LensNeRF: Rethinking Volume Rendering based on Thin-Lens Camera Model

Min-Jung Kim Gyojung Gu Jaegul Choo
Graduate School of AI, KAIST
{emjay73, gyojung.gu, jchoo}@kaist.ac.kr

Abstract

*Recent advances in Neural Radiance Field (NeRF) show promising results in rendering realistic novel view images. However, NeRF and its variants assume that input images are captured using a pinhole camera and that subjects in images are always all-in-focus by tacit agreement. In this paper, we propose aperture-aware NeRF optimization and rendering methods using a thin-lens model (dubbed **LensNeRF**), which allows defocus images of any aperture size as input and output. To generalize a pinhole camera model to a thin-lens camera model in NeRF framework, we define multiple rays originating from the aperture area, solving world-to-pixel scale ambiguity. Also, we propose in-focus loss that assigns the given pixel color to points on the focus plane to alleviate the color ambiguity caused by the use of multiple rays. For the rigorous evaluation of the proposed method, we collect a real forward-facing dataset with different F-numbers for each viewpoint. Experimental results demonstrate that our method successfully fuses an aperture-size adjustable thin-lens camera model into the NeRF architecture, showing favorable qualitative and quantitative results compared to baseline models. The dataset will be made available.*

1. Introduction

3D-aware novel view synthesis has gained tremendous attention in recent years. Especially, NeRF [45] has shed light on this trend, showing impressive results on novel view synthesis. NeRF samples 3D points along the rays under the pinhole camera assumption. Sampled points and ray directions are used to query view-independent density and view-dependent color from the implicit MLP model. Obtained density and color values are integrated using a volume rendering scheme to generate novel view images. The success of 3D-aware novel view synthesis has given rise to tons of NeRF-based methods [1, 3, 4, 21, 36, 38, 39, 44, 49]. However, most of these studies overlook one important fact: We do not use pinhole cameras in real life.

Although a simple pinhole camera model works well in many cases, there are some scenarios that cannot be explained using the pinhole assumption. Such an assumption hinders NeRF-based methods from using images with defocus blur. Specifically, defocus blur caused by rays coming through the open aperture area cannot be described using a pinhole camera model. There are some works that try to bridge the gap between a pinhole camera model and a general camera model. Deblur-NeRF [20] accepts defocus images as input, recovering sharp neural radiance fields from them. However, this work focuses on the problem of removing blur, rather than embracing a thin-lens camera model to NeRF framework. Deblur-NeRF cannot be expanded to the novel view synthesis task for general cameras with varying aperture sizes, since they are utilizing the view-dependent blur kernel. NeRFocus [44] tries to mimic the behavior of a thin-lens camera using synthetic gaussian blur. Although the NeRFocus model can synthesize defocus images, this work cannot decompose or explain blur in input images, taking only all-in-focus images as input. Unlike other methods, DoF-NeRF [46] can remove blurs from the input images, or synthesize a defocus image. However, since its

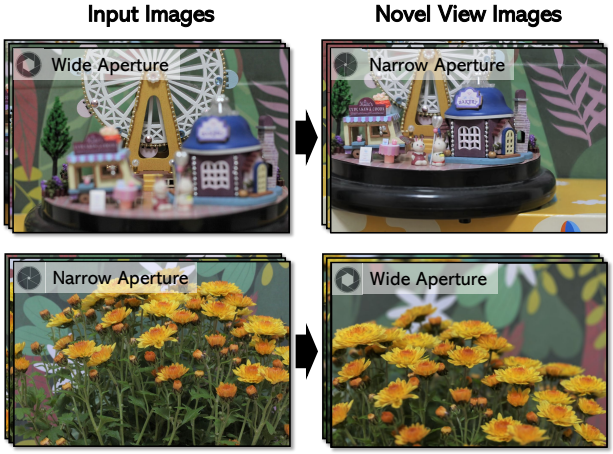

Figure 1. Our LensNeRF marries NeRF with a thin-lens camera model, enabling more general scenarios. LensNeRF facilitates the task of deblurring and defocusing in a single NeRF framework.

blur modeling relies on the geometric information learned from the pinhole-based NeRF model, DoF-NeRF fails to converge without initially training the pinhole-based NeRF model.

In this paper, we propose LensNeRF, a thin-lens camera-aware NeRF model. Unlike previous methods, LensNeRF not only grants defocus images as input but also renders novel view images with varying aperture sizes as output. To combine a thin-lens camera model with a NeRF model, we define multiple rays whose origins are distributed over the aperture area and the directions are refracted with respect to the ray origins accordingly. However, world space, where ray origins reside, is defined up-to-scale. Since the scale of world space is generally initialized using the first two frames chosen for structure-from-motion (SFM) registration, it causes scale ambiguity between the given scene and the camera space. This fact makes it difficult to define the exact aperture area for ray origin sampling. Even after we succeed in importing the thin-lens model into NeRF, there is color ambiguity caused by defocus blur. Estimating the true color of a blurred pixel is an ill-posed problem without problem relaxation.

Our LensNeRF resolves world-to-pixel scale ambiguity by explicitly optimizing it through sample positions, where the sample position is expressed with respect to the scale and aperture size. To alleviate color ambiguity caused by the defocus effect during training, the in-focus loss is proposed to maximally utilize color information from in-focus pixels. Our contributions boil down to the followings.

- We introduce the thin-lens camera model to NeRF framework for a multi-view scene optimization task, solving scale ambiguity.
- We propose an in-focus loss for a thin-lens camera model to alleviate color ambiguity caused by defocus blur and demonstrate its effectiveness on the deblur task.
- We collect a real-world dataset consisting of images with varying aperture sizes, which can be used for evaluating novel view, and novel F-number image synthesis tasks.
- By integrating the thin-lens camera model into the NeRF approach, we gain the ability to restore clear neural radiance fields from defocus images and to create defocus images from in-focus images.

2. Related Work

Novel View Synthesis Novel view synthesis has been actively explored in computer vision and graphics [7, 18, 19, 23, 31, 34, 41, 50]. After the introduction of Neural Radiance Field (NeRF) [24], NeRF-based view synthesis work has surged [1, 21, 21, 30, 49] because of its promising outcomes. These NeRF-based methods require multiple images taken from different positions and utilize a single ray to form a single pixel under the pinhole camera assumption. To overcome the aliasing artifacts, Mip-NeRF [3] constructs a cone-like frustum for rendering a pixel to simulate the continuous characteristic of a ray in the real-world. Accelerating NeRF train time without harming its performance has also gained attention. [8, 26, 38]. DVGO [38] optimizes the voxel grids directly with the shallow MLP, unifying the implicit and explicit representation. Nevertheless, the NeRF-based methodologies employed for generating new viewpoints differ in their physical principles of an actual camera in the real world, where multiple rays combine to form a single pixel.

Defocus When taking a picture using a camera with a large aperture, we can observe defocus blur in the image, especially around the region that is far from the in-focus plane. To mimic such a behavior of a real-world camera, there have been some attempts to synthesize the defocus images. The research on a light field camera has represented versatile applications like refocusing [25, 29] and novel view synthesis [11]. The studies on a depth estimation [2, 37, 42] show that the synthetic shallow depth-of-field images can be generated with a depth map estimated from stereo or monocular input. RawNeRF [22] uses raw images as train images and synthesizes novel defocus images by utilizing precomputed multiplane images during the rendering process. NeRFocus [44] acquires a defocus effect of a real-world camera by learning from synthetic Gaussian blurring. DoF-NeRF [46] estimates circle of confusion(CoC) based on the geometric information learned from the scene. It is worth noting that the methods mentioned earlier lack the ability to intuitively adjust blur size through conventional F-numbers. Instead, the adjustment of blur size necessitates the manipulation of hyper-parameter scales. In contrast, our proposed method allows users to input intuitive F-numbers, which in turn facilitates the synthesis of corresponding novel views, with adjusted F-number attributes.

Deblurring As a part of the image restoration, deblurring aims to recover the sharp image from the given blurry image. Conventional methods [6, 14, 47] for deblurring restore the sharp image by jointly optimizing the image and the blur kernel in the deconvolutional process. After the introduction of a neural network, the deblurring work [5, 15, 17, 27, 40, 48] that adopt a deep neural network have emerged. KPAC [35] utilizes the shared multiple atrous convolution filters for the efficient simulation of inverse kernels for the single image deblurring. For deblurred novel view synthesis task [9, 16, 20, 43], Deblur-NeRF [20] published the real-world dataset with blur (without ground-truth) and the NeRF architecture for deblurring. However, the primary emphasis of Deblur-NeRF lies in the elimination of image blur, rather than modifying the underlying camera model.

3. Preliminary

3.1. Volume Rendering

The concept of volume rendering is the integration of a neural radiance field through a ray. Following the notation in NeRF [24], estimated color $\hat{\boldsymbol{C}}$ of a ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$ is written as follows.

$$\hat{\boldsymbol{C}} \simeq \sum_{i=1}^{N} \boldsymbol{T}_i \boldsymbol{\alpha}_i \boldsymbol{c}_i, \tag{1a}$$

$$\boldsymbol{T}_i = \exp\left(- \sum_{j=1}^{j=i-1} \sigma_j \delta_j \right), \tag{1b}$$

Here, N is the number of sampled points over a ray, i is the index of a sample, δ_i is the distance between the adjacent samples, σ_i is the density of the i-th sample, $\boldsymbol{T}_i$ is the probability of not hitting any other sample points before the i-th sample, $\boldsymbol{\alpha}_i = 1 - \exp\left(-\sigma_i \delta_i\right)$ is the probability of hitting the i-th sample, and $\mathbf{c}_i$ is the estimated color of the i-th sample.

3.2. Thin-Lens Model

The assumption behind the thin-lens model is that the lens is so thin that the ray refracts only once on the surface of the lens, ignoring its thickness. The thin-lens model obeys the following rule.

$$\frac{1}{Z_C} + \frac{1}{Z_W} = \frac{1}{F}. \tag{2}$$

Eq. (2) implies the case when the object is located at Z_W amount away from the lens center, and rays originating from the object converge at a point with the distance Z_C. We can apply this equation directly to the camera model. When a thin-lens camera with a focal length L_I is focused at the point with distance L_F as shown in Fig. 4, we can rewrite the Eq. (2) as

$$\frac{1}{L_I} + \frac{1}{L_F} = \frac{1}{F}. \tag{3}$$

L_I and F in Eq. (3) are conventionally called *focal length* and *lens focal length* respectively. To prevent confusion, *focal length* will be named *camera focal length* afterward.

4. Method

Cameras in the real-world can manipulate their aperture size, adjusting the amount of light coming through the lens. Following this manner, the color of a pixel can be considered as the integration of rays coming through the open aperture area. Our goal is to embrace such a thin-lens camera model to NeRF. To this end, we define and utilize multiple rays originating from the aperture area. The overall architecture of the proposed method is described in Fig. 2. Modeling of ray origin and direction for the thin-lens camera is described in Sec. 4.1. Volume rendering scheme modified to fit in our thin-lens model is illustrated in Sec. 4.2 and how we circumvent scale ambiguity is elaborated in Sec. 4.3. The in-focus loss that can mitigate color ambiguity is elaborated in Sec. 4.4. Loss functions for optimizing neural radiance fields of a given scene are depicted in Sec. 4.5.

4.1. Thin-Lens Ray Modeling

In this section, given normalized coordinates $\boldsymbol{d}_{in}^{(i)}$ of a pixel $\boldsymbol{p}_i$, a bunch of refracted ray directions $\mathbf{d}_{out}^{(i,j)}$ that contribute to the formulation of the pixel $\boldsymbol{p}_i$ will be derived. Following the thin-lens camera ray tracing scheme [12, 33], ray origins and directions obey the Eq. (4),

$$\boldsymbol{d}_{out}^{(i,j)} = -\frac{1}{L_F}\boldsymbol{o}_j + \boldsymbol{d}_{in}^{(i)}, \tag{4}$$

which is also depicted in Fig. 4. Using Eq. (3), we can rewrite Eq. (4) as

$$\boldsymbol{d}_{out}^{(i,j)} = \left(\frac{1}{L_I} - \frac{1}{F}\right)\boldsymbol{o}_j + \boldsymbol{d}_{in}^{(i)}. \tag{5}$$

Note how Eq. (5) generalizes to the pinhole camera model when $\boldsymbol{o}_j$ accepts $[0,0,0]^T$ only, making $\boldsymbol{d}_{out}^{(i,j)}$ be the same as the $\boldsymbol{d}_{in}^{(i)}$. The camera focal length L_I and the lens focal length F are acquired through camera intrinsics and EXIF information, respectively. For the purpose of acquiring ray origins, a procedure of regular grid sampling is executed. This sampling process occurs within the boundaries of a square region that tightly encloses the aperture area. This approach is adopted to maintain computational tractability, satisfying

$$\boldsymbol{o}_j \in \mathcal{O}\{[X_j, Y_j, 0]^T \,|j \in \{0, 1, ..., N_o - 1\}\} \tag{6a}$$

$$X_j = \text{-}A_r + \frac{2\left(j \% \sqrt{N_o}\right) A_r}{\left(\sqrt{N_o} - 1\right)}, \tag{6b}$$

$$Y_j = \text{-}A_r + \frac{2\left(j // \sqrt{N_o}\right) A_r}{\left(\sqrt{N_o} - 1\right)}, \tag{6c}$$

where A_r is the aperture radius, and N_o is the number of sampled ray origins on the aperture area. Note that the ray origins are expressed with respect to the aperture radius A_r.

4.2. Thin-Lens Volume Rendering

From Sec. 4.1, we can define a ray $\mathbf{r}^{(i,j)} = \mathbf{o}_j + t\mathbf{d}_{out}^{(i,j)}$ that composes a pixel $\mathbf{p}_i$. Volume rendering rays $\mathbf{r}^{(i,j)}$ for all sampled ray origins $\mathbf{o}_j$ gives us a patch image $\hat{\mathbf{I}}_i \in \mathbb{R}^{3\times\sqrt{N_o}\times\sqrt{N_o}}$. Instead of directly using this patch image, we upscale it to obtain $\hat{\mathbf{I}}'_i \in \mathbb{R}^{3\times\sqrt{N'_o}\times\sqrt{N'_o}}$ so as to approximate a patch image obtained using densely sampled

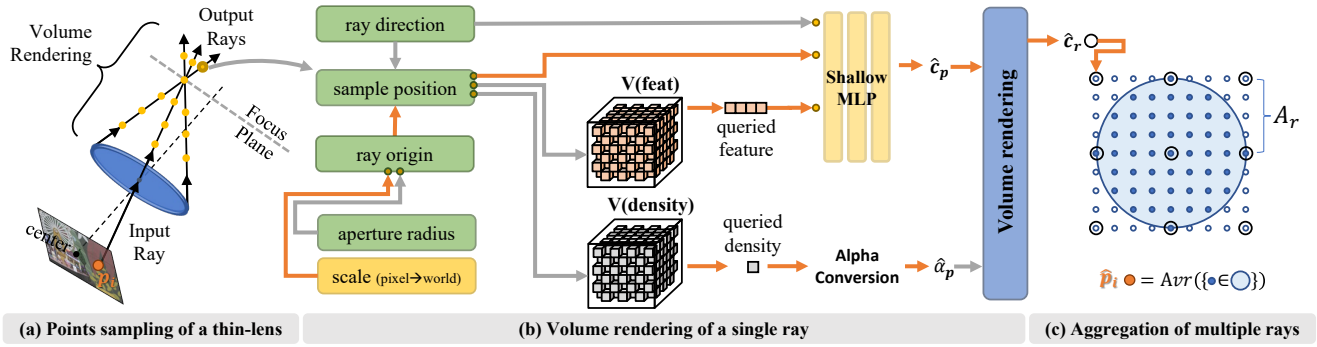

Figure 2. The overall architecture of our LensNeRF. (a) Unlike conventional NeRF, our LensNeRF samples points along multiple rays. (b) Aperture radius and scale converting pixel units to world units are used for computing sample position. We adapt DVGO [38] as our base method, directly optimizing features and densities in the voxel grids. The color $\hat{\mathbf{c}}_r$ is estimated following the volume rendering convention in Eq. (1) for each ray. (c) To estimate the color of queried pixel $\boldsymbol{p}_i$, we average the volume rendering results of multiple rays within the disk area as in Eq. (7). The figure shows the case when the N_o is 3×3 and N_o' is 9×9. Orange arrows represent the path where gradients flow during optimization.

ray origins. And then, the color values within the circular area are averaged to obtain the color of pixel $\mathbf{p}_i$. Estimated color of pixel $\mathbf{p}_i$ is written as follows:

$$\hat{\boldsymbol{C}}\left(\boldsymbol{p}_i\right) = \sum_{l=0}^{N_o'-1} \mathbb{1}\left(\boldsymbol{o}_l\right) \hat{\mathbf{I}}_i'\left(\mathbf{o}_l\right) / \sum_{l=0}^{N_o'-1} \mathbb{1}\left(\boldsymbol{o}_l\right), \tag{7}$$

where N_o' is the number of approximated ray origins, $\mathbf{o}_l$ is the l-th approximated ray origin, and $\mathbb{1}\left(\mathbf{o}_l\right)$ is the indicator that returns 1 if $\mathbf{o}_l$ is within the circular area with a radius of A_r, otherwise, 0. Eq. (7) is illustrated in Fig. 2(c). Here, we simply averaged colors within the aperture area instead of applying a weighted sum since the ray origins are sampled from evenly spaced grid positions and each ray covers roughly the same aperture area, endowing each ray with equal importance. A gamma correction function is omitted since we observed performance degradation. We can think of it as applying a gamma correction function $g(x) = x^{1/\gamma}$ with a gamma value of 1.

4.3. Scale Ambiguity

To sample ray origins within an aperture area with a radius of A_r, we should know what A_r is. Aperture radius can be obtained from the following EXIF exchangeable format:

$$A_r = \frac{\text{Lens Focal Length}}{2\,(\text{F-number})} = \frac{F}{2\,(\text{F-number})}. \tag{8}$$

Note that lens focal length F has the unit of *pixel*, and F-number has no unit since it is a ratio. Naturally, the unit of aperture radius A_r follows that of F, which is *pixel*. Since ray origins in Eq. (6) are expressed with respect to A_r, the unit of ray origins also is *pixel*. However, to sample points along the ray in 3D space for volume rendering, ray origins should be expressed in 3D space coordinates. Background color in Fig. 4 shows the unit difference between camera space and 3D world space.

To obtain scale $s_{3D\to pix}$ that converts the 3D space unit to the *pixel* unit, we assume that in-focus depth L_F in a 3D space unit is known. From this, we can rewrite Eq. (3) as follows:

$$\frac{1}{L_I} + \frac{1}{s_{3D\to pix} \cdot L_F} = \frac{1}{F}. \tag{9}$$

Note that L_I and F are obtainable from EXIF in *pixel* unit. Since we assume that in-focus depth L_F is known, the only unknown variable here is $s_{3D\to pix}$, making Eq. (9) solvable with respect to $s_{3D\to pix}$. To approximate L_F, sparse 3D points produced by COLMAP [32] are utilized. We approximate L_F as a median depth obtained by projecting 3D points to train images.

The derived $s_{3D\to pix}$ may have errors caused by rough estimates of in-focus depth L_F. To recoup this, residual for scale $\Delta s_{3D\to pix}$ is introduced and optimized explicitly.

$$s_{3D\to pix}^{(post)} = s_{3D\to pix}^{(\text{pre})} + s_{3D\to pix}^{(\text{pre})} \cdot \Delta s_{3D\to pix}. \tag{10}$$

To make gradients flow towards $\Delta s_{3D\to pix}$, the sample position is concatenated with shallow MLP inputs. Fig. 2(b) depicts how gradients flow towards $\Delta s_{3D\to pix}$ through the sample position and the ray origin.

4.4. In-Focus Loss

As stated in DeblurNeRF [20], deblurring images from view-consistent blurry pixels is an ill-posed problem. However, when there is an inconsistency in blur across multiple views, we can expect our model to decompose blur into multiple true-colors, trying to compensate for inconsistency. For additional ambiguity relaxation, we present the in-focus loss here.

When we define a sampled point $\mathbf{x}_{ik}$ along a ray through the lens center with respect to the ray origin $\mathbf{o}$ and direction $\mathbf{d}$, such as $\mathbf{x}_{ik} = \mathbf{o} + t_k\mathbf{d}$, the in-focus loss is defined as follows:

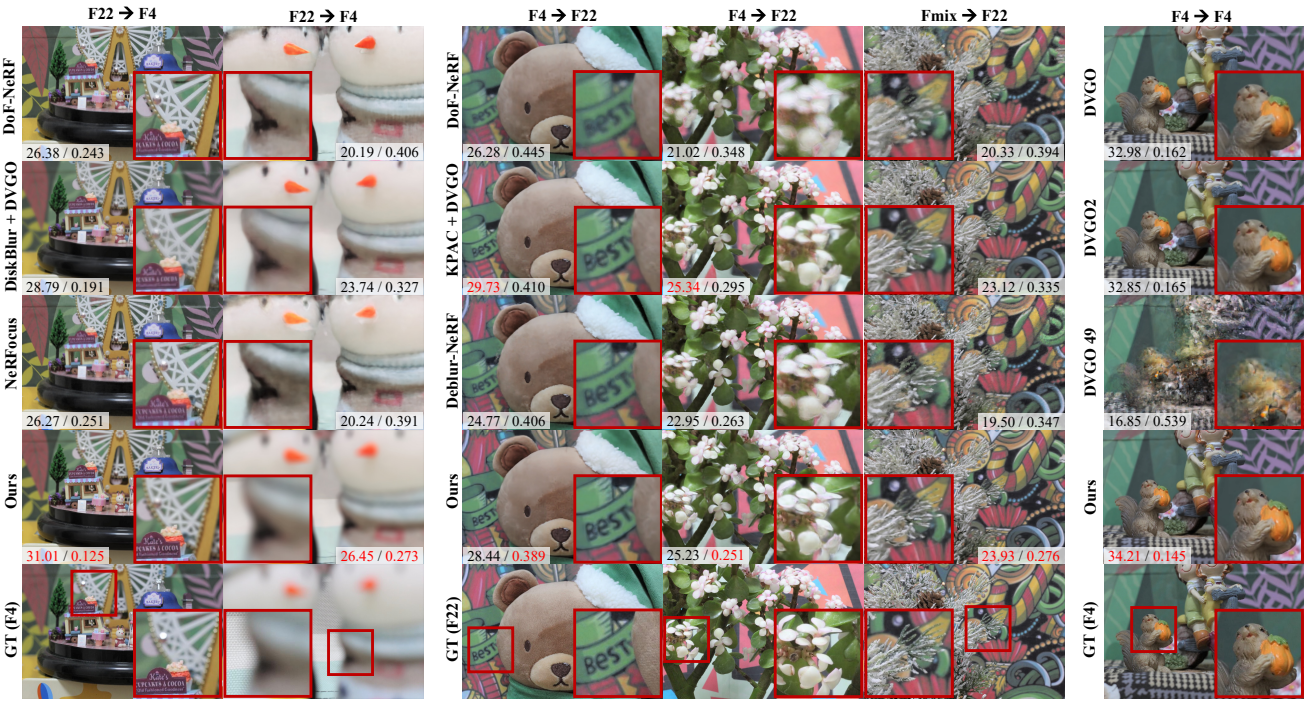

Figure 3. Qualitative results of novel view synthesis in a defocus, a deblur, and a classic task. (a) The purpose of the defocus task is to synthesize images captured with a wide aperture camera given images captured with a narrow aperture. (b) The deblur task aims to generate images captured with a small aperture camera given images captured with a wide aperture. (c) The goal of the classic task is to render novel view images under the assumption that the aperture size is fixed. The smaller F-number is, the wider the aperture size is. PSNR and LPIPS are written on the image and the best results are highlighted in red.

$$\mathcal{L}_{foc} = \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{K}} w_k \|\hat{\boldsymbol{c}}(\boldsymbol{x}_{ik}) - \boldsymbol{C}(\boldsymbol{p}_i)\|^2 / |\mathcal{I}|, \tag{11a}$$

$$w_k = \frac{\exp\left(-\tau \|t^{(foc)} - t_k\|^2\right)}{\sum_{\forall k} \exp\left(-\tau \|t^{(foc)} - t_k\|^2\right)}. \tag{11b}$$

Here, $\mathcal{I}$ is the indices of pixels in a batch, $\mathcal{K}$ is the indices of the sampled points along a single ray, $\boldsymbol{x}_{ik}$ is a sampled point, $\hat{\mathbf{c}}(\mathbf{x}_{ik})$ is the estimated color of the sample point, $\mathbf{C}(\mathbf{p}_i)$ is the true color of the i-th pixel, τ is the hyperparameter, and $t^{(foc)}$ is a scalar value that makes the sampled point be on the focus plane. This loss forces the color of the sampled points to have the color of the input pixel if the points are close to the focus plane. Fig. 4 illustrates how our in-focus loss works. Please note that w_k gets bigger as t_k becomes closer to the $t^{(foc)}$.

4.5. Training Objective

To optimize our LensNeRF framework, six different types of loss functions are employed.

$$\mathcal{L} = \lambda_p \mathcal{L}_{pho} + \lambda_e \mathcal{L}_{ent} + \lambda_t \mathcal{L}_{tv} + \lambda_d \mathcal{L}_{dis} + \lambda_f \mathcal{L}_{foc}. \tag{12}$$

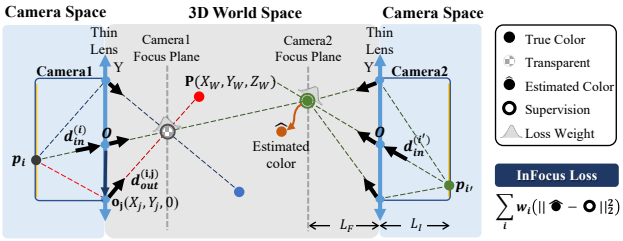

Figure 4. Illustration of our in-focus loss. The in-focus loss forces the focus plane to have the same color as the input image. Note that assigning the green color of $\mathbf{p}_{i'}$ to the focus plane of *Camera2* not only helps to determine the true color of a given 3D point but also helps to narrow down possible color compositions for gray pixel $\mathbf{p}_i$, such as (unknown, unknown, green), removing (gray, gray, gray) from the candidates.

First coming four types of losses $\mathcal{L}_{pho}$, $\mathcal{L}_{ent}$, $\mathcal{L}_{tv}$, $\mathcal{L}_{dist}$ are inherited from our cornerstone, DVGO [38, 39]. $\mathcal{L}_{pho}$ is photometric loss that minimizes mean square error between the target image and rendered image. $\mathcal{L}_{ent}$ is entropy loss responsible for enforcing accumulated alpha values in Eq. (7) to concentrate either on foreground or background. $\mathcal{L}_{tv}$ is TV loss that takes the role of preventing superfluous sharpness observed in explicit NeRF representation. $\mathcal{L}_{dist}$

is distortion loss initially proposed in Mip-NeRF360 [4] that is in charge of preventing floaters and background collapse. Please refer to the DVGO [38, 39] paper for a more detailed explanation. $\mathcal{L}_{foc}$ is in-focus loss in Sec. 4.4.

4.6. Iterative Scale Calibration

Algorithm 1: Iterative scale calibration.

Data: Input images I_n with EXIF, Camera intrinsics and extrinsics from COLMAP
Result: Trained NeRF model F_Θ

$s_{3D\to pix} \leftarrow s^{(init)}_{3D\to pix}$
for $i \leftarrow 0$ **to** N_{outer} **do**
  $\Theta \leftarrow$ random initialization
  $\Theta_{dir} \leftarrow 0$ and $lr_s \leftarrow$ 1e-4 $*$ (1e-2)$^{min(i,2)}$
  /* NeRF&Scale optimization */
  for $j \leftarrow 0$ **to** N_{inner1} **do**
    $\Theta_{\setminus dir} \leftarrow \Theta_{\setminus dir} - \nabla_{\Theta_{\setminus dir}}\mathcal{L}$
    $\Delta s_{3D\to pix} \leftarrow \nabla_{s_{3D\to pix}}\mathcal{L}$
  if $i+1 \neq N_{outer}$ **then**
    continue
  /* NeRF optimization */
  for $j \leftarrow N_{inner1}$ **to** N_{inner2} **do**
    $\Theta \leftarrow \Theta - \nabla_\Theta \mathcal{L}$

Algorithm 1 shows the iterative training scheme applied to LensNeRF. Jointly optimizing residual scale $\Delta s_{3D\to pix}$ and NeRF parameters Θ from scratch generally fails to converge. To handle this problem, we introduce an outer loop for calibrating $s_{3D\to pix}$. The NeRF parameters Θ are reinitialized when the outer loop starts. The learning rate lr_s for $s_{3D\to pix}$ diminishes as the index of the outer loop increases. Outer loops are early stopped when the inner loop iteration exceeds N_{inner1} except the last outer loop. We initialize MLP layer weights that are multiplied with the viewing direction as zero and freeze it for N_{inner1}, guiding NeRF network to learn view-independent color first, and develop view-dependent color from the view-independent color estimated by the previous inner loop.

5. Experiments

5.1. Dataset Acquisition

There are several datasets introduced by precedent studies for the novel view synthesis task [13, 23, 45]. However, none of these datasets is captured using a camera with a wide aperture setting. To properly evaluate our work, we construct a dataset for novel view synthesis with varying aperture sizes. Specifically, Canon EOS 550D with 50mm prime lens is used to capture images. For every position where the image is taken, the aperture value is manipulated to have four different F-numbers (F4, F5.6, F8, F22). When training and testing, a single F-number is selected from the four F-numbers for each viewpoint based on the task that we want to achieve. Exposure time is set accordingly so that the exposure value (Ev=Av+Tv=2 $\log_2$(F-number)-$\log_2$(exposure time)) is constant. Camera pose is obtained by COLMAP [32] using images with an F-number of 22. Mostly, we capture images of forward-facing scenes focused at the center. Starting from the first image, every eighth image is kept for evaluation. The number of collected scenes is nine in total.

5.2. Implementation Details

We use DVGO [38, 39] as our cornerstone, which imparts memory efficiency to the optimization process, providing us the room for multiple rays. Images are resized to 1296×864 both for the training and evaluation phase. As in DVGO framework, density and color are expressed using explicit-implicit hybrid representation. The density is explicitly queried from the density voxel grid. The color is acquired by employing shallow MLP to the concatenation of the queried feature representation, the viewing direction, and the sample position. Based on the values assigned for N_o, we define three models, named LensNeRF-D(Dense)/M(Moderate)/S(Sparse), having $49 = 7 \times 7$, $25 = 5 \times 5$, and $9 = 3 \times 3$ as N_o value respectively. N'_o is fixed as 11 during the training phase and dynamically changed based on the target F-number during the test phase, ranging from 1 to 11. To train our model, a single NVIDIA A100 GPU is used for each scene. Training takes 8 hours for the D(Dense) model. For the aforementioned hyper-parameters in Algorithm 1, we use 3 for N_{outer}, $40k$ for N_{inner1}, and $120k$ for N_{inner2}. We use a batch of 2048 rays for training. For the aforementioned hyper-parameters in Eq. (12), we use 0.4 for λ_f. The τ in Eq. (11) is set to 5.0. Other hyperparameter values follow those of our baseline, DVGO [38]. The lens focal length is obtained from the EXIF embedded in an image. Novel view synthesis takes 29s, 14s, 6.5s for LensNeRF-D/M/S respectively.

5.3. Comparative Study

Thanks to the introduction of the thin-lens camera model to NeRF, our LensNeRF can handle input images with any F-numbers and has the ability to synthesize novel view images with any F-number of interest. Since there are not many studies that fully match our experimental scenarios, we define three tasks and compare our method with the baselines of each task. We define the case where the F-number of input images is bigger than that of output images as a defocus task, the case where the F-number of input images is smaller than that of output images as a deblur task, and the case where the F-number of input images is identical to that of output images as a classic task. The brightness of the resulting images in qualitative results is adjusted for ease of comparison.

Defocus Task	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Methods	F22 → F4			F22 → F5.6		
DiskBlur + DVGO [38]	27.7064	0.8628	0.2946	28.1376	0.8537	0.2903
NeRFocus [44]	24.8951	0.7122	0.3460	25.2636	0.7156	0.3291
DoF-NeRF [46]	24.6593	0.7017	0.3411	25.3044	0.7128	0.3144
LensNeRF-D	**28.5347**	**0.8955**	**0.2344**	**28.7060**	**0.8849**	**0.2355**
LensNeRF-M	28.3465	0.8940	0.2368	28.4842	0.8835	0.2387
LensNeRF-S	27.7758	0.8838	0.2562	27.8823	0.8759	0.2487
Methods	F22 → F8			Fmix → F4		
DiskBlur + DVGO [38]	28.2544	0.8438	0.2932	N/A	N/A	N/A
NeRFocus [44]	25.3963	0.7132	0.3245	N/A	N/A	N/A
DoF-NeRF [46]	25.5785	0.7162	0.2981	24.1304	0.6976	0.3614
LensNeRF-D	**28.6910**	**0.8725**	**0.2432**	**29.4934**	**0.8974**	**0.2321**
LensNeRF-M	28.3881	0.8706	0.2474	28.9642	0.8955	0.2348
LensNeRF-S	27.8331	0.8643	0.2552	28.4776	0.8895	0.2431

Table 1. Quantitative results of novel view synthesis in the defocus task. All metrics are averaged over scenes. The **best** result is in boldface and the second-best result is underlined.

5.3.1 Defocus Task

Here, we compare our method with three baseline methods, DiskBlur+DVGO [38], NeRFocus [44] and DoF-NeRF [46]. DiskBlur+DVGO is a two-stage method that first synthesizes defocus images that match the target F-number, as in other studies [22, 37, 44], and then trains DVGO [38] using these synthetically blurred images.

Quantitative Results We summarize the quantitative results of our method and baseline methods in Tab. 1. Our method surpasses the other baseline performance in PSNR, SSIM, and LPIPS. Please note that as the input and output F-number difference is more dramatic, proposed LensNeRF outperforms baseline methods by a large margin.

Qualitative Results Qualitative results on the defocus task are shown in Fig. 3(a). From the first column, we can observe that the other baseline methods learn weak blur all over the pixels. Baseline results from the second column fail to synthesize the proper defocus blur. On the other hand, the proposed LensNeRF successfully model the defocus blur that reflects the depth variation. Overall, our LensNeRF model can synthesize novel view images with realistic defocus blur.

5.3.2 Deblur Task

In the deblur task, we compare our method with KPAC [35]+DVGO [38], DeblurNeRF [20], and DoF-NeRF [46]. Similar to DiskBlur+DVGO, KPAC+DVGO is a two-stage method. First, given images with defocus blur, KPAC(3-level) removes blur for each image, and then these blur-free images are used for DVGO training. Although we report PSNR as an evaluation metric in deblur task, the PSNR is known to be generous to blur [1, 28], which is not perfectly suitable for deblur task evaluation. Check the PSNR of the images in Fig. 3(b). Although our result is more crispy, there are some cases where the PSNR of KPAC+DVGO is higher than ours. To increase the credibility of our evaluation, we add one more metric, DISTS [10], which is known to be good for deblur task evaluation. For ease of comparison, we define AVG as in Mip-NeRF [3] to aggregate all metrics. One difference from AVG in Mip-NeRF [3] is that our AVG reflects DISTS. AVG aggregats all metric measurements using the following formula: $10^{-\text{PSNR}/10} \times \sqrt{1-SSIM} \times \text{LPIPS} \times \text{DISTS}$. The lower AVG value represents a better result.

Deblur Task	PSNR ↑	SSIM ↑	LPIPS ↓	DISTS ↓	AVG ↓
Methods	F4 → F22				
KPAC [35] + DVGO [38]	**26.8882**	0.7920	0.3651	0.1891	0.0896
Deblur-NeRF [20]	23.4495	0.6363	0.3373	0.1743	0.1125
DoF-NeRF [46]	23.9006	0.6341	0.4068	0.2269	0.1228
LensNeRF-D	26.4639	**0.8001**	**0.3239**	**0.1624**	**0.0854**
LensNeRF-M	25.9477	0.7964	0.3281	0.1659	0.0889
LensNeRF-S	25.9926	0.7915	0.3398	0.1714	0.0904
Methods	F5.6 → F22				
KPAC [35] + DVGO [38]	**27.5523**	0.8068	0.3331	0.1683	0.0811
Deblur-NeRF [20]	23.1585	0.6309	0.3158	0.1577	0.1100
DoF-NeRF [46]	24.8471	0.6663	0.3682	0.2009	0.1088
LensNeRF-D	27.1412	**0.8141**	**0.2990**	**0.1491**	**0.0781**
LensNeRF-M	27.1385	0.8126	0.3024	0.1503	0.0785
LensNeRF-S	26.8368	0.8059	0.3124	0.1561	0.0817
Methods	F8 → F22				
KPAC [35] + DVGO [38]	**27.8307**	0.8156	0.3110	0.1540	0.0763
Deblur-NeRF [20]	23.3319	0.6399	0.2869	0.1432	0.1034
DoF-NeRF [46]	25.0532	0.6757	0.3488	0.1857	0.1036
LensNeRF-D	27.4203	**0.8236**	**0.2817**	**0.1387**	**0.0738**
LensNeRF-M	27.2740	0.8217	0.2855	0.1393	0.0749
LensNeRF-S	27.1298	0.8170	0.2923	0.1443	0.0769
Methods	Fmix → F22				
KPAC [35] + DVGO [38]	**27.9776**	0.8147	0.3165	0.1578	0.0765
Deblur-NeRF [20]	23.1888	0.6305	0.2878	0.1437	0.1048
DoF-NeRF [46]	23.1072	0.6290	0.4053	0.2615	0.1333
LensNeRF-D	27.8825	**0.8275**	**0.2803**	**0.1366**	**0.0713**
LensNeRF-M	27.4536	0.8247	0.2862	0.1392	0.0740
LensNeRF-S	27.1647	0.8198	0.2898	0.1421	0.0761

Table 2. Quantitative results of the deblur task. All metrics are averaged over scenes. The **best** result is in boldface and the second-best result is underlined.

Quantitative Results Quantitative results on the deblur task can be found in Tab. 2. KPAC+DVGO generally shows high PSNR, SSIM, LPIPS, and DISTS values, meaning that the resulting images are numerically similar to the ground-truth images, and Deblur-NeRF generally has low PSNR, SSIM, LPIPS, and DISTS values, meaning that the resulting images are visually plausible to human perception. Our LensNeRF shows harmoniously good results in all four metrics in general, showing superior results in an aggregated score, AVG.

Qualitative Results Fig. 3(b) shows the qualitative result of the deblur task. From the qualitative results, we confirm that the proposed LensNeRF can synthesize crispy novel view images only from defocus images. Check out not only the region highlighted by the red box but how much the background region is clear enough compared to the other base-

Classic Task	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Methods	F4 → F4			F5.6 → F5.6		
DVGO	29.2202	0.8709	0.2745	29.2462	0.8625	0.2688
DVGO2	29.2957	0.8721	0.2790	29.2812	0.8640	0.2736
DVGO49	23.0537	0.7691	0.4046	22.9716	0.7499	0.4097
LensNeRF-D	**30.1185**	**0.8951**	**0.2360**	29.3277	**0.8834**	**0.2372**
LensNeRF-M	29.6116	0.8933	0.2385	**29.4338**	0.8822	0.2401
LensNeRF-S	29.2035	0.8882	0.2453	28.7302	0.8757	0.2482
Methods	F8 → F8			F22 → F22		
DVGO	29.1323	0.8534	0.2676	28.6261	0.8266	0.2906
DVGO2	**29.2261**	0.8550	0.2736	**28.7760**	0.8291	0.2982
DVGO49	22.7101	0.7326	0.4130	22.1168	0.6903	0.4551
LensNeRF-D	29.1582	**0.8706**	**0.2458**	28.1451	**0.8365**	**0.2644**
LensNeRF-M	28.7341	0.8685	0.2486	27.7949	0.8338	0.2691
LensNeRF-S	28.5137	0.8639	0.2565	27.5721	0.8288	0.2739

Table 3. Quantitative results of the classic task. All metrics are averaged over scenes. The **best** result is in boldface and the second-best result is underlined.

line methods.

5.3.3 Classic Task

The classic task assumes that the input images and the output images are taken from the same F-number. A conventional NeRF optimization using multi-view images can be considered as a special case of the classic task, where input and output images are captured using a pinhole camera. Here, we expand the camera model other than the pinhole camera. From Tab. 3, we can confirm that the proposed LensNeRF has better interpretability power of a given scene, synthesizing more ground-truth-like novel view than our base model DVGO [38, 39]. The larger the aperture size is, the more superior our LensNeRF is compared to the others. To show the effect of sampling multiple points, we report the result of two models, DVGO2 and DVGO49. Each represents the DVGO model with an increased number of point samples, where DVGO2 uses two times more points than those of DVGO and DVGO49 uses 49 times more. The experiment shows that doubling the number of sampled points achieves a slight performance gain. Sampling 49 times more points fails to converge to plausible results. From this, we can infer that the achievement of our LensNeRF is not simply due to using more points. Qualitative results can be found in Fig. 3(c).

5.4. Ablation Study

For the ablation study, we subsample three scenes (AmusementPark, Gink, Sheep) and use them for analysis.

In-Focus Loss and Scale Optimization We conduct an ablation study to show the effectiveness of each component. As in-focus loss is designed to support deblur task, we perform the ablation study on the most representative deblur task, F4→F22. Also, to show the effectiveness of the scale optimization, we further perturbed initial in-focus depth L_F estimated from Sec. 4.3 by 30%. From Tab. 4, we can verify that using in-focus loss is advantageous for the deblurring task. Scale optimization also contributes to the impressive performance enhancement of LensNeRF framework. Check Fig. 5 for the effectiveness of applying in-focus loss and scale optimization in a qualitative manner.

Deblur Task			PSNR ↑	SSIM ↑	LPIPS ↓	DISTS ↓	AVG ↓
Methods	IL	SO	F4 → F22				
LensNeRF (none)	X	X	19.4070	0.6232	0.4828	0.2351	0.1681
LensNeRF (IL)	O	X	21.7125	0.7075	0.4616	0.2214	0.1389
LensNeRF (SO)	X	O	25.6596	0.8418	**0.2747**	0.1395	0.0802
LensNeRF (full)	O	O	**26.1644**	**0.8447**	0.2753	**0.1386**	**0.0777**

Table 4. Ablation study of novel view synthesis with the deblur task. IL refers to in-focus loss and SO refers to scale optimization. All metrics are averaged over scenes. The **best** result is in boldface and the second-best result is underlined.

Figure 5. Effectiveness of using in-focus loss and scale optimization for deblur task. IL means in-focus loss, and SO means scale optimization. As we can see, with in-focus loss, color ambiguity caused by defocus blur can be mitigated.

6. Discussion and Conclusion

Limitations LensNeRF is specially designed to interpret the thin-lens model, so our model cannot remove motion blur as Deblur-NeRF [20] does. Also, LensNeRF requires comparatively large memory and computation, utilizing multiple rays that originate from a thin-lens aperture area. Lastly, LensNeRF requires EXIF metadata such as camera focal length L_I and lens focal length F.

Conclusion In this paper, we propose LensNeRF, a NeRF framework that generalizes volume rendering procedure for a thin-lens camera model. To this end, we suggest a way to solve scale ambiguity, bridging the world coordinates and the pixel coordinates. Introducing a thin-lens camera model allows us to perform not only the classic NeRF optimization task but also the deblur task and the defocus task. To boost the performance of LensNeRF, especially for the deblur case, newly designed in-focus loss is introduced to fully exploit view inconsistency over images and to alleviate color ambiguity caused by defocus blur. Experimental results show that our LensNeRF achieves surpassing or on-par performance compared to the baselines.

Acknowledgements This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-00075, Artificial Intelligence Graduate School Program (KAIST)), and the Ministry of Culture, Sports and Tourism and Korea Creative Content Agency (Project Number: R2021040097, Contribution Rate: 50).

References

[1] 1, 2, 7

[2] Jonathan T Barron, Andrew Adams, YiChang Shih, and Carlos Hernández. Fast bilateral-space stereo for synthetic defocus. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2015. 2

[3] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. 1, 2, 7

[4] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022. 1, 6

[5] Ayan Chakrabarti. A neural approach to blind motion deblurring. In *European Conference on Computer Vision*, 2016. 2

[6] T.F. Chan and Chiu-Kwong Wong. Total variation blind deconvolution. *IEEE Transactions on Image Processing*, 7(3):370–375, 1998. 2

[7] Anpei Chen, Minye Wu, Yingliang Zhang, Nianyi Li, Jie Lu, Shenghua Gao, and Jingyi Yu. Deep surface light fields. *In Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 2018. 2

[8] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision (ECCV)*, 2022. 2

[9] Peng Dai, Yinda Zhang, Xin Yu, Xiaoyang Lyu, and Xiaojuan Qi. Hybrid neural rendering for large-scale scenes with motion blur. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 154–164, 2023. 2

[10] Keyan Ding, Kede Ma, Shiqi Wang, and Eero P Simoncelli. Image quality assessment: Unifying structure and texture similarity. *IEEE transactions on pattern analysis and machine intelligence*, 44(5):2567–2581, 2020. 7

[11] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM Transactions on Graphics (TOG)*, 35(6):1–10, 2016. 2

[12] Takuhiro Kaneko. Ar-nerf: Unsupervised learning of depth and defocus effects from natural images with aperture rendering neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18387–18397, 2022. 3

[13] 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), 2017. 6

[14] Dilip Krishnan, Terence Tay, and Rob Fergus. Blind deconvolution using a normalized sparsity measure. In *CVPR 2011*, pages 233–240, 2011. 2

[15] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiri Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018. 2

[16] Dogyoon Lee, Minhyeok Lee, Chajin Shin, and Sangyoun Lee. Dp-nerf: Deblurred neural radiance field with physical scene priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12386–12396, 2023. 2

[17] Junyong Lee, Hyeongseok Son, Jaesung Rim, Sunghyun Cho, and Seungyong Lee. Iterative filter adaptive network for single image defocus deblurring. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2021. 2

[18] Yiyi Liao, Katja Schwarz, Lars Mescheder, and Andreas Geiger. Towards unsupervised learning of generative models for 3d controllable image synthesis. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 2

[19] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Transactions on Graphics (TOG)*, 2019. 2

[20] Li Ma, Xiaoyu Li, Jing Liao, Qi Zhang, Xuan Wang, Jue Wang, and Pedro V. Sander. Deblur-nerf: Neural radiance fields from blurry images. *arXiv preprint arXiv:2111.14292*, 2021. 1, 2, 4, 7, 8

[21] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7210–7219, 2021. 1, 2

[22] Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul P. Srinivasan, and Jonathan T. Barron. NeRF in the dark: High dynamic range view synthesis from noisy raw images. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2022. 2, 7

[23] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 2019. 2, 6

[24] 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 *European conference on computer vision*, pages 405–421. Springer, 2020. 2, 3

[25] Kaushik Mitra and Ashok Veeraraghavan. Light field denoising, light field superresolution and stereo camera based refocussing using a gmm light field patch prior. In *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 22–28. IEEE, 2012. 2

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

[27] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene

deblurring. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2

[28] Peter Ndajah, Hisakazu Kikuchi, Masahiro Yukawa, Hidenori Watanabe, and Shogo Muramatsu. Ssim image quality metric for denoised images. In *Proc. 3rd WSEAS Int. Conf. on Visualization, Imaging and Simulation*, pages 53–58, 2010. 7

[29] Ren Ng. *Digital light field photography*. stanford university, 2006. 2

[30] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 2

[31] Gernot Riegler and Vladlen Koltun. Free view synthesis. *European Conference on Computer Vision*, 2020. 2

[32] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4104–4113, 2016. 4, 6

[33] Peter Shirley. Ray tracing in one weekend., December 2020. https://raytracing.github.io/books/RayTracingInOneWeekend.html. 3

[34] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Niessner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2019. 2

[35] Hyeongseok Son, Junyong Lee, Sunghyun Cho, and Seungyong Lee. Single image defocus deblurring using kernel-sharing parallel atrous convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2642–2650, 2021. 2, 7

[36] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7495–7504, 2021. 1

[37] Pratul P Srinivasan, Rahul Garg, Neal Wadhwa, Ren Ng, and Jonathan T Barron. Aperture supervision for monocular depth estimation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018. 2, 7

[38] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022. 1, 2, 4, 5, 6, 7, 8

[39] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Improved direct voxel grid optimization for radiance fields reconstruction. *arXiv preprint arXiv:2206.05085*, 2022. 1, 5, 6, 8

[40] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[41] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: image synthesis using neural textures. *ACM Transactions on Graphics (TOG)*, 2019. 2

[42] Neal Wadhwa, Rahul Garg, David E Jacobs, Bryan E Feldman, Kanazawa Nori, Robert Carroll, Yair Movshovitz-Attias, Jonathan T Barron, Yael Pritch, and Marc Levoy. Synthetic depth-of-field with a single-camera mobile phone. *ACM Transactions on Graphics (TOG)*, 2018. 2

[43] Peng Wang, Lingzhe Zhao, Ruijie Ma, and Peidong Liu. Bad-nerf: Bundle adjusted deblur neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4170–4179, 2023. 2

[44] Yinhuai Wang, Shuzhou Yang, Yujie Hu, and Jian Zhang. Nerfocus: Neural radiance field for 3d synthetic defocus. *arXiv preprint arXiv:2203.05189*, 2022. 1, 2, 7

[45] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. Nerf--: Neural radiance fields without known camera parameters. *arXiv preprint arXiv:2102.07064*, 2021. 1, 6

[46] Zijin Wu, Xingyi Li, Juewen Peng, Hao Lu, Zhiguo Cao, and Weicai Zhong. Dof-nerf: Depth-of-field meets neural radiance fields. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 1718–1729, 2022. 1, 2, 7

[47] Li Xu, Shicheng Zheng, and Jiaya Jia. Unnatural l0 sparse representation for natural image deblurring. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1107–1114, 2013. 2

[48] Kaihao Zhang, Wenhan Luo, Yiran Zhong, Lin Ma, Bjorn Stenger, Wei Liu, and Hongdong Li. Deblurring by realistic blurring. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2737–2746, 2020. 2

[49] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 1, 2

[50] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. stereo magnification: learning view synthesis using multiplane images. *ACM Transactions on Graphics (TOG)*, 2018. 2