RefSR-NeRF: Towards High Fidelity and Super Resolution View Synthesis

Xudong Huang[*], Wei Li[*], Jie Hu, Hanting Chen, Yunhe Wang
Huawei Noah's Ark Lab
{huangxudong9, wei.lee, chenhanting, hujie23, yunhe.wang}@huawei.com

Abstract

We present Reference-guided Super-Resolution Neural Radiance Field (RefSR-NeRF) that extends NeRF to super resolution and photorealistic novel view synthesis. Despite NeRF's extraordinary success in the neural rendering field, it suffers from blur in high resolution rendering because its inherent multilayer perceptron struggles to learn high frequency details and incurs a computational explosion as resolution increases. Therefore, we propose RefSR-NeRF, an end-to-end framework that first learns a low resolution NeRF representation, and then reconstructs the high frequency details with the help of a high resolution reference image. We observe that simply introducing the pre-trained models from the literature tends to produce unsatisfied artifacts due to the divergence in the degradation model. To this end, we design a novel lightweight RefSR model to learn the inverse degradation process from NeRF renderings to target HR ones. Extensive experiments on multiple benchmarks demonstrate that our method exhibits an impressive trade-off among rendering quality, speed, and memory usage, outperforming or on par with NeRF and its variants while being 52× speedup with minor extra memory usage. Code will be available at: Mindspore and Pytorch

1. Introduction

Neural Radiance Field (NeRF) [29], which was first proposed by Mildenhall *et al.* in 2020, is leading a trend in neural rendering field for its realism and representation parsimony, showing great potential in various downstream industrial applications such as immersive view synthesis [1, 2, 41], 3D scene reconstruction [24, 60], autonomous driving [38, 18, 31, 50], aerial surveying [9, 21], digital human deformation [47, 62, 52, 36], robot navigation and environment simulation [38]. In essence, NeRF synthesizes photorealistic renderings by encoding the volumetric density and color of a scene within the weights of a coordinate-based multilayer perceptron (MLP) [2], and its magic manifests in reconstructing an intact 3D spacial representation from a handful of sparse observations, while simultaneously retaining a highly compact scene representation [58].

While this approach works well when the training and testing images observe the scene content with low resolution, NeRF, and its follow-ups exhibit significant blurry effects when the resolution goes up. This can be attributed to the following two reasons. First of all, MLPs perform poorly to regress the high frequency details from uniformly-sampled low-dimension 5D coordinates [39]. Although a positional encoding that uses *Fourier Transformation* will greatly leverage this inherent low frequency bias in neural networks, there is still a big room for photorealism. In addition, as the resolution increases, or in other words, when the scene becomes more complex, NeRF requires a larger MLP to encode more high frequency information [38]. However, simply enlarging the MLP only achieves minor gains in detail restoration [33] and will significantly exacerbate the rendering efficiency.

Moreover, due to the dense sampling strategy and frequent MLP queries, rendering a NeRF is agonizingly slow. It takes more than one day to train for a scene and 30 seconds to render an 800 × 800 image even running on a high-performance desktop GPU, and this issue would be more unacceptable when the resolution continues to ascend. To accelerate the rendering speed, several follow-up works are proposed from different perspectives. One of the most strait-forward solutions is introducing voxel-grid representation to NeRF to model local properties of geometries [22, 48, 37, 58, 10, 49, 13, 30, 5]. Despite the fact that this paradigm achieves two or three orders of magnitudes of speed up [37, 10], they consume massive storage and compromise rendering quality, which is unbearable for resource-limited mobile devices.

To tackle the issue of lacking high frequency details, a surge of interest is to introduce more advanced positional encoding algorithms [1, 39]. However, these approaches brought minor improvements. To this end, we propose that since MLPs have a natural defect in learning high frequency details, it is better to let MLPs learn low frequency

[*]Equal Contribution

information only and introduce a high resolution reference frame to provide high frequency details in each scene. This begs our final solution, as shown in Figure 1, an end-to-end reference-guided super-resolution NeRF framework which is a deft combination of NeRF and the experience from the RefSR community. Specifically, We first downsample the HR training images to low resolution ones, then we optimize the low resolution NeRF with patch shuffle, followed by a lightweight RefSR model which takes an HR reference image as its input to execute the upsampling and produce our final HR renderings. We observe that simply introducing the pre-trained RefSR models from the literature tends to produce unsatisfied artifacts due to divergence in the degradation model from HR to NeRF rendering. This prompted us to design a novel effective and lightweight RefSR model. To sum up, RefSR-NeRF realizes a well-exemplified trade-off among rendering quality, speed, and memory usage, outperforming or on par with NeRF and its variants while being 52× speedup with minor extra memory and storage usage. The main contributions of this paper can be summarized as follows:

- We propose a novel end-to-end RefSR-NeRF framework that extends NeRF to high resolution and photorealistic novel view synthesis.
- RefSR-NeRF can act as a novel NeRF acceleration paradigm, which can significantly alleviate the problem of NeRF computation and cache exploding as resolution increases.
- Extensive experiments show that RefSR-NeRF qualitatively and quantitatively outperforms baseline works by a large margin while being 52× faster.

2. Related Work

There are plenty of follow-up works of NeRF since it was first proposed and tremendous progress has been made in its improvements on rendering quality [8, 1, 2, 41, 16], efficiency [37, 11, 10, 15, 53, 7, 20], controllability [35, 42], scalability [38, 34], generalization [59, 40, 44], as well as novel applications [19, 26]. Despite NeRF can render novel views at any resolution by spatial interpolation, it suffers from losing detail and intensive computation overhead. NeRF-SR [43] is somewhat similar to ours, in which it directly renders HR images using super-sampling and introduces a reference image to further finetune the results. However, in essence, super-sampling can't save any computational resources so NeRF-SR is infeasible for resource-limited mobile devices. We observe that little research has been engrossed in accelerating NeRF from a resolution compression perspective. In this section, we briefly review the advances in efficiency improvement and the field of reference-based super-resolution.

2.1. Efficient NeRF

The lengthy training and inference time to render a novel view image has become a long-standing problem in NeRF community. To leverage this deficiency, KiloNeRF [33] splits one big MLP into around one thousand tiny MLPs and each tiny MLP encodes a partitioned small region independently. NeX [48] proposes a depth oracle network that predicts ray sample locations for each viewing ray to reduce the number of samples. Another general method is to use discretized volumetric representations to store the geometries feature or properties offline and reload them at the rendering stage [37, 22, 49, 58]. DIVeR [49] uses a voxel-based field of features to represent the whole scene and exploits deterministic rather than stochastic estimates of the volume rendering integral. NSVF [22] defines a set of voxel-bounded implicit fields organized in a sparse voxel octree to guide and reduce sampling. FastNeRF [11] replace multilayer perceptron by caching the NeRF into dense voxel grids. PlenOctrees [58] exploit a sparse voxel-based octree where each node stores both density and appearance values to model the radiance at a point. Efficient-NeRF [15] further pre-tabulates 3D scenes into dense and sparse voxels to speed up NeRF rendering. In addition to opacity and color information, SNeRG [13] stores view-dependent learned feature vectors to sparse voxel grids. By pre-computing and caching the properties(*i.e* density, spherical harmonics coefficients [58, 10] or learned features) into specific data structures, the rendering can be sped up significantly but such approach incurs a significant memory overhead and is prone to suboptimal geometry solutions [37]. Contrastively, our method achieves orders of magnitudes of speedup and maintains the simplicity of NeRF.

2.2. Reference-based Image Super-Resolution

Reference-based image super-resolution (RefSR) aims to exploit auxiliary reference (Ref) images to super-resolve low resolution (LR) images by transferring high frequency details of reference frames [17, 23, 54, 51, 56, 57, 32]. CrossNet [63] estimates the optical flow (OF) between Ref and LR and aligns them into the same resolution by cross-scale warping. TTSR [55] regards the LR and reference images as queries and keys in a transformer and proposes hard and soft attention for texture transfer and synthesis. C2-Matching [17] proposes a contrastive correspondence network to learn the relationship between LR and reference images, then exploits a teacher-student correlation distillation and a residual feature aggregation to synthesize HR images. To match the correspondence between LR and reference images with significant differences, Cao *et al.* [3] propose a deformable attention Transformer to aggregate features and relevant textures. Combined with RefSR, NeRF can accelerate rendering at low resolution and restore high frequency information with the assistance of reference frames. How-

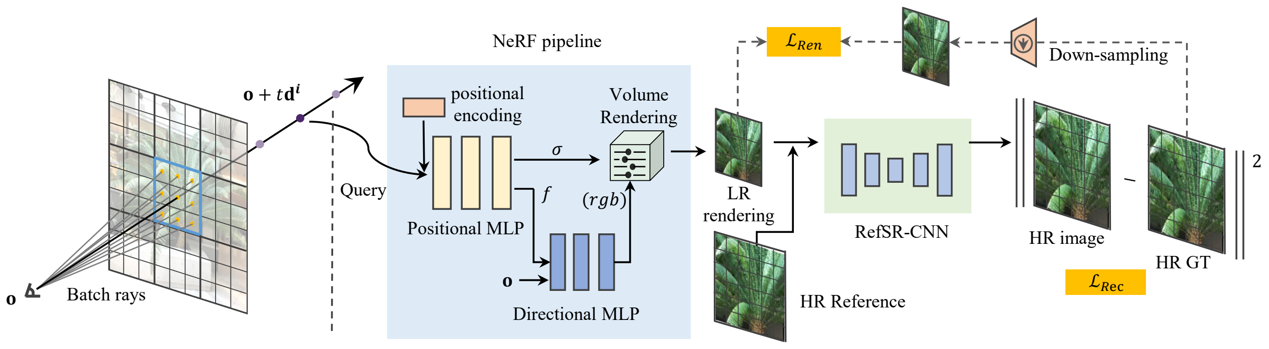

Figure 1. An overview of the proposed RefSR-NeRF framework. RefSR-NeRF first learns a low resolution scene representation and then reconstructs high frequency details with the help of a pre-determined high resolution reference image. Different from vanilla NeRF [29] or its variants that randomly sample rays during training process, RefSR-NeRF samples an image patch at a time to be compatible with end-to-end training with the following RefSR model.

ever, The off-the-shelf SR methods are for specific downsampling degradation models and do not apply to NeRF whose degradation models are uncertain. To this end, we propose an aggregation module to fit the downsampling in NeRF accurately.

3. Methodology

We build our RefSR-NeRF implementation on the vanilla NeRF [26]. Generally, NeRF and its variants adopt random batch rays sampling to train for each scene, which is not compatible with the following RefSR model training. Instead, we build this end-to-end pipeline by using a patch rays sampling strategy, so as to provide spatial contextual information to the RefSR model to make end-to-end training possible. Another impediment is that NeRF rendering process exhibits a new degradation model, so introducing the off-the-shelf RefSR models is prone to producing artifacts and is of too many parameters for the scene by scene optimization paradigm in NeRF.

3.1. NeRF

NeRF approximates the continuous 5D scene representation with an MLP network by taking a 3D location $\mathbf{x} = (x, y, z)$ and 2D viewing direction $\mathbf{d} = (\theta, \phi)$ as input, and producing an emitted color $\mathbf{c} = (r, g, b)$ and volume density σ [29]. For parsimony, this mapping process can be written as: $F_\Theta : (\mathbf{x}, \mathbf{d}) \longmapsto (\mathbf{c}, \sigma)$. Incorporated with the ray tracing method, NeRF renders each pixel of a camera as follows: A ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$ is emitted from the camera's center projection $\mathbf{o}$ along the direction $\mathbf{d}$ such that it passes through the pixel [1]. A sampling strategy (Monte Caro in most cases) is adopted to determine a vector of sorted distance $\mathbf{t}$ between the camera's predefined near and far planes t_n and t_f. In vanilla NeRF [29], the MLP consists of two sub-MLPs: MLP_{pos} to represent the position-dependent component density σ, and MLP_{dir} to represent position and view-dependent RGB color $\mathbf{c}$:

$$\forall t_k \in \mathbf{t}, \quad [\sigma_k, \mathbf{f}_k] = \text{MLP}_{pos}(\gamma^{L_\mathbf{x}}(\mathbf{x}(t_k)); \Theta_{pos}), \tag{1}$$

$$\mathbf{c}_k = \text{MLP}_{dir}(\gamma^{L_\mathbf{d}}([\mathbf{d}(t_k)), \mathbf{f}_k]; \Theta_{dir}) \tag{2}$$

Where Θ_{pos} and Θ_{dir} denote the weights of these two sub-MLPs, respectively. The $\mathbf{f}_k$ is a position-dependent feature vector generated by positional MLP_{pos} and is provided as input to the directional MLP_{dir}. γ^L represents the positional encoding which is to be discussed in the following section.

Despite the fact that neural networks are universal function approximator [14], Ben.Mildenhall *et al.* found that MLPs perform poorly to regress the high frequency details from an uniformly-sampled and low-dimension 5D coordinates, and a simple but non-trivial positional encoding operation that uses *Fourier Transformation* principle will greatly leverage this inherent low frequency bias in neural networks. Tancik *et al.* further analyze and prove this point in [39]. Specifically, the positional encoding used in vanilla NeRF [29] can be written as:

$$\begin{aligned}\gamma(p) = &[\sin(2^0\pi p),\ \cos(2^0\pi p),\\ &\quad ..\qquad\qquad ...\\ &\sin(2^{L-1}\pi p),\ \cos(2^{L-1}\pi p)]\end{aligned} \tag{3}$$

Note that $\gamma(\cdot)$ is applied to each dimension of the 3D position $\mathbf{x}$ and $\mathbf{d}$ scaled by powers of 2 from 0 to 2^{L-1}, where L is a hyperparameter that determines the bandwidth of the interpolation kernel (more details can be found in Tancik *et al.* [39]). Empirically, L is set to 10 for $\gamma(\mathbf{x})$ and 4 for $\gamma(\mathbf{d})$ in [29] in the context of position $\mathbf{x}$ and view direction $\mathbf{d}$ are normalized to lie in [-1, 1].

Once we obtain the estimated densities and colors along a ray, we can utilize the volume rendering integral using numerical quadrature, as per Max [25]:

$$\mathbf{C}(\mathbf{r}; \Theta; \mathbf{t}) = \sum_k T_k(1 - \exp(\sigma_k(t_{k+1} - t_k)))\mathbf{c}_k, \quad (4)$$

$$T_k = \exp(-\sum_{k'<k} \sigma_{k'}(t_{k'+1} - t_{k'})) \quad (5)$$

Where the $\mathbf{C}(\mathbf{r}; \Theta; \mathbf{t})$ is the final rendered color of the pixel. The full implementation of NeRF utilizes a coarse-to-fine sampling of t_k along each ray by treating the integral weights $w_i = T_i(1 - \exp(\sigma_k(t_{k+1} - t_k)))$ as a probability distribution in order to better concentrate samples in areas of high density.

3.2. Patch-Based NeRF Training

According to the camera pose, NeRF samples rays corresponding to each pixel and renders specific color information via MLP and volume rendering integral. However, the inherent nature of MLP and the training strategy of random sampling leads to poor performance of NeRF in high frequency scenarios. Conversely, RefSR extracts features of continuous pixels in the reference image to reconstruct high frequency information of the LR. This becomes a problem when we combine implicit representation learning with continuous feature representation, specifically, it makes no sense for individual rays(randomly sampled by NeRF) to be rendered to specific colors and then super-resolution to scale regions. There is a gap between sampling strategies.

Therefore, we present a patch-based NeRF to connect the NeRF to the SR module in the training stage, as shown in Figure 2. We first obtain the rays from a continuous region rather than stochastic selection. The final ray color is then obtained by the MLP and the volumetric integration. Finally, rendering regions are sent to the subsequent network to train the SR module. The formulation can be described as:

$$\forall \mathbf{r} \in \mathbf{R}, \quad HR = \text{RefSR}(\mathbf{C}(\mathbf{r}; \Theta; \mathbf{t})) \quad (6)$$

where $\mathbf{R}$ represents the sampled batching rays from training images, Θ denotes weights of MLP and $\mathbf{t}$ is the sampling intervals along each ray. We empirically found that feeding fixed-order batching rays that indicate a fixed region in image space to MLP leads to turbulence during the training process and falls into a suboptimal solution. Therefore, we adopt a batch shuffling across the whole training image after each epoch.

3.3. RefSR Model for NeRF Degradation

Reference-based Super Resolution (RefSR) methods aim to super-resolve HR images from the LR ones with the assistance of reference images. Combined with RefSR, NeRF is able to achieve considerable rendering acceleration and restore high frequency information from reference images [23, 55]. However, recent works in the RefSR community mainly focus on modeling the specific down-sampling degradation procedure[4], *e.g.,* JEPG compression, bicubic or bilinear interpolation, none of which has explored this emerging but sophisticated degradation process caused by MLPs in NeRF rendering pipeline.

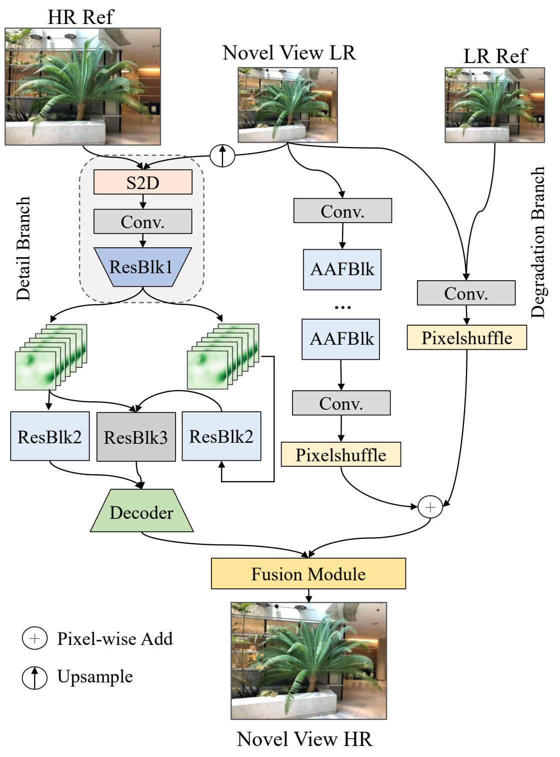

Figure 2. An overview of the reference-based SR model, which consists of a two-branch backbone and a fusion module. The backbones extract the feature maps from degradation-dominant (LR-Ref and novel view LR) and detail-dominant information (HR Ref and novel view LR) respectively. The two feature maps are then fused and further refined by a fusion module.

Therefore, as shown in Figure 2, we present a reference-based SR module to exploit reference images to super-resolve the LR images generated by NeRF model. The main considerations of our approach are how to model the down-sampling degradation procedure between HR and NeRF's LR views, and to transfer high frequency details from a content-consistent but misaligned reference image to the final HR image. Specifically, we design a two-branch network that thoroughly exploits detail-dominant (DeT) and degradation-dominant (DeG) information from the respective branch and fuses these two outputs effectively. We build two encoder-decoder networks in two branches to perform a down-sampling degradation prediction and a high frequency details restoration.

The detail-dominant branch aims to transfer the high frequency details from Ref inputs. For the purpose of effectiveness, an upsampling of NeRF's output (LR^{up}) is also sent to assist in detail restoration. In the current branch, inputs are transformed by space to depth layer (S2D) and extracted deep features through a series of convolution layers and two basic residual blocks(ResBlocks [12]). The outputs of the first ResBlock are F_{LR}^{up}, F_{Ref}, and the second are $F_{LR}^{'up}$, $F_{Ref}^{'}$, where

$$\begin{aligned} F_{LR}^{up}, F_{Ref} &= ResBlk(Conv(S2D((LR^{up}, Ref)))) \\ F_{LR}^{'up}, F_{Ref}^{'} &= ResBlk1(F_{LR}^{up}, F_{Ref}) \end{aligned} \tag{7}$$

The details are then initially fused by feeding the $F_{Ref}^{'}$ and F_{LR}^{up} into a ResBlock. Finally, the decoder layer upsamples the fused detail features and $F_{LR}^{'up}$ to the target scale, which generates the output of the detail-dominant branch, where

$$Out_{detail} = Decoder(F_{LR}^{'up}, ResBlk2(F_{Ref}^{'}, F_{LR}^{up})) \tag{8}$$

For the degradation-dominant branch, we exploit Attentive Auxiliary Feature Block(AAFBlk)[45] as the central feature extractor because of its excellent adaptive feature alignment ability. The degradation branch takes the LR and Ref at low resolution as inputs. The LR view of Ref is synthesized by MLP with the Ref camera pose as input. We add a skip connection by a conv layer and pixelshuffle layer for input and the output adds to the result of the main feature extractor directly. As two outputs are generated, we fuse them by a fusion module. Note that we select the middle frame (fixed) in the training set as the reference image during the training and inference process. So there is only minor extra disk storage.

3.4. Loss Function

The total loss function in our experiment mainly consists of two parts, the rendering loss between the LR rendering of NeRF and the LR ground truth images and the reconstruction loss $\mathcal{L}_{Rec}$ between the super-resolved HR images and the HR ground truth. The formulation can be denoted as:

$$\mathcal{L}_{Ren} = ||I_{LR} - (\overline{I}_{LR})||_1 \tag{9}$$

in which $\mathcal{L}_{Ren}$ denotes the LR rendering loss. In order to maintain the spatial structure information while improving the resolved HR quality, we use a Charbonier loss function to better deal with outliers.

$$\mathcal{L}_{Rec} = \sqrt{||I_{HR} - \overline{I}_{HR}||^2 + \epsilon^2} \tag{10}$$

in which $\mathcal{L}_{Rec}$ denotes the HR reconstruction loss and ϵ is a hyperparameter and is set to 1e-12 default.

$$\mathcal{L}_{total} = \lambda_1 \cdot \mathcal{L}_{Ren} + \lambda_2 \cdot \mathcal{L}_{Rec} \tag{11}$$

3.5. Implementation Details

We built our LR NeRF based on vanilla NeRF and implement our RefSR model using Pytorch and train it on a V100 NVIDIA GPU with 32GB memory. During training, Adam is selected as the optimizer. The batch size of rays sampled across the images in the training set is set to 1024, which means the LR NeRF renders a 32×32 image patch in each iteration and is sent to the following RefSR model for super resolving HR rendering. Our loss is simply the sum of the rendering loss (the mean squared error between the rendered LR image and ground truth LR image) and the reconstruction loss (the mean squared error between the resolved HR image and ground truth HR image).

4. Experiments

We provide a quantitative and qualitative comparison of both the synthetic scenes from [29] and real-world forward-facing scenes from [27] to demonstrate the superior performance and trade-offs in our algorithm. We evaluate the rendering quality of the renderings via the widely used metrics including PSNR, SSIM [46] , and VGG LPIPS [61]. Furthermore, we also report the rendering Frame Per Second (FPS) and storage usage for efficiency evaluation. Quantitative comparison are shown in Table 1 and Table 2, and visual results are depicted in Figure 3 and Figure 4. Note that in most experimental settings, our approach shows higher image quality while achieving two orders of magnitudes of speed up. We provide extensive ablation studies to verify the contribution of each component of our proposed framework.

4.1. The Real World Forward-Facing Dataset

The Local Light Field Fusion (LLFF) [28] dataset consists of 8 real-world scenes captured from lots of forward-facing views. Each scene has 20 to 62 images with a resolution of 4032×3024. We follow the same training and testing data split strategy as the original NeRF [29]. Each scene uses $7/8$ of the images for training and takes the remaining $1/8$ for testing. Compared with the NeRF synthetic dataset, *e.g.,* the synthesis 360° dataset [29], the LLFF dataset has more complex scenes that present the real-world application of NeRF, thus can be extremely challenging for such high resolution and complex view synthetic. We set the input resolution as 504×378 and super-resolve this LR rendering under the scale of ×4 and ×8. The downsampling method is the same as the one provided in the official LLFF [28] code.

We compared with bicubic interpolation, vanilla NeRF, and NeRF-SR [43] approaches. For vanilla NeRF, in addition to training and testing models at high resolution, termed NeRF-HR, we also provide the model trained at low resolution and render at high resolution using spacial super-

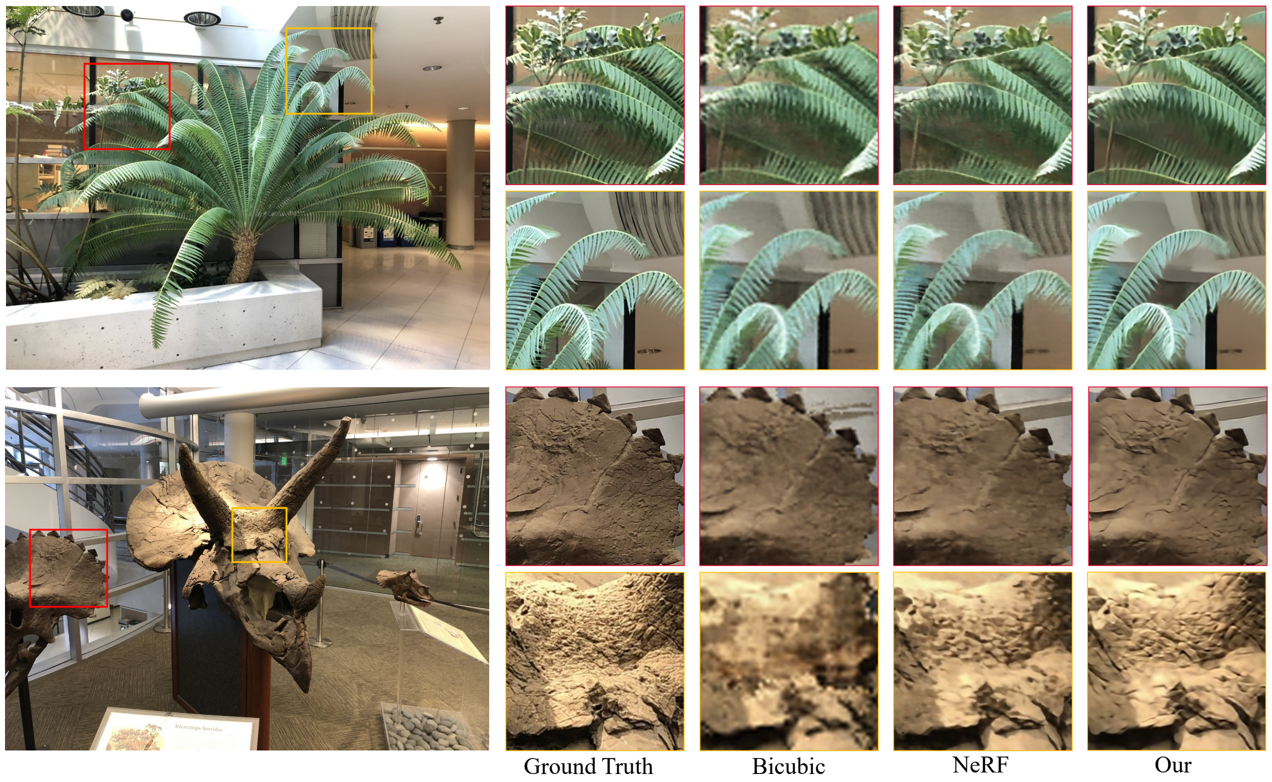

Figure 3. Qualitative comparison of our method with bicubic interpolation and NeRF on the dataset of LLFF with the input resolution at 504×378 with×4 super-resolving to 2016×1512. Our method can restore the high frequency details meanwhile the render time of our method is over two orders of magnitude faster than NeRF.

sampling (NeRF-LR). Table 1 demonstrates that our approach can outperform NeRF-LR and NeRF-HR in render quality, and be comparable with NeRF-based methods (*e.g.* NeRF-SR [43]) while achieving considerable rendering acceleration. Specifically, for the test scale ×2, our approach achieves at least 5× render acceleration with a slight degradation in render quality. For the test scale ×4, our approach achieves over 30× speed up while keeping the comparable result of the state-of-the-art method. For the test scale ×8, the rendering quality of our method is superior to NeRF-HR (0.66 dB), and the rendering speed is over 50× faster than NeRF-HR. Note that the rendering quality of our method decreases slowly as the test scale increases, which means that our method is more robust when synthesizing high resolution views.

We also compare the performance of our method with baselines. As illustrated in Figure 3 and Figure 4 , which show the visualization of different methods at the rendering resolution of 2K and 4K. Our method produces accurate and clear textures on all tested scales, for instance in the $orchids$ scene, Bicubic and NeRF tend to produce noisy and blurry renderings in cropped flower patch (the red rectangle) and leaves patch (the yellow rectangle), indicating that our method outperforms NeRF in high frequency information restoration.

4.2. Synthetic 360 ° Dataset

The NeRF Synthetic 360° dataset [29] contains 8 synthetic scenes, each of which includes 100 ground truth images and 200 testing images. All images are of 800×800 resolution with calibrated camera positions. For the NeRF synthesis 360° dataset, we report two experimental: scales ×4 with 100×100 LR resolution and scales ×4 with 100×100 LR resolution. The down-sampling strategy from HR to LR is *Mogrify* algorithm from the ImageMagic toolkit. We attach the quantitative comparisons and visual comparisons in supplementary material.

4.3. Quality, Speed, and Memory Trade-off

In Table 2 and Figure 6, we compare the rendering quality, rendering speed, and memory usage of our method along with MLP-based methods (LLFF, NeRF, Jax-NeRF) and caching-based methods (Plenoxel, FastNeRF, Efficient-NeRF, SNeRG, Mobile-NeRF). We can see that non-caching methods except LLFF have a slower inference speed (at least 600×) compared to the caching-based methods, while caching-based methods have a considerably high memory overhead (at least 62×). LLFF meets both high

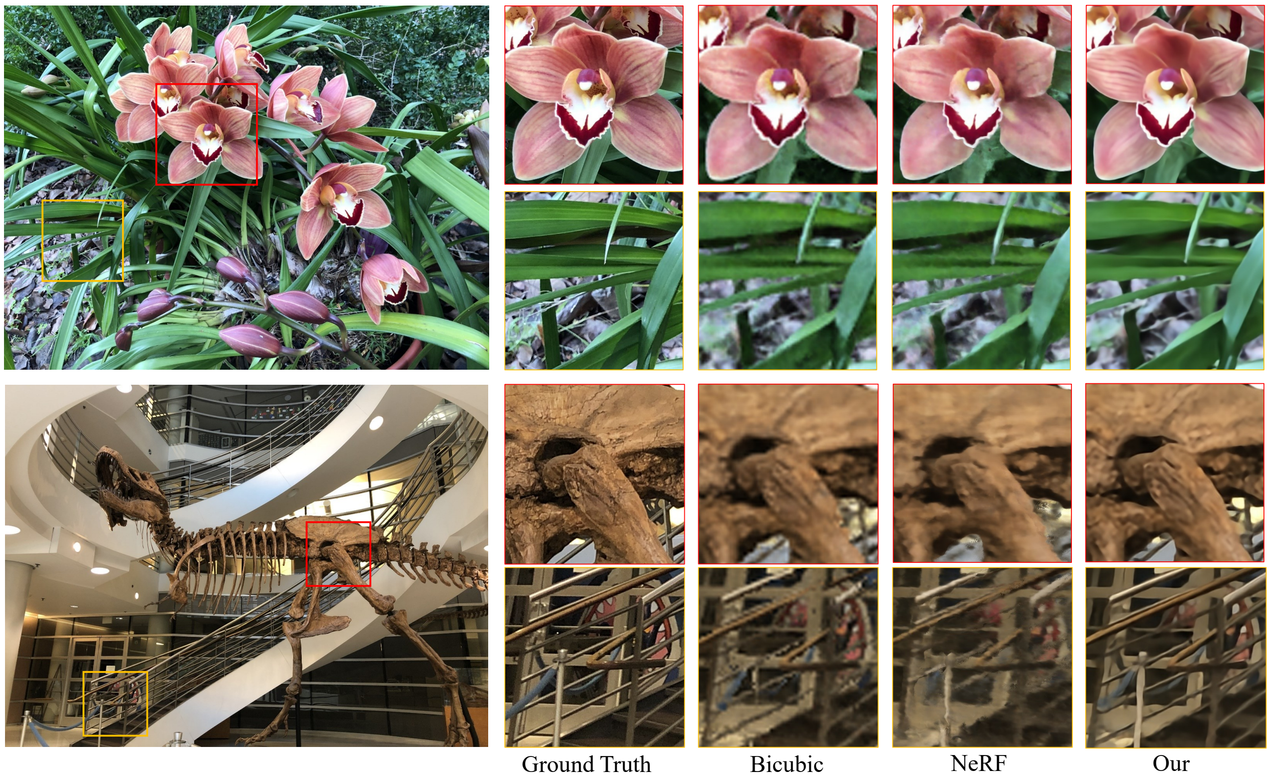

Figure 4. Qualitative comparison of our method with bicubic interpolation and NeRF on the dataset of LLFF with the input resolution at 504×378 with×8 super-resolving to 4032×3024. Our method can restore the high frequency details meanwhile the render time of our method is over two orders of magnitude faster than NeRF.

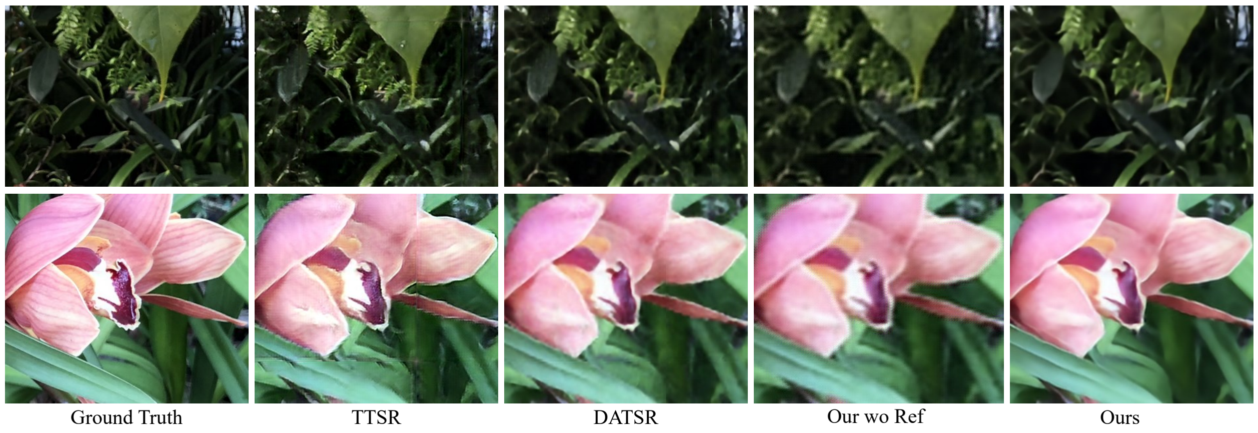

Figure 5. Qualitative comparison to the state-of-the-art RefSR method DATSR [3] and TTSR [55], which exhibit unpredictable artifacts or blurry effect. However, our method presents accurate and clear texture information in these challenging real-world scenes.

rendering speed and low memory consumption but performs poorly. Therefore, there is a clear trade-off between the memory consumption of a model and its inference speed and quality. We can see that our method exhibits an impressive trade-off among rendering quality, speed, and memory usage, outperforming or on par with NeRF and its variants while being 5× speedup with minor extra memory and storage usage. Note that with the increase in resolution, our approach maintains high rendering speed and low consumption, while rendering quality decreases slightly.

4.4. Ablation Study

In this section, we present extensive ablation studies of our method on the LLFF dataset for test scale ×4, as shown

Table 1. Quantitative comparison for novel view synthesis on the real forward-facing dataset [27]. With the input resolution 504×378, we report the quantitative metrics for upsampling scale at ×2, ×4 and ×8. In addition to considerable acceleration, our method achieves more robust performance with increasing rendering scales.

Method	Metrics (NeRF-LLFF×2)			
	PSNR↑	SSIM↑	LPIPS↓	Speed (*sec*) ↓
Bicubic	24.90	0.844	0.301	8.0
NeRF-LR [29]	23.30	0.663	0.296	38.9
NeRF-SR [43]	27.26	0.842	0.103	39.1
NeRF-HR [29]	26.50	0.811	0.251	38.9
Ours	26.23	0.874	0.243	8.5
Method	Metrics (NeRF-LLFF×4)			
	PSNR↑	SSIM↑	LPIPS↓	Speed (*sec*) ↓
Bicubic	24.19	0.815	0.438	8.0
NeRF-LR [29]	24.47	0.701	0.388	155.3
NeRF-SR [43]	25.59	0.759	0.165	245.5
NeRF-HR [29]	25.33	0.842	0.397	155.3
Ours	25.37	0.847	0.398	8.8
Method	Metrics (NeRF-LLFF×8)			
	PSNR↑	SSIM↑	LPIPS↓	Speed (*sec*) ↓
Bicubic	23.29	0.823	0.518	8.0
NeRF-LR [29]	21.71	0.800	0.514	468.4
NeRF-HR [29]	24.22	0.840	0.490	468.4
Ours	24.88	0.850	0.466	8.9

Table 2. A comparison with NeRF acceleration counterpart, all figures are reported on LLFF [28] dataset with render resolution 1008×756.

Method	Metrics (NeRF-LLFF×2)			Speed (FPS)	Disk Storage
	PSNR↑	SSIM↑	LPIPS↓		
Plenoxel [10]	26.29	0.839	0.210	-	19.3 GB
FastNeRF [11]	26.04	0.856	0.085	∼200	57.0 GB
Effi-NeRF [15]	27.39	0.912	0.082	218.83	4.3 GB
SNeRG [13]	25.63	0.818	0.183	27.38	310 MB
Mobile-NeRF [6]	25.91	0.825	0.183	-	451 MB
LLFF [27]	24.13	0.798	0.212	60	5.0 MB
NeRF [29]	26.50	0.811	0.250	0.030	5.0 MB
Jax-NeRF [8]	26.92	0.831	0.173	0.040	5.0 MB
Ours	26.23	0.874	0.243	0.153	38.3 MB

Table 3. The ablations of the proposed RefSR model. Metrics are reported on NeRF-LLFF ×4 experiment setting. Scratch represents training the initial model while pre-trained represents using the trained model for inference(fine-tuned).

Method	Scratch	Pre-trained	Fine-tuned	Metrics (NeRF-LLFF×4)	
				PSNR↑	SSIM↑
NeRF-LR	✓			24.47	0.701
NeRF+TTSR	✓			23.43	0.720
NeRF+DATSR		✓		24.34	0.826
NeRF+DATSR	✓			24.71	0.834
NeRF+DATSR		✓	✓	25.21	0.851
Ours wo Ref	✓			24.93	0.824
Ours	✓			**25.37**	0.849

in Table 3. We report rendering quality for a more comprehensive comparison. In addition, we ablate the contribution

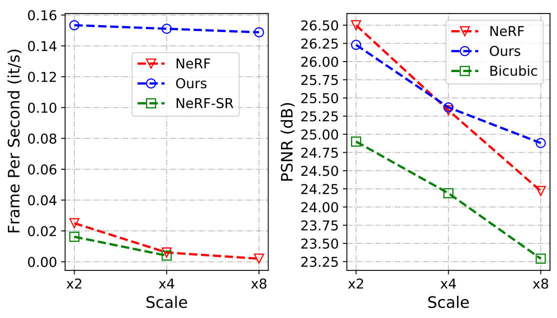

Figure 6. RefSR-NeRF surpasses NeRF by a large margin (as shown in Figure 6 (right)) and simultaneously gets a more than 52× speedup (as shown in Figure 6 (left)) at ×8 upsampling scale. In addition, our method achieves competitive quality and speed acceleration at ×2 and ×4 upsampling scale. NeRF-SR [43] is another reference based scheme based on super-sampling, but their rendering speed is much slower than ours.

of the proposed RefSR module by comparing it with existing SOTA RefSR models.

The baseline model is low resolution NeRF, which is directly trained on input images (504×378) and renders high resolution outputs. With the analysis of the results, we draw the following conclusions: First, our method without the reference module achieved a slight improvement over the baseline. Second, our Ref-based module is beneficial for restoring high frequency details. To prove the second conclusion, we add ablation experiments from recent RefSR methods. DATSR [3] and TTSR [55] are excellent methods in the RefSR task, while the former achieves state-of-the-art performance. However, these two methods are not suitable for down-sampling degradation caused by MLP.

In Figure 5 , we also provide a visual comparison of ablation experiments. The results show that DATSR and TTSR do not generate results well, while our method without Ref can reconstruct scenes clearly. With the assistance of the Ref, the output of our method is more approximate to ground truth in high frequency detail.

5. Discussion and Conclusion

In this paper, we present a straightforward yet non-trivial RefSR-NeRF framework for high fidelity and super resolution view synthesis. The proposed low resolution NeRF parametrization accompanied by our RefSR model is able to achieve an impressive render acceleration and meanwhile preserve fine details. In addition, this paradigm can be well generalized across a wide range of NeRF methods and benefits the community. Extensive experiments and results demonstrate the effectiveness of our RefSR-NeRF method.

Acknowledgments We gratefully acknowledge the support of MindSpore, CANN (Compute Architecture for Neural Networks) and Ascend AI Processor used for this research.

References

[1] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *ICCV*, pages 5855–5864, October 2021. 1, 2, 3

[2] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, pages 5470–5479, June 2022. 1, 2

[3] J. Cao, J. Liang, K. Zhang, Y. Li, Y. Zhang, W. Wang, and L. Van Goo. Reference-based image super-resolution with deformable attention transformer. *arXiv preprint arXiv:2207.11938*, 2022. 2, 7, 8

[4] L. Cao, A. Jiang, W. Li, H. Wu, and N. Ye. Oodhdr-codec: Out-of-distribution generalization for hdr image compression. In *AAAI*, volume 36, pages 158–166, 2022. 4

[5] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su. Tensorf: Tensorial radiance fields. *arXiv preprint arXiv:2203.09517*, 2022. 1

[6] Z. Chen, T. Funkhouser, P. Hedman, and A. Tagliasacchi. Mobilenerf: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures. *arXiv preprint arXiv:2208.00277*, 2022. 8

[7] F. Cole, K. Genova, A. Sud, D. Vlasic, and Z. Zhang. Differentiable surface rendering via non-differentiable sampling. In *ICCV*, pages 6088–6097, 2021. 2

[8] B. Deng, J. T. Barron, and P. P. Srinivasan. JaxNeRF: an efficient JAX implementation of NeRF, 2020. 2, 8

[9] D. Du, Y. Qi, H. Yu, Y. Yang, K. Duan, G. Li, W. Zhang, Q. Huang, and Q. Tian. The unmanned aerial vehicle benchmark: Object detection and tracking. In *ECCV*, pages 370–386, 2018. 1

[10] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, pages 5501–5510, June 2022. 1, 2, 8

[11] S. J. Garbin, M. Kowalski, M. Johnson, J. Shotton, and J. Valentin. Fastnerf: High-fidelity neural rendering at 200fps. In *CVPR*, pages 14346–14355, 2021. 2, 8

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 5

[13] P. Hedman, P. P. Srinivasan, B. Mildenhall, J. T. Barron, and P. Debevec. Baking neural radiance fields for real-time view synthesis. In *CVPR*, pages 5875–5884, 2021. 1, 2, 8

[14] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989. 3

[15] T. Hu, S. Liu, Y. Chen, T. Shen, and J. Jia. Efficientnerf efficient neural radiance fields. In *CVPR*, pages 12902–12911, 2022. 2, 8

[16] X. Huang, Q. Zhang, Y. Feng, H. Li, X. Wang, and Q. Wang. Hdr-nerf: High dynamic range neural radiance fields. In *CVPR*, pages 18398–18408, 2022. 2

[17] Y. Jiang, K. C. Chan, X. Wang, C. C. Loy, and Z. Liu. Robust reference-based super-resolution via c2-matching. In *CVPR*, pages 2103–2112, 2021. 2

[18] W. Li, C. Pan, R. Zhang, J. Ren, Y. Ma, J. Fang, F. Yan, Q. Geng, X. Huang, H. Gong, et al. Aads: Augmented autonomous driving simulation using data-driven algorithms. *Science robotics*, 4(28):eaaw0863, 2019. 1

[19] Y. Li, S. Li, V. Sitzmann, P. Agrawal, and A. Torralba. 3d neural scene representations for visuomotor control. *arXiv preprint arXiv:2107.04004*, 2021. 2

[20] D. B. Lindell, J. N. Martel, and G. Wetzstein. Autoint: Automatic integration for fast neural volume rendering. In *CVPR*, pages 14556–14565, 2021. 2

[21] A. Liu, R. Tucker, V. Jampani, A. Makadia, N. Snavely, and A. Kanazawa. Infinite nature: Perpetual view generation of natural scenes from a single image. In *ICCV*, pages 14458–14467, 2021. 1

[22] L. Liu, J. Gu, K. Z. Lin, T.-S. Chua, and C. Theobalt. Neural sparse voxel fields. *NeurIPS*, 2020. 1, 2

[23] L. Lu, W. Li, X. Tao, J. Lu, and J. Jia. Masa-sr: Matching acceleration and spatial adaptation for reference-based image super-resolution. In *CVPR*, pages 6368–6377, 2021. 2, 4

[24] R. Martin-Brualla, N. Radwan, M. S. M. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *CVPR*, pages 7210–7219, June 2021. 1

[25] N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995. 4

[26] B. Mildenhall, P. Hedman, R. Martin-Brualla, P. P. Srinivasan, and J. T. Barron. Nerf in the dark: High dynamic range view synthesis from noisy raw images. In *CVPR*, pages 16190–16199, 2022. 2, 3

[27] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019. 5, 8

[28] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019. 5, 8

[29] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 3, 5, 6, 8

[30] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *CoRR*, abs/2201.05989, 2022. 1

[31] J. Ost, F. Mannan, N. Thuerey, J. Knodt, and F. Heide. Neural scene graphs for dynamic scenes. In *CVPR*, pages 2856–2865, 2021. 1

[32] J. Qiao, S. Lin, Y. Zhang, W. Li, H. Jie, G. He, C. Wang, and Z. Ma. Dcs-risr: Dynamic channel splitting for efficient real-world image super-resolution. *arXiv preprint arXiv:2212.07613*, 2022. 2

[33] C. Reiser, S. Peng, Y. Liao, and A. Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *ICCV*, pages 14335–14345, October 2021. 1, 2

[34] K. Rematas, A. Liu, P. P. Srinivasan, J. T. Barron, A. Tagliasacchi, T. Funkhouser, and V. Ferrari. Urban radiance fields. In *CVPR*, pages 12932–12942, 2022. 2

[35] P. P. Srinivasan, B. Deng, X. Zhang, M. Tancik, B. Mildenhall, and J. T. Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *CVPR*, pages 7495–7504, 2021. 2

[36] S.-Y. Su, F. Yu, M. Zollhoefer, and H. Rhodin. A-nerf: Surface-free human 3d pose refinement via neural rendering. *arXiv preprint arXiv:2102.06199*, 2021. 1

[37] C. Sun, M. Sun, and H.-T. Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *CVPR*, pages 5459–5469, June 2022. 1, 2

[38] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *CVPR*, pages 8248–8258, 2022. 1, 2

[39] M. Tancik, P. P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. T. Barron, and R. Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 2020. 1, 3

[40] A. Trevithick and B. Yang. Grf: Learning a general radiance field for 3d representation and rendering. In *ICCV*, pages 15182–15192, October 2021. 2

[41] D. Verbin, P. Hedman, B. Mildenhall, T. Zickler, J. T. Barron, and P. P. Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *CVPR*, pages 5491–5500, June 2022. 1, 2

[42] C. Wang, M. Chai, M. He, D. Chen, and J. Liao. Clip-nerf: Text-and-image driven manipulation of neural radiance fields. In *CVPR*, pages 3835–3844, 2022. 2

[43] C. Wang, X. Wu, Y.-C. Guo, S.-H. Zhang, Y.-W. Tai, and S.-M. Hu. Nerf-sr: High-quality neural radiance fields using super-sampling. *arXiv preprint arXiv:2112.01759*, 2021. 2, 5, 6, 8

[44] Q. Wang, Z. Wang, K. Genova, P. P. Srinivasan, H. Zhou, J. T. Barron, R. Martin-Brualla, N. Snavely, and T. Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *CVPR*, pages 4690–4699, 2021. 2

[45] X. Wang, Q. Wang, Y. Zhao, J. Yan, L. Fan, and L. Chen. Lightweight single-image super-resolution network with attentive auxiliary feature learning. In *Proceedings of the Asian conference on computer vision*, 2020. 5

[46] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 5

[47] C.-Y. Weng, B. Curless, P. P. Srinivasan, J. T. Barron, and I. Kemelmacher-Shlizerman. Humannerf: Free-viewpoint rendering of moving people from monocular video. In *CVPR*, pages 16210–16220, 2022. 1

[48] S. Wizadwongsa, P. Phongthawee, J. Yenphraphai, and S. Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. In *CVPR*, pages 8534–8543, 2021. 1, 2

[49] L. Wu, J. Y. Lee, A. Bhattad, Y.-X. Wang, and D. Forsyth. Diver: Real-time and accurate neural radiance fields with deterministic integration for volume rendering. In *CVPR*, pages 16200–16209, June 2022. 1, 2

[50] Y. Xiangli, L. Xu, X. Pan, N. Zhao, A. Rao, C. Theobalt, B. Dai, and D. Lin. Citynerf: Building nerf at city scale. *arXiv preprint:2112.05504*, 2021. 1

[51] Y. Xie, J. Xiao, M. Sun, C. Yao, and K. Huang. Feature representation matters: End-to-end learning for reference-based image super-resolution. In *ECCV*, pages 230–245. Springer, 2020. 2

[52] H. Xu, T. Alldieck, and C. Sminchisescu. H-nerf: Neural radiance fields for rendering and temporal reconstruction of humans in motion. *NeurIPS*, 34:14955–14966, 2021. 1

[53] Q. Xu, Z. Xu, J. Philip, S. Bi, Z. Shu, K. Sunkavalli, and U. Neumann. Point-nerf: Point-based neural radiance fields. In *CVPR*, pages 5438–5448, June 2022. 2

[54] X. Yan, W. Zhao, K. Yuan, R. Zhang, Z. Li, and S. Cui. Towards content-independent multi-reference super-resolution: Adaptive pattern matching and feature aggregation. In *ECCV*, pages 52–68. Springer, 2020. 2

[55] F. Yang, H. Yang, J. Fu, H. Lu, and B. Guo. Learning texture transformer network for image super-resolution. In *CVPR*, pages 5791–5800, 2020. 2, 4, 7, 8

[56] X. Yang, Z. Daquan, S. Liu, J. Ye, and X. Wang. Deep model reassembly. *arXiv preprint arXiv:2210.17409*, 2022. 2

[57] X. Yang, J. Ye, and X. Wang. Factorizing knowledge in neural networks. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIV*, pages 73–91. Springer, 2022. 2

[58] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *ICCV*, pages 5752–5761, October 2021. 1, 2

[59] A. Yu, V. Ye, M. Tancik, and A. Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *CVPR*, pages 4578–4587, June 2021. 2

[60] K. Zhang, G. Riegler, N. Snavely, and V. Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint:2010.07492*, 2020. 1

[61] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, pages 586–595, 2018. 5

[62] F. Zhao, W. Yang, J. Zhang, P. Lin, Y. Zhang, J. Yu, and L. Xu. Humannerf: Generalizable neural human radiance field from sparse inputs. *arXiv preprint arXiv:2112.02789*, 2021. 1

[63] H. Zheng, M. Ji, H. Wang, Y. Liu, and L. Fang. Crossnet: An end-to-end reference-based super resolution network using cross-scale warping. In *ECCV*, pages 88–104, 2018. 2