Efficient View Synthesis with Neural Radiance Distribution Field

Yushuang Wu[1,2*] Xiao Li[3†] Jinglu Wang[3] Xiaoguang Han[2,1†] Shuguang Cui[2,1] Yan Lu[3]
[1]FNii, CUHKSZ [2]SSE, CUHKSZ [3]Microsoft Research Asia
yushuangwu@link.cuhk.edu.cn {xili11, jinglwa, yanlu}@microsoft.com
{shuguangcui, hanxiaoguang}@cuhk.edu.cn

Abstract

Recent work on Neural Radiance Fields (NeRF) has demonstrated significant advances in high-quality view synthesis. A major limitation of NeRF is its low rendering efficiency due to the need for multiple network forwardings to render a single pixel. Existing methods to improve NeRF either reduce the number of required samples or optimize the implementation to accelerate the network forwarding. Despite these efforts, the problem of multiple sampling persists due to the intrinsic representation of radiance fields. In contrast, Neural Light Fields (NeLF) reduce the computation cost of NeRF by querying only one single network forwarding per pixel. To achieve a close visual quality to NeRF, existing NeLF methods require significantly larger network capacities which limits their rendering efficiency in practice. In this work, we propose a new representation called Neural Radiance Distribution Field (NeRDF) that targets efficient view synthesis in real-time. Specifically, we use a small network similar to NeRF while preserving the rendering speed with a single network forwarding per pixel as in NeLF. The key is to model the radiance distribution along each ray with frequency basis and predict frequency weights using the network. Pixel values are then computed via volume rendering on radiance distributions. Experiments show that our proposed method offers a better trade-off among speed, quality, and network size than existing methods: we achieve a ∼254× speed-up over NeRF with similar network size, with only a marginal performance decline. Our project page is at yushuang-wu.github.io/NeRDF.

1. Introduction

The problem of digitally representing a 3D scene for novel view synthesis from arbitrary directions is an important research topic with many applications, ranging from immersive conferencing to augmented reality. The breakthroughs made by the pioneering work of NeRF [23] demonstrate considerable advancements in the view synthesis field, as it represents 3D scenes using implicit radiance fields modeled via neural networks. Despite these advances, a significant drawback of NeRF is its computationally intensive nature, requiring hundreds of network evaluations per pixel, resulting in slow rendering speed (*e.g.* ∼0.2 FPS on a high-end GPU). Subsequent research has aimed to enhance the rendering speed by improving importance sampling strategies, reducing the number of samples, or optimizing the code implementation. However, these do not address the fundamental problem of multiple sampling inherent in radiance fields, which limits the extent to which rendering speed can be improved (usually 5-30 FPS contingent on implementation) at the cost of increased memory requirements and additional implementation efforts. To achieve efficient view synthesis, the research community is exploring alternative representations such as neural light fields (NeLF) [38, 22, 41, 35, 1]. A NeLF maps rays directly into the RGB space, predicting pixel color based on the ray parameters (*e.g.* the origin and direction), thereby reducing the intrinsic computational complexity to one single network forwarding per pixel. Recent advances have indicated that the synthesis quality of NeLF can be comparable to that of NeRF. However, this usually comes at the cost of much larger networks - for instance, R2L [41] utilizes an 88-layer MLP network that is 11× larger than NeRF. The increased size of networks used in NeLF methods results in significantly higher computational and memory costs, as well as limited rendering efficiency in practice.

In this paper, we examine the challenge of achieving efficient view synthesis in practical settings, which requires a careful balance among multiple considerations including perceptual quality (measured by PSNR), computational efficiency (measured by FPS), and memory requirements (*e.g.* model size). To achieve this goal, we start from a key observation: previous NeLF methods have no explicit perception to the 3D geometry information. As known, NeRF attends to the radiance information at spatial locations along each camera ray as illustrated in Fig. 1, which enables NeRF

∗ This work was done when Yushuang Wu was an intern at MSRA.
† Corresponding author.

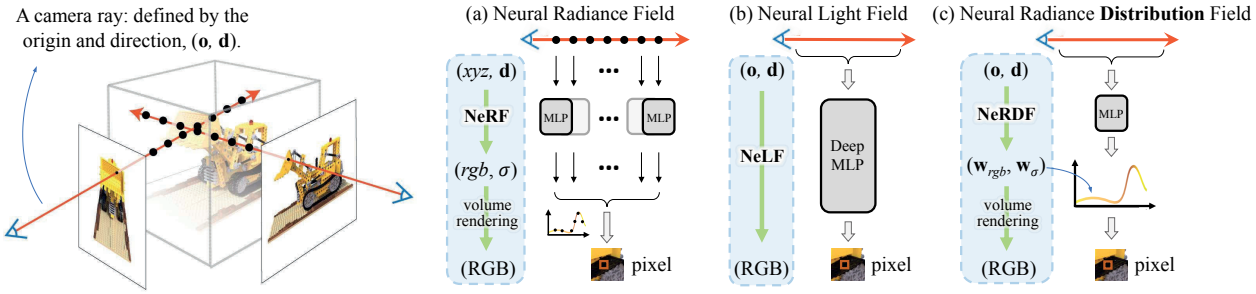

Figure 1. The overview of our Neural Radiance Distribution Field (NeRDF) and the comparison of (a) NeRF, (b) NeLF, and (c) NeRDF. NeRF requires hundreds of network forwarding per ray to predict the volume density and color, and output the pixel RGB via volume rendering. NeLF takes only one single forwarding per ray to predict the pixel RGB but strongly depends on a much larger network. Our NeRDF absorbs both advantages that takes only one single forwarding per ray as NeLF with only a small network as NeRF. The key idea is to directly predict the radiance distribution from the ray input.

to have a perception of the 3D geometry information of a scene. However, NeLF learns the direct mapping from the huge ray space to the pixel RGB space without any 3D prior. This paradigm hinders NeLF from learning the actual intrinsic 3D layout/structure of a scene, which results in an over-dependence on a large network capacity, as previous NeLF methods [1, 41] suffer from.

Based on this observation, we propose a novel implicit representation, termed as **Neural Radiance Distribution Field (NeRDF)**. NeRDF is based on a simple yet effective key idea: from the input ray space as in NeLF, NeRDF yet learns the radiance distribution of a given ray, so that the spatial 3D geometry information can be perceived. Specifically, we train the network to produce the radiance distribution along a ray, parameterized using a set of trigonometric functions. The final pixel color is re-synthesized via volume rendering from the output radiance distribution as in NeRF. Thus, the proposed NeRDF combines the strengths of both NeRF-based and NeLF-based methods. NeRDF models the parametric ray radiance distribution in order for a significantly more compact target space than direct modeling the ray-to-pixel mapping. This enables NeRDF to represent scenes with much smaller neural networks that are comparable to those in NeRF. Additionally, the prediction of the radiance distribution of a ray only takes one single network forwarding, so only one evaluation is required to render a pixel as in NeLF.

The learning of NeRDF is based on a knowledge distillation framework inspired by [41], where a teacher NeRF synthesizes dense and diverse views that then serve as the pseudo-training data. We further contribute three novel designs to enhance the framework: (i) an input ray encoding method that captures rich ray information, (ii) an online view sampling strategy that expands the diversity of the pseudo-training data, and (iii) a volume density constraint loss that promotes the learning of a strong 3D prior.

We have evaluated the efficiency of the NeRDF method on the Real Forward-Facing (LLFF) dataset [23]. On average, without any specific optimization in network inference, our proposed NeRDF method achieves a comparable visual quality to existing methods while rendering at a speed of ∼21 FPS and using only an 8-layer MLP network. This is ∼5× faster than the R2L method [41] and ∼100× faster than the original NeRF. Additionally, our method can benefit from any off-the-shelf inference optimization methods. By employing tiny-cuda-nn [25] as our inference backend, our NeRDF achieves a rendering speed of ∼369 FPS, which is a ∼1400× speed-up over an unoptimized NeRF and a 10-15× speed-up compared with previous methods using the same backend. We also demonstrate that the NeRDF method provides a superior trade-off between speed, memory, and visual quality compared with previous NeRF-based and NeLF-based methods. Our contributions are as follows:

- A new neural representation for 3D scenes, Neural Radiance Distribution Field (NeRDF), that outputs radiance distribution along rays.
- A method for NeRDF learning with a compact neural network that has high rendering speed, low memory cost, and plausible quality.
- An efficient view synthesis solution that has a good trade-off among visual quality, speed, and memory.

2. Related Work

Neural 3D representations. Recently, the neural field representation of 3D scenes has attracted significant attention from the literature [23, 36, 21, 43]. These techniques utilize multi-layer perceptrons to generate implicit fields such as sign distance functions or volume radiance fields. Of particular note is the Neural Radiance Fields (NeRF) [23, 3, 4]. NeRF has demonstrated its effectiveness on the task of view synthesis from a limited number of input views, leading to an explosion number of follow-up works that extend its capabilities to other tasks. These include the handling of dynamic scenes [10, 46, 27, 33, 28], human digitization [39, 14, 37, 31, 20], shape and appearance modeling [9, 17, 51, 5], 3D-aware synthesis [6, 8], and many others. A comprehensive overview of NeRF and its related applications can be found in [47]. Our method aims to train a succinct neural representation, which will allow

Table 1. Comparison between different neural representations. As show, previous methods struggle at fulfilling low memory, high speed, and high quality at the same time. Our method breaks this impossible trinity.

Methods	NeRF-based [49, 11, 26, 16, 25, 24]		NeLF-based [41, 1, 44]		**NeRDF (Ours)**
	Hybrid NeRF	Accelerated Backend	Small Network	Large Network	
Visual Quality	High	High	Low	High	**High**
Rendering Speed	Medium	Medium	Fast	Slow	**Very Fast**
Memory Cost	Large	Low	Low	High	**Low**

for more efficient view synthesis and thereby benefit many of NeRF's applications.

Efficient view synthesis. To improve the rendering efficiency of NeRF, numerous existing works have reduced the number of queried samples along rays through the use of auxiliary data structures, such as octrees [19, 49], hashtables [24], the application of additional networks for importance sampling [18, 26, 11, 16], or pre-baking the radiance field [12]. These methods accelerate NeRF to some degree, but come with additional computational and memory costs. Concurrently, tiny-cuda-nn [25] attempts to increase the inference speed of shallow MLPs through dedicated optimization at the CUDA level. However, they do not address the fundamental issue of multiple sampling. An alternative to these approaches is the use of other neural representations, with the neural light field (NeLF) becoming a popular choice as it requires only a single evaluation per pixel [41, 1, 44]. While NeLF-based methods can produce high-quality view synthesis results comparable to NeRF, they also entail significant memory costs due to their deeper networks used [41] or additional networks to map rays into latent spaces [1]. Despite these high memory costs, their rendering efficiency still remains limited with larger networks. Our method focuses on efficient view synthesis with a straightforward design, carefully balancing visual quality, speed, and memory costs.

Knowledge distillation. Knowledge distillation (KD), as commonly used in the field of network compression, involves training a small model (known as the student) to mimic the outputs and intermediate feature representations of a larger pre-trained model (referred to as the teacher) [13, 7, 29, 50, 40]. In the context of view synthesis, the R2L method [41] exploits KD by transferring knowledge from a pre-trained teacher NeRF network to a student NeLF network, by matching the rendered pixel value between the two models [41]. Our proposed approach extends the knowledge distillation process outlined in [41] by incorporating an additional matching between the volume density distribution of the target NeRDF and the teacher NeRF.

3. Method

In this section, we first provide the preliminary of view synthesis based on neural implicit representations including NeRF and NeLF in Sec. 3.1. Next, we introduce our proposed representation of NeRDF in Sec. 3.2. Lastly, we detail the learning process of a compact NeRDF via distillation from a teacher NeRF in Sec. 3.3.

3.1. Preliminary

View synthesis with neural fields. Given sparse multi-view images as observations of a 3D scene, the objective is to train a neural network that implicitly captures the scene's structure, which allows for the synthesis when given an arbitrary, unseen view of the scene.

Neural Radiance Fields (NeRF). We begin by reviewing the fundamentals of Neural Radiance Fields (NeRF) as described in [23]. NeRF is a Multi-layer Perceptron (MLP) network that maps 3D coordinates of a point to its radiance and volume density values. Mathematically, this can be represented as the following function:

$$F_{\text{NeRF}} : (x, y, z, \theta, \phi) \rightarrow (\mathbf{c}, \sigma), \tag{1}$$

where (x, y, z) are the spatial coordinates, (θ, ϕ) represent the 2D viewing direction, $\mathbf{c} = (r, g, b)$ is the emitted color, and σ is the volume density. Given a camera ray defined by $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d} \in \mathbb{R}^3$, where $\mathbf{o}$ and $\mathbf{d}$ are the origin and direction of the ray respectively, the rendered color $C(\mathbf{r})$ can be calculated with volume rendering:

$$C(\mathbf{r}) = \int_0^\infty T(t)\, \sigma(\mathbf{r}(t))\, \mathbf{c}(\mathbf{r}(t), \mathbf{d}) dt, \tag{2}$$

where

$$T(t) = \exp\left(- \int_0^t \sigma(\mathbf{r}(t)) ds \right) \tag{3}$$

In practice, the integration of Eq. (2) is approximated discretely through Monte Carlo sampling of points along the camera ray. The network F_{NeRF} is queried at each of these points, providing predictions of $\sigma(\mathbf{r}(t))$ and $\mathbf{c}(\mathbf{r}(t), \mathbf{d})$ using Eq. (1). It is typical that a NeRF requires approximately 100-200 network queries to compute the RGB pixel color value $C(\mathbf{r})$ for a given camera ray $\mathbf{r}$.

Neural Light Fields (NeLF). To reduce computational complexity, a scene can be also represented as a Neural Light Field (NeLF). Different from NeRF, a NeLF directly maps a 4D-oriented camera ray to the RGB color space, as

described in [2]:

$$F_{\text{NeLF}} : \mathbf{r} \in \mathbb{R}^4 \rightarrow \mathbf{c} \in \mathbb{R}^3. \tag{4}$$

Thus, NeLF simplifies the rendering process by eliminating the need for analyzing 3D radiance information and conducting volume rendering, requiring only a single network evaluation to compute the pixel color of a given ray. However, the learning of a NeLF that produces high-quality synthesis greatly challenges the network capacity. For example, the state-of-the-art NeLF-based method R2L, described in [41], uses an 88-layer MLP network (11$\times$ deeper than in NeRF), resulting in a rendering speed of only $\sim$5 FPS. Reducing the depth of the neural network for R2L leads to severe quality degradation, as evidenced in the results of [41] and our observations as shown in Fig. 3.

3.2. Neural Radiance Distribution Field

Tab. 1 summarizes the challenge in achieving a balance among the quality, rendering efficiency, and memory cost for existing neural field methods. Our goal is to overcome this challenge and enable high-quality, real-time view synthesis with low memory overhead (network size).

3D prior. In the comparison between NeRF and NeLF, a key factor contributing to NeRF's ability to achieve high-quality view synthesis with a smaller network than NeLF is its reliance on a 3D prior. The multi-view consistency and 3D geometric information of a scene are naturally taken into account during perceiving the radiance information at each spatial location, resulting in a compact representation with small networks. However, this advantage comes at the cost of requiring numerous evaluations to compute the color of each pixel via volume rendering in Eq. (2). In contrast, NeLF directly maps from ray space to RGB space, without considering any spatial radiance information. This leads to a strongly view-dependent representation, with each pixel from different views being addressed independently without any 3D prior information. Therefore, larger networks are typically required to memorize all views for synthesis, as seen in [41].

Neural Radiance Distribution Field (NeRDF). Different from NeRF and NeLF, our proposed representation, the Neural Radiance Distribution Field (NeRDF), takes the camera ray of each pixel as input and maps it into the radiance distribution along that ray:

$$F_{\text{NeRDF}} : \mathbf{r} \rightarrow (\mathbf{w}^C, \mathbf{w}^\sigma), \tag{5}$$

where $\mathbf{w}^C$ and $\mathbf{w}^\sigma$ are a set of parameters that define the radiance opacity and color distributions for the given ray $\mathbf{r}$. The pixel color is then computed via volume rendering (as in Eq. (2)) by a dense sampling from the parametric radiance distributions.

Fig. 1 shows the overall pipeline of NeRF, NeLF, and our proposed NeRDF. Our proposed NeRDF had the following advantages: (i) As in NeRF, NeRDF is built upon the volume rendering which obtains the pixel color through the ray marching of radiance values. This gives the neural representation the ability to model the 3D prior, which significantly narrows down the vast target space of NeLF, allowing for a smaller network capacity required during the learning process. (ii) As in NeLF, the input of NeRDF is defined in the ray space. As a result, only a single network evaluation is required to output the full radiance distribution of a ray, reducing the computation cost of multiple queries.

Parameterize radiance distribution. The radiance opacity/color distribution along a ray is generally arbitrary. To parameterize it, we conduct discrete Fourier analysis by integrating pre-defined trigonometric functions with discrete frequencies. The opacity and color function of a ray, $\sigma(t)$ and $C(t)$, respectively, can be represented as follows:

$$\sigma(t) = \sum_{i=0}^{2K-1} w_i^\sigma \cdot \mathcal{T}_i(t), \tag{6}$$

$$C(t) = \sum_{i=0}^{2K-1} w_i^C \cdot \mathcal{T}_i(t), \tag{7}$$

where t represents the distance stamp along the ray, $w_i^{\sigma(C)}$ refers to the coefficients predicted by NeRDF as outlined in Eq. (5), K is the number of frequencies. The frequency basis $\mathcal{T}_i(t)$ is defined as:

$$\mathcal{T}_i(t) = \begin{cases} \cos(\frac{i\pi}{T}t) & \text{for even } i, \\ \sin(\frac{(i+1)\pi}{T}t) & \text{for odd } i. \end{cases} \tag{8}$$

We use an MLP network to predict $\mathbf{w}^\sigma$ and $\mathbf{w}^C$ for the volume density distribution and the radiance color distribution, respectively. In total, this leads to an output with $8K$ channels. As demonstrated in Fig. 2, we show an example of the volume density distribution of a ray produced by NeRF (left) and NeRDF (right). The results show that NeRDF reproduces a similar radiance distribution to NeRF's.

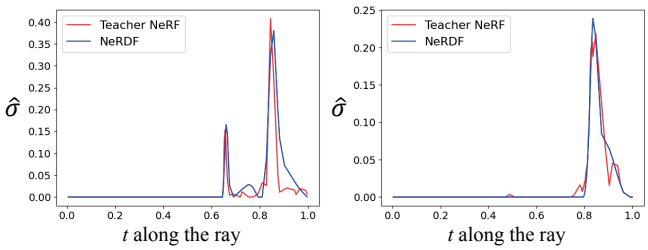

Figure 2. The radiance distribution (normalized opacity) predicted by NeRDF (blue) compared with NeRF (red). Overall, the output of NeRDF faithfully reproduces the distribution shape from NeRF.

3.3. NeRDF Learning

Given a set of multi-view posed images as training data, the proposed NeRDF is trained by minimizing an L_2 norm loss function between the synthesized RGB values, $\hat{C}(\mathbf{r})$, and the ground-truth RGB values, $C(\mathbf{r})$. The loss function is defined as follows:

$$L_{render} = \sum_{\mathbf{r}} ||\hat{C}(\mathbf{r}) - C(\mathbf{r})||_2^2 \tag{9}$$

The predicted RGB value, $\hat{C}(\mathbf{r})$, is calculated by performing two steps: (i) sampling multiple points along the ray and obtaining their opacity and color from the parametric radiance distribution using Eq. (6) and Eq. (7), and (ii) computing the volume integration with Eq. (2).

To learn a high-quality NeRDF representation with compact neural networks, there are still several technical challenges: (i) small changes in the input ray space can result in significant variations in the output RGB values, particularly at the edges of foreground objects. (ii) the training views are not sufficient to learn a NeRDF with an input defined in the ray space. (iii) there is a need to constrain the predicted radiance distributions of correlated rays to guarantee a valid 3D prior is learned. We address these challenges with the following novel designs.

Input ray encoding. To tackle the first challenge, we adopt a strategy of compound encoding to project the input into a higher-dimensional space, thereby incorporating more complex information about the rays. Given a ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$, we take into account the different impacts of the origin and direction on the viewed pixels, and encode $\mathbf{o}$ and $\mathbf{d}$ differently. Specifically, we utilize the typical frequency-based positional encoding method as in NeRF [23] for encoding the origin, and a spherical-harmonic (SH) based encoding inspired by [25, 42] for the direction, where the coefficients of the SH functions are used as the embedding vector of the direction. To include the path information of the input ray, we additionally sample N points along the ray $\mathbf{r}$. This is achieved by conducting a stratified sampling technique as described in [41]. Finally, we embed this $3N$-dimensional vector using the same frequency-based positional encoding technique as applied in [23] and concatenate it with the encoded origin and direction vector, forming the final input.

Online view sampling. To tackle the second challenge of only sparse views available for NeDRF learning, we utilize a teacher NeRF to synthesize pseudo data in the form of numerous views. In the recent work of R2L [41], a teacher NeRF is employed to pre-produce around 10,000 pseudo images with sufficient view coverage for offline training. However, we adopt a more efficient approach called online view sampling (OVS), which involves randomly sampling camera poses for pseudo-data generation during training. Besides the improved training efficiency, OVS also provides two additional benefits: (1) a wider range of sampled views can be obtained, and (2) not only the final pixel RGB but also the opacity and color information along the ray can be leveraged to improve the learning of 3D prior.

Volume density constraint. To alleviate the issue of inconsistent geometry across different views when individually fitting the radiance distribution along each ray, a volume density constraint (VDC) loss is introduced. The VDC loss minimizes the L_2 norm distance of the volume density at each point between two models, the student NeRDF and teacher NeRF:

$$L_{vdc} = \lambda \sum_{\mathbf{r}} \sum_{t} ||\hat{\sigma}_1(t) - \hat{\sigma}_2(\mathbf{r}(t))||_2^2, \tag{10}$$

where λ is the weight for balancing the VDC loss, $\hat{\sigma}_1(t)$ and $\hat{\sigma}_2(\mathbf{r}(t))$ denote the normalized volume density at the same point predicted by NeRDF (Eq. (6)) and NeRF, respectively. The proposed VDC not only ensures multi-view consistency in NeRDF, but also extends the knowledge distillation from using the final pixel RGB only to additionally involving the "intermediate" features, the volume density. This is in line with common practices in other knowledge distillation tasks for improving performance, as noted in [7].

Implementation details. The proposed method is implemented using the PyTorch framework [30]. An 8-layer MLP with a width of 384, unless specified otherwise, is used to build a NeRDF. The MLP network structure is similar to that of R2L [41], with skip connections added between neighboring layers. The parameter K in the Fourier trigonometric function approximation is set to 12. The volume rendering in Eq. (2) is approximated using 64 uniform samples from the radiance distribution, and the sampling process was implemented with Taichi [15]. The input ray encoding uses $N = 16$ and 8 degrees in the SH embedding to encode the ray direction $\mathbf{d}$ into a 64-dimensional vector. The positional encoding uses 10 frequencies, as in NeRF. The OVS training uses random sampling of 1 view origin and 2,048 directions to form a batch of training rays. The weight λ in the VDC loss of Eq. (10) is set to 0.1. The entire NeDRF training was conducted on a single RTX3090 GPU with a learning rate of 5×10^{-4} for 400k-600k iterations. Other implementation details are provided in supplementary materials.

4. Experiments

To demonstrate the advantages of our proposed NeRDF, we perform comparisons against multiple NeRF-based and NeLF-based efficient view synthesis methods in Sec. 4.2. We show that the proposed NeRDF leads to a better trade-off among speed, memory cost, and quality. We also perform a set of ablation studies in Sec. 4.3 to validate our proposed NeRDF as well as our training design.

4.1. Experiment Setup

Datasets. We conduct experiments on the Real Forward-Facing (LLFF) dataset [23]. The LLFF dataset consists of 8 scenes captured with a handheld cellphone. The number of images for each data ranges from 20 to 62. Following [41] and [18], we downsample all images into the resolution of 504×378 pixels in both training and test. Results on a higher resolution (1008×756) are provided in the supplementary materials.

Metrics. To better analyze the trade-off between different factors, we measure (1) the PSNR (dB) for evaluating the synthesis quality; (2) the network size (and the memory cost of additional data structure, if any) for evaluating the memory cost; (3) the running time in FPS for evaluating the rendering speed.

Baselines. We compare our method with the following efficient view synthesis methods based on NeRF: TermiNeRF [32], DONeRF [26], AdaNeRF [16], NeX [45], and Instant-NGP [24]. We also compare our method with NeLF-based methods including RSEN [1] and R2L [41]. We do not compare our method with baking-based methods such as [48, 12, 34], as they target speed-up NeRF by incurring a high memory cost. Our method, on the other hand, aims to keep the memory cost as low as possible.

4.2. Main Results

Comparison with NeLF-based methods. The quantitative results of our method and NeLF-based methods are listed in Tab. 2. These results are averaged over 8 scenes of the LLFF dataset and include synthesis quality, speed, and memory cost, evaluated on an RTX3090 GPU. Our method outperforms RSEN [1] in terms of rendering speed, achieving a speed-up of roughly 6-10 times while maintaining similar or superior visual quality. In terms of synthesis quality, both our method and R2L [41] produce high-quality results when using large networks (R2L-88 v.s. NeRDF-48). However, R2L struggles to produce results of adequate quality with smaller networks (R2L-16), whereas our method can still generate plausible results even with an 8-layer network (NeRDF-8). This makes our method possible to perform real-time, high-quality view synthesis with low memory cost. The advantage of our method in rendering speed with a small network is further demonstrated in the PSNR-FPS trade-off curve in Fig. 4, where our method outperforms NeLF-based methods in the high-FPS region.

Comparison with NeRF-based methods. The quantitative results of our method and NeRF-based methods on the LLFF dataset are presented in Tab. 3. Our NeRDF-8 has already exhibited significant speed advantages and outperformed almost all NeRF-based methods, except for NeX [45]. However, NeX has a higher memory cost due to its storage of multi-plane feature maps and a higher computational cost (in MFLOPs) of 42.71 compared with our 2.60. The PSNR-FPS trade-off curves of NeRF-based methods and our NeRDF are illustrated in Fig. 5, where our method demonstrates a significantly better trade-off than previous NeRF-based methods.

Table 2. Quantitative comparisons with NeLF-based methods on the LLFF dataset. For methods with multiple configurations available, we report two of them which correspond to "high-quality" and "high-speed" respectively.

Method		PSNR (dB)	FPS (RTX3090)	Memory Cost (MB)
NeLF-based	R2L-16	24.42	20.12	4.6
	R2L-88	27.79	4.44	23.0
	RSEN-4	25.58	3.75	5.4
	RSEN-32	27.45	0.45	
Ours	NeRDF-8	26.53	21.18	5.1
	NeRDF-48	27.19	4.39	28.0

Table 3. Quantitative comparisons with NeRF-based methods on the LLFF dataset. For methods with multiple configurations available, we report two of them which correspond to "high-quality" and "high-speed" respectively. "RTX3090*" represents the optimized version test on an RTX3090.

Method		PSNR (dB)	FPS (RTX3090*)	Memory Cost (MB)
NeRF-based	NeRF (Teacher)	27.75	1.45	3.8
	TermiNeRF-2	21.68	65.49	4.1
	TermiNeRF-16	23.55	11.34	
	AdaNeRF-2	21.82	101.01	4.1
	AdaNeRF-11	26.24	27.70	
	DONeRF-2	20.89	65.49	4.1
	DONeRF-16	22.91	11.34	
	InstantNGP-14	24.77	39.17	2.0
	InstantNGP-19	25.58	24.73	64.0
	NeX	27.99	∼350	89.0
Ours	NeRDF-8	26.53	369.00	5.1

Qualitative results. Finally, we conduct a qualitative comparison of our view synthesis results with other methods in Fig. 3. The results indicate that our method (NeRDF-8, Fig. 3e) produces plausible visual quality, although it has a slightly lower PSNR compared with other NeRF-based methods. The comparison also shows that previous NeLF-based methods, such as R2L in Fig. 3d, fail to synthesize acceptable visual quality under similar speed requirements and memory cost constraints.

4.3. Discussions

In this section, we perform a set of ablation studies on the FERN data, unless specified otherwise, from the LLFF dataset [23] to validate the key components of NeRDF and our training strategy.

Inference time breakdown. Although NeRDF only requires a single network forward pass per pixel, it includes

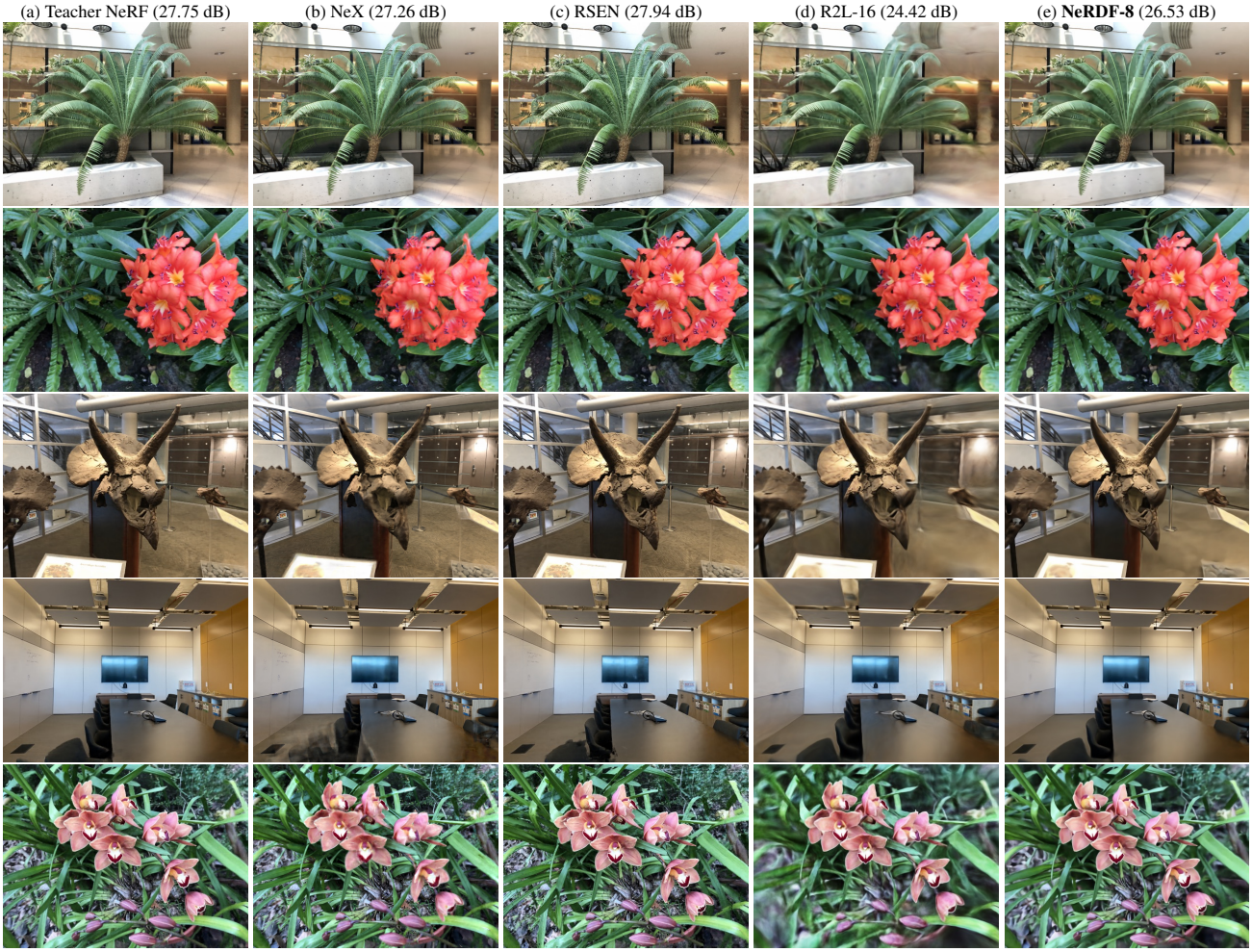

Figure 3. Qualitative comparisons. Compared with a 16-layer R2L (R2L-16) that produces much blur, ours has a high quality. Best viewed on a monitor.

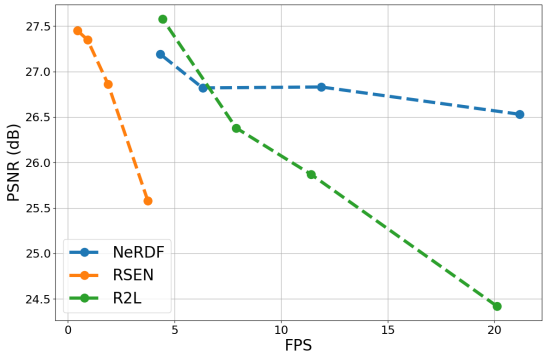

Figure 4. Trade-off curves between PSNR and FPS of NeLF-based methods and ours. Our method (the blue curve) has significant advantages in the high-FPS (>10) region over other NeLF-based methods.

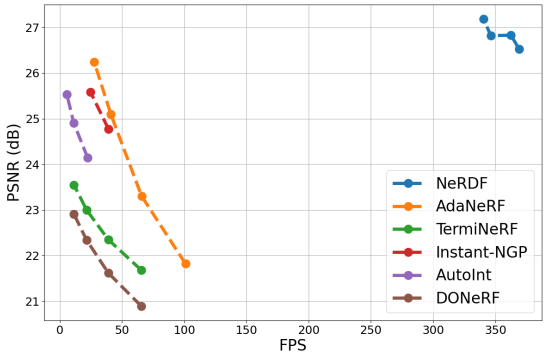

Figure 5. Trade-off curves between PSNR and FPS of NeRF-based methods and ours (both with inference optimization). Our method (the blue curve) has a significantly better trade-off than other NeRF-based methods.

the ray-marching process for pixel color rendering. Tab. 4 displays the breakdown of inference time for NeRDF-8. The results indicate that the majority of the inference time is still devoted to network inference. The additional time required for volumetric rendering only accounts for 5.3% of the total runtime for an 8-layer MLP. This highlights our advantage over NeRF-based methods, as we eliminate the most time-consuming part of multiple network forward

Table 4. The run time breakdown (render a 504×378 image on an RTX3090 GPU) for each component of a NeRDF-8 with $K = 12$.

Component	Input encoding	Network	Rendering	Total
Time(ms)	8.7	36.0	2.5	47.2

Table 5. The synthesis quality (PSNR, dB) comparison between R2L and different variants of NeRDF on FERN and FLOWER.

Idx	Distribution	OVS	VDC	FERN	FLOWER
(a)				22.61	22.01
(b)	✓			24.04	24.05
(c)	✓	✓		24.99	26.81
(d)	✓	✓	✓	**25.06**	**26.91**

passes while preserving the volumetric rendering component for synthesis quality. Besides, the training time and the optimization details are provided in supplementary materials.

NeRDF components. The proposed NeRDF method outperforms the NeLF-based method in representing scenes with small network capacities by incorporating three designs: outputting radiance distribution, incorporating online view sampling, and enforcing a view density constraint. To validate these designs, we conducted ablation experiments on two data, FERN and FLOWER, from the LLFF dataset. We start from an 8-layer MLP in R2L [41] (netwidth is 256) and gradually add our components to observe the impact of each component. The results, shown in Tab. 5, demonstrate that using radiance distribution as the output (Tab. 5b) leads to 1.4-2.0 dB improvement compared with directly predicting pixel RGB values (Tab. 5a). The online view sampling (OVS) further enhances the performance of NeRDF (Tab. 5c) with 0.9-2.8 dB improvement. Finally, introducing the view density constraint (VDC) increases the performance by approximately 0.1 dB (Tab. 5d). As seen in Fig. 6, the disparity maps generated by NeRDF with VDC are more reasonable compared with those generated without VDC, further validating the benefits of the proposed VDC for view-consistency. These results indicate that NeRDF effectively learns the 3D prior and provides a better scene representation than NeLF with a compact neural network.

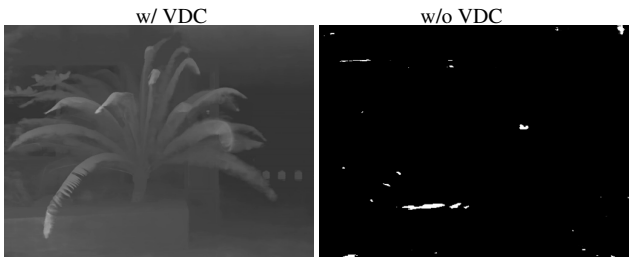

Figure 6. Visualization of synthesized disparity maps by NeRDF with and without the volume density constraint (VDC). VDC helps predict reasonable disparity.

Table 6. The synthesis quality of using different numbers of frequencies.

K	4	8	**12**	16	24
PSNR(dB)	25.63	25.64	**25.69**	25.66	25.67

Table 7. The synthesis quality of different variants in embedding the ray origin, direction, and on-the-path points in the input ray encoding.

Idx	Points	Direction	Origin	PSNR(dB)
(a)		✓	✓	24.57
(b)	✓			25.56
(c)	✓	✓		25.68
(d)	✓		✓	25.57
(e)	✓	✓	✓	**25.69**

The number of Fourier frequencies. The radiance distributions in NeRDF are approximated using a group of trigonometric functions (TrigF) through discrete Fourier transformations. We analyze the impact of the number of frequencies in TrigF (denoted as K), with results listed in Tab. 6. We find the best K is 12, and the performance of NeRDF is robust to the choice of K ranging from 4 to 24. We also provide the results of using a Gaussian Mixture Model as the alternative in supplementary materials.

Input ray encoding. We analyze various options for encoding input rays in Tab. 7. As shown, it results in a decrease of 1.1dB without encoding points sampled along the ray's path (as used in [41]), as shown in Tab. 7a. Besides, encoding both the direction and on-the-path points (Tab. 7c) results in a 0.1dB improvement compared with encoding points only (Tab. 7b). Omitting the direction encoding, as depicted in Tab. 7d, results in a drop of ∼0.1dB compared with using all three components, as shown in Tab. 7e.

Limitations and Future works. Our method is not without limitations. NeRDF requires further expansion in order to effectively handle 360-degree scenes with high fidelity. As future avenues of research, we propose to further improve the view-synthesis quality as well as extend NeRDF to handle dynamic scenes.

5. Conclusion

We have proposed a novel 3D scene representation, Neural Radiance Distribution Field (NeRDF), for real-time efficient view synthesis. NeRDF models the radiance distribution along rays parameterized using a set of trigonometric functions, and synthesizes images with volumetric rendering. Our NeRDF requires only one single network forward pass per pixel for view synthesis. NeRDF is learned by distilling both color and volume density information from a teacher NeRF. Our method offers a better trade-off among speed, cost, and quality compared with existing efficient view synthesis methods, with a significantly high rendering speed of ∼350 FPS and producing visually plausible results with a small network.

References

[1] Benjamin Attal, Jia-Bin Huang, Michael Zollhoefer, Johannes Kopf, and Changil Kim. Learning neural light fields with ray-space embedding networks. *arXiv preprint arXiv:2112.01523*, 2021. 1, 2, 3, 6

[2] Benjamin Attal, Jia-Bin Huang, Michael Zollhöfer, Johannes Kopf, and Changil Kim. Learning neural light fields with ray-space embedding. In *CVPR*, 2022. 4

[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 *ICCV*, 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 *CVPR*, 2022. 2

[5] Sai Bi, Zexiang Xu, Pratul Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. *arXiv preprint arXiv:2008.03824*, 2020. 2

[6] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *CVPR*, 2021. 2

[7] Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. *NeurIPS*, 2017. 3, 5

[8] Yu Deng, Jiaolong Yang, Jianfeng Xiang, and Xin Tong. Gram: Generative radiance manifolds for 3d-aware image generation. In *CVPR*, 2022. 2

[9] Yu Deng, Jiaolong Yang, and Tong Xin. Deformed implicit field: Modeling 3d shapes with learned dense correspondence. In *CVPR*, 2021. 2

[10] Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B. Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *ICCV*, 2021. 2

[11] Jiemin Fang, Lingxi Xie, Xinggang Wang, Xiaopeng Zhang, Wenyu Liu, and Qi Tian. Neusample: Neural sample field for efficient view synthesis. *arXiv preprint arXiv:2111.15552*, 2021. 3

[12] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. FastNeRF: High-fidelity neural rendering at 200fps. In *ICCV*, 2021. 3, 6

[13] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 3

[14] Yang Hong, Bo Peng, Haiyao Xiao, Ligang Liu, and Juyong Zhang. Headnerf: A real-time nerf-based parametric head model. In *CVPR*, 2022. 2

[15] Yuanming Hu, Tzu-Mao Li, Luke Anderson, Jonathan Ragan-Kelley, and Frédo Durand. Taichi: a language for high-performance computation on spatially sparse data structures. *TOG*, 2019. 5

[16] Andreas Kurz, Thomas Neff, Zhaoyang Lv, Michael Zollhöfer, and Markus Steinberger. Adanerf: Adaptive sampling for real-time rendering of neural radiance fields. In *ECCV*, 2022. 3, 6

[17] Xiu Li, Xiao Li, and Yan Lu. Estimating neural reflectance field from radiance field using tree structures. *arXiv preprint arXiv:2210.04217*, 2022. 2

[18] David B Lindell, Julien NP Martel, and Gordon Wetzstein. AutoInt: Automatic integration for fast neural volume rendering. In *CVPR*, 2021. 3, 6

[19] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *NeurIPS*, 2020. 3

[20] Lingjie Liu, Marc Habermann, Viktor Rudnev, Kripasindhu Sarkar, Jiatao Gu, and Christian Theobalt. Neural Actor: Neural free-view synthesis of human actors with pose control. *TOG*, 2021. 2

[21] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, 2019. 2

[22] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *TOG*, 2019. 1

[23] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2, 3, 5, 6

[24] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv preprint arXiv:2201.05989*, 2022. 3, 6

[25] Thomas Müller, Fabrice Rousselle, Jan Novák, and Alexander Keller. Real-time neural radiance caching for path tracing. *arXiv preprint arXiv:2106.12372*, 2021. 2, 3, 5

[26] Thomas Neff, Pascal Stadlbauer, Mathias Parger, Andreas Kurz, Joerg H. Mueller, Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, and Markus Steinberger. DONeRF: Towards Real-Time Rendering of Compact Neural Radiance Fields using Depth Oracle Networks. *Computer Graphics Forum*, 2021. 3, 6

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

[28] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M Seitz. HyperNeRF: A higher-dimensional representation for topologically varying neural radiance fields. *TOG*, 2021. 2

[29] Nikolaos Passalis and Anastasios Tefas. Learning deep representations with probabilistic knowledge transfer. In *ECCV*, 2018. 3

[30] Adam Paszke, Sam Gross, Soumith Chintala, and Gregory Chanan. PyTorch: Tensors and dynamic neural networks in Python with strong GPU acceleration, 2017. 5

[31] Sida Peng, Junting Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Xiaowei Zhou, and Hujun Bao. Animatable neural radiance fields for modeling dynamic human bodies. In *ICCV*, 2021. 2

[32] Martin Piala and Ronald Clark. Terminerf: Ray termination prediction for efficient neural rendering. In *3DV*, 2021. 6

[33] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural radiance fields for dynamic scenes. In *CVPR*, 2021. 2

[34] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. KiloNeRF: Speeding up neural radiance fields with thousands of tiny mlps. In *ICCV*, 2021. 6

[35] Vincent Sitzmann, Semon Rezchikov, Bill Freeman, Josh Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. *NeurIPS*, 2021. 1

[36] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *NeurIPS*, 2019. 2

[37] Shih-Yang Su, Frank Yu, Michael Zollhöfer, and Helge Rhodin. A-nerf: Articulated neural radiance fields for learning human shape, appearance, and pose. *NeurIPS*, 2021. 2

[38] Mohammed Suhail, Carlos Esteves, Leonid Sigal, and Ameesh Makadia. Light field neural rendering. In *CVPR*, 2022. 1

[39] Gusi Te, Xiu Li, Xiao Li, Jinglu Wang, Wei Hu, and Yan Lu. Neural capture of animatable 3d human from monocular video. *arXiv preprint arXiv:2208.08728*, 2022. 2

[40] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. *arXiv preprint arXiv:1910.10699*, 2019. 3

[41] Huan Wang, Jian Ren, Zeng Huang, Kyle Olszewski, Menglei Chai, Yun Fu, and Sergey Tulyakov. R2l: Distilling neural radiance field to neural light field for efficient novel view synthesis. In *ECCV*, 2022. 1, 2, 3, 4, 5, 6, 8

[42] Liao Wang, Jiakai Zhang, Xinhang Liu, Fuqiang Zhao, Yanshun Zhang, Yingliang Zhang, Minye Wu, Jingyi Yu, and Lan Xu. Fourier plenoctrees for dynamic radiance field rendering in real-time. In *CVPR*, 2022. 5

[43] 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 *NeurIPS*, 2021. 2

[44] Peng Wang, Yuan Liu, Guying Lin, Jiatao Gu, Lingjie Liu, Taku Komura, and Wenping Wang. Progressively-connected light field network for efficient view synthesis. *arXiv preprint arXiv:2207.04465*, 2022. 3

[45] Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. In *CVPR*, 2021. 6

[46] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *CVPR*, 2021. 2

[47] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. In *Computer Graphics Forum*, 2022. 2

[48] 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*, 2021. 6

[49] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *ICCV*, 2021. 3

[50] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016. 3

[51] 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. *TOG*, 2021. 2