Ray Priors through Reprojection: Improving Neural Radiance Fields for Novel View Extrapolation

Jian Zhang[1*] Yuanqing Zhang[1,2*] Huan Fu[1†] Xiaowei Zhou[2] Bowen Cai[1]
Jinchi Huang[1] Rongfei Jia[1] Binqiang Zhao[1] Xing Tang[1]

[1]Tao Technology Department, Alibaba Group
[2]State Key Lab of CAD&CG, Zhejiang University

Abstract

Neural Radiance Fields (NeRF) [22] have emerged as a potent paradigm for representing scenes and synthesizing photo-realistic images. A main limitation of conventional NeRFs is that they often fail to produce high-quality renderings under novel viewpoints that are significantly different from the training viewpoints. In this paper, instead of exploiting few-shot image synthesis, we study the novel view extrapolation setting that (1) the training images can well describe an object, and (2) there is a notable discrepancy between the training and test viewpoints' distributions. We present RapNeRF (RAy Priors) as a solution. Our insight is that the inherent appearances of a 3D surface's arbitrary visible projections should be consistent. We thus propose a random ray casting policy that allows training unseen views using seen views. Furthermore, we show that a ray atlas pre-computed from the observed rays' viewing directions could further enhance the rendering quality for extrapolated views. A main limitation is that RapNeRF would remove the strong view-dependent effects because it leverages the multi-view consistency property.

1. Introduction

A primary target of the computer graphics community is to enable photo-realistic rendering of virtual worlds efficiently. Physics-inspired graphics techniques well approach real-time rendering and photo-realistic imagery creation but suffer from expensive manual content generations of geometries, materials, and other aspects of scenes. The past several years have seen an explosion of interest in neural rendering [17, 18, 21, 38, 39, 47], which models physical knowledge in deep networks to address reconstruction and rendering in a single formulation for controllable image generation.

Leveraging neural volume rendering, a recent advance Neural Radiance Fields (NeRF [22]) learn to represent 3D scenes from images and impressively support photo-realistic image synthesis. The visual quality of the generated images is even competitive with ones produced by physically-based rendering pipelines. One of NeRF's main limitations is that it requires many images to reconstruct a scene's geometry and texture details. Thereby, several subsequent works focus on investigating few-shot or unsupervised radiance fields reconstruction [4, 11, 52]. These works assume that we only observe several images of a scene. In an extreme setting, some geometries and appearances of the scene are never observed.

In this paper, we investigate NeRF from an object reconstruction perspective like [9, 33, 51], and restrict our focus on solid and non-transparent objects. We find that, even with enough images that can well describe an object, conventional NeRFs often fail to produce high-quality renderings for novel viewpoints that are significantly different from the training viewpoints. This observation motivates us to study the novel view extrapolation setting as explained in Figure 1. We take inspiration from the insight that the inherent appearances of a 3D surface's arbitrary visible projections should be consistent. It has been well studied in unsupervised 3D object reconstruction and texture optimization works [10, 15, 41, 55]. We dig into the multi-view consistency property on NeRF's formulation and present RapNeRF as a solution.

In specific, we propose a random ray casting (RRC) policy that randomly generates rays within a cone for each training ray in an online fashion. This training strategy is simple yet effective in creating supervisions for potential unseen views using seen views. RRC relies on the target objects' rough 3D meshes (R3DMs), which can be extracted from their pre-trained NeRFs. Furthermore, we prudently rethink the tradeoff between strong view-dependent effects

*These authors contribute equally to this work.
†Corresponding author.

Figure 1. **Observation & Setting.** *Left:* We study the novel view extrapolation setting that (1) the training images can well describe the objects, and (2) the test viewpoints are significantly different from the training viewpoints. We take a specific object as an example to illustrate the training (Red) and test (Blue) viewpoints in MobileObject. The viewpoints labeled in "Gray" are discarded. *Right:* For novel view extrapolation, NeRF [22] produces images that usually contains artifacts, while RapNeRF can generate high-quality renderings.

and multi-view consistent renderings. We show that a ray atlas (RA) computed from the training ray's viewing directions could further improve the rendering quality of extrapolated views. RapNeRF is empowered by both RRC and RA in a unified formulation.

To study the novel view exploration setting, we resplit the synthetic training and test images of NeRF's objects to construct the Synthetic-NeRF* [22]. We also capture eight scenes with real objects via a mobile phone to build a MobileObject dataset. A sample is illustrated in Figure 1 (right). Experiments demonstrate the superiority of RapNeRF in synthesizing promising novel views compared to state-of-the-art approaches. We conduct various ablation studies to discuss the core components of RapNeRF. Last but not least, a major limitation of RapNeRF is that it trades some view-dependent effects for better novel view exploration performance. We provide it a remedy by studying the deferred NeRF architecture in [9].

2. Related Work

Neural Rendering (NR) bypasses mesh reconstruction to perform view synthesis of real scenes directly. It constructs an implicit scene representation from a few input images token in different viewpoints and lighting conditions [38]. This implicit scene representation can be utilized to synthesize high-quality novel images when giving some guidance. Among the NR literature, Neural Volume Rendering (NVR) supports producing photo-realistic renderings of scenes [1–3, 6, 20, 22, 29, 31, 33, 34, 37, 45]. A good practice is from Neural Radiance Fields (NeRF) [22]. It proposes to represent a continuous scene as the neural radiance fields and leverages volume rendering to achieve high-quality view synthesis.

Leveraging the success of NeRF, there are many subsequent works that have been presented for better and more efficient view synthesis [8,13,16,17,19,23,27,36,42,51,53]. For example, Neural Sparse Voxel Fields (NSVF) [17] studies a progressive training strategy to exploit sparse voxel octrees for local geometry properties modeling. The obtained model largely improve NeRF in both rendering quality and speed. Recently, PlenOctree [51] shows another milestone which realizes real-time view synthesis with preserved visual quality. Unlike NeRF's representation, PlenOctree investigates the spherical harmonic function for color computation. Other routines include deformable or dynamic scene synthesis [7, 28, 30, 32, 40], learnable camera poses [14, 44, 50], and editable view synthesis [25, 46].

Several works learn few-shot view synthesis by conditioning a NeRF on image inputs [52] and exploiting the semantic consistency of multi-view features [11]. In the few-shot setting, they only observe several images of a scene. Some geometries and appearances are not covered by these images. This makes the neural reconstruction process particularly challenging. In contrast, we study a relaxed novel view extrapolation setting that the training images are enough to well describe an object. There is a great concurrent work RegNeRF [24] regularizes the geometry and appearance leverages the multi-view consistency property to obtain 3D-consistent representations.

3. Methodology

Our main goal is to close the visual quality gap between interpolated and extrapolated views synthesized by NeRF [22] series. In this section, we begin with a brief review of NeRF in Sec. 3.1. Then, we present the proposed random ray casting (RRC) policy in Sec. 3.2 and explain the ray atlas (RA) in Sec. 3.3. RRC and RA rely on objects' rough 3D meshes, which are extracted from the pre-trained NeRFs. Finally, we present how we train RapNeRF in Sec. 3.4.

3.1. Preliminaries: NeRF

A volume scene representation can be seen as a radiance field (or a 5D vector-valued function) that takes a 3D location $\mathrm{x} = (x, y, z)$ and 2D viewing direction $\mathrm{d} = (\theta, \Phi)$ as input, and output an emitted color (or radiance) $\mathrm{c} = (r, g, b)$ and a volume density σ. NeRF [22] adopts a single MLP

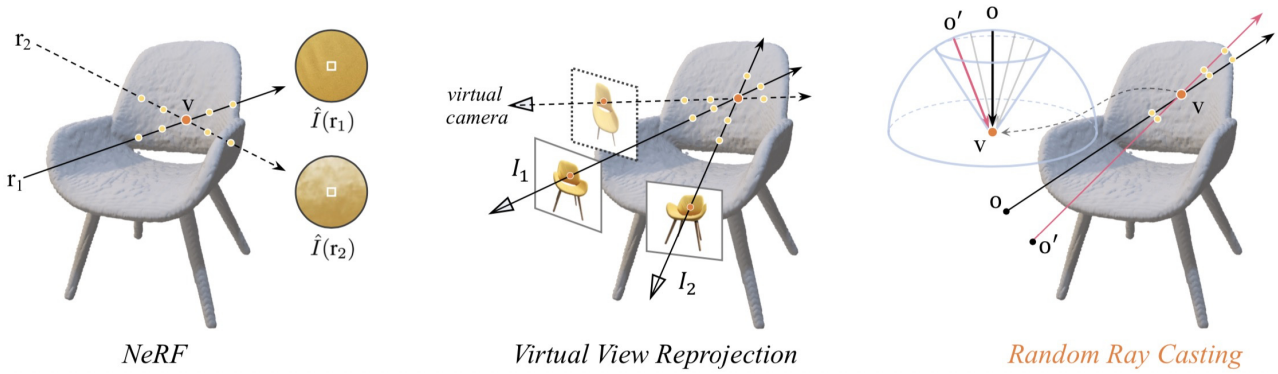

Figure 2. **Random Ray Casting.** *Left:* r_1 lies in the training space, and r_2 is distant from training rays. The radiance accumulation operation along r_2 is more likely to provide an adverse color estimation of v compared to $\hat{I}(\mathrm{r}_1)$. *Middle*: A straightforward virtual view reprojection idea, which is inconvenient. *Right*: For a specific training ray (casting from o and passing through v), the random ray casting (RRC) policy randomly generate an unseen virtual ray (casting from o' passing through v) within a cone, then assign it a pseudo label based on the training ray in an online manner. RRC enables training unseen rays using seen rays. See Sec. 3.2 for detailed explanations.

network to approximate the 5D function as the neural radiance fields:

$$\sigma, \mathrm{c} = F(\mathrm{d}, \mathrm{x}). \tag{1}$$

The formulation can be further decomposed as $F_\sigma : \mathrm{x} \rightarrow (\sigma, \mathrm{f})$ and $F_\mathrm{c} : (\mathrm{d}, \mathrm{f}) \rightarrow \mathrm{c}$.

To render a pixel of image I, NeRF casts a ray r from the camera's center of projection o along the direction d passing through the pixel. It samples N points along the ray and approximate the pixel's color $\hat{I}(\mathrm{r})$ following:

$$\begin{aligned} \hat{I}(\mathrm{r}) &= \sum_{i=1}^{N} T_i (1 - \exp(-\sigma_i \delta_i)) \mathrm{c}_i, \\ T_i &= \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j), \end{aligned} \tag{2}$$

where $\delta_i = t_{i+1} - t_i$ denotes the distance between two consecutive samples, c_i and σ_i are the radiance and volume density of a sample point $\mathrm{r}(t_i) = \mathrm{o} + t_i \mathrm{d}$, and T_i represents the accumulated transmittance from $\mathrm{r}(t_1)$ to $\mathrm{r}(t_i)$. In practice, t_i are bounded by a predefined intervals $[t_n, t_f]$. NeRF minimizes the squared error between the rendered and true pixel colors ($\hat{I}(\mathrm{r})$ *vs.* $I(\mathrm{r})$) to learn its MLP.

3.2. Random Ray Casting

We start with a further discussion of the aforementioned visual quality gap from the ray casting and neural mapping perspectives. As shown in Figure 2 (left), r_1 and r_2 are two rays that view a 3D point v in two directions, where the former lies in the training space, and the latter (a test ray) is distinct from the training rays. We may have a sense that the radiance of some samples along r_2 would be imprecise considering both the distribution shift and the mapping function $F_\mathrm{c} : (\mathrm{r}, \mathrm{f}) \rightarrow \mathrm{c}$. In further, the radiance accumulation operation along r_2 is more likely to provide an adverse color estimation compared to $\hat{I}(\mathrm{r}_1)$. We can see from the synthesized small regions around $\hat{I}(\mathrm{r}_2)$ and $\hat{I}(\mathrm{r}_1)$, the former contains more artifacts.

Our intuition is to create supervisions for potential unseen views by exploiting the multi-view consistency property. The property has been well studied in reprojection-based unsupervised 3D object reconstruction and texture optimization approaches [10, 15, 41, 48, 55]. Here in the NeRF formulation, we can naively follow the pipeline of generating some virtual cameras and their views, computing the involved pixel-wise rays, finding the corresponding rays for each virtual ray that hit the same 3D surface point from the training ray pool. It likes a pseudo-label generation process for virtual rays through multi-view projection. See Figure 2 (middle) for an illustration of this virtual view reprojection solution. In practice, the offline workflow is inconvenient.

Our random ray casting (RRC) policy allows pseudo-label assignment for randomly generated virtual rays in an online manner. Specifically, for an interested pixel in one training image I, we are given its viewing direction d, camera origin o, and depth value t_z in the world coordinate system, and ray $\mathrm{r} = \mathrm{o} + t\mathrm{d}$. Here, $t_\mathrm{z} = \sum_{i=1}^{N} T_i (1 - \exp(-\sigma_i \delta_i)) t_i$ is pre-computed and stored using the pre-trained NeRF. Let $\mathrm{v} = \mathrm{o} + t_\mathrm{z}\mathrm{d}$ denote the closest 3D surface point hit by r. In the training phase, as shown in Figure 2 (right), we regard v as a new origin, and randomly cast a ray from v within the cone whose central ray is the vector $\overrightarrow{\mathrm{vo}} = -t_\mathrm{z}\mathrm{d}$. This can be easily implemented by converting $\overrightarrow{\mathrm{vo}}$ to the spherical space, and introducing some randomness $\Delta\varphi$ and $\Delta\theta$ to φ and θ. Here, φ and θ are the *Azimuth* and *Elevation* of $\overrightarrow{\mathrm{vo}}$, respectively. $\Delta\varphi$ and $\Delta\theta$ are uniformly sampled from a pre-defined interval $[-\eta, \eta]$. With this operation, we obtain $\theta' = \theta + \Delta\theta$ and $\varphi' = \varphi + \Delta\varphi$, thus can generate a virtual ray r' casting from a random origin o'

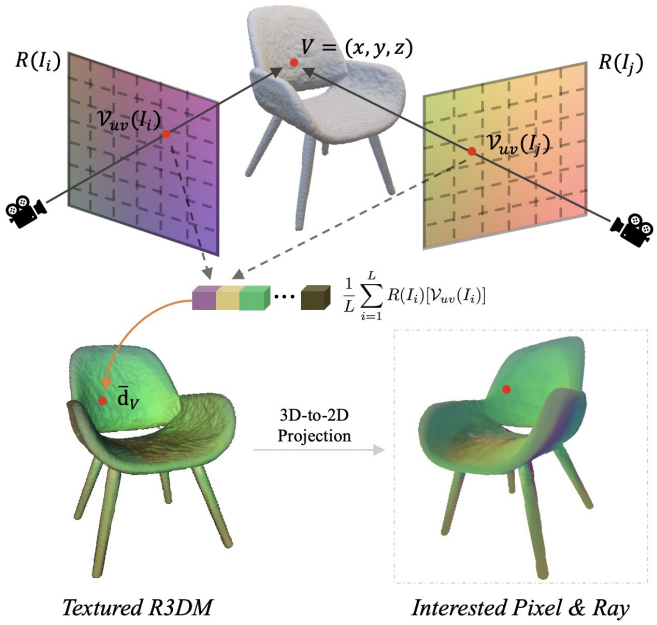

Figure 3. **Ray Atlas.** An illustration of how we capture a ray atlas from the training rays and use it to texture a chair's rough 3D mesh (R3DM). $R(I_i)$ is the ray map of the training image I_i. $\mathcal{V}_{uv}(I_i)$ is the 2D position of image I_i corresponding to vertex V. The global ray direction $\bar{\mathrm{d}}_V$ of vertex V is computed following Eqn. 3.

that also passes through v. Thereby, we can treat the ground truth color intensity $I(\mathrm{r})$ as the pseudo label of $\hat{I}(\mathrm{r}')$.

3.3. Ray Atlas

The vanilla NeRF utilizes "direction embedding" to encode the lighting effects of a scene. We find the scene fitting process makes the trained color prediction MLP rely heavily on the viewing direction. It is not a problem for novel view interpolation. Nevertheless, it might not be good for novel view extrapolation as there are some discrepancies between the training and test ray distributions. A naive idea is to directly remove the direction embedding (denoted as "NeRF w/o dir"). However, we find it often produces images with artifacts such as unexpected ripple and non-smooth colors. That means the rays' viewing directions might also contribute to surface smoothing. We compute a ray atlas and show it can further enhance the rendering quality of extrapolated views while not involving more issues to interpolated views. A ray atlas is like a texture atlas, but instead, it stores a global ray direction for each 3D vertex.

In particular, for each image (*e.g.,* image I), we capture its rays' viewing directions for all spatial locations, resulting in a ray map $R(I)$. We extract a rough 3D mesh (R3DM) from the pre-trained NeRF, and map the ray directions to the 3D vertexes. Taking a vertex $V = (x, y, z)$ as an example, its global ray direction $\bar{\mathrm{d}}_V$ should be expressed as:

$$\begin{aligned} \bar{\mathrm{d}}_V &= \frac{1}{L}\sum_{i=1}^{L} R(I_i)[\mathcal{V}_{uv}(I_i)], \\ \mathcal{V}_{uv}(I_i) &= \frac{1}{z} K \mathcal{T}_{w2c}(I_i) V, \end{aligned} \tag{3}$$

where K is the camera intrinsic parameter, $\mathcal{T}_{w2c}(I_i)$ is the camera to world transformation matrix of image I_i, $\mathcal{V}_{uv}(I_i)$ denotes the projected 2D location in image I_i of vertex V, and L represents the number of training images that contributed to the reconstruction of vertex V. We normalize $\bar{\mathrm{d}}_V$ before storing it. Then, for each pixel with an arbitrary camera pose, we can capture a global ray prior $\bar{\mathrm{d}}$ by projecting the R3DM, which is textured by the ray maps, to 2D. See Figure 3 for an illustration.

When training RapNeRF, we adopt $\bar{\mathrm{d}}$ of the interested pixel $I(\mathrm{r})$ to replace its d in F_c for its color prediction. This alternative mechanism occurs with a probability of 0.5. In our experiments, we sample points along the original ray r, and use $\bar{\mathrm{d}}$ for ray embedding computation. We find sampling points along the direction $\bar{\mathrm{d}}$ would sometimes make the training unstable. In the test phase, the radiance c of a sample x is approximated as:

$$\mathrm{c} = F_c(\bar{\mathrm{d}}, F_\sigma(\mathrm{x})). \tag{4}$$

3.4. Training RapNeRF

Our RapNeRF is trained in two stages. For an object to be reconstructed, we first train a NeRF [22] in N_1 iterations to recovery the geometry. Then, we incorporate the proposed random ray cast (RRC) policy and ray mapping (RA) approach to fine-tune the pre-trained NeRF in another N_2 iterations. For each iteration, RRC and RA occur with the probabilities of 0.7 and 0.5, respectively. We set η in RRC to $30°$. In both stages, we employ an additional opacity constraint [26] to enforce the accumulated opacity (transmittance) along a ray to be 1 if the ray trace through the object regions, and 0 if the ray belongs to the backgrounds. Let $m(\mathrm{r})$ denote the mask label (1 or 0) of a pixel ray. The opacity constraint can be expressed as:

$$\mathcal{L}_o = \sum_{\mathrm{r}} |m(\mathrm{r}) + T_N(\mathrm{r}) - 1|, \tag{5}$$

where $T_N(\mathrm{r})$ can be seen as the ratio of light that can pass through the object. $\mathcal{L}_o$ could reduce some noisy volume densities around the objects' surface regions. It is worth mentioning that other NeRF works [9, 17, 46, 51] also incorporate opacity regularization techniques to remove background voxels for object reconstruction.

4. Experiments

In this section, we conduct experiments to investigate the performance of our RapNeRF for novel view extrapolation. First, we briefly introduce the Synthetic-NeRF*

Method	Synthetic-NeRF* [22]			MobileObject		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF [22]	25.73	0.906	0.090	24.05	0.948	0.089
NeRF w/o dir [22]	26.14	0.918	0.067	26.69	0.951	0.053
NSVF [17]	26.37	0.893	0.088	22.65	0.899	0.125
IDR [49]	20.45	0.909	0.113	23.43	0.947	0.074
IBRNet [42]	23.33	0.870	0.153	18.99	0.870	0.185
PlenOctree [51]	24.19	0.875	0.100	21.76	0.903	0.105
SNeRG† [9]	24.68	0.904	0.074	26.32	0.952	0.058
RapNeRF	**27.63**	**0.929**	**0.046**	**28.90**	**0.963**	**0.045**
RapNeRF†	26.40	0.912	0.069	28.65	0.961	0.047

Table 1. **Benchmark Comparisons.** RapNeRF† remedies the degenerated view-dependent effects of RapNeRF by incorporating RRC and RA into a deferred NeRF variant (SNeRG† [9]). It further shows RRC and RA can be easily integrated into other NeRFs. Our approaches obtain best performance on datasets of both synthetic and real images. IBRNet [42] and IDR [49] also exploit the multi-view consistency property. See Sec. 5 for limitation discussions and Sec. 4.2 for experimental details. Per-object results are reported in the supplementary.

Components			Metrics		
NeRF [22]	RRC	RA	PSNR ↑	SSIM ↑	LPIPS ↓
√			24.05	0.948	0.089
√	√		27.55	0.963	0.045
√		√	25.29	0.954	0.056
√	√	√	**28.90**	**0.963**	**0.045**

Table 2. **Ray Priors.** We study the performance gains ordained by the proposed random ray casting (RRC) and ray atlas (RA) approaches on the MobileObject dataset. When computing $\hat{I}(\mathrm{r})$, RapNeRF adopts the direction embedding of d (if only use RRC) or the direction embedding of $\bar{\mathrm{d}}$ from the ray atlas (if use RA).

[22] and MobileObject datasets towards our studied setting in Sec. 4.1. Then, we make qualitative and quantitative comparisons with recent representative NeRF variants in Sec. 4.2. Finally, we perform various ablation studies based on MobileObject to discuss our method in Sec. 4.3. We refer to the supplemental materials for more experimental results.

4.1. Datasets

Synthetic-NeRF*. The original Synthetic-NeRF [22] dataset contains eight objects, where each object is rendered in a resolution of 800×800, with 100 views for training and 200 for testing. The viewpoints are sampled on the upper hemisphere or a full sphere. The ground-truth camera poses and object masks are provided. In this paper, we simply sort the cameras' locations along the z axis in ascending order, and choose the first 100 images for training, and the remained 200 images for testing.

MobileObject. We capture eight object-centric videos using a mobile device, where the viewpoints are on the upper space. To allow better SFM, we put the target objects into complexity scenes that are with rich textures before recording the videos. The image size is 960×540 or 540×960. For each object, we uniformly sample about 200∼300 images from the video sequence and compute the camera poses using COLMAP [35]. The blur images are pre-removed. Then, we compute the average z value of the cameras' positions, and choose its neighboured 100 images for training based on the z value distance. To construct the test set, we utilize the following metric to measure the distance between a camera pose $y \in \mathcal{SO}(3)$ and the training poses $\mathcal{X}$:

$$\mathcal{D}_y = \min_{x\in\mathcal{SO}(3)}\{\|\log(x) - \log(y)\| \mid x \in \mathcal{X}\}, \qquad (6)$$

where $\mathcal{SO}(3)$ is the 3D rotation group. A large $\mathcal{D}_y$ represents a significant viewpoint discrepancy. We compute $\mathcal{D}_y$ for each remained image, and choose the ones with larger distance for testing. We select up to 60∼100 test images for each object according to its video length. Other images have been filtered out.

4.2. Benchmark Comparisons

We make comparisons with NeRF [22] and its recent represent variants, including NSVF [17], IBRNet [42], PlenOctree [51], and SNeRG [9]. We perform per-object fine-tuning for IBRNet using their released model pre-trained on a large database. PlenOctree here means its NeRF-SH version. We slightly reformulate the deferred NeRF architecture in SNeRG by predicting a specular color for each sampled point along a ray from f and d. We have not used its sparse radiance grid data structure as it imposes a quality loss of about 2dB. We also examine IDR [49], an impressive 3D reconstruction work that exploits the multi-view consistency property leveraging differentiable rendering. We

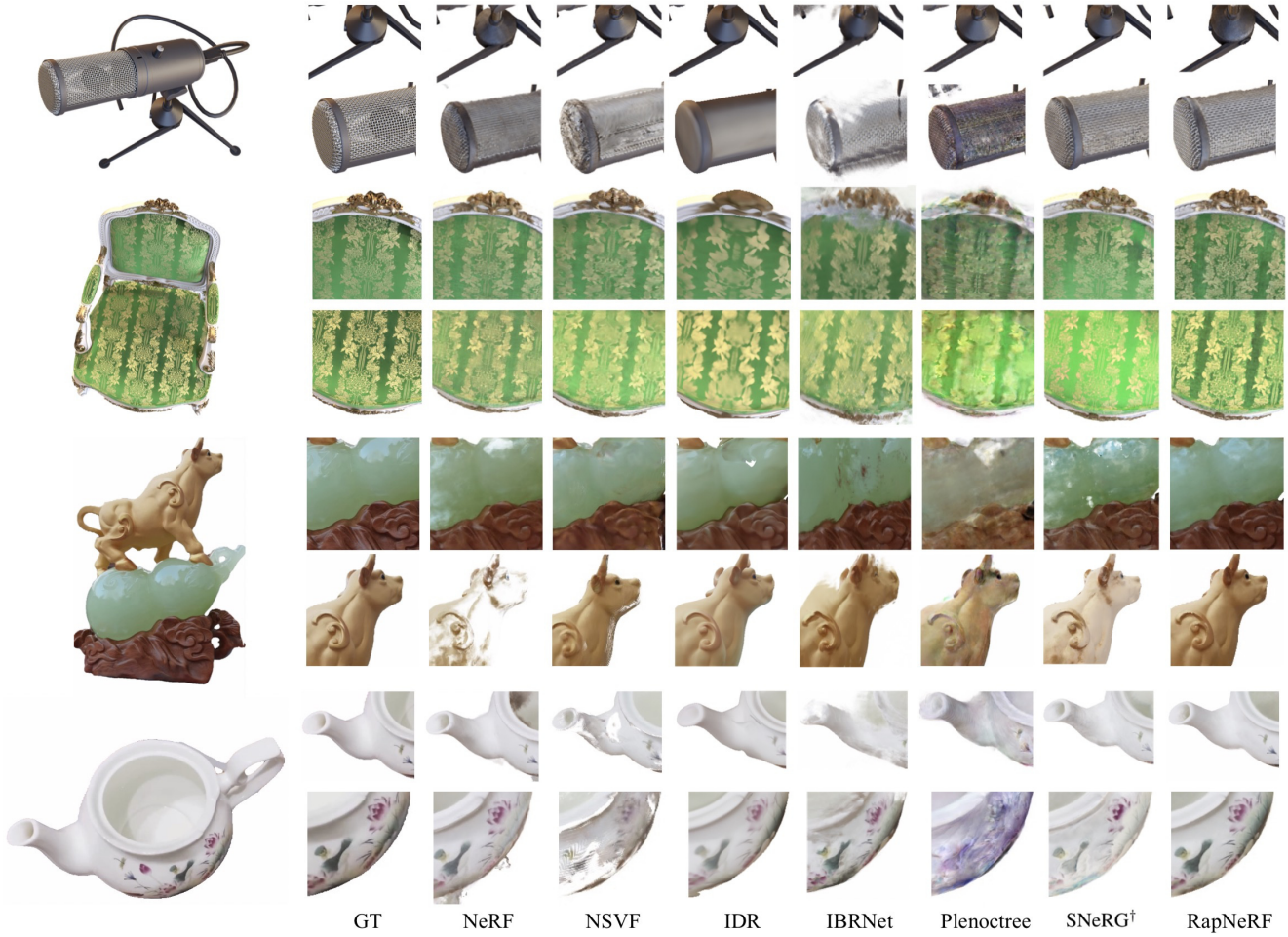

Figure 4. **Qualitative Comparisons.** We compare RapNeRF with several recent impressive methods on Synthetic-NeRF* (Top) and MobileObject (Bottom). RapNeRF better recover the fine details of these objects. The images synthesized by other methods often contains artifacts. We refer to the supplemental materials for more qualitative comparisons. Zoom in for a better view.

optimize NeRF in $N_1 + N_2$ iterations for a fair comparison, since RapNeRF contains a pre-training stage. For other methods, we train them longer to ensure their models are converged.

We use PSNR, SSIM [43], and LPIPS [54] to measure the rendering quality. As reported in Table 1, RapNeRF obtains the best performance on all metrics for novel view extrapolation. It is interesting to see the vanilla NeRF achieves higher PSNR than NSVF on the real MobielObject dataset and vice verve on Synthetic-NeRF*. As discussed in the NSVF paper [17], it shows fewer tolerances to camera pose errors than NeRF. IBRNet's performance drops much on real scenes (18.99 *vs*. 23.33 for synthetic scenes). The main reason is that the pose distance between training and test views of MobileObject's scenes is large. IBRNet and a similar work MVSNeRF [4] can only well reconstruct the scene content that has been seen by the reference views (a small scene frustum). IDR here yields a mean PSNR of 23.43 on real scenes. We notice that it performs similarly on the DTU dataset [12] (23.20 as reported in its paper). That means, though IDR is good at recovering surfaces, but is not as great as NeRF for photo-realistic view synthesis. We also show some qualitative results in Figure 4. Other methods often produce renderings that contain artifacts and distortions, while RapNeRF can generate images with great visual quality.

4.3. Ablation Studies

Ray Priors. We study the effectiveness of the core components in RapNeRF, *i.e.*, random ray casting (RRC) and ray atlas (RA), for novel view extrapolation. As depicted in Table 2, while RA yields a PSNR improvement of 1.24 over the NeRF baseline, RRC outperforms NeRF (24.05) by a remarkable margin (+3.5 on PSNR). Moreover, the complete model obtains a further performance gain and produces a significant PSNR of 28.90. Some qualitative

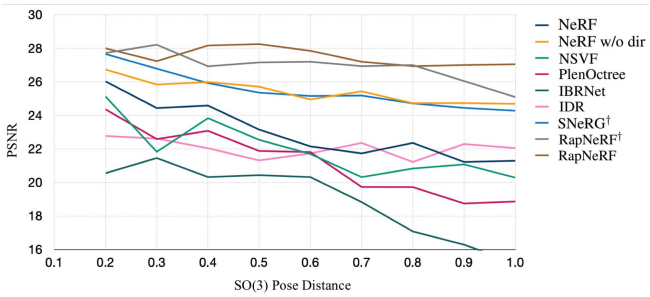

Figure 5. **PSNR vs. SO(3) Pose Distance.** We draw the curves by computing the PSNR score and SO(3) pose distance (D_y) for each test image in the whole MobileObject dataset. RapNeRF yields consistently superior PSNR while D_y becomes larger.

studies are presented in Figure 6. We find, in comparisons with RA, RRC bakes more lighting effects, but it would generate smoother renderings under extrapolated viewpoints. RA would produce artifacts such as slight bump surfaces and white noises. RRC can remedy issues caused by RA.

Pose Distance. We study the robustness of our method to pose distance. We first present the "PSNR vs. SO(3) pose distance" curves obtained by different SOTA approaches in Figure 5. Here, we draw the curves by computing the PSNR score and SO(3) pose distance (D_y) for each test image in the whole MobileObject dataset. We also make a comparison with NeRF using different test splits in Table 5. Specifically, for each object in MobileObject, we combine its test and discarded images (labeled in "Blue" and "Gray" in Figure 1). Then we calculate the distance between these images and the training set based on Eqn. 6 and resplit them into three disjoint sets. From Figure 5 and Table 5, the performance of the compared approaches decreases drastically as D_y becoming larger. It is worth mentioning that, for the "Close" setting, there are still some shifts between test viewpoints and training viewpoints (like the "Gray" and "Red" examples). Thereby, it is not surprising that NeRF can only reach 26.23 on PSNR. Luckily, RapNeRF yields consistently promising PSNR even for the "Far" setting. RapNeRF could generate a visually appealing rendering in these challenging cases, but other methods produce severe artifacts.

η in RRC. Our random ray casting (RRC) policy allows pseudo-label assigning for randomly generated virtual rays during the training process. We introduce azimuth randomness $\Delta\varphi$ and elevation randomness $\Delta\theta$ to the ray vectors in spherical space. Here, $\Delta\varphi$ and $\Delta\theta$ are uniformly sampled from a pre-defined interval $[-\eta, \eta]$. We show quantitative results on *Skull* with different elevation threshold η for $\Delta\theta$ in Table 3. The selection $\eta = 30°$ achieves the best performance. The image quality shows a slight decrease when η is greater than $30°$. A possible reason is that RRC does not consider the self-occlusions of objects.

η	10°	20°	30°	40°	50°	60°
PSNR ↑	26.89	27.32	27.86	26.72	27.38	27.29
SSIM ↑	0.963	0.965	0.967	0.964	0.966	0.965
LPIPS↓	0.036	0.032	0.029	0.030	0.029	0.029

Table 3. Quantitative results on *Skull* with threshold η in RRC.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF-TF [22]	31.01	0.947	0.081
NeRF-TF (w/o dir)	27.66	0.925	0.117
JaxNeRF [5]	31.65	0.952	0.051
RapNeRF-PL	30.08	0.949	0.067
SNeRG-Jax [9]	30.47	0.951	0.049
RapNeRF†-PL	31.29	0.951	0.055

Table 4. **Novel View Interpolation.** JaxNeRF is the Jax NeRF implementation. NeRF-TF is the official tensorflow NeRF implementation. RapNeRF is implemented via Pytorch-Lightning.

5. Limitation

As our key insight is the multi-view consistency property, RapNeRF would sacrifice view-dependent effects for novel view interpolation to secure much better performance for novel view extrapolation. We argue it is acceptable for applications such as 6-DOF immersive viewing and next-generation AR/VR systems. Specifically, when showcasing an object, a person prefers not to capture it in very high-frequency lighting conditions. Furthermore, studies like relighting might take some inspiration from RapNeRF since they need to decompose view-dependent effects from objects' base colors.

For pure view synthesis, we provide a remedy to the slightly degenerated view-dependent effects by exploiting the deferred NeRF architecture in SNeRG [9]. In particular, after the NeRF pre-training stage, we add a tiny MLP that maps from a 3D point's geometry feature f (as explained in Sec. 3.1) and its corresponding direction embedding to a specular color. We remove the direction embedding in RRC and force RapNeRF only to learn the diffuse color. The color (or radiance) for a 3D point is now computed by the addition of its diffuse color and specular color. We denote this variant as RapNeRF†. As reported in Table 1, RapNeRF† imposes a quality loss of about 1dB in comparison with RapNeRF, but still outperforms other methods by a large margin on MobileObject (real scenes). Other re-

Figure 6. We visualize the impacts of the proposed random ray casting (RRC) policy and ray atlas (RA). We can find that (1) fine-tuning NeRF with either RRC or RA could well remedy the artifact issue; and (2) RA and RRC are compatible with each other.

Method	Close			Middle			Far		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF [22]	26.23	0.960	0.075	23.42	0.944	0.095	22.51	0.939	0.099
NeRF w/o dir [17]	27.46	0.958	0.047	26.91	0.955	0.050	25.71	0.938	0.603
RapNeRF†	30.24	0.968	0.041	28.31	0.958	0.049	27.42	0.955	0.052
RapNeRF	**30.88**	**0.969**	**0.037**	**29.06**	**0.963**	**0.044**	**28.74**	**0.963**	**0.045**

Table 5. **Viewpoints Distance.** We further split the test and discarded images (labeled in "Blue" and "Gray") of each object into three disjoint subsets based on Eqn. 6. The performance of NeRF [22] decreases drastically when the viewpoint distance between training and test images becomes larger. RapNeRF yields consistently promising scores for each setting.

sults are presented in Table 5 and Figure 5. Furthermore, RapNeRF† can be seen as training a deferred NeRF variant [9] with RRC and RA. It further shows RRC and RA can be flexibly integrated into other NeRFs to improve their novel view exploration ability.

In Table 4, we evaluate RapNeRF and RapNeRF† on the standard NeRF blender dataset. RapNeRF imposes a quality loss of 1.6dB compared to JaxNeRF. RapNeRF may fake view-dependent effects by hiding some reflected content inside the objects' surface as analyzed in [9]. Moreover, RapNeRF† does not degrade the performance of SNeRG for novel view interpolation.

6. Conclusion

In this paper, we study *Neural Radiance Fields* (NeRF) for novel view extrapolation where the test viewpoints are significantly different from the training viewpoints. We find that NeRF [22] often produces low-quality renderings with many artifacts under extrapolated viewpoints, even the training images can well describe the scenes. We take inspiration from the insight that the inherent appearances of a 3D surface's arbitrary visible projections should be consistent, and proposes RapNeRF as a solution. It is empowered by random ray casting (RRC) and ray atlas (RA). The former allows pseudo supervision for unseen views in an online manner, and the latter prudently rethinks the tradeoff between strong view-dependent effects and multi-view consistent renderings. We reconstruct Synthetic-NeRF [22] for our studied setting and build a MobileObject dataset that contains eight objects with real images. The comparisons with NeRF and its recent variants demonstrate the superiority of RapNeRF for novel view extrapolation. In the future, we would like to study neural radiance fields reconstruction from sparse multi-view images.

Acknowledgement

This work was partially supported by Alibaba Group through Alibaba Innovative Research Program. This work is done when Yuanqing Zhang is an intern at Alibaba Group.

References

[1] 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. *arXiv preprint arXiv:2103.13415*, 2021. 2

[2] Sai Bi, Zexiang Xu, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Deep reflectance volumes: Relightable reconstructions from multi-view photometric images. *arXiv preprint arXiv:2007.09892*, 2020. 2

[3] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. Nerd: Neural reflectance decomposition from image collections. *arXiv preprint arXiv:2012.03918*, 2020. 2

[4] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. *arXiv preprint arXiv:2103.15595*, 2021. 1, 6

[5] Boyang Deng, Jonathan T. Barron, and Pratul P. Srinivasan. JaxNeRF: an efficient JAX implementation of NeRF, 2020. 7

[6] Emilien Dupont, Miguel Bautista Martin, Alex Colburn, Aditya Sankar, Josh Susskind, and Qi Shan. Equivariant neural rendering. In *International Conference on Machine Learning*, pages 2761–2770. PMLR, 2020. 2

[7] Guy Gafni, Justus Thies, Michael Zollhöfer, and Matthias Nießner. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. *arXiv preprint arXiv:2012.03065*, 2020. 2

[8] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. Fastnerf: High-fidelity neural rendering at 200fps. *arXiv preprint arXiv:2103.10380*, 2021. 2

[9] Peter Hedman, Pratul P Srinivasan, Ben Mildenhall, Jonathan T Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis. 2021. 1, 2, 4, 5, 7, 8

[10] Jingwei Huang, Justus Thies, Angela Dai, Abhijit Kundu, Chiyu Jiang, Leonidas J Guibas, Matthias Nießner, Thomas Funkhouser, et al. Adversarial texture optimization from rgb-d scans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1559–1568, 2020. 1, 3

[11] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. *arXiv preprint arXiv:2104.00677*, 2021. 1, 2

[12] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 406–413, 2014. 6

[13] Adam R Kosiorek, Heiko Strathmann, Daniel Zoran, Pol Moreno, Rosalia Schneider, Soňa Mokrá, and Danilo J Rezende. Nerf-vae: A geometry aware 3d scene generative model. *arXiv preprint arXiv:2104.00587*, 2021. 2

[14] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. *arXiv preprint arXiv:2104.06405*, 2021. 2

[15] Chen-Hsuan Lin, Oliver Wang, Bryan C Russell, Eli Shechtman, Vladimir G Kim, Matthew Fisher, and Simon Lucey. Photometric mesh optimization for video-aligned 3d object reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 969–978, 2019. 1, 3

[16] David B Lindell, Julien NP Martel, and Gordon Wetzstein. Autoint: Automatic integration for fast neural volume rendering. *arXiv preprint arXiv:2012.01714*, 2020. 2

[17] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *NeurIPS*, 2020. 1, 2, 4, 5, 6, 8

[18] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 38(4):65:1–65:14, July 2019. 1

[19] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhoefer, Yaser Sheikh, and Jason Saragih. Mixture of volumetric primitives for efficient neural rendering. *arXiv preprint arXiv:2103.01954*, 2021. 2

[20] Quan Meng, Anpei Chen, Haimin Luo, Minye Wu, Hao Su, Lan Xu, Xuming He, and Jingyi Yu. Gnerf: Gan-based neural radiance field without posed camera. *arXiv preprint arXiv:2103.15606*, 2021. 2

[21] Moustafa Meshry, Dan B Goldman, Sameh Khamis, Hugues Hoppe, Rohit Pandey, Noah Snavely, and Ricardo Martin-Brualla. Neural rerendering in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6878–6887, 2019. 1

[22] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2, 4, 5, 7, 8

[23] Thomas Neff, Pascal Stadlbauer, Mathias Parger, Andreas Kurz, Chakravarty R Alla Chaitanya, Anton Kaplanyan, and Markus Steinberger. Donerf: Towards real-time rendering of neural radiance fields using depth oracle networks. *arXiv preprint arXiv:2103.03231*, 2021. 2

[24] Michael Niemeyer, Jonathan T Barron, Ben Mildenhall, Mehdi SM Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. *arXiv preprint arXiv:2112.00724*, 2021. 2

[25] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. *arXiv preprint arXiv:2011.12100*, 2020. 2

[26] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3504–3515, 2020. 4

[27] Atsuhiro Noguchi, Xiao Sun, Stephen Lin, and Tatsuya Harada. Neural articulated radiance field. *arXiv preprint arXiv:2104.03110*, 2021. 2

[28] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan Goldman, Steven Seitz, and Ricardo Martin-Brualla. Deformable neural radiance fields. *arXiv preprint arXiv:2011.12948*, 2020. 2

[29] Despoina Paschalidou, Osman Ulusoy, Carolin Schmitt, Luc Van Gool, and Andreas Geiger. Raynet: Learning volumetric 3d reconstruction with ray potentials. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3897–3906, 2018. 2

[30] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. *arXiv preprint arXiv:2011.13961*, 2020. 2

[31] Amit Raj, Michael Zollhoefer, Tomas Simon, Jason Saragih, Shunsuke Saito, James Hays, and Stephen Lombardi. Pva: Pixel-aligned volumetric avatars. *arXiv preprint arXiv:2101.02697*, 2021. 2

[32] Daniel Rebain, Wei Jiang, Soroosh Yazdani, Ke Li, Kwang Moo Yi, and Andrea Tagliasacchi. Derf: Decomposed radiance fields. *arXiv preprint arXiv:2011.12490*, 2020. 2

[33] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. *arXiv preprint arXiv:2103.13744*, 2021. 1, 2

[34] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *European Conference on Computer Vision*, pages 623–640. Springer, 2020. 2

[35] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 5

[36] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33, 2020. 2

[37] 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. *arXiv preprint arXiv:2012.03927*, 2020. 2

[38] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, et al. State of the art on neural rendering. In *Computer Graphics Forum*, volume 39, pages 701–727. Wiley Online Library, 2020. 1, 2

[39] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *ACM Trans. Graph.*, 38(4), July 2019. 1

[40] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a deforming scene from monocular video. *https://arxiv.org/abs/2012.12247*, 2020. 2

[41] Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Multi-view consistency as supervisory signal for learning shape and pose prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2897–2905, 2018. 1, 3

[42] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. *arXiv preprint arXiv:2102.13090*, 2021. 2, 5

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

[44] 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. 2

[45] Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. *arXiv preprint arXiv:2103.05606*, 2021. 2

[46] Fanbo Xiang, Zexiang Xu, Miloš Hašan, Yannick Hold-Geoffroy, Kalyan Sunkavalli, and Hao Su. Neutex: Neural texture mapping for volumetric neural rendering. *arXiv preprint arXiv:2103.00762*, 2021. 2, 4

[47] Zexiang Xu, Kalyan Sunkavalli, Sunil Hadap, and Ravi Ramamoorthi. Deep image-based relighting from optimal sparse samples. *ACM Transactions on Graphics (ToG)*, 37(4):1–13, 2018. 1

[48] Guandao Yang, Yin Cui, Serge Belongie, and Bharath Hariharan. Learning single-view 3d reconstruction with limited pose supervision. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 86–101, 2018. 3

[49] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems*, 33, 2020. 5

[50] Lin Yen-Chen, Pete Florence, Jonathan T Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. inerf: Inverting neural radiance fields for pose estimation. *arXiv preprint arXiv:2012.05877*, 2020. 2

[51] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. *arXiv preprint arXiv:2103.14024*, 2021. 1, 2, 4, 5

[52] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021. 1, 2

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

[54] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 6

[55] Rui Zhu, Hamed Kiani Galoogahi, Chaoyang Wang, and Simon Lucey. Rethinking reprojection: Closing the loop for

pose-aware shape reconstruction from a single image. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 57–65, 2017. 1, 3