Multiscale Tensor Decomposition and Rendering Equation Encoding for View Synthesis

Kang Han
James Cook University
kang.han@my.jcu.edu.au

Wei Xiang*
La Trobe University
w.xiang@latrobe.edu.au

Abstract

Rendering novel views from captured multi-view images has made considerable progress since the emergence of the neural radiance field. This paper aims to further advance the quality of view synthesis by proposing a novel approach dubbed the neural radiance feature field (NRFF). We first propose a multiscale tensor decomposition scheme to organize learnable features so as to represent scenes from coarse to fine scales. We demonstrate many benefits of the proposed multiscale representation, including more accurate scene shape and appearance reconstruction, and faster convergence compared with the single-scale representation. Instead of encoding view directions to model view-dependent effects, we further propose to encode the rendering equation in the feature space by employing the anisotropic spherical Gaussian mixture predicted from the proposed multiscale representation. The proposed NRFF improves state-of-the-art rendering results by over 1 dB in PSNR on both the NeRF and NSVF synthetic datasets. A significant improvement has also been observed on the real-world Tanks & Temples dataset. Code can be found at https://github.com/imkanghan/nrff.

1. Introduction

View synthesis aims to synthesize unrecorded views from multiple captured views using computer vision techniques. A great deal of effort has been made to solve this problem in the past few decades [29]. The recently proposed neural radiance field (NeRF) [18] made a breakthrough in this area by modeling a scene via a multilayer perceptron (MLP). The NeRF achieves an impressive photo-realistic view synthesis quality with 6 degrees of freedom for the first time. The NeRF also represents a scene in a very compact form. That is, only a small number of parameters in the MLP, whose size is even smaller than the captured images. However, this advantage in model size comes at the expense of extensive computations. Numerous evaluations of the MLP are required to render a single pixel, incurring a challenge for both training and testing.

*Corresponding author.

Representing a scene via learnable features is shown to be an effective alternative approach for photo-realistic view synthesis [6,7,19,25]. Several data structures are employed to efficiently organize learnable features to achieve compact representations. Multiresolution hash encoding (MHE) [19] and tensor decomposition in TensoRF [6] are two typical works in this direction. MHE organizes learnable features in multiresolution hash tables. As each hash table corresponds to a distinct grid resolution, a point is thus indexed into different positions of the hash tables to mitigate the negative effects of hash collisions. However, this structure breaks the local coherence in nature scenes, even though the spatial hash function in MHE preserves the coherence to some extent. By comparison, TensoRF decomposes a 3D tensor into 2D plane and 1D line tensors, where the local coherence is largely preserved. However, TensoRF's decomposition is performed only in a single scale, whereas multiscale methods are much more desirable for wide-ranging computer vision tasks [1, 14, 16, 26, 27]. We thus propose a multiscale tensor decomposition (MTD) method to represent scenes from coarse to fine scales. We show that the proposed MTD method is able to reconstruct more accurate scene shapes and appearances, and also converges faster than the single-scale TensoRF. As a result, the proposed MTD method achieves better view synthesis quality than TensoRF, even with fewer learnable features.

View direction encoding is the key to the success of neural rendering in modeling complex view-dependent effects. Frequency (or position encoding) [18] and spherical harmonics [30] are the two mostly used view direction encoding methods. The encoded feature vector of a view direction is then fed to an MLP to predict a view-dependent color. This approach models the 5D light field function (3D spatial position with 2D view direction) [13]. In computer graphics, the light field is usually modeled by the rendering equation [10], where the outgoing radiance is the interaction result of the incoming light at a point with a specific material. An accurate solution to the rendering equation involves Monte Carlo sampling and integration, which is computationally expensive, especially for the scenario of inverse rendering [9]. In this paper, we propose to encode the rendering equation in the feature space in lieu of the color space using the predicted anisotropic spherical Gaussian mixture. In this way, the following MLP is aware of the rendering equation so as to better model complex view-dependent effects. As we use both neural and learnable feature representations as well as the rendering equation encoding (REE) in the feature space, we dub the proposed method the neural radiance feature field (NRFF). In summary, we make the following contributions:

- We propose a novel multiscale tensor decomposition scheme to represent scenes from coarse to fine scales, enabling better rendering quality and faster convergence with fewer learnable features;
- In lieu of direct encoding of view directions, we propose to encode the rendering equation in the feature space to facilitate the modeling of view-dependent effects.

2. Related work

We divide view synthesis methods into neural and learnable feature representations depending on whether extra learnable parameters are used to represent a scene in addition to weights and biases in neural networks.

2.1. Neural representations

Neural representations mean representing a scene by neural networks, typically MLPs [18] or transformers [24]. Mildenhall *et al.* [18] first proposed this idea for view synthesis in the NeRF and achieved photo-realistic view synthesis results. The MLP in the NeRF is optimized to predict the volume density and the view-dependent appearance of a 3D spatial point observed from a given 2D view direction. Each component in this 5D input is encoded by a set of functions, *e.g.*, sine and cosine, with varying periods before being fed to the MLP. Such position or frequency encoding is one of the key factors to NeRF's success. The input encoding has been further explored in [28] by a neural tangent kernel and extended in mip-NeRF [2] to achieve anti-aliasing view synthesis. Neural representations have the advantage of representing a scene in a very compact form. MLPs are also used to predict the light source visibility of a point to enable relighting [23, 37]. However, these methods are computationally expensive because numerous evaluations of the networks are needed to render a single pixel.

Encoding view directions is important for neural representations to achieve photo-realistic view synthesis. Except for the aforementioned position encoding, spherical harmonics are also used to encode view directions with various frequency components [30, 35]. This approach composed of view direction encoding and the following MLP modeling is the dominant solution in the current neural rendering approaches. Such view direction encoding methods provide view direction information in various frequencies but neglect the rich information contained in the well-known rendering equation [10]. In this paper, instead of encoding view directions, we propose to encode the rendering equation to facilitate the learning of complex view-dependent effects for the following MLP.

2.2. Learnable feature representations

Learnable features are parameters that are also optimized by gradient descent in addition to weights and biases in neural networks. Learnable features are usually organized by the data structures of grids, sparse grids, trees, and hash tables. For a given input, interpolation is performed to obtain the corresponding features. The interpolated features can be directly interpreted as some properties, e.g., densities or colors, or optionally fed into neural networks to predict the designed outputs. Compared with pure neural representations, learnable feature representations are computationally efficient at the expense of memory footprint. As the features are also optimized for the considered scene, the task of inferring scene properties for the subsequent MLP is much easier in comparison with predicting from input coordinate encoding. As a result, with learnable feature representations, small MLPs are able to achieve a competitive rendering quality similar to pure neural representations.

Efficient data structures to arrange learnable features are crucial in terms of both computational cost and memory consumption. The 3D dense grid is a significant waste of memory because most of the voxels are empty. Its number of parameters increases by $\mathcal{O}(N^3)$. Thus, the 3D dense grid is only practical at low resolution, e.g., $N = 160$ in [25], limiting its rendering quality. The Octree [15, 35] and sparse 3D grid [7] are also employed but data structures need to be updated progressively. Because scene geometry only emerges during training. The recently proposed MHE [19] is a very compact learnable feature representation but hash collision and the break of spatial coherence limit its rendering quality. Concurrent tensor decomposition in TensoRF [6] preserves spatial coherence but is only performed at a single scale. The benefits of multiscale schemes [14, 16, 26] studied in the literature inspire us to propose the MTD scheme to represent scenes at varying scales.

3. Method

The proposed NRFF obtains the view-dependent color of a point through two main steps. For a point $\mathbf{x} = (x, y, z)$ sampled from a cast ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$, where $\mathbf{o}$ and $\mathbf{d}$ are the camera center and view direction, respectively, we

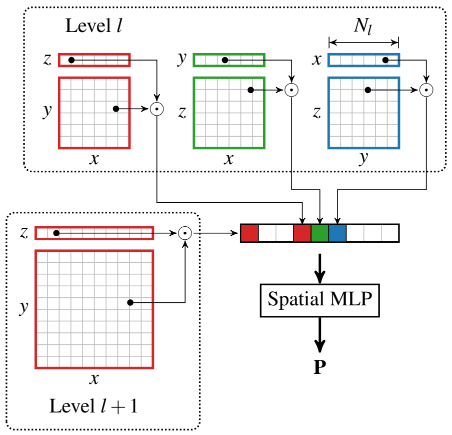

Figure 1. Multiscale tensor decomposition representation. At each level, a 3D tensor representation is decomposed to three sets of plane feature maps and line feature vectors. The resolution of decomposed tensors increases with the level, enabling scene representation at different scales. The concatenated feature vectors from all levels are used to predict parameters $\mathbf{P}$ by a spatial MLP.

first compute its feature vector from the proposed multiscale representation. The feature vector is fed into a spatial MLP to predict light parameters used to encode the rendering equation. Next, we apply the proposed REE and then use a directional MLP to predict the final color.

3.1. Multiscale tensor decomposition

We propose a multiscale tensor decomposition (MTD) scheme to represent a scene from coarse to fine scales. Similar multiscale ideas have been used in 3D shape representation [27], coarse-to-fine point rasterization [1] and many other computer vision works in the literature [14, 16, 26]. We start with a base resolution of N_{min} and progressively increase the level resolution to the maximum resolution of N_{max} by a factor b, in line with the strategy in MHE [19]:

$$N_l = \lfloor N_{\text{min}} b^l \rfloor \tag{1}$$

$$b = \exp\left(\frac{\ln N_{\text{max}} - \ln N_{\text{min}}}{L-1}\right) \tag{2}$$

where N_l is the resolution at level l and L is the number of multiscale levels. Feature vectors of point $\mathbf{x}$ are obtained from the proposed MTD independently at different levels. As shown in Fig. 1, we use the tensor decomposition mechanism [6] that decomposes a 3D tensor representation into three plane feature maps and three line feature vectors. We apply linear interpolation (bilinear interpolation for 2D) to the plane feature map $\mathbf{F}^l_{xy}$ and the feature vector $\mathbf{F}^l_z$ using the corresponding decomposed coordinates $\mathbf{x}_{xy}, \mathbf{x}_z$ to obtain the following two feature vectors:

$$\begin{aligned} \mathbf{f}^l_{xy} &= \text{Interp2D}(\mathbf{F}^l_{xy}, \mathbf{x}_{xy}) \\ \mathbf{f}^l_z &= \text{Interp1D}(\mathbf{F}^l_z, \mathbf{x}_z). \end{aligned} \tag{3}$$

The output feature vector at level l is obtained as follows:

$$\mathbf{f}^l_{xy,z} = \mathbf{f}^l_{xy} \odot \mathbf{f}^l_z \tag{4}$$

where $\odot$ denotes the element-wise multiplication. Feature vectors from other levels are obtained similarly. The output feature vectors $[..., \mathbf{f}^l_{xy,z}, \mathbf{f}^l_{xz,y}, \mathbf{f}^l_{yz,x}, \mathbf{f}^{l+1}_{xy,z}, ...]$ from all levels are concatenated and then fed into a spatial MLP to predict parameters $\mathbf{P}$, which will be detailed in Sec. 3.2.

The proposed multiscale scheme brings about three main benefits compared with the single-scale tensor decomposition in TensoRF [6]. First, it enables better exploration of the local smoothness of nature scenes at varying scales. Coarse-scale representations are inherently smooth, while fine-scale representations provide rich local details. It should be noted that the goal of the multiscale scheme here is different from that of MHE [19]. MHE uses multiresolution mainly for mitigating the negative effects of hash collisions as points are indexed to different positions in the hash tables at varying resolutions. Second, the number of feature channels at each scale could be significantly smaller than that in the single-scale representation, enabling high-resolution representations to explore richer details. For example, a multiscale representation with 16 levels, a maximum resolution of 512, and 4 feature channels has 8.5M parameters, which are fewer than 13M parameters in a single-scale TensoRF with a resolution of 300 and 48 feature channels. In Sec. 4.3, we show that even with fewer parameters, the proposed MTD method outperforms the single-scale TensoRF in terms of rendering quality. Third, scene geometry appears fast in coarse-scale representations, leading to faster convergence than the single-scale representation.

3.2. Rendering equation encoding

A light field can be defined as the radiance at a point in a given direction [13]. It is thus represented by a 5D function $L(\mathbf{x}, \boldsymbol{\omega}_o)$, where $\mathbf{x} \in \mathbb{R}^3$ is the spatial position and $\boldsymbol{\omega}_o \in \mathbb{R}^2$ (spherical coordinate) is the outgoing radiance direction. This 5D light field is the result of the interaction of the scene shape, material, and lighting, which is usually modeled by the rendering equation [10] consisting of the diffuse and specular components:

$$\begin{aligned} L(\boldsymbol{\omega}_o; \mathbf{x}) &= \mathbf{c}_d + \mathbf{s} \int_\Omega L_i(\boldsymbol{\omega}_i; \mathbf{x}) \rho_s(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \mathbf{x})(\mathbf{n} \cdot \boldsymbol{\omega}_i)\, d\boldsymbol{\omega}_i \\ &= \mathbf{c}_d + \mathbf{s} \int_\Omega f(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \mathbf{x}, \mathbf{n})\, d\boldsymbol{\omega}_i \end{aligned} \tag{5}$$

where $\mathbf{c}_d$ indicates the diffuse color and $\mathbf{s}$ is the weight of the specular color. Symbol $\cdot$ indicates the dot product in the Cartesian coordinate system. L_i is the incoming radiance from direction $\boldsymbol{\omega}_i$, and ρ_s represents the specular component of the spatially-varying bidirectional reflectance distribution function (BRDF). For ease of exposition, we define f as a function describing the outgoing radiance after the ray interaction. The integral is solved over the hemisphere Ω defined by the normal vector $\mathbf{n}$ at point $\mathbf{x}$. In computer graphics, $L_i, \rho_s, \mathbf{n}$ are usually known functions or parameters that describe scene lighting, material, and shape. An accurate solution to the rendering equation is achieved by computationally intensive Monte Carlo estimation in the color space, *e.g.*, computing the discrete summation by evaluating L_i, ρ_s at sampled $\boldsymbol{\omega}_i$ for a given $\boldsymbol{\omega}_o$.

In the inverse rendering problem, $L_i, \rho_s, \mathbf{n}$ are unknown functions or parameters. The most popular method in the inverse rendering to solve the equation is to treat it as a function of $\boldsymbol{\omega}_o$, and then employ an MLP to directly predict the integral result from the encoded $\boldsymbol{\omega}_o$. However, this simplification neglects the rich information described in the rendering equation and gives the MLP a complicated function to learn. Recent studies have also attempted to estimate the unknown properties to achieve relightable view rendering [3, 4, 17, 37]. But their rendering quality is inferior to methods [2,6,19,30] that focus only on view rendering with fixed lighting conditions. We instead propose to encode the rendering equation in the feature space and let the MLP predict the integrated color from the resultant encoding. By doing this, the following MLP is aware of the rendering equation, making the learning task much easier.

While encoding the rendering equation in the color space has a clear physical meaning, difficulties in three aspects limit its performance. First, the MLP yields the color parameters in the rendering equation by its final layer. Before the final layer, the MLP does not even know the outgoing radiance direction. This means the MLP does not benefit from the rendering equation as its input does not include information relevant to the rendering equation. Instead, the MLP only learns a spatial function of the position of the input point. Second, using the Monte Carlo integration technique to solve the rendering equation requires many samples to achieve a satisfactory quality in the color space, while extensive sampling is expensive in the inverse rendering problem [9, 37]. In the feature space, a feature vector consisting of a small number of sampled features could be a comprehensive representation. We show that 128 samples in the feature space are sufficient to render high-quality views. Last, in the color space, approximating the rendering equation by some basis functions (typically spherical Gaussians [31, 34] or spherical harmonics [22]) leads to a closed-form solution so that sampling over $\boldsymbol{\omega}_i$ can be avoided. However, for the inverse rendering problem, the parameters of the basis functions are unknown and predicted from the MLP. Deriving the final color using the computation (*e.g.*, the product of spherical harmonic coefficients [22]) of predicted parameters does not provide much additional useful information for the MLP.

We thus encode the rendering equation in the feature space by viewing f as a function of $\boldsymbol{\omega}_o$ for a sampled $\boldsymbol{\omega}_i$. In this perspective, we can apply a feature function to each sampled $\boldsymbol{\omega}_i$. We choose the basic function based on three considerations: 1) the function shape can be controlled by parameters such that each point can have its independent encoding; 2) the function can model all-frequency information (spherical harmonics are band-limited); 3) the function can be in diverse forms. Thus, we use the anisotropic spherical Gaussian (ASG) [34] to encode the rendering equation:

$$\mathbf{c}_s'(\boldsymbol{\omega}_o;\mathbf{x}) = \sum_{i=0}^{N-1} G_i(\boldsymbol{\omega}_o;\mathbf{x},[\boldsymbol{\omega}_i,\boldsymbol{\omega}_i^\lambda,\boldsymbol{\omega}_i^\mu],[\lambda_i,\mu_i],\mathbf{a}_i) \quad (6)$$
$$= \sum_{i=0}^{N-1} \mathbf{a}_i S(\boldsymbol{\omega}_o;\boldsymbol{\omega}_i)\exp\left(-\lambda_i(\boldsymbol{\omega}_o\cdot\boldsymbol{\omega}_i^\lambda)^2 - \mu_i(\boldsymbol{\omega}_o\cdot\boldsymbol{\omega}_i^\mu)^2\right)$$

where $\mathbf{c}_s'(\boldsymbol{\omega}_o;\mathbf{x})$ is a feature representation of the specular integral in Eq. (5); $\mathbf{a}_i$ is a feature vector; $[\boldsymbol{\omega}_i,\boldsymbol{\omega}_i^\lambda,\boldsymbol{\omega}_i^\mu]$ (lobe, tangent and bi-tangent) are predefined orthonormal axes satisfying $\boldsymbol{\omega}_i\cdot\boldsymbol{\omega}_i^\lambda = \boldsymbol{\omega}_i\cdot\boldsymbol{\omega}_i^\mu = \boldsymbol{\omega}_i^\lambda\cdot\boldsymbol{\omega}_i^\mu = 0$; $\lambda_i, \mu_i > 0$ are the bandwidths for $\boldsymbol{\omega}_i^\lambda, \boldsymbol{\omega}_i^\mu$ axes, controlling the shape of the ASG function; $S(\boldsymbol{\omega}_o;\boldsymbol{\omega}_i) = \max(\boldsymbol{\omega}_o\cdot\boldsymbol{\omega}_i, 0)$ is a smooth term. G_i is thus a function of $\boldsymbol{\omega}_o$ defined at the sampled $\boldsymbol{\omega}_i$.

A problem of using $\boldsymbol{\omega}_o = -\mathbf{d}$ to encode the rendering equation is that the ASG does not match the behavior of physical specular reflection. According to the law of reflection, the most significant energy from an incoming radiance in direction $\boldsymbol{\omega}_i$ is in the area centered at the reflective direction defined to have the same angle to the surface normal as the incoming radiance, but on the opposite side [8]. However, the energy centers of the ASG functions are in the sampled incoming radiance directions $\boldsymbol{\omega}_i$. We tackle this problem by reparameterizing the view direction to the opposite reflective direction, and treat the reparameterized direction as the outgoing radiance direction $\boldsymbol{\omega}_o$:

$$\boldsymbol{\omega}_o = 2(\mathbf{d}\cdot\mathbf{n})\mathbf{n} - \mathbf{d}. \quad (7)$$

After reparameterization, the REE matches the physical specular reflection behavior as $\boldsymbol{\omega}_o$ aligns with $\boldsymbol{\omega}_i$. This reparameterization has also been shown to be able to simplify view interpolation, as studied in [30, 33].

As depicted in Fig. 2, we sample $N = 8 \times 16$ lobes on a unit sphere and determine tangent and bi-tangent axes according to their orthonormal constraint. For a sampled $\boldsymbol{\omega}_i = (\theta, \phi)$ in the spherical coordinate system, we define $\boldsymbol{\omega}_i^\lambda = (\theta + \pi/2)$ and rotate $\boldsymbol{\omega}_i^\lambda$ around $\boldsymbol{\omega}_i$ by $\pi/2$ using the quaternion operation to obtain $\boldsymbol{\omega}_i^\mu$. Two ASG examples in Fig. 2 show that such ASGs have a strong representation ability to model the rendering equation in the feature space. Simply solving Eq. (6) by computing the sum of encoded feature vectors greatly reduces the channels of the feature representation, limiting its representative ability. Instead,

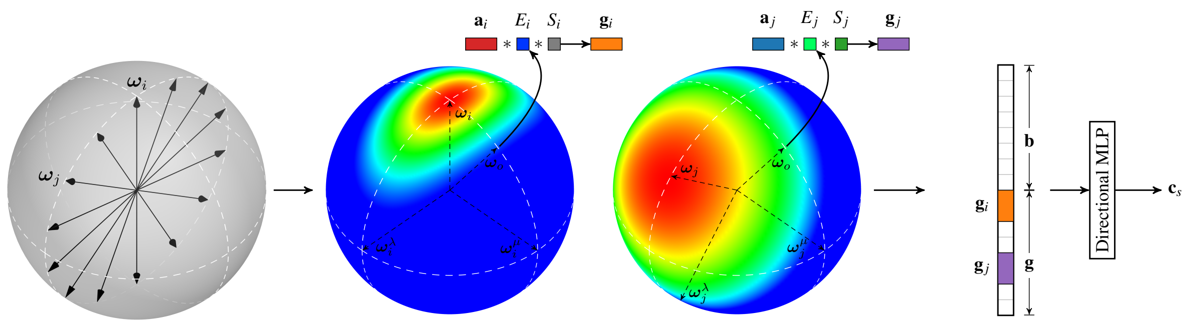

Figure 2. Illustration of the proposed rendering equation encoding. The rendering equation is encoded in the feature space by the learned ASG mixture with predefined orthonormal axes. The axes are defined by a set of radiance directions uniformly sampled on a unit sphere. Here only sampled $\boldsymbol{\omega}$ on a plane are shown for better visualization. For a sampled $\boldsymbol{\omega}_i$, an ASG function of the reparameterized view direction $\boldsymbol{\omega}_o$ is employed to determine the feature response $\mathbf{g}_i$. E_i and S_i are exponential and smooth terms, respectively, and $*$ denotes multiplication. Each ASG function is controlled by learned bandwidths λ and μ. The encoded feature vector $\mathbf{g}$ along with a bottleneck feature vector $\mathbf{b}$ depending only on the spatial position, are fed into a directional MLP to predict the specular color $\mathbf{c}_s$.

we form a comprehensive feature vector $\mathbf{g}$ by concatenating the encoded feature vectors:

$$\mathbf{g} = [\mathbf{g}_0, \mathbf{g}_1, ..., \mathbf{g}_{N-1}]. \tag{8}$$

Together with a spatial bottleneck feature vector $\mathbf{b}$, we apply a directional MLP to predict the specular color $\mathbf{c}_s$. The required parameters for the ASG encoding are from $\mathbf{P}$, which are predicted by the spatial MLP from the spatial feature vector obtained using Eq. (4) as aforementioned in Sec. 3.1. In summary, $\mathbf{P}$ include the following parameters: $\{\mathbf{c}_d, \mathbf{s}, \mathbf{n}, \mathbf{b}, \mathbf{a}_i, \lambda_i, \mu_i\}$. Finally, we apply the sigmoid function to obtain the final color:

$$\mathbf{c} = \text{Sigmoid}(\mathbf{c}_d + \mathbf{s} \odot \mathbf{c}_s). \tag{9}$$

We can also interpret the proposed REE as a more advanced view direction encoding method. Our REE has two-fold benefits compared with popular frequency encoding [18] and sphere harmonics [30]. First, every point now has its independent encoding functions controlled by the predicted bandwidths in the ASGs, while the encoding functions are fixed for all points in existing works. Second, a diverse of ASG functions can be produced to achieve much richer encoding compared with a few fixed basis encoding functions in existing methods [18]. In addition, the proposed method can also be seen as an underlying technique to model the surface light field. Our method is more accurate and compact than traditional surface light field methods [33], and provides richer information to the downstream networks compared with recent neural methods that use frequency encoding [20] or raw view direction [5].

3.3. Volume rendering

We use the differentiable volume rendering technique [18] to render a ray according to predicted densities and view-dependent colors. The scene density and appearance fields are modeled separately by two MTD representations. For a point $\mathbf{x}_i$ sampled at depth t_i, its density σ_i is the result of the softplus activation of the sum of the feature vectors at all levels. The color of the considered point is obtained by the method described in Sec. 3.2. We compute the color composition weights based on densities as follows:

$$w_i = \exp\left(-\sum_{j=0}^{i-1} \sigma_j \Delta_j\right)(1 - \exp(-\sigma_i \Delta_i)) \tag{10}$$

where Δ is the sampling interval. We follow the method in TensoRF [6] that only computes the colors of sampled points whose weights are larger than a predefined threshold. This strategy is effective in reducing the computational cost and makes the appearance representation focus on meaningful points. The rendered pixel color $\hat{\mathbf{c}}$ is a weighted sum of the predicted colors:

$$\hat{\mathbf{c}} = \sum_{i=0}^{N-1} w_i \mathbf{c}_i. \tag{11}$$

3.4. Training loss

The training loss of the proposed method consists of the mean squared error of the rendered pixel value, a regularization term about the predicted surface normals [30], and a regularization term regarding the density features [6]. Mathematically, the training loss is written as:

$$\mathcal{L} = (\hat{\mathbf{c}} - \mathbf{c}_{gt}) + \alpha \frac{1}{N} \sum_{i=0}^{N-1} w_i \max(0, \mathbf{d} \cdot \mathbf{n}_i)^2 + \beta \frac{1}{M} \sum_{i=0}^{M-1} |\mathbf{F}_\sigma^i| \tag{12}$$

where $\mathbf{c}_{gt}$ is the ground truth color, N represents the number of samples in the cast ray, and M is the number of fea-

					NeRF Synthetic [18]			NSVF Synthetic [15]			Tanks & Temples [12]		
	#Features	#MLP	Batch size	Steps	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
NeRF [18]	N/A	1,191K	4096	300K	31.01	0.947	0.081	30.81	0.952	0.043*	25.78	0.864	0.198*
Mip-NeRF [2]	N/A	612K	4096	1M	33.09	0.961	0.043	-	-	-	-	-	-
Ref-NeRF [30]	N/A	902K	16384	250K	33.99	0.966	0.038	-	-	-	-	-	-
NSVF [15]	0.32∼3.2M	500K	8192	150K	31.75	0.953	0.047*	35.18	0.979	0.015*	28.48	0.901	0.155*
DVGO [25]	49M	22K	8192	30K	31.95	0.957	0.053	35.08	0.975	0.033	28.41	0.911	0.155
MHE [19]	12.6M	10K	4096	30K	33.18	-	-	-	-	-	-	-	-
TensoRF [6]	18.6M	36K	4096	30K	33.14	0.963	0.047	36.52	0.982	0.026	28.56	0.920	0.140
Ours	12.8M	549K	4096	30K	34.65	0.975	0.034	37.76	0.986	0.019	28.87	0.927	0.127
Ours	12.8M	549K	4096	60K	**35.02**	**0.977**	**0.031**	**38.25**	**0.988**	**0.017**	**29.05**	**0.931**	**0.119**

Table 1. Objective performance comparison. # denotes the number of learnable parameters. The LPIPS are evaluated using the VGG network, while * means results from the Alex network. Our LPIPS results with 60K training steps evaluated by the Alex network on the three datasets are 0.016, 0.007, and 0.092, respectively.

tures in the density field representation. The normal regularization term, i.e., the second term in the above equation, penalizes the densities which decrease along the ray. In other words, it encourages concentrated modeling of the scene surface. The third term is density regularization defined as the mean absolute value of all features, which encourages a sparse density field. α, β are loss weights to balance the impact of the two regularization terms, and we empirically use $\alpha = 0.3$ and $\beta = 0.0004$ for all experiments as in [6, 30].

4. Experiments

We implement the proposed method using PyTorch [21]. There are a total of 16 levels starting with a base resolution of 16 and growing to a maximum resolution of 512. The number of feature channels is 4 for the appearance field and 2 for the density field. The sizes of the bottleneck $\mathbf{b}$ and feature vector $\mathbf{a}_i$ are 128 and 2, respectively. The spatial MLP has 3 layers, while the directional one has 6 layers. All layers contain 256 hidden units and ReLU activation. We optimize the proposed model using the Adam algorithm [11] with a learning rate of 2e-3 for the MTDs and 1e-3 for two MLPs. The learning rates degrade log-linearly to 0.1 times their initial values.

We compare our method with methods based on both neural representations and learnable feature representations. The compared methods based on neural representations include NeRF [18], Mip-NeRF [2], and Ref-NeRF [30], while NSVF [15], DVGO [25], MHE [19], and TensoRF [6] belong to learnable feature representations. We evaluate the rendering quality of these methods using the PSNR, SSIM [32], and LPIPS [36]. Two synthetic datasets, namely the NeRF synthetic [18] and NSVF synthetic [15] datasets, and one real-world Tanks & Temples dataset [12] are used for evaluation. Model details including the number of parameters of learnable features and MLPs, batch size, and training steps are also presented for comparison.

4.1. Objective results

The proposed method significantly outperforms existing state-of-the-art view synthesis approaches as shown in Tab. 1. Over 1 dB improvement in PSNR has been observed on both the NeRF and NSVF synthetic datasets. Pure MLP-based methods are compact in representing a scene but are computationally expensive. Besides, they also require a large number of training steps to converge. For example, Ref-NeRF [30] takes 250K steps to converge when using a large batch size of 16384. Thanks to the proposed MTD and encoding the rendering equation in the feature space, we are able to use 12.8M learnable features, which is similar to that in MHE [19] and fewer than those in DVGO [25] and TensoRF [6], to achieve significantly better rendering quality than those compared methods. The proposed NRFF also outperforms the compared methods on the Tanks & Temples dataset, demonstrating the efficacy of our method in representing real-world scenes.

4.2. Subjective results

Subjective comparisons are presented in Fig. 3 to show that our method is able to recover accurate texture, specular surface, and geometry. For fair comparison, we use the results from TensoRF with decreased features (as detailed in the ablation study in Sec. 4.3) such that the model has a similar number of parameters in the learnable features and the MLP as ours. The comparison on scene *chair* in Fig. 3 shows that our method synthesizes sharper texture than TensoRF. This advantage stems from the high-resolution representation in our method, which provides rich local details for view synthesis. The rendered balls in the scene *materials* demonstrate the superiority of our REE method in modeling the specular surface compared with the position

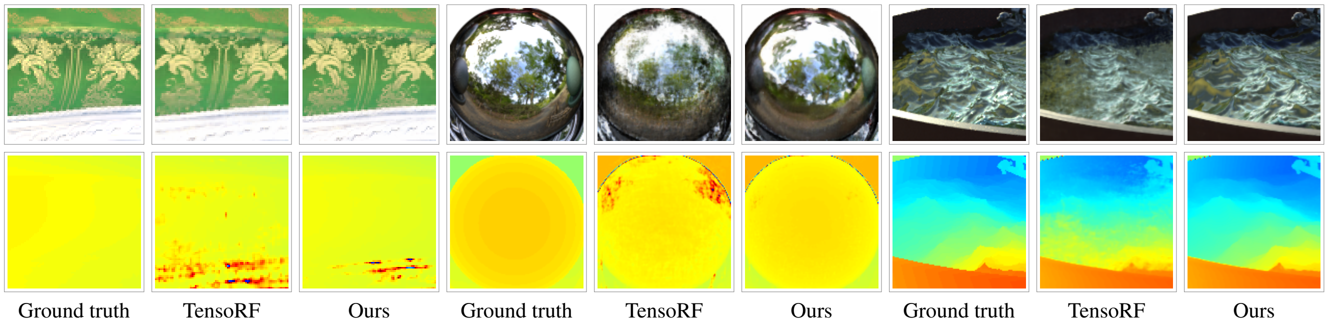

Figure 3. Subjective comparison of rendered views. The first row shows rendered novel views and the second row shows their corresponding depth maps. Our method recovers more accurate texture, specular surface, and geometry than TensoRF [6]. The scenes from left to right are *chair*, *materials*, and *ship* from the NeRF synthetic dataset [18].

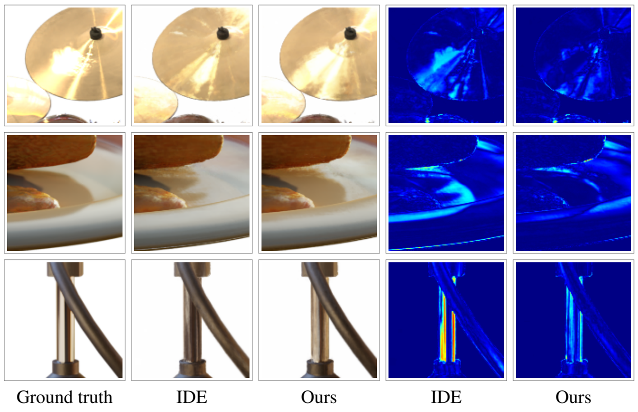

Figure 4. Visual comparison of two methods in modeling view-dependent effects. For fair comparison, all components are fixed except for the methods used to model view-dependent effects. From top to bottom: *drums*, *hotdog*, and *mic* from [18].

	#Feat.	#MLP	#L	Train	Test	PSNR↑	SSIM↑	LPIPS↓
TensoRF [6]	18.6M	36K	4	1.34h	0.99s	33.43	0.964	0.045
TensoRF, inc. MLP	18.6M	568K	10	2.02h	1.71s	34.10	0.970	0.038
TensoRF, dec. feat.	13.4M	557K	10	2.19h	1.72s	33.99	0.969	0.039
Ours, MTD, PE	12.8M	515K	9	3.32h	1.92s	34.58	0.975	0.034
Ours, MTD, IDE	12.8M	532K	9	3.52h	2.04s	34.61	0.974	0.034
Ours, MTD, color	12.8M	545K	9	3.36h	2.06s	33.53	0.965	0.043
Ours, full	12.8M	549K	9	3.39h	2.09s	**35.02**	**0.977**	**0.031**

Table 2. Ablation study on the NeRF synthetic dataset [18]. #L indicates the number of MLP layers. Training and testing times are averaged over all scenes and frames, respectively.

encoding of view directions employed in TensoRF [6]. Our multiscale representation also enables more accurate geometry reconstruction as shown in the depth map of the scene *ship*, resulting in more realistic view synthesis of the water surface. The visual comparison in Fig. 4 demonstrates that the proposed REE produces better reflection and illumination effects than the integrated directional encoding (IDE) proposed in Ref-NeRF [30]. Lastly, it is observed from Fig. 5 that our model yields a diverse of ASG functions to encode the rendering equation and reconstructs accurate light fields of scenes.

4.3. Ablation study

We investigate the effectiveness of the proposed modules in Tab. 2. We start with the single-scale TensoRF [6]. All reported results in this table are from models trained by 60K steps. Simply increasing the MLP's size in TensoRF to 10 layers greatly improves the rendering quality but also increases the training and testing times. This quality improvement highlights that both the learnable features and MLP are important for improving the rendering quality. When we decrease the number of learnable features in TensoRF to the same level as in our model, there is a small performance degradation (around 0.1 dB). As our encoding method produces a comparable larger encoding vector, for fair comparison, our models use MLPs with 9 layers to make the MLPs' parameters roughly consistent or fewer than that in TensoRF using 10 layers. Our MTD using the position encoding (PE) to encode view directions achieves better rendering quality than the single-scale TensoRF, even with fewer learnable features and a smaller MLP.

Further quality improvement is observed when using our MTD in conjunction with the proposed REE method (i.e., ours, full). As can be observed from Tab. 2 that our full model improves the PSNR from 34.58 dB (ours, MTD, PE) to 35.02 dB, yielding the state-of-the-art rendering quality. We also experiment on the IDE [30] using our MTD as the input coordinate encoding instead of integrated positional encoding in mip-NeRF [2]. We do not observe a significant performance improvement when using the IDE method. The model (ours, MTD, color) using the same form of the REE but in the color space performs poorly. This verifies the drawbacks of encoding the rendering equation in the color space, as discussed in Sec. 3.2.

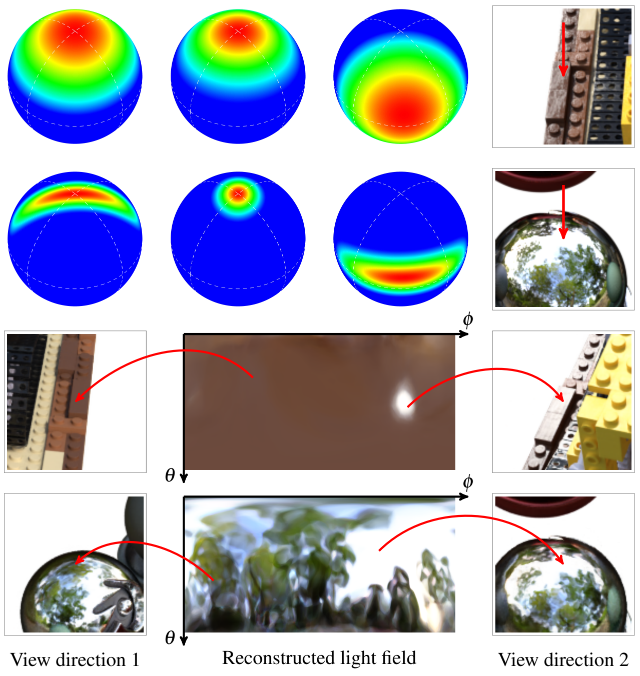

Figure 5. Visualization of the learned ASG functions and reconstructed light fields on the scenes of *lego* and *materials*. The first and second rows show the learned ASG functions used to encode the rendering equation at two points, which are on the rays cast from the pixels' position indicated by the red arrows in the rightmost image patches in the first two rows. The two points have their independent and diverse ASG functions. Our model produces more complex ASG functions on the specular surface (second row) to model the complex reflections. The reconstructed light fields and rendered images at different view directions imply successful modeling of complex view-dependent effects.

A detailed performance evaluation over training steps for varying the number of scale levels in Fig. 6 demonstrates the benefits of the proposed MTD scheme. For each setup, we adjust the number of feature channels and the maximum resolution to keep roughly the same number of parameters (12.8M) in the learnable features. As shown in Fig. 6, the model with two levels trained by 30K steps already surpasses that with one level trained by 60K steps, suggesting faster convergence speed of the multiscale representation than its single-scale counterpart. 1 dB improvement of the final PSNR is observed (from 31.3 dB with $L = 1$ to 32.3 dB with $L = 2$) when we have the two-level representation. Nearly 1 dB additional performance gain (from 32.3 dB with $L = 2$ to 33.2 dB with $L = 16$) becomes observable when increasing the number of levels to 16.

4.4. Limitations

Our method use a comparable large MLP than popular methods [6, 19] with learnable features. Representations with more scale levels introduce extra computations for interpolation weights compared with single-scale representation. On the NeRF synthetic dataset, training takes 3∼4 hours for each scene on one Nvidia Tesla V100 with 32 GB memory, and rendering an image of resolution 800×800 requires 2∼3 seconds. When using MLPs with similar size, the training time of our method is longer than the single-scale TensoRF but our rendering time is only slightly increased as presented in Tab. 2. The speed of the proposed method is slower than fast methods [19, 25], but faster than pure MLP methods [2, 18, 30]. We believe thorough optimization could overcome this limitation to some extent, considering that the hash encoding in [19] is fast thanks to the highly efficient implementation even with trilinear interpolation. Besides, in the testing stage, the plane feature maps could also be loaded to GPU texture memory to leverage hardware accelerated bilinear interpolation to fetch features more efficiently.

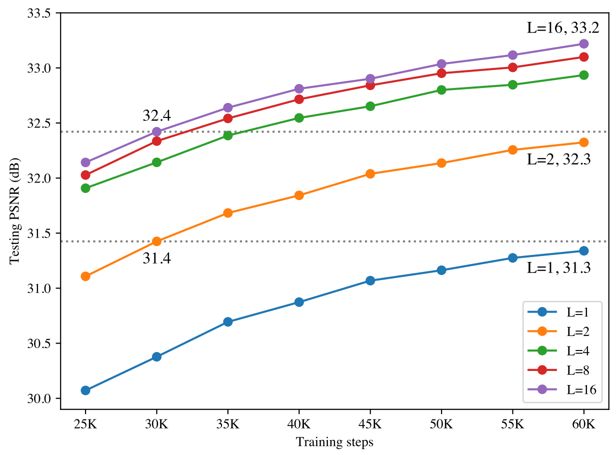

Figure 6. Performance comparison over training steps for varying the number of scale levels on scene *ship* from [18]. L indicates the number of levels. All models with different levels have roughly the same number of learnable features. Models with more levels not only converge faster but yield better final PSNRs.

5. Conclusion

We proposed the novel neural radiance feature field (NRFF) to achieve photo-realistic view synthesis. The proposed multiscale tensor decomposition scheme represents scenes from coarse to fine scales, leading to faster convergence and a better rendering quality than the single-scale tensor decomposition. Our proposed rendering equation encoding in the feature space provides more knowledge about the outgoing radiance to the MLP and overcomes the limitations of encoding the rendering equation in the color space. Extensive experimental results were presented to demonstrate the efficacy of the proposed NRFF on both the synthetic and real-world datasets.

References

[1] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *ECCV*, pages 696–712, 2020. 1, 3

[2] 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 *ICCV*, pages 5855–5864, 2021. 2, 4, 6, 7, 8

[3] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. NeRD: Neural reflectance decomposition from image collections. In *ICCV*, pages 12684–12694, 2021. 4

[4] Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan Barron, and Hendrik Lensch. Neural-PIL: Neural pre-integrated lighting for reflectance decomposition. *NIPS*, 34:10691–10704, 2021. 4

[5] Anpei Chen, Minye Wu, Yingliang Zhang, Nianyi Li, Jie Lu, Shenghua Gao, and Jingyi Yu. Deep surface light fields. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 1(1):1–17, 2018. 5

[6] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. TensoRF: Tensorial radiance fields. *ECCV*, 2022. 1, 2, 3, 4, 5, 6, 7, 8

[7] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, pages 5501–5510, 2022. 1, 2

[8] Eric Haines. Reflection and refraction formulas. In *Ray Tracing Gems II*, pages 105–108. Springer, 2021. 4

[9] Jon Hasselgren, Nikolai Hofmann, and Jacob Munkberg. Shape, light & material decomposition from images using monte carlo rendering and denoising. *arXiv preprint arXiv:2206.03380*, 2022. 2, 4

[10] James T Kajiya. The rendering equation. In *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*, pages 143–150, 1986. 1, 2, 3

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

[12] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017. 6

[13] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pages 31–42, 1996. 1, 3

[14] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 2117–2125, 2017. 1, 2, 3

[15] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *NIPS*, 33:15651–15663, 2020. 2, 6

[16] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 10012–10022, 2021. 1, 2, 3

[17] Linjie Lyu, Ayush Tewari, Thomas Leimkühler, Marc Habermann, and Christian Theobalt. Neural radiance transfer fields for relightable novel-view synthesis with global illumination. In *ECCV*, page 153–169, 2022. 4

[18] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, pages 405–421, 2020. 1, 2, 5, 6, 7, 8

[19] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (TOG)*, 41(4):102:1–102:15, July 2022. 1, 2, 3, 4, 6, 8

[20] Michael Oechsle, Michael Niemeyer, Christian Reiser, Lars Mescheder, Thilo Strauss, and Andreas Geiger. Learning implicit surface light fields. In *2020 International Conference on 3D Vision (3DV)*, pages 452–462. IEEE, 2020. 5

[21] 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. *NIPS*, 32, 2019. 6

[22] Ravi Ramamoorthi. Modeling illumination variation with spherical harmonics. *Face Processing: Advanced Modeling Methods*, pages 385–424, 2006. 4

[23] Pratul P Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *CVPR*, pages 7495–7504, 2021. 2

[24] Mohammed Suhail, Carlos Esteves, Leonid Sigal, and Ameesh Makadia. Light field neural rendering. In *CVPR*, pages 8269–8279, 2022. 2

[25] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *CVPR*, pages 5459–5469, 2022. 1, 2, 6, 8

[26] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, pages 5693–5703, 2019. 1, 2, 3

[27] 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 *CVPR*, pages 11358–11367, 2021. 1, 3

[28] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *NIPS*, pages 7537–7547, 2020. 2

[29] Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, W Yifan, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, et al. Advances in neural rendering. *Computer Graphics Forum*, 41(2):703–735, 2022. 1

[30] 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 *CVPR*, pages 5491–5500, 2022. 1, 2, 4, 5, 6, 7, 8

[31] Jiaping Wang, Peiran Ren, Minmin Gong, John Snyder, and Baining Guo. All-frequency rendering of dynamic, spatially-varying reflectance. In *ACM SIGGRAPH Asia*, pages 1–10, 2009. 4

[32] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 6

[33] Daniel N Wood, Daniel I Azuma, Ken Aldinger, Brian Curless, Tom Duchamp, David H Salesin, and Werner Stuetzle. Surface light fields for 3D photography. In *Proceedings of the 27th Nnnual Conference on Computer Graphics and Interactive Techniques*, pages 287–296, 2000. 4, 5

[34] Kun Xu, Wei-Lun Sun, Zhao Dong, Dan-Yong Zhao, Run-Dong Wu, and Shi-Min Hu. Anisotropic spherical gaussians. *ACM Transactions on Graphics (TOG)*, 32(6):1–11, 2013. 4

[35] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. PlenOctrees for real-time rendering of neural radiance fields. In *ICCV*, pages 5752–5761, 2021. 2

[36] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, pages 586–595, 2018. 6

[37] Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, Paul Debevec, William T Freeman, and Jonathan T Barron. NeRFactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics (TOG)*, 40(6):1–18, 2021. 2, 4