy Ilg, and Jan Eric Lenssen. Neural parametric gaussians for monocular non-rigid object reconstruction. *arXiv preprint arXiv:2312.01196*, 2023. 3

[7] Gang Zeng Diwen Wan, Ruijie Lu. Superpoint gaussian splatting for real-time high-fidelity monocular dynamic scene reconstruction. In *Forty-first International Conference on Machine Learning*, 2024. 3

[8] Yuanxing Duan, Fangyin Wei, Qiyu Dai, Yuhang He, Wenzheng Chen, and Baoquan Chen. 4d gaussian splatting: Towards efficient novel view synthesis for dynamic scenes, 2024. 3

[9] Bardienus Pieter Duisterhof, Zhao Mandi, Yunchao Yao, Jia-Wei Liu, Mike Zheng Shou, Shuran Song, and Jeffrey Ichnowski. Md-splatting: Learning metric deformation from 4d gaussians in highly deformable scenes. *ArXiv*, abs/2312.00583, 2023. 3

[10] Jiemin Fang, Taoran Yi, Xinggang Wang, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Matthias Nießner, and Qi Tian. Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia 2022 Conference Papers*, 2022. 1, 2, 5, 6, 7

[11] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12479–12488, 2023. 2, 3, 6, 7

[12] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, 2022. 2

[13] Wanshui Gan, Hongbin Xu, Yi Huang, Shifeng Chen, and Naoto Yokoya. V4d: Voxel for 4d novel view synthesis. *IEEE Transactions on Visualization and Computer Graphics*, 2023. 6, 7

[14] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 3

[15] Xiang Guo, Guanying Chen, Yuchao Dai, Xiaoqing Ye, Jiadai Sun, Xiao Tan, and Errui Ding. Neural deformable voxel grid for fast optimization of dynamic view synthesis. In *Proceedings of the Asian Conference on Computer Vision*, pages 3757–3775, 2022. 5, 6, 7

[16] Zhiyang Guo, Wengang Zhou, Li Li, Min Wang, and Houqiang Li. Motion-aware 3d gaussian splatting for efficient dynamic scene reconstruction, 2024. 3

[17] Hankyu Jang and Daeyoung Kim. D-tensorf: Tensorial radiance fields for dynamic scenes. *arXiv preprint arXiv:2212.02375*, 2022. 3

[18] Animesh Karnewar, Tobias Ritschel, Oliver Wang, and Niloy Mitra. Relu fields: The little non-linearity that could. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–9, 2022. 2

[19] Kai Katsumata, Duc Minh Vo, and Hideki Nakayama. An efficient 3d gaussian representation for monocular/multi-view dynamic scenes, 2023. 3, 4, 6, 7, 8

[20] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. 2, 3, 8

[21] Numair Khan, Min H. Kim, and James Tompkin. Differentiable diffusion for dense depth estimation from multi-view images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 2

[22] Georgios Kopanas, Julien Philip, Thomas Leimkühler, and George Drettakis. Point-based neural rendering with per-view optimization. *Computer Graphics Forum (Proceedings of the Eurographics Symposium on Rendering)*, 40(4), June 2021. 2

[23] Agelos Kratimenos, Jiahui Lei, and Kostas Daniilidis. Dynmf: Neural motion factorization for real-time dynamic view synthesis with 3d gaussian splatting. *arXiV*, 2023. 3

[24] Christoph Lassner and Michael Zollh"ofer. *arXiv:2004.07484*, 2020. 2

[25] Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. Spacetime gaussian feature splatting for real-time dynamic view synthesis. *arXiv preprint arXiv:2312.16812*, 2023. 3

[26] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 3

[27] Zhengqi Li, Qianqian Wang, Forrester Cole, Richard Tucker, and Noah Snavely. Dynibar: Neural dynamic image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3

[28] Yiqing Liang, Eliot Laidlaw, Alexander Meyerowitz, Srinath Sridhar, and James Tompkin. Semantic attention flow fields for monocular dynamic scene decomposition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 21797–21806, October 2023. 3

[29] Youtian Lin, Zuozhuo Dai, Siyu Zhu, and Yao Yao. Gaussian-flow: 4d reconstruction with dynamic 3d gaussian particle. *arXiv:2312.03431*, 2023. 3

[30] Qingming Liu, Yuan Liu, Jiepeng Wang, Xianqiang Lv, Peng Wang, Wenping Wang, and Junhui Hou. Modgs: Dynamic gaussian splatting from causually-captured monocular videos, 2024. 3

[31] Yu-Lun Liu, Chen Gao, Andreas Meuleman, Hung-Yu Tseng, Ayush Saraf, Changil Kim, Yung-Yu Chuang, Johannes Kopf, and Jia-Bin Huang. Robust dynamic radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 3

[32] Zhicheng Lu, Xiang Guo, Le Hui, Tianrui Chen, Ming 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*, 2024. 3

[33] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *3DV*, 2024. 2, 3, 4

[34] Youssef A Mejjati, Isa Milefchik, Aaron Gokaslan, Oliver Wang, Kwang In Kim, and James Tompkin. GaussiGAN: Controllable image synthesis with 3d gaussians from unposed silhouettes. In *British Machine Vision Conference*, 2021. 2

[35] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2, 3

[36] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022. 2, 3

[37] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. *ICCV*, 2021. 2, 3, 5, 6, 7

[38] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *ACM Trans. Graph.*, 40(6), dec 2021. 1, 2, 5, 6, 7, 8

[39] Sungheon Park, Minjung Son, Seokhwan Jang, Young Chun Ahn, Ji-Yeon Kim, and Nahyup Kang. Temporal interpolation is all you need for dynamic neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4212–4221, 2023. 3

[40] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 2, 3, 5, 6, 7, 8

[41] Helge Rhodin, Nadia Robertini, Christian Richardt, Hans-Peter Seidel, and Christian Theobalt. A versatile scene model with differentiable visibility applied to generative pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015. 2

[42] Darius Rückert, Linus Franke, and Marc Stamminger. Adop: Approximate differentiable one-pixel point rendering. *arXiv preprint arXiv:2110.06635*, 2021. 2

[43] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 2

[44] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 1

[45] Ruizhi Shao, Zerong Zheng, Hanzhang Tu, Boning Liu, Hongwen Zhang, and Yebin Liu. Tensor4d: Efficient neural 4d decomposition for high-fidelity dynamic reconstruction and rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16632–16642, 2023. 3

[46] 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. 3

[47] Carsten Stoll, Nils Hasler, Juergen Gall, Hans-Peter Seidel, and Christian Theobalt. Fast articulated motion tracking using a sums of gaussians body model. In *2011 International Conference on Computer Vision*, pages 951–958. IEEE, 2011. 2

[48] Cheng Sun, Min Sun, and H Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. 2022 ieee. In *CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5449–5459, 2021. 2

[49] Jiakai Sun, Han Jiao, Guangyuan Li, Zhanjie Zhang, Lei Zhao, and Wei Xing. 3dgstream: On-the-fly training of 3d gaussians for efficient streaming of photo-realistic free-viewpoint videos. 2024. 3

[50] Vadim Tschernezki, Diane Larlus, and Andrea Vedaldi. NeuralDiff: Segmenting 3D objects that move in egocentric videos. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2021. 8

[51] Chaoyang Wang, Lachlan Ewen MacDonald, László A. Jeni, and Simon Lucey. Flow supervision for deformable nerf. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21128–21137, June 2023. 2, 3

[52] Feng Wang, Zilong Chen, Guokang Wang, Yafei Song, and Huaping Liu. Masked space-time hash encoding for efficient dynamic scene reconstruction, 2023. 3

[53] Feng Wang, Sinan Tan, Xinghang Li, Zeyue Tian, and Huaping Liu. Mixed neural voxels for fast multi-view video synthesis. *arXiv preprint arXiv:2212.00190*, 2022. 3

[54] 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. 5

[55] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Sys-*

*tems & Computers, 2003*, volume 2, pages 1398–1402. Ieee, 2003. 5

[56] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. SynSin: End-to-end view synthesis from a single image. In *CVPR*, 2020. 2

[57] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Wang Xinggang. 4d gaussian splatting for real-time dynamic scene rendering. *arXiv preprint arXiv:2310.08528*, 2023. 1, 3, 4, 6, 7, 8

[58] Tianhao Wu, Fangcheng Zhong, Andrea Tagliasacchi, Forrester Cole, and Cengiz Oztireli. Dˆ 2nerf: Self-supervised decoupling of dynamic and static objects from a monocular video. *Advances in Neural Information Processing Systems*, 35:32653–32666, 2022. 3, 8

[59] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9421–9431, 2021. 3

[60] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5438–5448, 2022. 2

[61] Zhiwen Yan, Chen Li, and Gim Hee Lee. Nerf-ds: Neural radiance fields for dynamic specular objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8285–8295, 2023. 1, 5, 7, 8

[62] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction, 2023. 1, 3, 6, 7, 8

[63] Zeyu Yang, Hongye Yang, Zijie Pan, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. 2024. 3, 8

[64] Zeyu Yang, Hongye Yang, Zijie Pan, Xiatian Zhu, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting, 2023. 3

[65] Wang Yifan, Felice Serena, Shihao Wu, Cengiz Öztireli, and Olga Sorkine-Hornung. Differentiable surface splatting for point-based geometry processing. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH ASIA)*, 38(6), 2019. 2

[66] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 3

[67] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5752–5761, 2021. 2

[68] Heng Yu, Joel Julin, Zoltan A Milacski, Koichiro Niinuma, and Laszlo A Jeni. Cogs: Controllable gaussian splatting. *arXiv*, 2023. 3

[69] Qiang Zhang, Seung-Hwan Baek, Szymon Rusinkiewicz, and Felix Heide. Differentiable point-based radiance fields for efficient view synthesis. *arXiv preprint arXiv:2205.14330*, 2022. 2

[70] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 5

[71] Xiaoming Zhao, Alex Colburn, Fangchang Ma, Miguel Ángel Bautista, Joshua M. Susskind, and Alexander G. Schwing. Pseudo-Generalized Dynamic View Synthesis from a Video. In *ICLR*, 2024. 3

[72] Wojciech Zielonka, Timur Bagautdinov, Shunsuke Saito, Michael Zollhöfer, Justus Thies, and Javier Romero. Drivable 3d gaussian avatars. 2023. 3

[73] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. Surface splatting. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 371–378, 2001. 2, 3