GES : Generalized Exponential Splatting for Efficient Radiance Field Rendering

Abdullah Hamdi[1] **Luke Melas-Kyriazi**[1] **Jinjie Mai**[2] **Guocheng Qian**[2,4]
Ruoshi Liu[3] **Carl Vondrick**[3] **Bernard Ghanem**[2] **Andrea Vedaldi**[1]
[1]Visual Geometry Group, University of Oxford
[2]King Abdullah University of Science and Technology (KAUST)
[3]Columbia University [4]Snap Inc.
abdullah.hamdi@eng.ox.ac.uk

Abstract

*Advancements in 3D Gaussian Splatting have significantly accelerated 3D reconstruction and generation. However, it may require a large number of Gaussians, which creates a substantial memory footprint. This paper introduces GES (Generalized Exponential Splatting), a novel representation that employs Generalized Exponential Function (GEF) to model 3D scenes, requiring far fewer particles to represent a scene and thus significantly outperforming Gaussian Splatting methods in efficiency with a plug-and-play replacement ability for Gaussian-based utilities. GES is validated theoretically and empirically in both principled 1D setup and realistic 3D scenes. It is shown to represent signals with sharp edges more accurately, which are typically challenging for Gaussians due to their inherent low-pass characteristics. Our empirical analysis demonstrates that GEF outperforms Gaussians in fitting natural-occurring signals (*e.g.* squares, triangles, parabolic signals), thereby reducing the need for extensive splitting operations that increase the memory footprint of Gaussian Splatting. With the aid of a frequency-modulated loss, GES achieves competitive performance in novel-view synthesis benchmarks while requiring less than half the memory storage of Gaussian Splatting and increasing the rendering speed by up to 39%. The code is available on the project website* https://abdullahamdi.com/ges.

1. Introduction

The pursuit of more engaging and immersive virtual experiences across gaming, cinema, and the metaverse demands advancements in 3D technologies that balance visual richness with computational efficiency. In this regard, 3D Gaussian Splatting (GS) [18] is a recent alternative to neural radiance fields [10, 12, 29–31, 34, 35, 51] for learning and rendering 3D objects and scenes. GS represents a scene as a large mixture of small, coloured Gaussians. Its key advantage is the existence of a very fast differentiable renderer, which makes this representation ideally suited for real-time applications and significantly reduces the learning cost. Specifically, fast rendering of learnable 3D representations is of key importance for applications like gaming, where high-quality, fluid, and responsive graphics are essential.

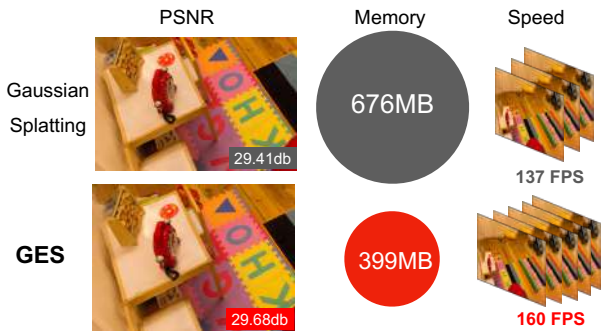


Figure 1. **GES: Generalized Exponential Splatting** We propose a faster and more memory-efficient alternative to Gaussian Splatting [18] that relies on Generalized exponential Functions (with additional learnable shape parameters) instead of Gaussians.

However, GS is not without shortcomings. We notice in particular that GS implicitly makes an assumption on the nature of the modeled signals, which is suboptimal. Specifically, Gaussians correspond to *low-pass filters*, but most 3D scenes are far from low-pass as they contain abrupt discontinuities in shape and appearance. Fig.2 demosntrates this inherent low-pass limitation of Gaussian-based methods. As a result, GS needs to use a huge number of very small Gaussians to represent such 3D scenes, far more than if a more appropriate basis was selected, which negatively impacts memory utilization.

To address this shortcoming, in this work, we intro-

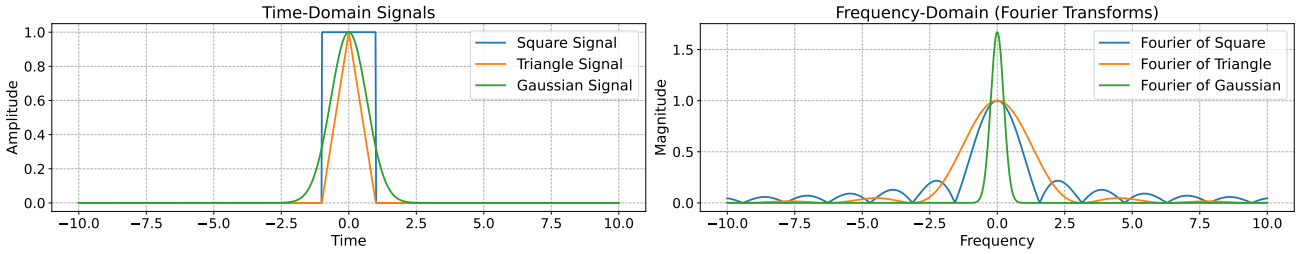


Figure 2. **The Inherent Low-Pass Limitation of Gaussians**. We illustrate the bandwidth constraint of Gaussian functions compared to square and triangle signals. The Gaussian functions' low-pass property restricts their ability to fit signals with sharp edges that have infinite bandwidth. This limitation constitutes a challenge for 3D Gaussian Splatting [18] in accurately fitting high-bandwidth 3D spatial data.

duce *GES* (Generalized Exponential Splatting), a new approach that utilizes the Generalized Exponential Function (GEF) for modeling 3D scenes (Fig.1). Our method is designed to effectively represent signals, especially those with sharp features, which previous Gaussian splatting techniques often smooth out or require extensive splitting to model [18]. Demonstrated in Fig.3, we show that while $N = 5$ randomly initialized Gaussians are required to fit a square, only 2 GEFs are needed for the same signal. This stems from the fact that Gaussian mixtures have a low-pass frequency domain, while many common signals, like the square, are not band-limited. This high-band modeling constitutes a fundamental challenge to Gaussian-based methods. To help GES to train gradually from low-frequency to high-frequency details, we propose a specialized frequency-modulated image loss. This allows GES to achieve more than 50% reduction in the memory requirement of Gaussian splatting and up to 39% increase in rendering speed while maintaining a competitive performance on standard novel view synthesis benchmarks.

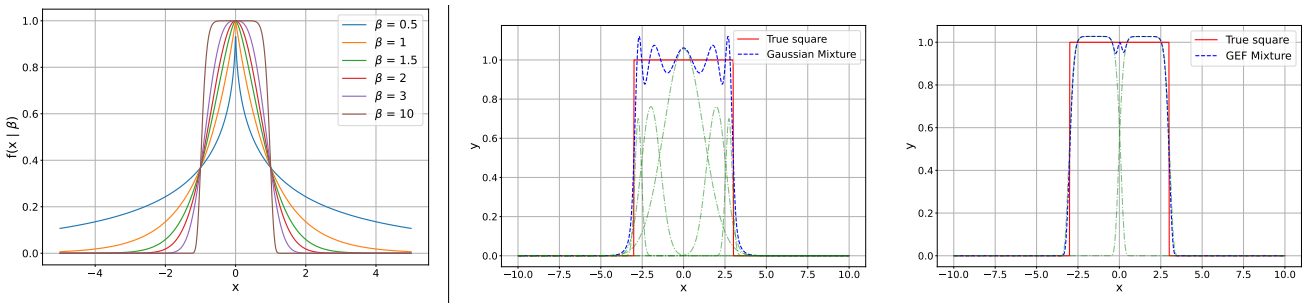
We summarize our contributions as follows:

- We present principled numerical simulations motivating the use of the Generalized Exponential Functions (GEF) instead of Gaussians for scene modeling.
- We propose Generalized Exponential Splatting (GES), a novel 3D representation that leverages GEF to develop a splatting-based method for realistic, real-time, and memory-efficient novel view synthesis.
- Equipped with a specialized frequency-modulated image loss and through extensive experiments on standard benchmarks on novel view synthesis, GES shows a 50% reduction in memory requirement and up to 39% increase in rendering speed for real-time radiance field rendering based on Gaussian Splatting. GES can act as a plug-and-play replacement for *any* Gaussian-based utilities.

2. Related work

Multi-view 3D reconstruction. Multi-view 3D reconstruction aims to recover the 3D structure of a scene from its 2D RGB images captured from different camera positions [1, 9]. Classical approaches usually recover a scene's geometry as a point cloud using SIFT-based [28] point matching [36, 38]. More recent methods enhance them by relying on neural networks for feature extraction (*e.g.* [14, 45, 46, 53]). The development of Neural Radiance Fields (NeRF) [26, 30] has prompted a shift towards reconstructing 3D as volume radiance [39], enabling the synthesis of photo-realistic novel views [3, 4, 40]. Subsequent works have also explored the optimization of NeRF in few-shot (*e.g.* [8, 15, 19]) and one-shot (*e.g.* [5, 52]) settings. NeRF does not store any 3D geometry explicitly (only the density field), and several works propose to use a signed distance function to recover a scene's surface [6, 23, 24, 41, 42, 47, 48], including in the few-shot setting as well (*e.g.* [54, 55]).

Differentiable rendering. Gaussian Splatting is a point-based rendering [2, 11] algorithm that parameterizes 3D points as Gaussian functions (mean, variance, opacity) with spherical harmonic coefficients for the angular radiance component [50]. Prior works have extensively studied differentiable rasterization, with a series of works [17, 25, 27] proposing techniques to define a differentiable function between triangles in a triangle mesh and pixels, which allows for adjusting parameters of triangle mesh from observation. These works range from proposing a differentiable renderer for mesh processing with image filters [22], and proposing to blend schemes of nearby triangles [33], to extending differentiable rasterization to large-scale indoor scenes [49]. On the point-based rendering [11] side, neural point-based rendering [17] allows features to be learned and stored in 3D points for geometrical and textural information. Wiles *et al.* combine neural point-based rendering with an adversarial loss for better photorealism [43], whereas later works use points to represent a radiance field, combining NeRF and point-based rendering [44, 56]. Our GES is a point-based rasterizer in which every point represents a generalized exponential with scale, opacity, and shape, affecting the rasterization accordingly.



(a) A family of GEFs $f_\beta(x)$ (b) Five Gaussians fitting a square (c) Two GEFs fitting a square

Figure 3. **Generalized Exponential Function (GEF)**. (*a*): We show a family of GEFs $f_\beta(x) = Ae^{-\left(\frac{|x-\mu|}{\alpha}\right)^\beta}$ with different β values for $\alpha = 1, \mu = 0$. When $\beta = 2$, the function reduces to the Gaussian function followed in 3D gaussian splatting [18]. In our GES , we learn β as another parameter of each splatting component. (*b,c*): The proposed GEF mixture, with learnable β, fits the same signal (square) with fewer components compared to Gaussian functions using gradient-based optimizations. (*b*): We show an example of the fitted mixture with $N = 5$ components when Gaussians are used *vs*. (*c*) when GEF is used with $N = 2$ components. GEF achieves less error loss (0.44) and approximates sharp edges better than the Gaussian counterpart (0.48 error) with less number of components. The optimized individual components (initialized with random parameters) are shown in green after convergence.

3. Properties of Generalized Exponentials

3.1. Generalized Exponential Function

Preliminaries. The Generalized Exponential Function (GEF) is similar to the probability density function (PDF) of the Generalized Normal Distribution (GND) [7]. This function allows for a more flexible adaptation to various data shapes by adjusting the shape parameter $\beta \in (0, \infty)$. The GEF is given by:

$$f(x|\mu, \alpha, \beta, A) = A \exp\left(-\left(\frac{|x-\mu|}{\alpha}\right)^\beta\right) \quad (1)$$

where $\mu \in \mathbb{R}$ is the location parameter, $\alpha \in \mathbb{R}$ is the scale parameter, $A \in \mathbb{R}^+$ defines a positive amplitude. The behavior of this function is illustrated in Fig.3. For $\beta = 2$, the GEF becomes a scaled Gaussian $f(x|\mu, \alpha, \beta = 2, A) = Ae^{-\frac{1}{2}\left(\frac{x-\mu}{\alpha/\sqrt{2}}\right)^2}$. The GEF, therefore, provides a versatile framework for modeling a wide range of data by varying β, unlike the Gaussian mixtures, which have a low-pass frequency domain. Many common signals, like the square or triangle, are band-unlimited, constituting a fundamental challenge to Gaussian-based methods. In this paper, we try to *learn* a positive β for every component of the Gaussian splatting to allow for a generalized 3D representation.

3.2. Assessing 1D GEF Mixtures in Simulation

We evaluate the effectiveness of a mixture of GEFs in representing various one-dimensional (1D) signal types. This evaluation is conducted by fitting the model to synthetic signals that replicate characteristics properties of common real-world signals. More details and additional simulation results are provided in *Supplementary Material*.

Simulation Setup. The experimental framework was based on a series of parametric models implemented in PyTorch [32], designed to approximate 1D signals using mixtures of different functions such as Gaussian (low-pass), Difference of Gaussians (DoG), Laplacian of Gaussian (LoG), and a GEF mixture model. Each model comprised parameters for means, variances (or scales), and weights, with the generalized model incorporating an additional parameter, β, to control the exponentiation of the GEF function.

Model Configuration. The models were configured with a varying number of components N, with tests conducted using $N = \{2, 5, 8, 10, 15, 20\}$. The weights of the components are chosen to be positive. All the parameters of all the N components were learned. Each model was trained using the Adam optimizer with a mean squared error loss function. The input x was a linearly spaced tensor representing the domain of the synthetic signal, and the target y was the value of the signal at each point in x. Training proceeded for a predetermined number of epochs, and the loss was recorded at the end of training.

Data Generation. Synthetic 1D signals were generated for various signal types over a specified range, with a given data size and signal width. The signals were used as the ground truth for training the mixture models. The ground truth signals used in the experiment are one-dimensional (1D) functions that serve as benchmarks for evaluating signal processing algorithms. The signal types under study are: *square*, *triangle*, *parabolic*, *half sinusoidal*, *Gaussian*, and *exponential* functions. We show Fig.3 an example of fitting a Gaussian when $N = 5$ and a Generalized mixture on the square signal when $N = 2$. Note how sharp edges constitute a challenge for Gaussians that have low pass bandwidth while a square signal has an infinite bandwidth known by

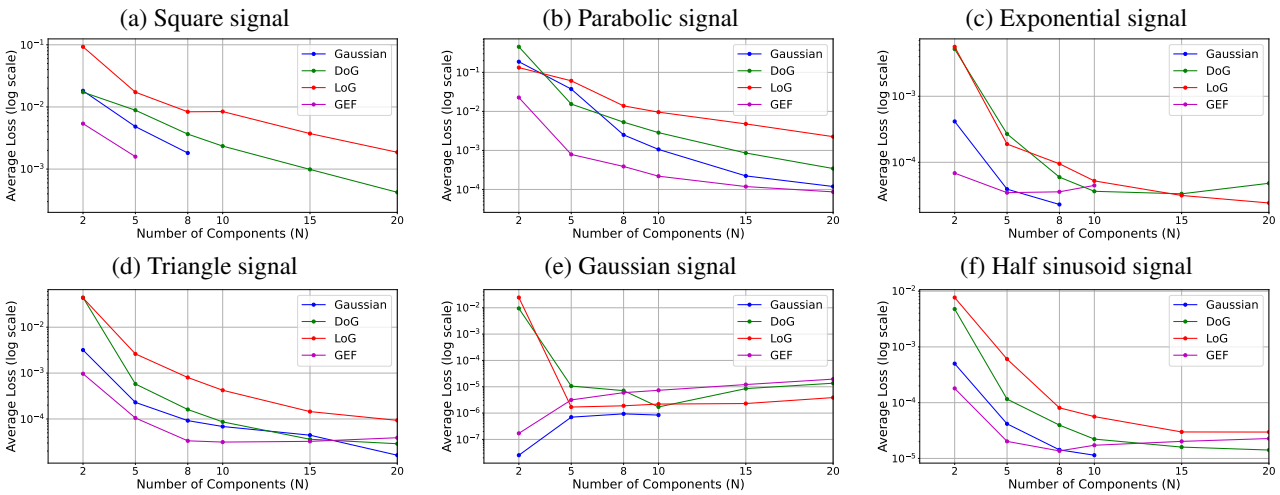


Figure 4. **Numerical Simulation Results of Different Mixtures.** We show a comparison of average loss for different mixture models optimized with gradient-based optimizers across varying numbers of components on various signal types (a-f). In the case of 'NaN' loss (gradient explosion), the results are not shown on the plots. Full simulation results are provided in *Supplementary Material*

the sinc function [16].

Simulation Results. The models' performance was evaluated based on the loss value after training. Additionally, the model's ability to represent the input signal was visually inspected through generated plots. Multiple runs per configuration were executed to account for variance in the results. For a comprehensive evaluation, each configuration was run multiple times (20 runs per configuration) to account for variability in the training process. During these runs, the number of instances where the training resulted in a 'nan' loss was removed from the loss plots, and hence some plots in Fig.4 do not have loss values at some N. As depicted in Fig.4, the GEF Mixture consistently yielded the lowest loss across the number of components, indicating its effective approximation of many common signals, especially band-unlimited signals like the square and triangle. The only exception is the Gaussian signal, which is (obviously) fitted better with a Gaussian Mixture.

4. Generalized Exponential Splatting (GES)

Having established the benefits of GEF of Eq.(1) over Gaussian functions, we will now demonstrate how to extend GEF into the Generalized Exponential Splatting (GES) framework, offering a plug-and-play replacement for Gaussian Splatting. We also start with a collection of static images of a scene and their corresponding camera calibrations obtained through Structure from Motion (SfM) [37], which additionally provides a sparse point cloud. Moving beyond Gaussian models [18], GES adopts an exponent β to tailor the focus of the splats, thus sharpening the delineation of scene edges. This technique is not only more efficient in memory usage but also can surpass Gaussian splatting in established benchmarks for novel view synthesis.

4.1. Differentiable GES Formulation

Our objective is to enhance novel view synthesis with a refined scene representation. We leverage a generalized exponential form, here termed Generalized Exponential Splatting, which for location $\mathbf{x}$ in 3D space and a positive definite matrix $\mathbf{\Sigma}$, is defined by:

$$L(\mathbf{x};\boldsymbol{\mu},\mathbf{\Sigma},\beta) = \exp\left\{-\frac{1}{2}\left((\mathbf{x}-\boldsymbol{\mu})^\intercal\mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)^{\frac{\beta}{2}}\right\}, \tag{2}$$

where $\boldsymbol{\mu}$ is the location parameter and $\mathbf{\Sigma}$ is the covariance matrix equivalance in Gaussian Splatting [18]. β is a shape parameter that controls the sharpness of the splat. When $\beta = 2$, this formulation is equivalent to Gaussian splatting [18]. Our approach maintains an opacity measure κ for blending and utilizes spherical harmonics for coloring, similar to Gaussian splatting [18].

For 2D image projection, we adapt the technique by Zwicker *et al*. [57], but keep track of our variable exponent β. The camera-space covariance matrix $\mathbf{\Sigma}'$ is transformed as follows: $\mathbf{\Sigma}' = \mathbf{JW\Sigma W}^\intercal\mathbf{J}^\intercal$, where $\mathbf{J}$ is the Jacobian of the transformation from world to camera space, and $\mathbf{W}$ is a diagonal matrix containing the inverse square root of the eigenvalues of $\mathbf{\Sigma}$. We ensure $\mathbf{\Sigma}$ remains positively semi-definite throughout the optimization by formulating it as a product of a scaling matrix $\mathbf{S}$ (modified by some positive modification function $\phi(\beta) > 0$ as we show later) and a rotation matrix $\mathbf{R}$, with optimization of these components facilitated through separate 3D scale vectors $\mathbf{s}$ and quaternion rotations $\mathbf{q}$.

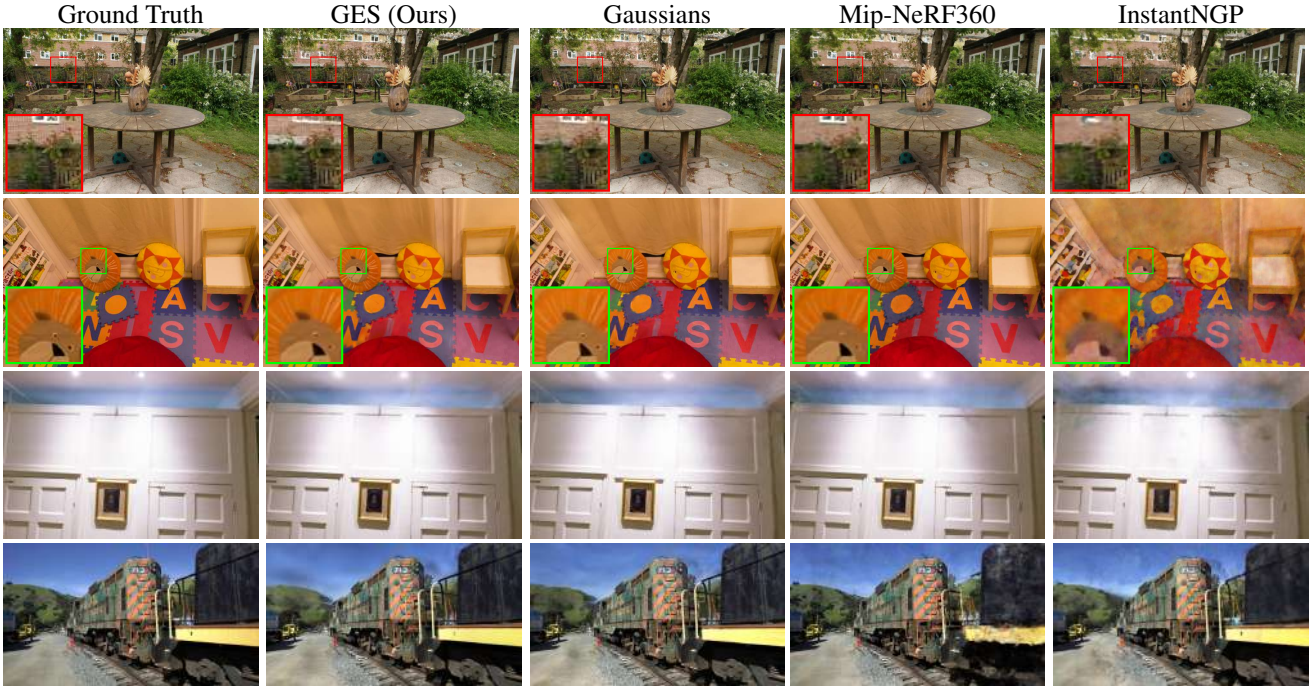


Figure 5. **Visual Comparison on Novel View Synthesis.** We display comparisons between our method and established baselines and the ground truth images. The depicted scenes are ordered as follows: GARDEN and ROOM from the Mip-NeRF360 dataset; DRJOHNSON from the Deep Blending dataset; and TRAIN from Tanks&Temples. Subtle differences in rendering quality are accentuated through zoomed-in details. These specific scenes were picked similarly to Gaussin Splatting [18] for a fair comparison. It might be difficult in general to see differences between GES and Gaussians because they have almost the same PSNR (despite GES requiring 50% less memory).

4.2. Fast Differentiable Rasterizer for GES

Intuition from Volume Rendering. The concept of volume rendering in the context of neural radiance fields [30] involves the integration of emitted radiance along a ray passing through a scene. The integral equation for the expected color $C(\mathbf{r})$ of a camera ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$, with near and far bounds t_n and t_f, respectively, is given by:

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\kappa(\mathbf{r}(t))c(\mathbf{r}(t), \mathbf{d})\, dt, \quad \text{where} \quad T(t) = \exp\left(-\int_{t_n}^{t} \kappa(\mathbf{r}(s))\, ds\right). \tag{3}$$

Here, $T(t)$ represents the transmittance along the ray from t_n to t, $\kappa(\mathbf{r}(t))$ is the volume density, and $c(\mathbf{r}(t), \mathbf{d})$ is the emitted radiance at point $\mathbf{r}(t)$ in the direction $\mathbf{d}$. The total distance $[t_n, t_f]$ crossed by the ray across non-empty space dictates the amount of lost energy and hence the reduction of the intensity of the rendered colors. In the Gaussian Splatting world [18], this distance $[t_n, t_f]$ is composed of the projected variances α of each component along the ray direction $\mathbf{o} + t\mathbf{d}$. In our GES of Eq.(2), if the shape parameter β of some individual component changes, the effective impact on Eq.(3) will be determined by the effective variance projection $\widehat{\alpha}$ of the same component modified by the modifcation function $\phi(\beta)$ as follows:

$$\widehat{\alpha}(\beta) = \phi(\beta)\alpha \quad . \tag{4}$$

Note that the modification function ϕ we chose does not depend on the ray direction since the shape parameter β is a global property of the splatting component, and we assume the scene to comprise many components. We tackle next the choice of the modification function ϕ and how it fits into the rasterization framework of Gaussian Splatting [18].

Approximate Rasterization. The main question is how to represent the GES in the rasterization framework. In effect, the rasterization in Gaussian Splatting [18] only relies on the variance splats of each component. So, we only need to simulate the effect of the shape parameter β on the covariance of each component to get the rasterization of GES . To do that, we modify the scales matrix of the covariance in each component by the scaler function $\phi(\beta)$ of that component. From probability theory, the exact conversion between the variance of the generalized exponential distribution and the variance of the Gaussian distribution is given by [7] as

$$\phi(\beta) = \frac{\Gamma(3/\beta)}{\Gamma(1/\beta)} \tag{5}$$

, where Γ is the Gamma function. This conversion in Eq.(5) ensures the PDF integrates to 1. In a similar manner, the

Figure 6. **Frequency-Modulated Image Masks.** For the input example image on the left, We show examples of the frequency loss masks M_ω used in Sec.4.3 for different numbers of target normalized frequencies ω ($\omega = 0\%$ for low frequencies to $\omega = 100\%$ for high frequencies). This masked loss helps our GES learn specific bands of frequencies. We use a linear schedule to determine these target ω values during the optimization of GES , $\omega = \frac{\text{current iteration}}{\text{total iterations}}$. Note that due to DoG filter sensitivity for high-frequencies, the mask for $0 < \omega \leq 50\%$ is defined as $1 - M_\omega$ of $50 < \omega \leq 100\%$. This ensures that all parts of the image will be covered by one of the masks M_ω, while focusing on the details more as the optimization progresses.

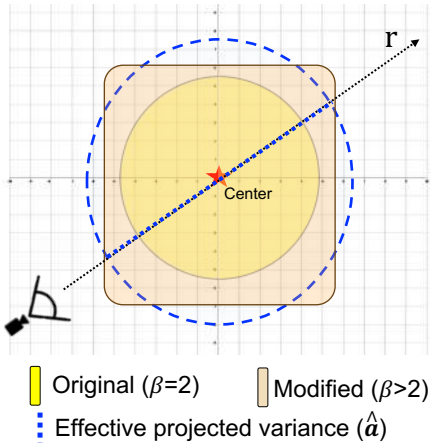


Figure 7. **Effective Variance of GES components**. We demonstrate the concept of effective variance projection $\widehat{\alpha}(\beta)$ for an individual splatting component intersecting a camera ray $\mathbf{r}$ under shape modification ($\beta > 2$). Note that $\widehat{\alpha}(\beta)$ is a scaled version of the original splat projected variance α.

integrals in Eq.(3) under Eq.(4) can be shown to be equivalent for Gaussians and GES using the same modification of Eq.(5). The modification will affect the rasterization *as if* we did perform the exponent change. It is a trick that allows using generalized exponential rasterization without taking the β exponent. Similarly, the Gaussian splatting [18] is *not learning rigid Gaussians*, it learns properties of point clouds that *act as if* there are Gaussians placed there when they splat on the image plane. Both our GES and Gaussians are in the same spirit of splatting, and representing 3D with splat properties. Fig.7 demonstrates this concept for an individual splatting component intersecting a ray $\mathbf{r}$ from the camera and the idea of effective variance projection $\widehat{\alpha}$. However, as can be in Fig.7, this scaler modification $\phi(\beta)$ introduces some view-dependent boundary effect error (*e.g.* if the ray $\mathbf{r}$ passed on the diagonal). We provide an upper bound estimate on this error in *Supplementary Material*.

4.3. Frequency-Modulated Image Loss

To effectively utilize the broad-spectrum capabilities of GES , it has been enhanced with a frequency-modulated image loss, denoted as $\mathcal{L}_\omega$. This loss is grounded in the rationale that GES , initially configured with Gaussian low-pass band splats, should primarily concentrate on low-frequency details during the initial stages of training. As training advances, with the splat formations adapting to encapsulate higher frequencies, the optimization's emphasis should gradually shift towards these higher frequency bands within the image. This concept bears a technical resemblance to the frequency modulation approach used in BARF [21], albeit applied within the image domain rather than the 3D coordinate space. The loss is guided by a frequency-conditioned mask implemented via a Difference of Gaussians (DoG) filter to enhance edge-aware optimization in image reconstruction tasks modulated by the normalized frequency ω. The DoG filter acts as a band-pass filter, emphasizing the edges by subtracting a blurred version of the image from another less blurred version, thus approximating the second spatial derivative of the image. This operation is mathematically represented as:

$$\text{DoG}(I) = G(I, \sigma_1) - G(I, \sigma_2), \quad 0 < \sigma_2 < \sigma_1$$

where $G(I, \sigma)$ denotes the Gaussian blur operation on image I with standard deviation σ. The choice of σ values dictates the scale of edges to be highlighted, effectively determining the frequency band of the filter. We chose $\sigma_1 = 2\sigma_2$ to ensure the validity of the band-pass filter, where the choice of σ_2 will determine the target frequency band of the filter. In our formulation, we use predetermined target normalized frequencies ω ($\omega = 0\%$ for low frequencies to $\omega = 100\%$ for high frequencies). We chose $\sigma_2 = 0.1 + 10\omega$ to ensure the stability of the filter and reasonable resulting masks. The filtered image is then used to generate an edge-aware mask M_ω through a pixel-wise comparison to a

Dataset	Mip-NeRF360 Dataset						Tanks&Temples						Deep Blending					
Method—Metric	$SSIM^{\uparrow}$	$PSNR^{\uparrow}$	$LPIPS^{\downarrow}$	Train$^{\downarrow}$	FPS$^{\uparrow}$	Mem$^{\downarrow}$	$SSIM^{\uparrow}$	$PSNR^{\uparrow}$	$LPIPS^{\downarrow}$	Train$^{\downarrow}$	FPS$^{\uparrow}$	Mem$^{\downarrow}$	$SSIM^{\uparrow}$	$PSNR^{\uparrow}$	$LPIPS^{\downarrow}$	Train$^{\downarrow}$	FPS$^{\uparrow}$	Mem$^{\downarrow}$
Plenoxels	0.626	23.08	0.463	26m	6.79	2.1GB	0.719	21.08	0.379	25m	13.0	2.3GB	0.795	23.06	0.510	28m	11.2	2.7GB
INGP	0.699	25.59	0.331	7.5m	9.43	48MB	0.745	21.92	0.305	7m	14.4	48MB	0.817	24.96	0.390	8m	2.79	48MB
Mip-NeRF360	0.792	27.69	0.237	48h	0.06	8.6MB	0.759	22.22	0.257	48h	0.14	8.6MB	0.901	29.40	0.245	48h	0.09	8.6MB
3D Gaussians-7K	0.770	25.60	0.279	6.5m	160	523MB	0.767	21.20	0.280	7m	197	270MB	0.875	27.78	0.317	4.5m	172	386MB
3D Gaussians-30K	0.815	27.21	0.214	42m	134	734MB	0.841	23.14	0.183	26m	154	411MB	0.903	29.41	0.243	36m	137	676MB
GES (ours)	0.794	26.91	0.250	32m	186	377MB	0.836	23.35	0.198	21m	210	222MB	0.901	29.68	0.252	30m	160	399MB

Table 1. **Comparative Analysis of Novel View Synthesis Techniques.** This table presents a comprehensive comparison of our approach with established methods across various datasets. The metrics, inclusive of SSIM, PSNR, and LPIPS, alongside training duration, frames per second, and memory usage, provide a multidimensional perspective of performance efficacy. Note that our training time numbers of the different methods may be computed on different GPUs; they are not necessarily perfectly comparable but are still valid. Note that non-explicit representations (INGP, Mip-NeRF360) have low memory because they rely on additional slow neural networks for decoding. Red-colored results are the best.

threshold value (after normalization) as follows.

$$M_\omega = \mathbb{1}\big(\text{DoG}_\omega(I_{\text{gt}})_{\text{normalized}} > \epsilon_\omega\big) \ , \\ \text{DoG}_\omega(I) = G(I, 0.2 + 20\omega) - G(I, 0.1 + 10\omega) \tag{6}$$

, where $0 \leq \epsilon_\omega \leq 1$ is the threshold (we pick 0.5) for a normalized response of the filter DoG_ω, I_{gt} is the ground truth image, and $\mathbb{1}$ is the indicator function. See Fig.6 for examples of the masks. The edge-aware frequency-modulated loss $\mathcal{L}_\omega$ is defined as:

$$\mathcal{L}_\omega = \|(I - I_{\text{gt}}) \cdot M_\omega\|_1, \tag{7}$$

where I is the reconstructed image, and $\|\cdot\|_1$ denotes the L1 norm. This term is integrated into the overall loss, as shown later. The mask is targeted for the specified frequencies ω. We use a linear schedule to determine these target ω values in Eq.(7) and Eq.(6) during the optimization of GES , $\omega = \frac{\text{current iteration}}{\text{total iterations}}$. The loss $\mathcal{L}_\omega$ aims to help in tuning the shape β based on the nature of the scene. It does so by focusing the GES components on low pass signals first during the training before focusing on high frequency with tuning β from their initial values. This helps the *efficiency* of GES as can be seen later in Table 2 (almost free 9% reduction in memory).

Due to DoG filter sensitivity for high-frequencies, the mask for $0\% < \omega \leq 50\%$ is defined as $1 - M_\omega$ of $50\% < \omega \leq 100\%$. This ensures that all parts of the image will be covered by one of the masks M_ω, while focusing on the details more as the optimization progresses.

4.4. Optimization of GES

We detail a novel approach for controlling shape density, which selectively prunes GES according to their shape attributes, thus eliminating the need for a variable density mechanism. This optimization strategy encompasses the β parameter as well as the splat's position $\mathbf{x}$, opacity κ, covariance matrix $\mathbf{\Sigma}$, and color representation through spherical harmonics coefficients [18]. Optimization of these elements is conducted using stochastic gradient descent, with the process accelerated by GPU-powered computation and specialized CUDA kernels.

Starting estimates for $\mathbf{\Sigma}$ and $\mathbf{x}$ are deduced from the SfM points, while all β values are initialized with $\beta = 2$ (pure Gaussian spalts). The loss function integrates an $\mathcal{L}_1$ metric combined with a structural similarity loss (SSIM), and the frequency-modulated loss$\mathcal{L}_\omega$:

$$\mathcal{L} = \lambda_{\text{L1}}\mathcal{L}_1 + \lambda_{\text{ssim}}\mathcal{L}_{\text{ssim}} + \lambda_\omega\mathcal{L}_\omega, \tag{8}$$

where $\lambda_{\text{ssim}} = 0.2$ is applied uniformly in all evaluations, and $\lambda_{\text{L1}} = 1 - \lambda_{\text{ssim}} - \lambda_\omega$. Expanded details on the learning algorithm and other specific procedural elements are available in *Supplementary Material*.

5. Experiments

5.1. Datasets and Metrics

In our experiments, we utilized a diverse range of datasets to test the effectiveness of our algorithm in rendering real-world scenes. This evaluation encompassed 13 real scenes from various sources. We particularly focused on scenes from the Mip-Nerf360 dataset [4], renowned for its superior NeRF rendering quality, alongside select scenes from the Tanks & Temples dataset [20], and instances provided by Hedman et al. [13] for their work in Deep Blending. We follow the same metrics and procedures in Guassian Splatting [18].

5.2. Implementation Details of GES

Our methodology maintained consistent hyperparameter settings across all scenes, ensuring uniformity in our evaluations. We deployed an A6000 GPU for most of our tests. Our Generalized Exponential Splatting (GES) was implemented over 40,000 iterations, and the density gradient threshold is set to 0.0003. The learning rate for the shape parameter was set at 0.0015, with a shape reset interval of 1000 iterations and a shape pruning interval of 100 iterations. The other hyperparameters and design choices (like opacity splitting and pruning) shared with Gaussian splitting [18] were kept the same. More details are provided in *Supplementary Material*.

Figure 8. **Frequency-Modulated Loss Effect.** We show the effect of the frequency-modulated image loss $\mathcal{L}_\omega$ on the performance on novel views synthesis. Note how adding $\mathcal{L}_\omega$ improves the optimization in areas with large contrast or a smooth background.

Figure 9. **A Fair Visual Comparison.** We show an example of Gaussians [18] and GES when constrained to *the same number* of splatting components for a fair visual comparison. It clearly shows that GES can better model tiny and sharp edges for that scene.

Ablation Setup	$PSNR^{\uparrow}$	$SSIM^{\uparrow}$	$LPIPS^{\downarrow}$	**Size (MB)**$^{\downarrow}$
Gaussians	27.21	0.815	0.214	734
GES w/o shape reset	26.57	0.788	0.257	374
GES w/o $\mathcal{L}_\omega$ loss	27.07	0.800	0.250	411
Full GES	26.91	0.794	0.250	377

Table 2. **Ablation Study on Novel View Synthesis.** We study the impact of several components in GES on the reconstruction quality and file size in the Mip-NeRF360 dataset.

6. Results

6.1. Novel View Synthesis Results

We evaluated *GES* against several state-of-the-art techniques in both novel view synthesis tasks. Table 1 encapsulate the comparative results in addition to Fig.5. Table 1 demonstrates that *GES* achieves a balance between high fidelity and efficiency in novel view synthesis. Although it does not always surpass other methods in SSIM or PSNR, it significantly excels in memory usage and speed. With only 377MB of memory and a processing speed of 2 minutes, *GES* stands out as a highly efficient method, particularly when compared to the 3D Gaussians-30K and Instant NGP, which require substantially more memory or longer processing times. Overall, the results underscore *GES* 's capability to deliver balanced performance with remarkable efficiency, making it a viable option for real-time applications that demand both high-quality output and operational speed and memory efficiency.

Note that it is difficult to see the differences in *visual effects* between GES and Gaussians in Fig.5 since they have almost the same PSNR but a different file size (Table 1). For a fair visual comparison, we restrict the number of components to be roughly the same (by controlling the splitting of Gaussians) and show the results in Fig.9. It clearly shows that GES can model tiny and sharp edges for that scene better than Gaussians.

6.2. Ablation and analysis

Shape parameters. In Table 2, we explore the effect of important hyperparameters associated with the new shape parameter on novel view synthesis performance. Additional detailed analysis is provided in *Supplementary Material*.

Effect of frequency-modulated image loss. We study the effect of the frequency loss $\mathcal{L}_\omega$ introduced in Sec.4.3 on the performance by varying λ_ω. In table 2 and in Fig.8 we demonstrate how adding this $\mathcal{L}_\omega$ improves the optimization in areas where large contrast exists or where the smooth background is rendered and also improves the efficiency of GES. We notice that increasing λ_ω in GES indeed reduces the size of the file, but can affect the performance. We chose $\lambda_\omega = 0.5$ as a middle ground between improved performance and reduced file size.

7. Conclusion and discussion

This paper introduced *GES* (Generalized Exponential Splatting), a new technique for 3D scene modeling that improves upon Gaussian Splatting in memory efficiency and signal representation, particularly for high-frequency signals. Our empirical results demonstrate its efficacy in novel view synthesis with substantial efficiency improvement.

Limitation. One obvious limitation in our approach is that performance typically drops trying to make the representation as memor-efficient and as compact as possible. This is more noticeable for more complex scenes due to the pruning operations that depend on β-tuning. Removing many of the components can eventually drop the PSNR performance (Table 1 last 2 rows). Future research could focus on enhancing GES 's performance in more complex and dynamic environments and exploring its integration with other technologies in 3D modeling.

Acknowledgments. This work was supported by KAUST Ibn Rushd Postdoc Fellowship program.

References

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 54(10):105–112, 2011. 2

[2] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, pages 696–712. Springer, 2020. 2

[3] 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 Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5855–5864, 2021. 2

[4] 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 (CVPR)*, pages 5470–5479, June 2022. 2, 7

[5] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16123–16133, 2022. 2

[6] François Darmon, Bénédicte Bascle, Jean-Clément Devaux, Pascal Monasse, and Mathieu Aubry. Improving neural implicit surfaces geometry with patch warping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6260–6269, 2022. 2

[7] J Armando Dominguez-Molina, Graciela González-Farías, Ramón M Rodríguez-Dagnino, and ITESM Campus Monterrey. A practical procedure to estimate the shape parameter in the generalized gaussian distribution. *available through link https://www.cimat.mx/BiblioAdmin/RTAdmin/reportes/enlinea/I-01-18_eng.pdf*, 1, 2003. 3, 5

[8] Yilun Du, Cameron Smith, Ayush Tewari, and Vincent Sitzmann. Learning to render novel views from wide-baseline stereo pairs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[9] Olivier D Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig? In *Computer Vision—ECCV'92: Second European Conference on Computer Vision Santa Margherita Ligure, Italy, May 19–22, 1992 Proceedings 2*, pages 563–578. Springer, 1992. 2

[10] 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 (CVPR)*, pages 5501–5510, June 2022. 1

[11] Markus Gross and Hanspeter Pfister. *Point-based graphics*. Elsevier, 2011. 2

[12] Abdullah Hamdi, Bernard Ghanem, and Matthias Nießner. Sparf: Large-scale learning of 3d sparse radiance fields from few input images. *arxiv*, 2022. 1

[13] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Trans. on Graphics (TOG)*, 37(6), 2018. 7

[14] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2821–2830, 2018. 2

[15] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5885–5894, 2021. 2

[16] A.J. Jerri. The shannon sampling theorem—its various extensions and applications: A tutorial review. *Proceedings of the IEEE*, 65(11):1565–1596, 1977. 4

[17] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3907–3916, 2018. 2

[18] 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. 1, 2, 3, 4, 5, 6, 7, 8

[19] Mijeong Kim, Seonguk Seo, and Bohyung Han. Infonerf: Ray entropy minimization for few-shot neural volume rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12912–12921, 2022. 2

[20] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 36(4), 2017. 7

[21] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 6

[22] Hsueh-Ti Derek Liu, Michael Tao, and Alec Jacobson. Paparazzi: surface editing by way of multi-view image processing. *ACM Trans. Graph.*, 37(6):221–1, 2018. 2

[23] Ruoshi Liu, Sachit Menon, Chengzhi Mao, Dennis Park, Simon Stent, and Carl Vondrick. What you can reconstruct from a shadow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17059–17068, 2023. 2

[24] Ruoshi Liu and Carl Vondrick. Humans as light bulbs: 3d human reconstruction from thermal reflection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12531–12542, 2023. 2

[25] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7708–7717, 2019. 2

[26] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *arXiv preprint arXiv:1906.07751*, 2019. 2

[27] Matthew M Loper and Michael J Black. Opendr: An approximate differentiable renderer. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VII 13*, pages 154–169. Springer, 2014. 2

[28] David G Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60:91–110, 2004. 2

[29] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7210–7219, 2021. 1

[30] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 405–421. Springer, 2020. 1, 2, 5

[31] 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, July 2022. 1

[32] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 3

[33] Felix Petersen, Amit H Bermano, Oliver Deussen, and Daniel Cohen-Or. Pix2vex: Image-to-geometry reconstruction using a smooth differentiable renderer. *arXiv preprint arXiv:1903.11149*, 2019. 2

[34] 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 (CVPR)*, pages 10318–10327, 2021. 1

[35] Guocheng Qian, Jinjie Mai, Abdullah Hamdi, Jian Ren, Aliaksandr Siarohin, Bing Li, Hsin-Ying Lee, Ivan Skorokhodov, Peter Wonka, Sergey Tulyakov, and Bernard Ghanem. Magic123: One image to high-quality 3d object generation using both 2d and 3d diffusion priors. *arXiv preprint arXiv:2306.17843*, 2023. 1

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

[37] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4

[38] 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. 2

[39] Andrea Tagliasacchi and Ben Mildenhall. Volume rendering digest (for nerf). *arXiv preprint arXiv:2209.02417*, 2022. 2

[40] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5481–5490. IEEE, 2022. 2

[41] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2

[42] Yiqun Wang, Ivan Skorokhodov, and Peter Wonka. Hf-neus: Improved surface reconstruction using high-frequency details. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:1966–1978, 2022. 2

[43] 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

[44] 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

[45] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 767–783, 2018. 2

[46] Yao Yao, Zixin Luo, Shiwei Li, Tianwei Shen, Tian Fang, and Long Quan. Recurrent mvsnet for high-resolution multi-view stereo depth inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5525–5534, 2019. 2

[47] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:4805–4815, 2021. 2

[48] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:2492–2502, 2020. 2

[49] Wang Yifan, Felice Serena, Shihao Wu, Cengiz Öztireli, and Olga Sorkine-Hornung. Differentiable surface splatting for point-based geometry processing. *ACM Transactions on Graphics (TOG)*, 38(6):1–14, 2019. 2

[50] Alex Yu, Sara Fridovich-Keil, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. *arXiv preprint arXiv:2112.05131*, 2(3):6, 2021. 2

[51] 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 (ICCV)*, 2021. 1

[52] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4578–4587, 2021. 2

[53] Zehao Yu and Shenghua Gao. Fast-mvsnet: Sparse-to-dense multi-view stereo with learned propagation and gauss-newton refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1949–1958, 2020. 2

[54] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2

[55] Jason Zhang, Gengshan Yang, Shubham Tulsiani, and Deva Ramanan. Ners: neural reflectance surfaces for sparse-view 3d reconstruction in the wild. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:29835–29847, 2021. 2

[56] Qiang Zhang, Seung-Hwan Baek, Szymon Rusinkiewicz, and Felix Heide. Differentiable point-based radiance fields for efficient view synthesis. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–12, 2022. 2

[57] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. Ewa volume splatting. *Proceedings Visualization, 2001. VIS'01.*, pages 29–538, 2001. 4