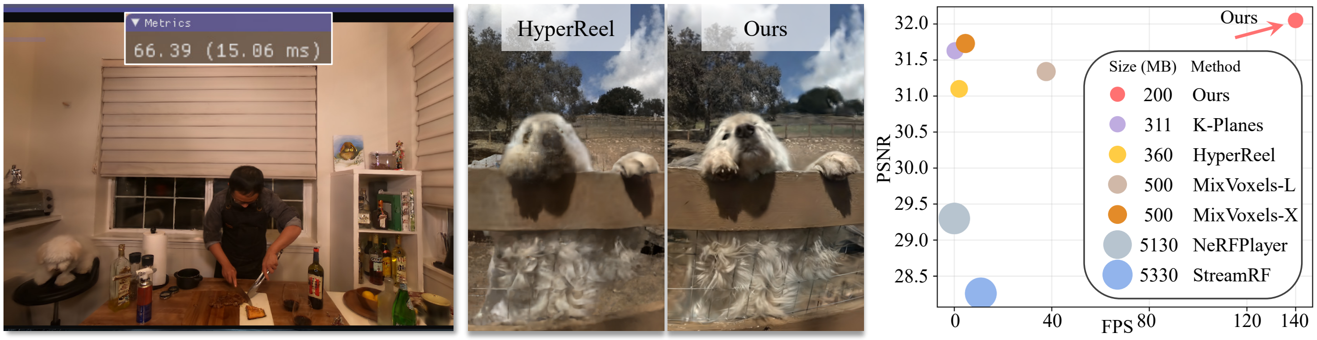
Spacetime Gaussian Feature Splatting for Real-Time Dynamic View Synthesis

Zhan Li[1,2*] Zhang Chen[1†] Zhong Li[1†] Yi Xu[1]
[1] OPPO US Research Center [2] Portland State University
lizhan@pdx.edu zhang.chen@oppo.com zhong.li@oppo.com yi.xu@oppo.com
https://oppo-us-research.github.io/SpacetimeGaussians-website/



(a) Real-Time 8K Demo at 66 FPS (b) High Quality Result (c) Comparison on Quality, Speed and Size

Figure 1. **Our dynamic scene representation achieves photorealistic quality, real-time high-resolution rendering and compact model size.** (a) Our lite-version model can render 8K 6-DoF video at 66 FPS on an Nvidia RTX 4090 GPU. (b) Example novel view rendering of a challenging scene. (c) Quantitative comparisons of rendering quality, speed and model size with prior arts on the Neural 3D Video Dataset.

Abstract

Novel view synthesis of dynamic scenes has been an intriguing yet challenging problem. Despite recent advancements, simultaneously achieving high-resolution photorealistic results, real-time rendering, and compact storage remains a formidable task. To address these challenges, we propose Spacetime Gaussian Feature Splatting as a novel dynamic scene representation, composed of three pivotal components. First, we formulate expressive Spacetime Gaussians by enhancing 3D Gaussians with temporal opacity and parametric motion/rotation. This enables Spacetime Gaussians to capture static, dynamic, as well as transient content within a scene. Second, we introduce splatted feature rendering, which replaces spherical harmonics with neural features. These features facilitate the modeling of view- and time-dependent appearance while maintaining small size. Third, we leverage the guidance of training error and coarse depth to sample new Gaussians in areas that are challenging to converge with existing pipelines. Experiments on several established real-world datasets demonstrate that our method achieves state-of-the-art rendering quality and speed, while retaining compact storage. At 8K resolution, our lite-version model can render at 60 FPS on an Nvidia RTX 4090 GPU.

1. Introduction

Photorealistic modeling of real-world dynamic scenes has been persistently pursued in computer vision and graphics. It allows users to freely explore dynamic scenes at novel viewpoints and timestamps, thus providing strong immersive experience, and can vastly benefit applications in VR/AR, broadcasting, education, *etc*.

Recent advances in novel view synthesis, especially Neural Radiance Fields (NeRF) [66], have greatly improved the convenience and fidelity of static scene modeling from casual multi-view inputs in non-lab environments. Since then, large quantities of work [6–8, 15, 18, 27, 41, 67, 82] have emerged aiming to enhance rendering quality and speed. Particularly, [18, 41] propose to use anisotropic radial basis functions as 3D representations, which are highly adaptive to scene structures and boost the modeling of de-

---

[†] Corresponding authors.
[*] Work done while Zhan was an intern at OPPO US Research Center.

tails. 3D Gaussian Splatting (3DGS) [41] further presents an efficient rasterization-based scheme for differentiable volume rendering. Instead of shooting rays from camera to the scene and sampling points along each ray, 3DGS rasterizes 3D Gaussians onto image plane via splatting, which brings about notable rendering speedup.

Despite the success on static scenes, directly applying the above methods per-frame to dynamic scenes is challenging, due to the overhead in model size and training time. State-of-the-art dynamic view synthesis methods [4, 12, 28, 48, 80, 87] adopt a holistic approach where multiple frames are represented in a single model. NeRFPlayer [80] and HyperReel [4] combine static spatial representations [15, 67] with temporal feature sharing/interpolation to improve model compactness. This strategy exploits the characteristic that adjacent frames in natural videos usually exhibit high similarity. In a similar vein, MixVoxels [87] uses time-variant latents and bridges them with spatial features by inner product. K-Planes [28] and HexPlane [12] factorize the 4D spacetime domain into multiple 2D planes for compact representation. One limitation of these methods is that their grid-like representations cannot fully adapt to the dynamics of scene structures, hindering the modeling of delicate details. Meanwhile, they struggle to produce real-time high-resolution rendering without sacrificing quality.

In this work, we present a novel representation for dynamic view synthesis. Our approach simultaneously achieves photorealistic quality, real-time high-resolution rendering and compact model size (see Fig. 1 for example results and comparisons with state-of-the-arts). At the core of our approach is Spacetime Gaussian (STG), which extends 3D Gaussian to 4D spacetime domain. We propose to equip 3D Gaussian with time-dependent opacity along with polynomially parameterized motion and rotation. As a result, STGs are capable of faithfully modeling static, dynamic as well as transient (*i.e*., emerging or vanishing) content in a scene.

To enhance model compactness and account for time-varying appearance, we propose splatted feature rendering. Specifically, for each Spacetime Gaussian, instead of storing spherical harmonic coefficients, we store features that encode base color, view-related information and time-related information. These features are rasterized to image space via differentiable splatting, and then go through a tiny multi-layer perceptrons (MLP) network to produce the final color. While smaller in size than spherical harmonics, these features exhibit strong expressiveness.

Additionally, we introduce guided sampling of Gaussians to improve rendering quality of complex scenes. We observe that distant areas which are sparsely covered by Gaussians at initialization tend to have blurry rendering results. To tackle this problem, we propose to sample new Gaussians in the 4D scene with the guidance of training error and coarse depth.

In summary, the contributions of our work are the following:

- We present a novel representation based on Spacetime Gaussian for high-fidelity and efficient dynamic view synthesis.
- We propose splatted feature rendering, which enhances model compactness and facilitates the modeling of time-varying appearance.
- We introduce a guided sampling approach for Gaussians to improve rendering quality at distant sparsely covered areas.
- Extensive experiments on various real-world datasets demonstrate that our method achieves state-of-the-art rendering quality and speed while keeping small model size. Our lite-version model enables 8K rendering at 60 FPS.

2. Related Work

Novel View Synthesis. Early approaches leverage image-based rendering techniques with proxy geometry/depth to sample novel views from source images [11, 13, 20, 31, 34, 44, 46, 103]. Chaurasia *et al*. [14] estimate a depth map to blend pixels from source views and employ superpixels to compensate for missing depth data. Hedman *et al*. [32] utilize RGBD sensors to improve rendering quality and speed. Penner and Zhang [72] leverage volumetric voxels for continuity in synthesized views and robustness to depth uncertainty. Hedman *et al*. [33] learn the blending scheme with neural networks. Flynn *et al*. [26] combine multi-plane images with learned gradient descent. Wiles *et al*. [92] splat latent features from point cloud for novel view synthesis.

Neural Scene Representations. In recent years, neural scene representations have achieved great progress in novel view synthesis. These methods allocate neural features to structures such as volume [62, 78], texture [16, 84], or point cloud [1]. The seminal work of NeRF [66] proposes to leverage differentiable volume rendering. It does not require proxy geometry and instead uses MLPs to implicitly encode density and radiance in 3D space. Later on, numerous works emerge to boost the quality and efficiency of differentiable volume rendering. One group of methods focuses on improving the sampling strategy to reduce the number of point queries [4, 68, 73] or applies light field-based formulation [3, 25, 53, 54, 79, 81, 89]. Another group trades space for speed by incorporating explicit and localized neural representations [8, 15, 17, 27, 36, 60, 67, 75, 82, 83, 96, 101]. Among them, to improve model compactness, Instant-NGP [67] uses hash grid while TensoRF [15] utilizes tensor decomposition.

Recently, 3D Gaussian Splatting (3DGS) [41] proposes to use anisotropic 3D Gaussians as scene representation and presents an efficient differentiable rasterizer to splat these Gaussians to the image plane. Their method enables fast high-resolution rendering, while preserving great rendering quality. Similar to 3DGS, NeuRBF [18] leverages anisotropic radial basis functions for neural representation and achieves high-fidelity rendering. However, the above methods focus on static scene representation.

Dynamic Novel View Synthesis. A widely adopted setting for dynamic free-viewpoint rendering is using multi-view videos as input. Classic methods in this area include [19, 21, 39, 40, 49–51, 98, 104]. More recently, Broxton *et al.* [10] use multi-sphere image as bootstrap and then convert it to layered meshes. Bansal *et al.* [5] separate static and dynamic contents and manipulate video with deep network in screen space. Bemana *et al.* [9] learn a neural network to implicitly map view, time or light coordinates to 2D images. Attal *et al.* [2] use multi-sphere representations to handle the depth and occlusions in 360-degree videos. Lin *et al.* [57, 58] propose 3D mask volume to addresses the temporal inconsistency of disocclusions. Neural Volumes [62] uses an encoder-decoder network to encode images into a 3D volume and decode it with volume rendering. Lombardi *et al.* [63] enhance Neural Volumes by decoding a mixture of dynamic geometric primitives from latent code and skipping samples in empty space for efficient ray marching. Extending static NeRF-related representations to dynamic scenes are also being actively explored [4, 12, 28, 38, 42, 47, 48, 56, 71, 77, 80, 87, 88, 90, 91]. DyNeRF [48] combines NeRF with time-conditioned latent codes to compactly represent dynamic scenes. StreamRF [47] accelerates the training of dynamic scenes by modeling the differences of consecutive frames. NeRFPlayer [80] decomposes scene into static, new and deforming fields and proposes streaming of feature channels. MixVoxels [87] represents scene with a mixture of static and dynamic voxels to accelerate rendering. HyperReel [4] utilizes sampling prediction network to reduce sampling points and leverages keyframe-based representation. K-Planes [28], HexPlane [12] and Tensor4D [77] factorize 4D spacetime domain into 2D feature planes for compact model size.

Another line of research tackles dynamic view synthesis from monocular videos [22, 23, 29, 30, 52, 55, 61, 69, 70, 74, 85, 86, 94]. Under this setting, a single camera moves around in the dynamic scene, providing only one observed viewpoint at each timestep. To address the sparsity of supervision, priors on motion, scene flow or depth are usually introduced. In this work, we focus on the dynamic representation itself and only consider the setting of multi-view input videos.

Recently, there are several work on this topic that are concurrent to ours [37, 45, 59, 64, 93, 95, 97, 99, 100]. 4K4D [97] combines 4D point clouds with K-Planes [28] and discrete image-based rendering, and uses differentiable depth peeling to train the model. Luiten *et al.* [64] models 4D scene with a set of moving 3D Gaussians, whose positions and rotations are discretely defined at each step. Their method demonstrates appealing results for 3D tracking, but its rendering quality is less favorable due to flickering artifacts. Yang *et al.* [100] leverage 4D Gaussians and 4D spherindrical harmonics for dynamic modelling. 4D Gaussians essentially represent motion with linear model. Comparatively, our polynomial motion model is more expressive, resulting in higher rendering quality. Our method also has higher rendering speed than their work. Yang *et al.* [99] and Wu *et al.* [93] prioritize on monocular dynamic view synthesis and employ deformation fields to deform a set of canonical 3D Gaussians. For multi-view videos setting, the performance of [99] is not extensively evaluated while [93] depicts inferior rendering quality and speed than ours.

3. Preliminary: 3D Gaussian Splatting

Given images at multiple viewpoints with known camera poses, 3D Gaussian Splatting [41] (3DGS) optimizes a set of anisotropic 3D Gaussians via differentiable rasterization to represent a static 3D scene. Owing to their efficient rasterization, the optimized model can render high-fidelity novel views in real-time.

3DGS [41] associates a 3D Gaussian i with a position μ_i, covariance matrix Σ_i, opacity σ_i and spherical harmonics (SH) coefficients $\mathbf{h}_i$. The final opacity of a 3D Gaussian at any spatial point $\mathbf{x}$ is

$$\alpha_i = \sigma_i \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_i)^T \Sigma_i^{-1}(\mathbf{x}-\mu_i)\right). \tag{1}$$

Σ_i is positive semi-definite and can be decomposed into scaling matrix S_i and rotation matrix R_i:

$$\Sigma_i = R_i S_i S_i^T R_i^T, \tag{2}$$

where S_i is a diagonal matrix and is parameterized by a 3D vector $\mathbf{s}_i$, and R_i is parameterized by a quaternion q.

To render an image, 3D Gaussians are first projected to 2D image space via an approximation of the perspective transformation [105]. Specifically, the projection of a 3D Gaussian is approximated as a 2D Gaussian with center μ_i^{2D} and covariance Σ_i^{2D}. Let W, K be the viewing transformation and projection matrix, μ_i^{2D} and Σ_i^{2D} are computed as

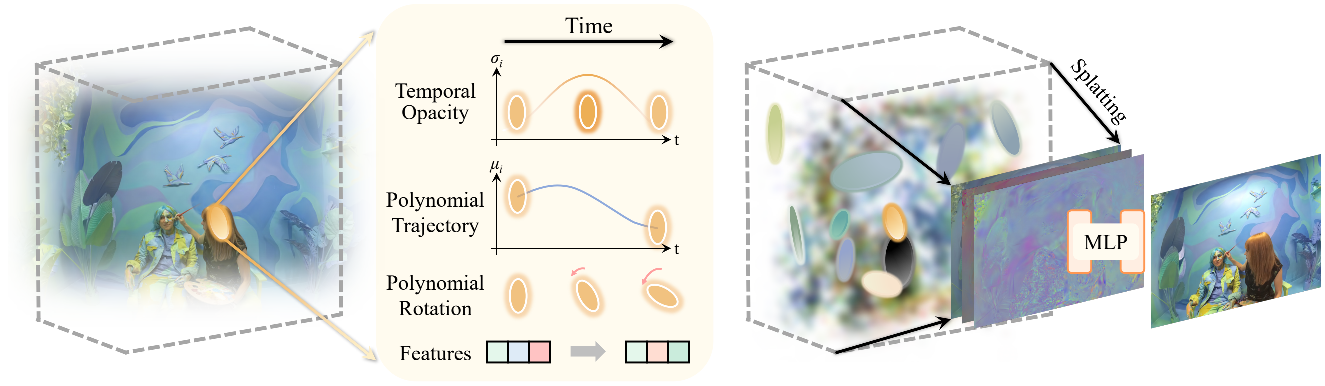
$$\mu_i^{2D} = (K((W\mu_i)/(W\mu_i)_z))_{1:2}, \tag{3}$$

$$\Sigma_i^{2D} = (JW\Sigma_i W^T J^T)_{1:2,1:2}, \tag{4}$$

where J is the Jacobian of the projective transformation.

After sorting the Gaussians in depth order, the color at a pixel is obtained by volume rendering:

$$\mathbf{I} = \sum_{i\in\mathcal{N}} \mathbf{c}_i \alpha_i^{2D} \prod_{j=1}^{i-1}(1-\alpha_j^{2D}), \tag{5}$$



(a) Spacetime Gaussians

(b) Feature Splatting and Rendering

Figure 2. **Overview of Spacetime Gaussians and splatted feature rendering.** (a) Our method leverages a set of Spacetime Gaussians (STG) to represent the dynamic scenes. On top of 3D Gaussian, each STG is further equipped with temporal opacity, polynomial motion/rotation and time-dependent features. (b) We visualize the splatted features as maps, which are converted to color image via MLP.

where α_i^{2D} is a 2D version of Eq. (1), with $\mu_i, \Sigma_i, \mathbf{x}$ replaced by $\mu_i^{2D}, \Sigma_i^{2D}, \mathbf{x}^{2D}$ (pixel coordinate). $\mathbf{c}_i$ is the RGB color after evaluating SH with view direction and coefficients $\mathbf{h}_i$.

4. Method

We propose a novel representation based on Spacetime Gaussians for modeling dynamic 3D scenes. Our approach takes multi-view videos as input and creates 6-DoF video that allows rendering at novel views. We first describe the formulation of our Spacetime Gaussian (STG) in Sec. 4.1. Then in Sec. 4.2, we present feature-based splatting for time-varying rendering. Sec. 4.3 details our optimization process and Sec. 4.4 introduces guided sampling of Gaussians.

4.1. Spacetime Gaussians

To represent 4D dynamics, we propose Spacetime Gaussians (STG) that combine 3D Gaussians with temporal components to model emerging/vanishing content as well as motion/deformation, as shown in Fig. 2 (a). Specifically, we introduce temporal radial basis function to encode temporal opacity, which can effectively model scene content that emerges or vanishes within the duration of video. Meanwhile, we utilize time-conditioned parametric functions for the position and rotation of 3D Gaussians to model the motion and deformation in the scene. For a spacetime point $(\mathbf{x}, t)$, the opacity of an STG is

$$\alpha_i(t) = \sigma_i(t) \exp\left(-\frac{1}{2}(\mathbf{x} - \mu_i(t))^T \Sigma_i(t)^{-1} (\mathbf{x} - \mu_i(t))\right), \tag{6}$$

where $\sigma_i(t)$ is temporal opacity, $\mu_i(t), \Sigma_i(t)$ are time-dependent position and covariance, and i stands for the ith STG. We detail each of the components below.

Temporal Radial Basis Function. We use a temporal radial basis function to represent the temporal opacity of an STG at any time t. Inspired by [18, 41] that use radial basis functions for approximating spatial signals, we utilize 1D Gaussian for the temporal opacity $\sigma_i(t)$:

$$\sigma_i(t) = \sigma_i^s \exp\left(-s_i^\tau |t - \mu_i^\tau|^2\right), \tag{7}$$

where μ_i^τ is temporal center, s_i^τ is temporal scaling factor, and σ_i^s is time-independent spatial opacity. μ_i^τ represents the timestamp for the STG to be most visible while s_i^τ determines its effective duration (*i.e.*, the time duration where its temporal opacity is high). We include σ_i^s to allow spatial opacity variation across STGs.

Polynomial Motion Trajectory. For each STG, we employ a time-conditioned function to model its motion. Motivated by [24, 35], we choose polynomial function:

$$\mu_i(t) = \sum_{k=0}^{n_p} b_{i,k}(t - \mu_i^\tau)^k, \tag{8}$$

where $\mu_i(t)$ denotes the spatial position of an STG at time t. $\{b_{i,k}\}_{k=0}^{n_p}, b_{i,k} \in \mathbb{R}$ are the corresponding polynomial coefficients and are optimized during training. Combining Eq. (7) and Eq. (8), complex and long motion can be represented by multiple short segments with simpler motion. In our implementation, we use $n_p = 3$ as we find it a good balance between representation capacity and model size.

Polynomial Rotation. Following [41], we use real-valued quaternion to parameterize the rotation matrix R_i in Eq. (2). Similar to motion trajectory, we adopt a polynomial func-

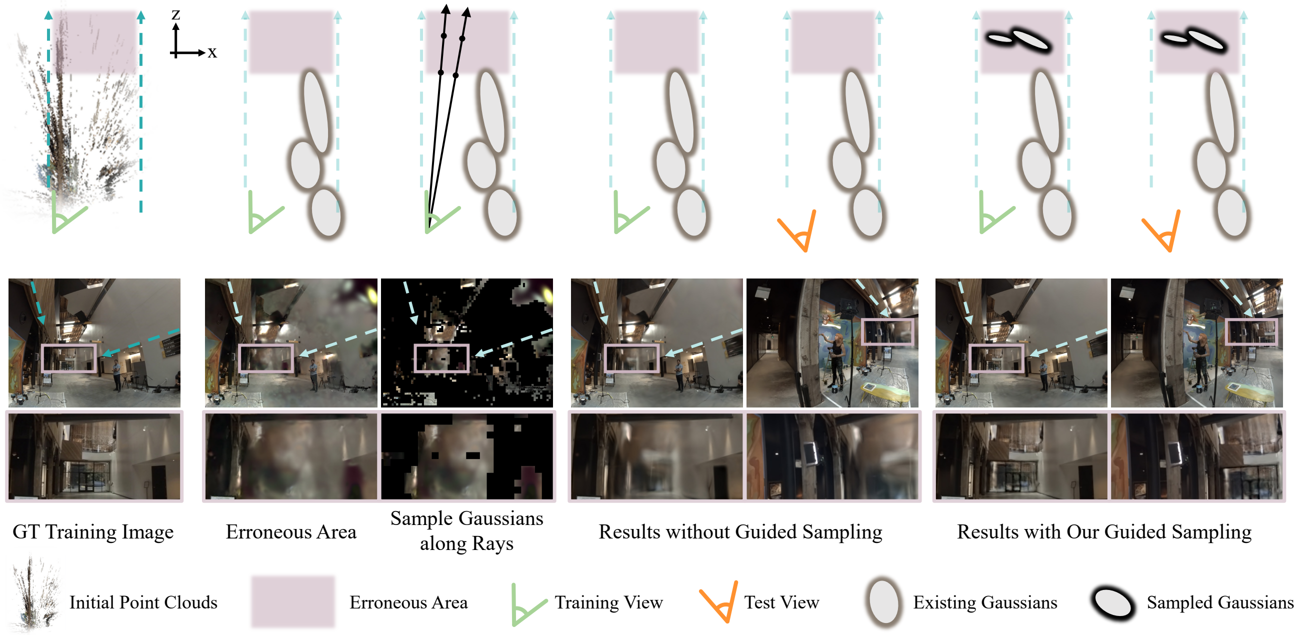


Figure 3. **Illustration of our guided sampling strategy for Gaussians.** Our strategy samples new Gaussians along rays by leveraging the guidance of training error and coarse depth.

tion to represent quaternion:

$$q_i(t) = \sum_{k=0}^{n_q} c_{i,k}(t-\mu_i^\tau)^k, \tag{9}$$

where $q_i(t)$ is the rotation (in quaternion) of an STG at time t, and $\{c_{i,k}\}_{k=0}^{n_q}, c_{i,k} \in \mathbb{R}$ are polynomial coefficients. After converting $q_i(t)$ to rotation matrix $R_i(t)$, the covariance $\Sigma_i(t)$ at time t can be obtained via Eq. (2). We set $n_q = 1$ in our experiments.

Note that we keep the scaling matrix S_i in Eq. (2) to be time-independent, since we experimentally do not observe improvement in rendering quality when applying time-conditioned function on this parameter.

4.2. Splatted Feature Rendering

To encode view- and time-dependent radiance both accurately and compactly, we store features instead of spherical harmonics coefficients (SH) in each STG. Specifically, the features $\mathbf{f}_i(t) \in \mathbb{R}^9$ of each STG consist of three parts:

$$\mathbf{f}_i(t) = \left[\mathbf{f}_i^{\text{base}}, \mathbf{f}_i^{\text{dir}}, \quad (t-\mu_i^\tau)\mathbf{f}_i^{\text{time}}\right]^T, \tag{10}$$

where $\mathbf{f}_\mathbf{i}^{\text{base}} \in \mathbb{R}^3$ contains base RGB color, and $\mathbf{f}_\mathbf{i}^{\text{dir}}, \mathbf{f}_\mathbf{i}^{\text{time}} \in \mathbb{R}^3$ encode information related to view direction and time. The feature splatting process is similar to Gaussian Splatting [41], except that the RGB color $\mathbf{c}_i$ in Eq. (5) is now replaced by features $\mathbf{f}_i(t)$. After splatting to image space, we split the splatted features at each pixel into $\mathbf{F}^{base}, \mathbf{F}^{dir}, \mathbf{F}^{time}$, whose channels correspond to the three parts in Eq. (10). The final RGB color at each pixel is obtained after going through a 2-layer MLP Φ:

$$\mathbf{I} = \mathbf{F}^{base} + \Phi(\mathbf{F}^{dir}, \mathbf{F}^{time}, \mathbf{r}), \tag{11}$$

where $\mathbf{r}$ is the view direction at the pixel and is additionally concatenated with the features as input. Fig. 2 (b) shows an illustration of the rendering process.

Compared to SH encoding, our feature-based approach requires fewer parameters for each STG (9 vs. 48 for 3-degree SH). At the same time, since the MLP network Φ is shallow and narrow, our method still achieves fast rendering speed.

To maximize rendering speed, we can also optionally drop Φ and only keep $\mathbf{F}^{base}$ during training and rendering. We refer to this configuration as our lite-version model.

4.3. Optimization

The parameters to be optimized include the MLP Φ and the parameters of each STG $(\sigma_i^s, s_i^\tau, \mu_i^\tau, \{b_{i,k}\}_{k=0}^{n_p}, \{c_{i,k}\}_{k=0}^{n_q}, \mathbf{s}_i, \mathbf{f}_i^{\text{base}}, \mathbf{f}_i^{\text{dir}}, \mathbf{f}_i^{\text{time}})$.

Following [41], we optimize these parameters through differentiable splatting and gradient-based backpropagation, and interleave with density control of Gaussians. We use rendering loss that compares rendered images with groundtruth images. The rendering loss consists of a $\mathcal{L}_1$ term and a D-SSIM term.

4.4. Guided Sampling of Gaussians

We observe that areas which have sparse Gaussians at initialization are challenging to converge to high rendering quality, especially if these areas are far away from the training cameras. Therefore, we further introduce a strategy to sample new Gaussians with the guidance of training error and coarse depth.

We sample new Gaussians along the rays of pixels that have large errors during training, as illustrated in Fig. 3. To ensure sampling effectiveness, we conduct sampling after training loss is stable. Since error maps can be noisy during training, we patch-wise aggregate training errors to prioritize on areas with substantial errors rather than outlier pixels. Then we sample a ray from the center pixel of each selected patches that have large errors. To avoid sampling in an excessively large depth range, we exploit the coarse depth map of Gaussians' centers to determine a more specific depth range. The depth map is generated during feature splatting and incurs little computational overhead. New Gaussians are then uniformly sampled within the depth range along the rays. We additionally add small noises to the centers of the newly sampled Gaussians. Among the sampled Gaussians, the unnecessary ones will have low opacity after steps of training and be pruned. For the scenes in our experiments, the above sampling process only needs to be conducted no more than 3 times.

Our guided sampling strategy is complimentary to the density control techniques in [41]. While density control gradually grows Gaussians near existing ones by splitting, our approach can sample new Gaussians at regions that have sparse or no Gaussians.

5. Implementation Details

We initialize our STGs with the structure-from-motion sparse point clouds from all available timestamps. For density control, we conduct more aggressive pruning than [41] to reduce the number of Gaussians and keep model size to be relatively small. We use Adam optimizer [43]. The training time for a 50-frame sequence is 40-60 minutes on a single NVIDIA A6000 GPU. We adapt the splatting process to support different camera models in real world datasets. See supplementary material for more implementation details.

6. Experiments

We evaluate our method on three real-world benchmarks: Neural 3D Video Dataset [48] (Sec. 6.1), Google Immersive Dataset [10] (Sec. 6.2), and Technicolor Dataset [76] (Sec. 6.3). We also conduct ablation studies on various aspects of our method (Sec. 6.4). Please refer to supplementary material and video for more results and real-time demo.

6.1. Neural 3D Video Dataset

The Neural 3D Video Dataset [48] contains six indoor multi-view video sequences captured by 18 to 21 cameras at 2704 × 2028 resolution. Following common practice, training and evaluation are conducted at half resolution, and the first camera is held out for evaluation [48]. The number of frames is 300 for each scene.

Table 1. **Quantitative comparisons on the Neural 3D Video Dataset.** "FPS" is measured at *1352 × 1014* resolution. "Size" is the total model size for 300 frames. Some methods only report part of the scenes. For fair comparison, we additionally report our results under their settings. ^{1} only includes the *Flame Salmon* scene. ^{2} excludes the *Coffee Martini* scene.

Method	PSNR↑	DSSIM$_1$↓	DSSIM$_2$↓	LPIPS↓	FPS↑	Size↓
Neural Volumes [62] ^{1}	22.80	-	0.062	0.295	-	-
LLFF [65] ^{1}	23.24	-	0.076	0.235	-	-
DyNeRF [48] ^{1}	**29.58**	-	**0.020**	0.083	0.015	**28 MB**
Ours ^{1}	29.48	0.038	0.022	**0.063**	**103**	300 MB
HexPlane [12] ^{2}	31.71	-	-	0.075	-	200 MB
Ours ^{2}	**32.74**	0.027	0.012	**0.039**	**140**	**190 MB**
StreamRF [47]	28.26	-	-	-	10.9	5310 MB
NeRFPlayer [80]	30.69	0.034	-	0.111	0.05	5130 MB
HyperReel [4]	31.10	0.036	-	0.096	2	360 MB
K-Planes [28]	31.63	-	0.018	-	0.3	311 MB
MixVoxels-L [87]	31.34	-	0.017	0.096	37.7	500 MB
MixVoxels-X [87]	31.73	-	0.015	0.064	4.6	500 MB
Ours	**32.05**	**0.026**	**0.014**	**0.044**	**140**	**200 MB**

Table 2. **Quantitative comparisons on the Google Immersive Dataset.** "Size/Fr" stands for model size per frame.

Method	PSNR↑	DSSIM$_1$↓	LPIPS↓	FPS↑	Size/Fr↓
NeRFPlayer [80]	25.8	0.076	0.196	0.12	17.1 MB
HyperReel [4]	28.8	0.063	0.193	4	**1.2 MB**
Ours	**29.2**	**0.042**	**0.081**	**99**	**1.2 MB**

We use PSNR, DSSIM and LPIPS [102] as evaluation metrics. As mentioned in [4, 28], there is an inconsistency in the DSSIM implementation across methods. For fair comparison, we do our best to group existing methods' DSSIM results into two categories (DSSIM$_1$ and DSSIM$_2$). Using the $structural\_similarity$ function from $scikit\text{-}image$ library, DSSIM$_1$ sets $data\_range$ to 1.0 while DSSIM$_2$ sets $data\_range$ to 2.0. We use FPS as metric for rendering speed. Metrics are averaged over all six scenes except noted otherwise.

As shown in Tab. 1, our method achieves 140 FPS and outperforms the others by a large margin. Our approach also has the best LPIPS in all comparisons and the best PSNR/DSSIM in most cases. Fig. 4 shows qualitative comparisons on a representative view that is widely used in other work. Compared to the other baselines, our result contains more vivid details (*e.g.*, the textures on the salmon) and artifact-less rendering (*e.g.*, the caustics on the cup). Please see supplementary material for comparisons with concurrent methods.

6.2. Google Immersive Dataset

Google Immersive Dataset [10] contains indoor and outdoor scenes captured with a 46-camera rig. The cameras are in fish-eye mode and are mounted on an outward-facing hemi-

Figure 4. **Qualitative comparisons on the Neural 3D Video Dataset.**

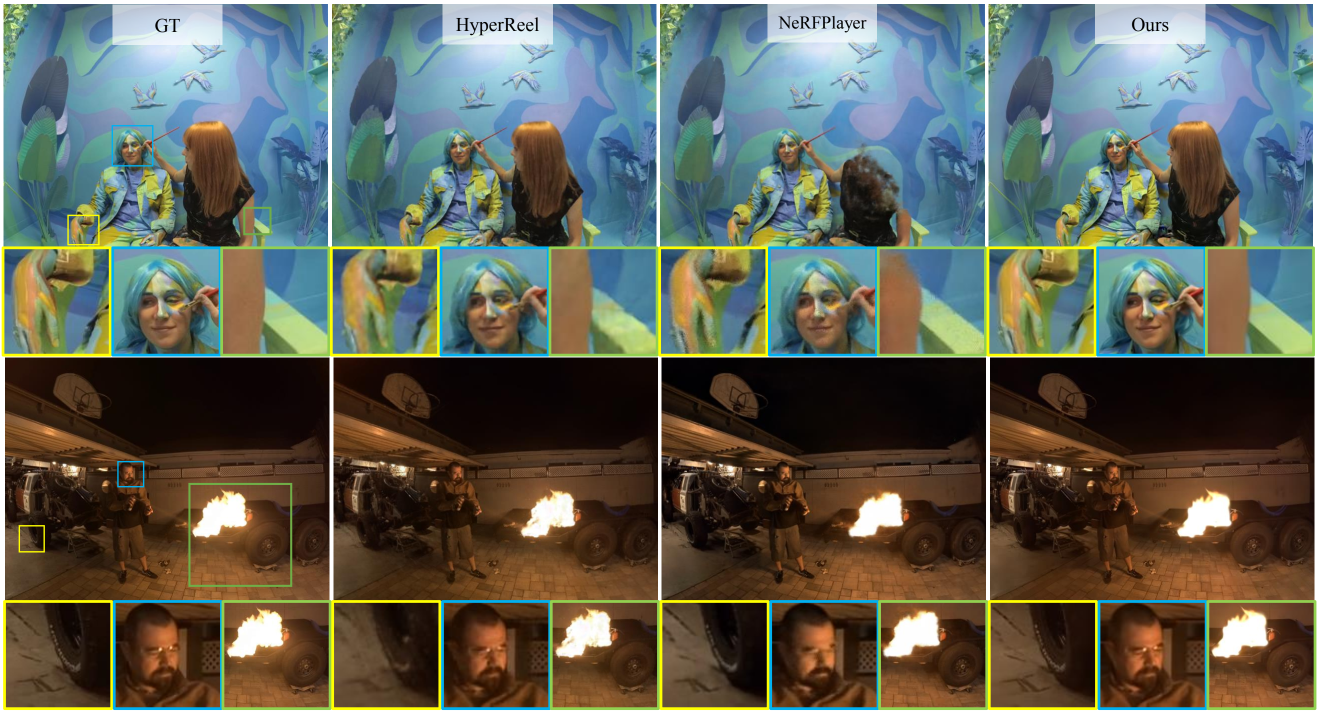


Figure 5. **Qualitative comparisons on the Google Immersive Dataset.**

sphere. Compared to outside-in setups, there is less overlap among views, hence posing additional challenges.

Following [4, 80], we evaluate on 7 selected scenes (*Welder*, *Flames*, *Truck*, *Exhibit*, *Face Paint 1*, *Face Paint 2*, *Cave*) and hold out the center camera as test view. The numerical results of NeRFPlayer [80] and HyperReel [4] are

Table 3. **Quantitative comparisons on the Technicolor Dataset.** "Size/Fr" stands for model size per frame.

Method	PSNR↑	DSSIM$_1$↓	DSSIM$_2$↓	LPIPS↓	FPS↑	Size/Fr↓
DyNeRF [48]	31.8	-	0.021	0.140	0.02	**0.6 MB**
HyperReel [4]	32.7	0.047	-	0.109	4.00	1.2 MB
Ours	**33.6**	**0.040**	**0.019**	**0.084**	**86.7**	1.1 MB

Table 4. **Ablation study of proposed components.** Conducted on all the five scenes from the Technicolor Dataset.

Method	PSNR↑	DSSIM$_1$↓	LPIPS↓
w/o Temporal Opacity	31.0	0.063	0.153
w/o Polynomial Motion	32.6	0.045	0.099
w/o Polynomial Rotation	33.4	0.042	0.085
w/o Feature Splatting	33.0	0.044	0.097
w/o Guided Sampling of Gaussians	33.3	0.041	0.085
Ours Full	**33.6**	**0.040**	**0.084**

Table 5. **Ablation study on the number of frames whose SfM point clouds are used in initialization.** Conducted on the *Theater* scene from the Technicolor Dataset.

Every N Frames	PSNR↑	DSSIM$_1$↓	LPIPS↓	Size↓
N = 1	**31.58**	0.059	0.124	110.2 MB
N = 4	31.51	**0.057**	**0.117**	46.7 MB
N = 16	31.04	0.060	0.139	**32.6 MB**

from their papers. The visual results of NeRFPlayer [80] are obtained from their authors while those of HyperReel [4] are produced by running their released codes.

As shown in Tab. 2, our method outperforms NeRFPlayer and HyperReel in both speed and quality. Compared to HyperReel, our method is over 10 times faster in rendering speed. Although our PSNR is only 0.4 dB higher, the improvements in DSSIM and LPIPS are significant. When compared to NeRFPlayer, the margin is larger for all metrics. Visual comparisons are shown in Fig. 5. Our method depicts sharper details and fewer artifacts than the others.

6.3. Technicolor Dataset

Technicolor Light Field Dataset [76] contains videos taken with a 4x4 camera array. Each camera is time-synchronized and the spatial resolution is 2048×1088. In alignment with HyperReel [4], we hold out the camera at second row second column and evaluate on five scenes (*Birthday*, *Fabien*, *Painter*, *Theater*, *Trains*) at full resolution.

Tab. 3 shows the comparisons, where our method achieves noticeable gain in quality and speed. Please refer to supplementary material for visual comparisons.

6.4. Ablation Study

To evaluate the effectiveness of proposed components, we conduct an ablation study in Tab. 4 using all the five scenes from Technicolor Dataset. Below we describe the configuration and performance of each ablation baseline.

Temporal Opacity. "w/o Temporal Opacity" fixes the center and scale of the temporal radial basis functions during training. This variant suffers from a significant performance drop, revealing the importance of temporal opacity.

Polynomial Motion and Rotation. "w/o Polynomial Motion" and "w/o Polynomial Rotation" fix the spatial position and rotation of STGs respectively. Both lead to a performance drop. Comparatively, motion is more important than rotation, which motivates us to use a lower-degree polynomial for rotation.

Feature Splatting. "w/o Feature Splatting" uses the base RGB color $\mathbf{F}^{base}$ as the final color. It can be seen that there is a moderate drop in quality due to reduced ability to model view- and time-dependent appearance.

Guided Sampling of Gaussians. "w/o Guided Sampling of Gaussians" does not encounter much performance drop in this dataset. The reason is that the scenes contain rich textures and can be well covered by SfM points. However, for other challenging scenes, guided sampling plays an important role (see Fig. 3 and supplementary material for examples).

Number of Frames used for Initialization. We further analyzed the number of frames used for initialization in Tab. 5. Using fewer frames slightly downgrade quality, but also significantly reduces model size. It reveals that the compactness of our method can be further enhanced with a good selection of frames for initialization.

6.5. Limitations

Although our representation achieves fast rendering speed, it cannot be trained on-the-fly. The support for on-the-fly training could benefit numerous streaming applications. To achieve this, advanced initialization techniques could be explored to accelerate the training process or alleviate the requirement of per-scene training. On the other hand, our method currently focuses on multi-view video inputs. It is promising to adapt our approach to monocular setting by combining with regularization or generative priors.

7. Conclusion

We present a novel representation based on Spacetime Gaussians for dynamic view synthesis. The proposed Spacetime Gaussians are enhanced with temporal opacity and parametric motion/rotation to model complex 4D content. To increase model compactness and encode view/time-dependent appearance, we introduce splatted feature rendering, which utilizes neural features and a lightweight MLP instead of spherical harmonics. Additionally, we leverage guided sampling of Gaussians to further improve the rendering quality of complex scenes. Experiments on real-world datasets show that our representation delivers state-of-the-art quality at high resolution and FPS, while maintaining a compact model size.

References

[1] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *European Conference on Computer Vision*, pages 696–712. Springer, 2020. 2

[2] Benjamin Attal, Selena Ling, Aaron Gokaslan, Christian Richardt, and James Tompkin. Matryodshka: Real-time 6dof video view synthesis using multi-sphere images. In *European Conference on Computer Vision*, pages 441–459. Springer, 2020. 3

[3] Benjamin Attal, Jia-Bin Huang, Michael Zollhöfer, Johannes Kopf, and Changil Kim. Learning neural light fields with ray-space embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19819–19829, 2022. 2

[4] 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. 2, 3, 6, 7, 8

[5] Aayush Bansal, Minh Vo, Yaser Sheikh, Deva Ramanan, and Srinivasa Narasimhan. 4d visualization of dynamic events from unconstrained multi-view videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5366–5375, 2020. 3

[6] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. 1

[7] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022.

[8] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19697–19705, 2023. 1, 2

[9] Mojtaba Bemana, Karol Myszkowski, Hans-Peter Seidel, and Tobias Ritschel. X-fields: Implicit neural view-, light-and time-image interpolation. *ACM Transactions on Graphics (TOG)*, 39(6):1–15, 2020. 3

[10] Michael Broxton, John Flynn, Ryan Overbeck, Daniel Erickson, Peter Hedman, Matthew Duvall, Jason Dourgarian, Jay Busch, Matt Whalen, and Paul Debevec. Immersive light field video with a layered mesh representation. *ACM Transactions on Graphics (TOG)*, 39(4):86–1, 2020. 3, 6

[11] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, page 425–432, New York, NY, USA, 2001. Association for Computing Machinery. 2

[12] 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

[13] Rodrigo Ortiz Cayon, Abdelaziz Djelouah, and George Drettakis. A bayesian approach for selective image-based rendering using superpixels. In *2015 International Conference on 3D Vision*, pages 469–477. IEEE, 2015. 2

[14] Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Transactions on Graphics (TOG)*, 32(3):1–12, 2013. 2

[15] 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. 1, 2

[16] Zhang Chen, Anpei Chen, Guli Zhang, Chengyuan Wang, Yu Ji, Kiriakos N Kutulakos, and Jingyi Yu. A neural rendering framework for free-viewpoint relighting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5599–5610, 2020. 2

[17] Zhang Chen, Yinda Zhang, Kyle Genova, Sean Fanello, Sofien Bouaziz, Christian Häne, Ruofei Du, Cem Keskin, Thomas Funkhouser, and Danhang Tang. Multiresolution deep implicit functions for 3d shape representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13087–13096, 2021. 2

[18] Zhang Chen, Zhong Li, Liangchen Song, Lele Chen, Jingyi Yu, Junsong Yuan, and Yi Xu. Neurbf: A neural fields representation with adaptive radial basis functions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4182–4194, 2023. 1, 3, 4

[19] 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. 3

[20] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, page 11–20, New York, NY, USA, 1996. Association for Computing Machinery. 2

[21] Yuqi Ding, Zhong Li, Zhang Chen, Yu Ji, Jingyi Yu, and Jinwei Ye. Full-volume 3d fluid flow reconstruction with light field piv. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 3

[22] Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14304–14314. IEEE Computer Society, 2021. 3

[23] 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*, pages 1–9, 2022. 3

[24] Liangji Fang, Qinhong Jiang, Jianping Shi, and Bolei Zhou. Tpnet: Trajectory proposal network for motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6797–6806, 2020. 4

[25] Brandon Yushan Feng and Amitabh Varshney. Signet: Efficient neural representation for light fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14224–14233, 2021. 2

[26] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. Deepview: View synthesis with learned gradient descent. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2367–2376, 2019. 2

[27] 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. 1, 2

[28] 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

[29] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5712–5721, 2021. 3

[30] Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. Monocular dynamic view synthesis: A reality check. *Advances in Neural Information Processing Systems*, 35:33768–33780, 2022. 3

[31] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Siggraph*, pages 43–54, 1996. 2

[32] Peter Hedman, Tobias Ritschel, George Drettakis, and Gabriel Brostow. Scalable inside-out image-based rendering. *ACM Trans. Graph.*, 35(6), 2016. 2

[33] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. In *SIGGRAPH Asia 2018 Technical Papers*, page 257. ACM, 2018. 2

[34] Benno Heigl, Reinhard Koch, Marc Pollefeys, Joachim Denzler, and Luc J. Van Gool. Plenoptic modeling and rendering from image sequences taken by hand-held camera. In *Mustererkennung 1999, 21. DAGM-Symposium*, page 94–101, Berlin, Heidelberg, 1999. Springer-Verlag. 2

[35] Adam Houenou, Philippe Bonnifait, Véronique Cherfaoui, and Wen Yao. Vehicle trajectory prediction based on motion model and maneuver recognition. In *2013 IEEE/RSJ international conference on intelligent robots and systems*, pages 4363–4369. IEEE, 2013. 4

[36] Wenbo Hu, Yuling Wang, Lin Ma, Bangbang Yang, Lin Gao, Xiao Liu, and Yuewen Ma. Tri-miprf: Tri-mip representation for efficient anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19774–19783, 2023. 2

[37] 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. *arXiv preprint arXiv:2312.14937*, 2023. 3

[38] Mustafa Işık, Martin Rünz, Markos Georgopoulos, Taras Khakhulin, Jonathan Starck, Lourdes Agapito, and Matthias Nießner. Humanrf: High-fidelity neural radiance fields for humans in motion. *ACM Transactions on Graphics (TOG)*, 42(4):1–12, 2023. 3

[39] Ramesh Jain and Koji Wakimoto. Multiple perspective interactive video. In *Proceedings of the international conference on multimedia computing and systems*, pages 202–211. IEEE, 1995. 3

[40] Takeo Kanade, Peter Rander, and PJ Narayanan. Virtualized reality: Constructing virtual worlds from real scenes. *IEEE multimedia*, 4(1):34–47, 1997. 3

[41] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (TOG)*, 42(4):1–14, 2023. 1, 2, 3, 4, 5, 6

[42] Seoha Kim, Jeongmin Bae, Youngsik Yun, Hahyun Lee, Gun Bang, and Youngjung Uh. Sync-nerf: Generalizing dynamic nerfs to unsynchronized videos. In *AAAI Conference on Artificial Intelligence*, 2024. 3

[43] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[44] Johannes Kopf, Michael F Cohen, and Richard Szeliski. First-person hyper-lapse videos. *ACM Transactions on Graphics (TOG)*, 33(4):78, 2014. 2

[45] Agelos Kratimenos, Jiahui Lei, and Kostas Daniilidis. Dynmf: Neural motion factorization for real-time dynamic view synthesis with 3d gaussian splatting. *arXiv preprint arXiv:2312.00112*, 2023. 3

[46] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 31–42. ACM, 1996. 2

[47] Lingzhi Li, Zhen Shen, zhongshu wang, Li Shen, and Ping Tan. Streaming radiance fields for 3d video synthesis. In *Advances in Neural Information Processing Systems*, 2022. 3, 6

[48] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5521–5531, 2022. 2, 3, 6, 8

[49] Zhong Li, Yu Ji, Wei Yang, Jinwei Ye, and Jingyi Yu. Robust 3d human motion reconstruction via dynamic template construction. In *2017 International Conference on 3D Vision (3DV)*, pages 496–505. IEEE, 2017. 3

[50] Zhong Li, Minye Wu, Wangyiteng Zhou, and Jingyi Yu. 4d human body correspondences from panoramic depth maps.

In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2877–2886, 2018.

[51] Zhong Li, Yu Ji, Jingyi Yu, and Jinwei Ye. 3d fluid flow reconstruction using compact light field piv. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 120–136. Springer, 2020. 3

[52] 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*, pages 6498–6508, 2021. 3

[53] Zhong Li, Liangchen Song, Celong Liu, Junsong Yuan, and Yi Xu. Neulf: Efficient novel view synthesis with neural 4d light field. In *Eurographics Symposium on Rendering*, 2022. 2

[54] Zhong Li, Liangchen Song, Zhang Chen, Xiangyu Du, Lele Chen, Junsong Yuan, and Yi Xu. Relit-neulf: Efficient relighting and novel view synthesis via neural 4d light field. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 7007–7016, 2023. 2

[55] 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)*, pages 4273–4284, 2023. 3

[56] Haotong Lin, Sida Peng, Zhen Xu, Tao Xie, Xingyi He, Hujun Bao, and Xiaowei Zhou. Im4d: High-fidelity and real-time novel view synthesis for dynamic scenes. *arXiv preprint arXiv:2310.08585*, 2023. 3

[57] Kai-En Lin, Lei Xiao, Feng Liu, Guowei Yang, and Ravi Ramamoorthi. Deep 3d mask volume for view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1749–1758, 2021. 3

[58] Kai-En Lin, Guowei Yang, Lei Xiao, Feng Liu, and Ravi Ramamoorthi. View synthesis of dynamic scenes based on deep 3d mask volume. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 3

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

[60] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020. 2

[61] 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*, pages 13–23, 2023. 3

[62] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 38(4):65:1–65:14, 2019. 2, 3, 6

[63] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhoefer, Yaser Sheikh, and Jason Saragih. Mixture of volumetric primitives for efficient neural rendering. *ACM Transactions on Graphics (ToG)*, 40(4):1–13, 2021. 3

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

[65] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 2019. 6

[66] 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. 1, 2

[67] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41 (4):102:1–102:15, 2022. 1, 2

[68] Thomas Neff, Pascal Stadlbauer, Mathias Parger, Andreas Kurz, Joerg H Mueller, Chakravarty R Alla Chaitanya, Anton Kaplanyan, and Markus Steinberger. Donerf: Towards real-time rendering of compact neural radiance fields using depth oracle networks. In *Computer Graphics Forum*, pages 45–59. Wiley Online Library, 2021. 2

[69] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 3

[70] 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 Transactions on Graphics (TOG)*, 40: 1 – 12, 2021. 3

[71] Sida Peng, Yunzhi Yan, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Representing volumetric videos as dynamic mlp maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4252–4262, 2023. 3

[72] Eric Penner and Li Zhang. Soft 3d reconstruction for view synthesis. *ACM Transactions on Graphics (TOG)*, 36(6): 1–11, 2017. 2

[73] Martin Piala and Ronald Clark. Terminerf: Ray termination prediction for efficient neural rendering. In *2021 International Conference on 3D Vision (3DV)*, pages 1106–1114. IEEE, 2021. 2

[74] 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. 3

[75] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with

thousands of tiny mlps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14335–14345, 2021. 2

[76] Neus Sabater, Guillaume Boisson, Benoit Vandame, Paul Kerbiriou, Frederic Babon, Matthieu Hog, Remy Gendrot, Tristan Langlois, Olivier Bureller, Arno Schubert, et al. Dataset and pipeline for multi-view light-field video. In *Proceedings of the IEEE conference on computer vision and pattern recognition Workshops*, pages 30–40, 2017. 6, 8

[77] 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

[78] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2437–2446, 2019. 2

[79] Vincent Sitzmann, Semon Rezchikov, Bill Freeman, Josh Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. *Advances in Neural Information Processing Systems*, 34: 19313–19325, 2021. 2

[80] 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. 2, 3, 6, 7, 8

[81] Mohammed Suhail, Carlos Esteves, Leonid Sigal, and Ameesh Makadia. Light field neural rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8269–8279, 2022. 2

[82] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022. 1, 2

[83] Towaki Takikawa, Joey Litalien, Kangxue Yin, Karsten Kreis, Charles Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11358–11367, 2021. 2

[84] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019. 2

[85] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12959–12970, 2021. 3

[86] Chaoyang Wang, Ben Eckart, Simon Lucey, and Orazio Gallo. Neural trajectory fields for dynamic novel view synthesis. *arXiv preprint arXiv:2105.05994*, 2021. 3

[87] Feng Wang, Sinan Tan, Xinghang Li, Zeyue Tian, Yafei Song, and Huaping Liu. Mixed neural voxels for fast multi-view video synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19706–19716, 2023. 2, 3, 6

[88] Feng Wang, Zilong Chen, Guokang Wang, Yafei Song, and Huaping Liu. Masked space-time hash encoding for efficient dynamic scene reconstruction. *Advances in Neural Information Processing Systems*, 36, 2024. 3

[89] Huan Wang, Jian Ren, Zeng Huang, Kyle Olszewski, Menglei Chai, Yun Fu, and Sergey Tulyakov. R2l: Distilling neural radiance field to neural light field for efficient novel view synthesis. In *European Conference on Computer Vision*, pages 612–629. Springer, 2022. 2

[90] Liao Wang, Jiakai Zhang, Xinhang Liu, Fuqiang Zhao, Yanshun Zhang, Yingliang Zhang, Minye Wu, Jingyi Yu, and Lan Xu. Fourier plenoctrees for dynamic radiance field rendering in real-time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13524–13534, 2022. 3

[91] Liao Wang, Qiang Hu, Qihan He, Ziyu Wang, Jingyi Yu, Tinne Tuytelaars, Lan Xu, and Minye Wu. Neural residual radiance fields for streamably free-viewpoint videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 76–87, 2023. 3

[92] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7467–7477, 2020. 2

[93] 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. *arXiv preprint arXiv:2310.08528*, 2023. 3

[94] 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*, pages 9421–9431, 2021. 3

[95] Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. *arXiv preprint arXiv:2311.12198*, 2023. 3

[96] 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

[97] Zhen Xu, Sida Peng, Haotong Lin, Guangzhao He, Jiaming Sun, Yujun Shen, Hujun Bao, and Xiaowei Zhou. 4k4d: Real-time 4d view synthesis at 4k resolution. *arXiv preprint arXiv:2310.11448*, 2023. 3

[98] Jason C Yang, Matthew Everett, Chris Buehler, and Leonard McMillan. A real-time distributed light field camera. *Rendering Techniques*, 2002(77-86):2, 2002. 3

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

[100] Zeyu Yang, Hongye Yang, Zijie Pan, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. In *International Conference on Learning Representations (ICLR)*, 2024. 3

[101] 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

[102] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 6

[103] Ke Colin Zheng, Alex Colburn, Aseem Agarwala, Maneesh Agrawala, David Salesin, Brian Curless, and Michael F Cohen. Parallax photography: creating 3d cinematic effects from stills. In *Proceedings of Graphics Interface 2009*, pages 111–118. Canadian Information Processing Society, 2009. 2

[104] C Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM transactions on graphics (TOG)*, 23(3):600–608, 2004. 3

[105] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. Ewa volume splatting. In *Proceedings Visualization, 2001. VIS'01.*, pages 29–538. IEEE, 2001. 3