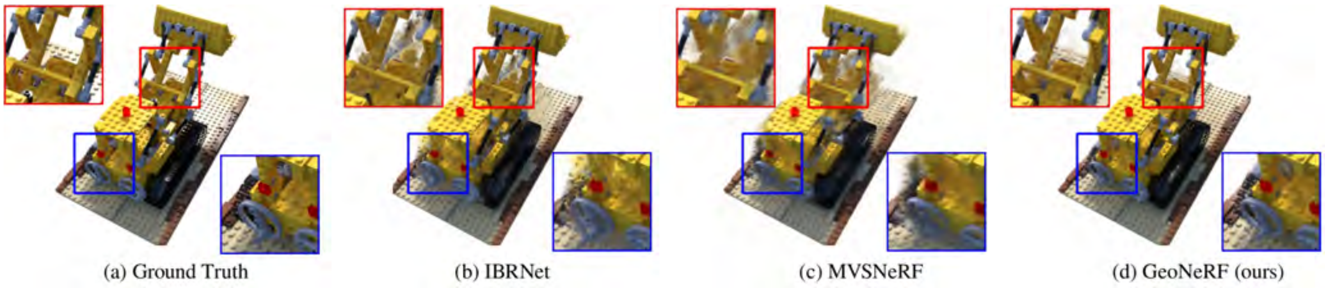
arXiv:2111.13539v1 [cs.CV] 26 Nov 2021

GeoNeRF: Generalizing NeRF with Geometry Priors

Mohammad Mahdi Johari
Idiap Reserach Institute, EPFL
mohammad.johari@idiap.ch

Yann Lepoittevin
ams OSRAM
yann.lepoittevin@ams.com

François Fleuret
University of Geneva, EPFL
francois.fleuret@unige.ch



(a) Ground Truth (b) IBRNet (c) MVSNeRF (d) GeoNeRF (ours)

Figure 1. Our generalizable GeoNeRF model infers complex geometries of objects in a novel scene without per-scene optimization and synthesizes novel images of higher quality than the existing works: IBRNet [53] and MVSNeRF [6].

Abstract

We present GeoNeRF, a generalizable photorealistic novel view synthesis method based on neural radiance fields. Our approach consists of two main stages: a geometry reasoner and a renderer. To render a novel view, the geometry reasoner first constructs cascaded cost volumes for each nearby source view. Then, using a Transformer-based attention mechanism and the cascaded cost volumes, the renderer infers geometry and appearance, and renders detailed images via classical volume rendering techniques. This architecture, in particular, allows sophisticated occlusion reasoning, gathering information from consistent source views. Moreover, our method can easily be fine-tuned on a single scene, and renders competitive results with per-scene optimized neural rendering methods with a fraction of computational cost. Experiments show that GeoNeRF outperforms state-of-the-art generalizable neural rendering models on various synthetic and real datasets. Lastly, with a slight modification to the geometry reasoner, we also propose an alternative model that adapts to RGBD images. This model directly exploits the depth information often available thanks to depth sensors. The implementation code will be publicly available [1].

1. Introduction

Novel view synthesis is a long-standing task in computer vision and computer graphics. Neural Radiance Fields (NeRF) [36] made a significant impact on this research area by implicitly representing the 3D structure of the scene and rendering high-quality novel images. Our work addresses the main drawback of NeRF, which is the requirement to train from scratch for every scene separately. The per-scene optimization of NeRF is lengthy and requires densely captured images from each scene.

Approaches like pixelNeRF [59], GRF [50], MINE [28], SRF [7], IBRNet [53], MVSNeRF [6], and recently introduced NeRFormer [42] address this issue and generalize NeRF rendering technique to unseen scenes. The common motivation behind such methods is to condition the NeRF renderer with features extracted from source images from a set of nearby views. Despite the generalizability of these models to new scenes, their understanding of the scene geometry and occlusions is limited, resulting in undesired artifacts in the rendered outputs. MVSNeRF [6] constructs a low-resolution 3D cost volume inspired by MVSNet [56], which is widely used in the Multi-View Stereo research, to condition and generalize the NeRF renderer. However, it has difficulty rendering detailed images and does not deal with occlusions in a scene. In this work, we take MVSNeRF as a baseline and propose the following improvements.

- We introduce a geometry reasoner in the form of cascaded cost volumes (Section 3.1) and train it in a semi-supervised fashion (Section 3.4) to obtain fine and high-resolution priors for conditioning the renderer.

- We combine an attention-based model which deals with information coming from different source views

[1] https://www.idiap.ch/paper/geonerf

at any point in space, by essence permutation invariant, with an auto-encoder network which aggregates information along a ray, leveraging its strong Euclidean and ordering structure (Section 3.3).

- Thanks to the symmetry and generalizability of our geometry reasoner and renderer, we detect and exclude occluded views for each point in space and use the remaining views for processing that point (Section 3.3).

In addition, with a slight modification to the architecture, we propose an alternate model that takes RGBD images (RGB+Depth) as input and exploits the depth information to improve its perception of the geometry (Section 3.5).

Concurrent to our work, the followings also introduce a generalizable NeRF: RGBD-Net [37] builds a cost volume for the target view instead of source views, NeuralMVS [44] proposes a coarse to fine approach to increase speed, and NeuRay [31] proposes a method to deal with occlusions.

2. Related Work

Multi-View Stereo. The purpose of Multi-View Stereo (MVS) is to estimate the dense representation of a scene given multiple overlapping images. This field has been extensively studied: first, with now called traditional methods [8, 17, 26, 46] and more recently with methods relying on deep learning such as MVSNet [56], which outperformed the traditional ones. MVSNet estimates the depth from multiple views by extracting features from all images, aggregating them into a variance-based cost volume after warping each view onto the reference one, and finally, post-processing the cost volumes with a 3D-CNN. The memory needed to post-process the cost volume being the main bottleneck of [56], R-MVSNet [57] proposed regularizing the cost volume along the depth direction with gated recurrent units while slightly sacrificing accuracy. To further reduce the memory impact, [18] proposed CasMVSNet, where the cost volume is built at gradually finer scales, with the depth output computed in a coarse to fine manner without any compromise on the accuracy. Replacing the variance-based metric with group-wise correlation similarity is another approach to further decrease the memory usage of MVS networks [55]. We found MVS architectures suitable for inferring the geometry and occlusions in a scene and conditioning a novel image renderer.

Novel View Synthesis. Early work on synthesizing novel views from a set of reference images was done by blending reference pixels according to specific weights [10, 27]. The weights were computed according to ray-space proximity [27] or approximated geometry [4, 10]. To improve the computed geometry, some used the optical flow [5, 14] or soft blending [41]. Others synthesized a radiance field directly on a mesh [9, 20] or on a point cloud [1, 34]. An advantage of these methods is that they can synthesize new views with a small number of references, but their performance is limited by the quality of 3D reconstruction [21, 45], and problems often arise in low-textured or reflective regions where stereo reconstruction tends to fail. Leveraging CNNs to predict volumetric representations stored in voxel grids [19, 23, 41] or Multi-Plane Images [15, 16, 49, 61] produces photo-realistic renderings. Those methods rely on discrete volumetric representations of the scenes limiting their outputs' resolution. They also need to be trained on large datasets to store large numbers of samples resulting in extensive memory overhead.

Neural Scene Representations. Recently, using neural networks to represent the geometry and appearance of scenes has allowed querying color and opacity in continuous space and viewing directions. NeRF [36] achieves impressive results for novel view synthesis by optimizing a 5D neural radiance field for a scene. Building upon NeRF many improvements were made [3, 12, 29, 33, 39, 40, 43, 47, 48], but the network needs to be optimized for hours or days for each new scene. Later works, such as GRF [50], pixelNeRF [59] and MINE [28], try to synthesize novel views with very sparse inputs, but their generalization ability to challenging scenes with complex specularities is highly restricted. MVSNeRF [6] proposes to use a low-resolution plane-swept cost volume to generalize rendering to new scenes with as few as three images without retraining. Once the cost volume is computed, MVSNeRF uses a 3D-CNN to aggregate image features. This 3D-CNN resembles the generic view interpolation function presented in IBRNet [53] that allows rendering novel views on unseen scenes with few images. Inspired by MVSNeRF, our work first constructs cascaded cost volumes per source view and then aggregates the cost volumes of the views in an attention-based approach. The former allows capturing high-resolution details, and the latter addresses occlusions.

3. Method

We use volume rendering techniques to synthesize novel views given a set of input source views. Our proposed architecture is presented in Figure 2, and the following sections provide details of our method.

3.1. Geometry Reasoner

Given a set of V nearby views $\{I_v\}_{v=1}^{V}$ with size $H \times W$, our geometry reasoner constructs cascaded cost volumes for each input view individually, following the same approach in CasMVSNet [18]. First, each image goes through a Feature Pyramid Network (FPN) [30] to generate semantic 2D

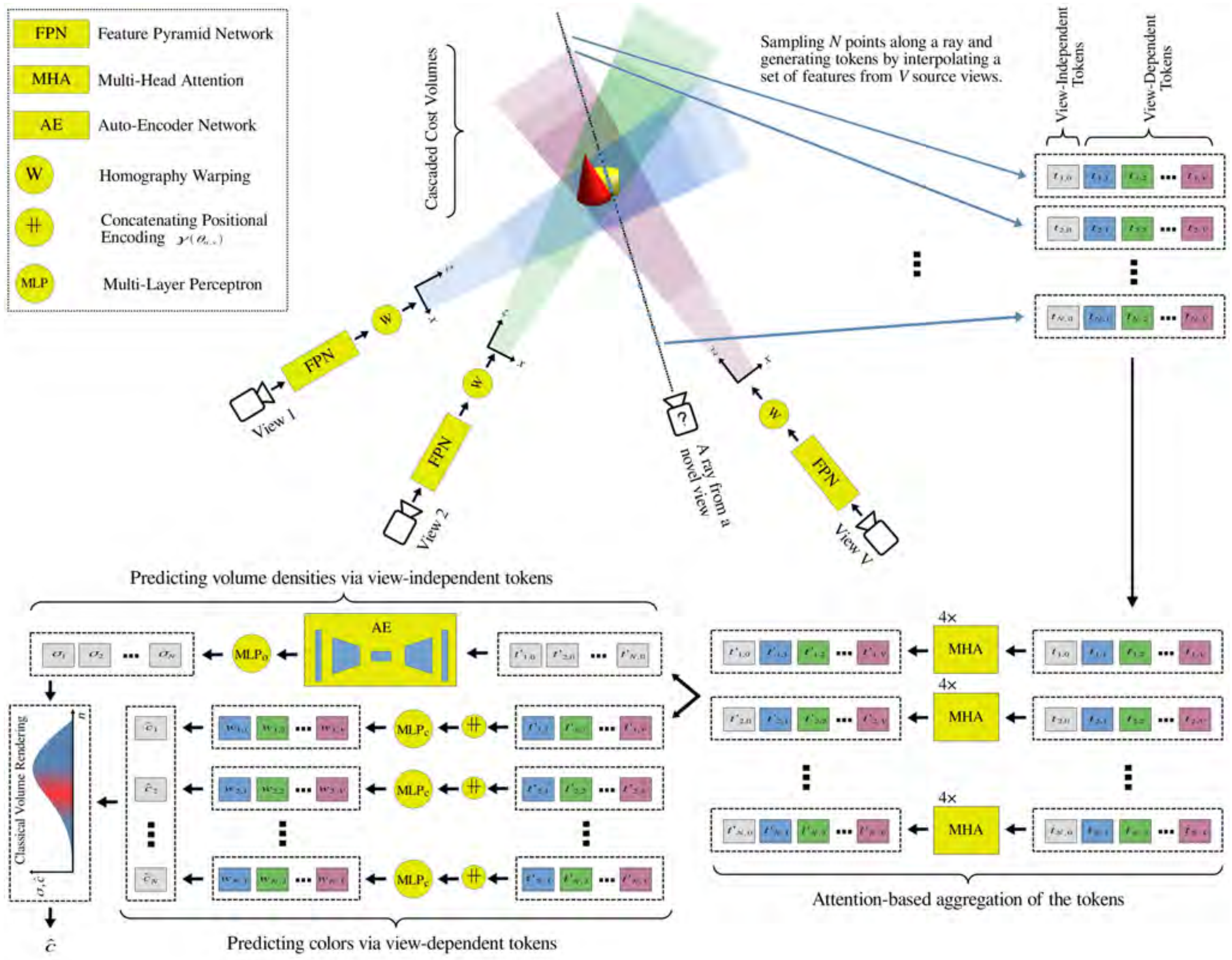

FPN Feature Pyramid Network
MHA Multi-Head Attention
AE Auto-Encoder Network
W Homography Warping
Concatenating Positional Encoding
MLP Multi-Layer Perceptron
Cascaded Cost Volumes
Sampling N points along a ray and generating tokens by interpolating a set of features from V source views.
View-Independent Tokens
View-Dependent Tokens
View 1
View 2
View V
A ray from a novel view
FPN
W
4×
MHA
AE
MLP$_\sigma$
MLP$_c$
Predicting volume densities via view-independent tokens
Attention-based aggregation of the tokens
Predicting colors via view-dependent tokens
Classical Volume Rendering
$\hat{c}$

Figure 2. The overview of GeoNeRF. 2D feature pyramids are first generated via Feature Pyramid Network (FPN) [30] for each source view v. We then construct cascaded cost volumes at three levels for each view by homography warping of its nearby views (see Section 3.1). Guided by the distribution of the cascaded cost volumes in the 3D space, $N = N_c + N_f$ points $\{x_n\}_{n=1}^{N}$ are sampled along a ray for a novel pose (see Section 3.2). By interpolating both 2D and 3D features ($\boldsymbol{f}_{\boldsymbol{n},\boldsymbol{v}}^{(0)}$, $\{\boldsymbol{\Phi}_{\boldsymbol{n},\boldsymbol{v}}^{(l)}\}_{l=0}^{2}$) from FPN and cascaded cost volumes for each sample point x_n, one view independent token $\boldsymbol{t}_{\boldsymbol{n},0}$ and V view-dependent tokens $\{\boldsymbol{t}_{\boldsymbol{n},\boldsymbol{v}}\}_{v=1}^{V}$ are generated. These $V+1$ tokens go through four stacked Multi-Head Attention (MHA) layers and yield more refined tokens $\{\boldsymbol{t}'_{\boldsymbol{n},\boldsymbol{v}}\}_{v=0}^{V}$. The MHA layers are shared among all sample points on a ray. Thereafter, the view-independent tokens $\{\boldsymbol{t}'_{\boldsymbol{n},\boldsymbol{0}}\}_{n=1}^{N}$ are regularized and aggregated along the ray samples through the AE network, and volume densities $\{\boldsymbol{\sigma}_{\boldsymbol{n}}\}_{n=1}^{N}$ of the sampled points are estimated. Other tokens $\{\boldsymbol{t}'_{\boldsymbol{n},\boldsymbol{v}}\}_{v=1}^{V}$, supplemented with the positional encodings $\{\gamma(\theta_{n,v})\}_{v=1}^{V}$, predict the color weights $\{\boldsymbol{w}_{\boldsymbol{n},\boldsymbol{v}}\}_{v=1}^{V}$ with respect to source views, and the color $\hat{\boldsymbol{c}}_{\boldsymbol{n}}$ of each point is estimated in a weighted sum fashion (see Section 3.3). Finally, the color of the ray $\hat{\boldsymbol{c}}$ is rendered using classical volume rendering.

features at three different scale levels.

$$\boldsymbol{f}_{\boldsymbol{v}}^{(\boldsymbol{l})} = \text{FPN}\left(I_v\right) \in \mathbb{R}^{\frac{H}{2^l} \times \frac{W}{2^l} \times 2^l C} \quad \forall l \in \{0, 1, 2\} \tag{1}$$

where FPN is the Feature Pyramid Network, C is the channel dimension at level 0, and l indicates the scale level. Once 2D features are generated, we follow the same approach in CasMVSNet to construct plane sweeps and cascaded cost volumes at three levels via differentiable homography warping. CasMVSNet originally estimates depth maps $\hat{\boldsymbol{D}}^{(\boldsymbol{l})}$ of the input images at three levels. The coarsest level ($l = 2$) consists of $D^{(2)}$ plane sweeps covering the whole depth range in the camera's frustum. Then, subsequent levels narrow the hypothesis range (decrease $D^{(l)}$) but increase the spatial resolution of each voxel by creating $D^{(l)}$ finer plane sweeps on both sides of the estimated depths from the previous level. As a result, the finer the cost volume is, the thinner the depth range it covers. We make two modifications to the CasMVSNet architecture and use it as the geometry reasoner. Firstly, we provide an additional output head to the network to produce multi-level semantic 3D features $\boldsymbol{\Phi}^{(\boldsymbol{l})}$ along with the estimated depth maps $\hat{\boldsymbol{D}}^{(\boldsymbol{l})}$. Secondly, we replace the variance-based metric in CasMVSNet with **group-wise correlation similarity** from [55] to construct light-weight volumes. Group-wise correlation decreases memory usage and inference time.

To be more specific, for each source view I_v, we first form a set of its nearby views Γ_v. Then, by constructing $D^{(l)}$ depth plane sweeps and homography warping techniques, we create multi-level cost volumes $\boldsymbol{P_v^{(l)}}$ from 2D feature pyramids $\boldsymbol{f^{(l)}}$ of images in Γ_v using the group-wise correlation similarity metric from [55]. We finally further process and regularize the cost volumes using 3D hourglass networks $R_{3D}^{(l)}$ and generate depth maps $\boldsymbol{\hat{D}_v^{(l)}} \in \mathbb{R}^{\frac{H}{2^l} \times \frac{W}{2^l} \times 1}$ and 3D feature maps $\boldsymbol{\Phi_v^{(l)}} \in \mathbb{R}^{D^{(l)} \times \frac{H}{2^l} \times \frac{W}{2^l} \times C}$:

$$\boldsymbol{\hat{D}_v^{(l)}}, \boldsymbol{\Phi_v^{(l)}} = R_{3D}^{(l)} \left(\boldsymbol{P_v^{(l)}} \right) \;\; \forall l \in \{0, 1, 2\} \tag{2}$$

3.2. Sampling Points on a Novel Ray

Once the features from the geometry reasoner are generated, we render novel views with the ray casting approach. For each camera ray at a novel camera pose, we first sample N_c points along the ray uniformly to cover the whole depth range. Furthermore, we estimate a stepwise probability density function $p_0(x)$ along the ray representing the probability that a point x is covered by a full-resolution partial cost volume $\boldsymbol{P^{(0)}}$. Voxels inside the thinnest, full-resolution cost volumes $\{\boldsymbol{P_v^{(0)}}\}_{v=1}^V$ contain the most valuable information about the surfaces and geometry. Therefore, we sample N_f more points from $p_0(x)$ distribution. Unlike previous works [2, 36, 42, 53, 59] that require training two networks simultaneously (one coarse and one fine network) and rendering volume densities from the coarse network to resample more points for the fine one, we sample a mixture of $N = N_c + N_f$ valuable points before rendering the ray without any computation overhead or network duplication thanks to the design of our geometry reasoner.

3.3. Renderer

For all sample points $\{x_n\}_{n=1}^N$, we interpolate the full-resolution 2D features $\boldsymbol{f_{n,v}^{(0)}}$ and the three-level 3D features $\{\boldsymbol{\Phi_{n,v}^{(l)}}\}_{l=0}^2$ from all source views. We also define an occlusion mask $M_{n,v}$ for each point x_n with respect to each view v. Formally, if a point x_n stands behind the estimated full-resolution depth map $\boldsymbol{\hat{D}_v^{(0)}}$ (being occluded) or the projection of x_n to the camera plane of view v lies outside of the image plane (being outside of the camera frustum), we set $M_{n,v} = 0$ and discard view v from the rendering process of point x_n. Next, we create a view-independent token $t_{n,0}$ and V view-dependent tokens $\{t_{n,v}\}_{v=1}^V$ by utilizing the interpolated features for each point x_n:

$$\begin{aligned} \boldsymbol{t_{n,v}} &= \text{LT}\left(\left[\boldsymbol{f_{n,v}^{(0)}}; \{\boldsymbol{\Phi_{n,v}^{(l)}}\}_{l=0}^2 \right] \right) \;\; \forall v \in \{1, ..., V\} \\ \boldsymbol{t_{n,0}} &= \text{LT}\left(\left[mean\{\boldsymbol{f_{n,v}^{(0)}}\}_{v=1}^V; var\{\boldsymbol{f_{n,v}^{(0)}}\}_{v=1}^V \right] \right) \end{aligned} \tag{3}$$

where $\text{LT}(\cdot)$ and $[\cdot\;;\;\cdot]$ denote respectively linear transformation and concatenation. $\boldsymbol{t_{n,0}}$ could be considered as a global understanding of the scene at point x_n, while $\boldsymbol{t_{n,v}}$ represents the understanding of the scene from source view v. The global and view-dependent tokens are aggregated through four stacked Multi-Head Attention (MHA) layers, which are introduced in Transformers [13, 51]:

$$\{\boldsymbol{t'_{n,v}}\}_{v=0}^V = {}^{4\times}\text{MHA}\left(\boldsymbol{t_{n,0}}, \{\boldsymbol{t_{n,v}}, M_{n,v}\}_{v=1}^V \right) \tag{4}$$

Our MHA layers also take the occlusion masks $M_{n,v}$ as inputs and force the occluded views' attention scores to zero to prevent them from contributing to the aggregation.

The global view-independent output tokens $\{\boldsymbol{t'_{n,0}}\}_{n=1}^N$ now have access to all necessary data to learn the geometry of the scene and estimate volume densities. We further regularize these tokens through an auto-encoder-style (AE) network in the ray dimension (n). The AE network learns the global geometry along the ray via convolutional layers and predicts more coherent volume densities $\boldsymbol{\sigma_n}$:

$$\{\boldsymbol{\sigma_n}\}_{n=1}^N = \text{MLP}_\sigma \left(\text{AE}\left(\{\boldsymbol{t'_{n,0}}\}_{n=1}^N \right) \right) \tag{5}$$

where MLP_σ is a simple two-layer perceptron. We argue that convolutionally processing the tokens with the AE network along the ray dimension (n) is a proper inductive bias and significantly reduces the computation resources compared to methods like IBRNet [53] and NeRFormer [42], which employ an attention-based architecture because the geometry of a scene is naturally continuous, and accordingly, closer points are more likely related.

View-dependent tokens $\{\boldsymbol{t'_{n,v}}\}_{v=1}^V$, together with two additional inputs, are used for color prediction. We project each point x_n to source views' image planes and interpolate the color samples $c_{n,v}$. We also calculate the angle between the novel camera ray and the line that passes through the camera center of source view v and x_n. This angle $\theta_{n,v}$ represents the similarity between the camera pose of source view v and the novel view. Each point's color is estimated via a weighted sum of the non-occluded views' colors:

$$\begin{aligned} \boldsymbol{w_{n,v}} &= \text{Softmax}\left(\left\{ \text{MLP}_c\left([\boldsymbol{t'_{n,v}}; \gamma(\theta_{n,v})] \right), M_{n,v} \right\}_{v=1}^V \right) \\ \boldsymbol{\hat{c}_n} &= \sum_{v=1}^V \boldsymbol{w_{n,v}} c_{n,v} \;\; \forall n \in \{1, 2, ..., N\} \end{aligned} \tag{6}$$

where $\gamma(\cdot)$ is the sinusoidal positional encoding proposed in [36], and MLP_c is a simple two-layer perceptron. The Softmax function also takes the occlusion masks $M_{n,v}$ as input to exclude occluded views.

Once volume densities and colors are predicted, our model renders, as in NeRF [36], the color of the camera ray at a novel pose using the volume rendering approach:

$$\boldsymbol{\hat{c}} = \sum_{n=1}^N \exp\left(-\sum_{k=1}^{n-1} \boldsymbol{\sigma_k} \right) \left(1 - \exp\left(-\boldsymbol{\sigma_n}\right)\right) \boldsymbol{\hat{c}_n} \tag{7}$$

In addition to the rendered color, our model also outputs the estimated depth $\hat{\boldsymbol{d}}$ for each ray:

$$\hat{\boldsymbol{d}} = \sum_{n=1}^{N} \exp\left(-\sum_{k=1}^{n-1} \boldsymbol{\sigma_k}\right)\left(1 - \exp\left(-\boldsymbol{\sigma_n}\right)\right) z_n \tag{8}$$

where z_n is the depth of point x_n with respect to the novel pose. This auxiliary output is helpful for training and supervising our generalizable model (see Section 3.4).

3.4. Loss Functions

The primary loss function when we train our generalizable model on various scenes and the **only** loss when we fine-tune it on a specific scene is the mean squared error between the rendered colors and ground truth pixel colors:

$$\mathcal{L}_c = \frac{1}{|R|} \sum_{r \in R} \left\| \hat{\boldsymbol{c}}(r) - c_{gt}(r) \right\|^2 \tag{9}$$

where R is the set of rays in each training batch and c_{gt} is the ground truth color.

DS-NeRF [11] shows that depth supervision can help NeRF train faster with fewer input views. Moreover, numerous works [24, 38, 52, 58] show that despite the high-quality color rendering, NeRF has difficulty reconstructing 3D geometry and surface normals. Accordingly, for training samples coming from datasets with ground truth depths, we also output the predicted depth $\hat{\boldsymbol{d}}$ for each ray and supervise it if the ground truth depth of that pixel is available:

$$\mathcal{L}_d = \frac{1}{|R_d|} \sum_{r \in R_d} \left\| \hat{\boldsymbol{d}}(r) - d_{gt}(r) \right\|_{s1} \tag{10}$$

where R_d is the set of rays from samples with ground truth depths and d_{gt} is the pixel ground truth depth and $||\cdot||_{s1}$ is the smooth L_1 loss.

Lastly, we supervise cascaded depth estimation networks in our geometry reasoner. For datasets with ground truth depth, the loss is defined as:

$$\mathcal{L}_D^{(l)} = \frac{2^{-l}}{|V|} \sum_{v=1}^{V} \left\langle \left\| \hat{\boldsymbol{D}}_{\boldsymbol{v}}^{(\boldsymbol{l})} - D_v^{(l)} \right\|_{s1} \right\rangle \tag{11}$$

where $D_v^{(l)}$ is the ground truth depth map of view v resized to scale level l, and $< \cdot >$ denotes averaging over all pixels. For training samples without ground truth depths, we self-supervise the depth maps. We take the rendered ray depths as pseudo-ground truth and warp their corresponding colors and estimated depths from all source views using camera transformation matrices. If the ground truth pixel color of a ray is consistent with the warped color of a source view, and it is located in a textured neighborhood, we allow $\hat{\boldsymbol{d}}$ to supervise the geometry reasoner for that view. Formally:

$$\mathcal{L}_D^{(l)} = \frac{2^{-l}}{|V||R|} \sum_{v=1}^{V} \sum_{r \in R} M_v(r) \left\| \hat{\boldsymbol{D}}_{\boldsymbol{v}}^{(\boldsymbol{l})}(r_v) - \hat{\boldsymbol{d}}(r_v) \right\|_{s1}$$

$$\text{where} \quad r_v = T_{\rightarrow v}\left(r, \hat{\boldsymbol{d}}(r)\right) \tag{12}$$

$$\text{and} \quad M_v(r) = \begin{cases} 1 & \text{if } \; |I_v(r_v) - c_{gt}(r)| < \epsilon_c \\ & \quad \text{and } V_5\left(I_v(r_v)\right) > \epsilon_t \\ 0 & \text{otherwise} \end{cases}$$

Given a ray r at a novel pose with rendered depth $\hat{\boldsymbol{d}}(r)$, $T_{\rightarrow v}\left(r, \hat{\boldsymbol{d}}(r)\right)$ transforms the ray to its correspondent ray from source view v using camera matrices. $\hat{\boldsymbol{d}}(r_v)$ denotes the rendered depth of the correspondent ray with respect to source view v, and $M_v(r)$ validates the texturedness and color consistency. We keep pixels whose variance $V_5(\cdot)$ in their 5×5 pixels neighborhood is higher than ϵ_t, and whose color differs less than ϵ_c from the color of the ray r. The aggregated loss function for our generalizable model is:

$$\mathcal{L} = \mathcal{L}_c + 0.1\mathcal{L}_d + \lambda \sum_{l=0}^{2} \mathcal{L}_D^{(l)} \tag{13}$$

where λ is 1.0 if the supervision is with ground truth depths and is 0.1 if it is with pseudo-ground truth rendered depths. For **fine-tuning** on a single scene, regardless of the availability of depth data, we only use $\mathcal{L}_c$ as the loss function.

3.5. Compatibility with RGBD data

Concerning the ubiquitousness of the embedded depth sensors in devices nowadays, we also propose an RGBD compatible model, GeoNeRF$_{\text{+D}}$, by making a small modification to the geometry reasoner. We assume an **incomplete**, low-resolution, noisy depth map $D_v \in \mathbb{R}^{\frac{H}{4} \times \frac{W}{4} \times 1}$ is available for each source view v. When we construct the coarsest cost volume $\boldsymbol{P}_{\boldsymbol{v}}^{(\boldsymbol{2})}$ with $D^{(2)}$ depth planes, we also construct a binary volume $B_v \in \mathbb{R}^{D^{(2)} \times \frac{H}{4} \times \frac{W}{4} \times 1}$ and concatenate it with $\boldsymbol{P}_{\boldsymbol{v}}^{(\boldsymbol{2})}$ before feeding them to the $R_{3D}^{(2)}$ network:

$$B_v(d, h, w) = \begin{cases} 1 & \text{if } Q\left(D_v\left(h, w\right)\right) \equiv d \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

where $Q(\cdot)$ maps and quantizes real depth values to depth plane indices. B_v plays the role of coarse guidance of the geometry in GeoNeRF$_{\text{+D}}$. As a result of this design, the model is robust to quality and sparsity of the depth inputs.

4. Experiments

Training datasets. We train our model on the real DTU dataset [22] and real forward-facing datasets from LLFF [35] and IBRNet [53]. We exclude views with

Method	Settings	DTU MVS[§] [22]			Realistic Synthetic [36]			Real Forward Facing [35]		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
pixelNeRF [59]	No per-scene Optimization	19.31	0.789	0.382	7.39	0.658	0.411	11.24	0.486	0.671
IBRNet [53]		26.04	0.917	0.190	25.49	0.916	0.100	25.13	0.817	0.205
MVSNeRF [6]		26.63	0.931	0.168	23.62	0.897	0.176	21.93	0.795	0.252
GeoNeRF		**31.34**	**0.959**	**0.060**	**28.33**	**0.938**	**0.087**	**25.44**	**0.839**	**0.180**
IBRNet [53]	Per-scene Optimization	31.35	0.956	0.131	28.14	0.942	0.072	**26.73**	0.851	0.175
MVSNeRF [6]		28.50	0.933	0.179	27.07	0.931	0.168	25.45	**0.877**	0.192
NeRF [36]		27.01	0.902	0.263	**31.01**	0.947	0.081	26.50	0.811	0.250
GeoNeRF$_{10k}$		**31.66**	**0.961**	**0.059**	30.42	**0.956**	**0.055**	26.58	0.856	**0.162**
GeoNeRF$_{1k}$		31.52	0.960	**0.059**	29.83	0.952	0.061	26.31	0.852	0.164

[§] The evaluations of the baselines on the DTU MVS dataset [22] are borrowed from the paper MVSNeRF [6]. Also, metrics are calculated for all methods on this dataset on foreground pixels, whose ground truth depths stand inside the scene bound.

Table 1. Quantitative comparison of our proposed GeoNeRF with existing generalizable NeRF models in terms of PSNR (higher is better), SSIM [54] (higher is better), and LPIPS [60] (lower is better) metrics. Highlights are **best** and second best. GeoNeRF is superior to the existing approaches in all experiments in which the methods are evaluated without any per-scene optimization (the top row). Notably, GeoNeRF outperforms others with a significant margin on datasets with relatively sparser source views: DTU MVS [22] and NeRF realistic synthetic [36]. In particular, our method generalizes outstandingly well on the NeRF synthetic dataset [36], although our training dataset only contains real scenes which greatly differ from the synthetic scenes. The bottom row of the table presents the evaluation of the methods when they are fine-tuned on each scene separately, as well as a comparison with vanilla NeRF [36], which is per-scene optimized for 200k–500k iterations. After fine-tuning for only 10k iterations, our GeoNeRF$_{10k}$ produces competitive results with NeRF. Remarkably, even after fine-tuning for 1k iterations (approximately one hour on a single V100 GPU), GeoNeRF$_{1k}$ reaches 98.15% of the GeoNeRF$_{10k}$'s performance on average, which is another evidence for efficient convergence of our model on novel scenes.

incorrect exposure from the DTU dataset [22] following the practice in pixelNeRF [59] and use the same 88 scenes for training as in pixelNeRF [59] and MVSNeRF [6]. Ground truth depths of DTU [22], provided by [56], are the only data that is directly used for depth supervision. For samples from forward-facing datasets (35 scenes from LLFF [35] and 67 scenes from IBRNet [53]), depth supervision is in the self-supervised form.

Evaluation datasets. We evaluate our model on the 16 test scenes of DTU MVS [22], the 8 test scenes of real forward-facing dataset from LLFF [35], and the 8 scenes in NeRF realistic synthetic dataset [36]. We follow the same evaluation protocol in NeRF [36] for the synthetic dataset [36] and LLFF dataset [35] and the same protocol in MVSNeRF [6] for the DTU dataset [22]. Specifically, for LLFF [35], we hold out $\frac{1}{8}$ of the views of the unseen scenes, and for DTU [22], we hold out 4 views of the unseen scenes for testing and leave the rest for fine-tuning.

Implementation details. We train the generalizable GeoNeRF for 250k iterations. For each iteration, one scene is randomly sampled, and 512 rays are randomly selected as the training batch. For training on a specific scene, we only fine-tune the model for 10k iterations (GeoNeRF$_{10k}$), in contrast to NeRF [36] that requires 200k–500k optimization steps per scene. Since our renderer's architecture is agnostic to the number of source views, we flexibly employ a different number of source views V for training and evaluation to reduce memory usage. We use $V = 6$ source views for training the generalizable model and $V = 9$ for evaluation. For fine-tuning, we select V based on the images' resolution and available GPU memory. Specifically, we set $V = 9$ for the DTU dataset [22] and $V = 7$ for the other two datasets. We fix the number of sample points on a ray to $N_c = 96$ and $N_f = 32$ for all scenes. We utilize Adam [25] as the optimizer with a learning rate of 5×10^{-4} for training the generalizable model and a learning rate of 2×10^{-4} for fine-tuning. A cosine learning rate scheduler [32] without restart is also applied to the optimizer. For the details of our networks' architectures, refer to the supplementary.

Experimental Results. We evaluate our model and provide a comparison with the original vanilla NeRF [36] and the existing generalizable NeRF models: pixelNeRF [59], IBRNet [53], and MVSNeRF [6]. The authors of NeRFormer [42] did not publish their code, did not benchmark their method on NeRF benchmarking datasets, nor test state-of-the-art generalizable NeRF models on their own dataset. They perform on par with NeRF in their experiments with scene-specific optimization and train and test their generalizable model on specific object categories separately. A quantitative comparison is provided in Table 1 in terms of PSNR, SSIM [54], and LPIPS [60]. The results show the superiority of our GeoNeRF model with respect to the previous generalizable models. Moreover, when fine-tuned on the scenes for only 10k iterations, GeoNeRF$_{10k}$ produces competitive results with NeRF, which requires lengthy per-scene optimization. We further show that even

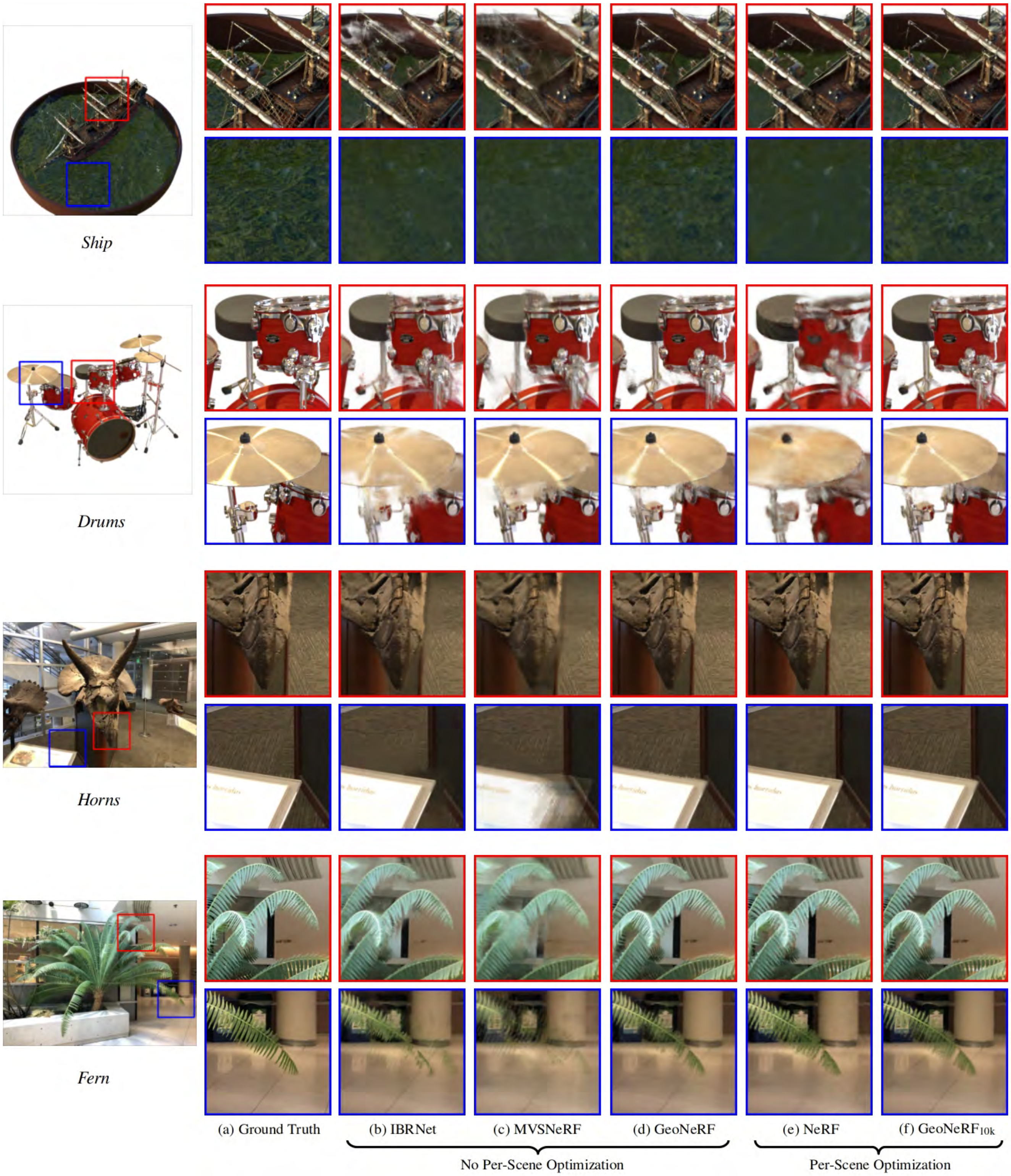


Figure 3. Qualitative comparison of the methods on the NeRF synthetic dataset [33] (*Ship* and *Drums*) and the real forward-facing dataset [35] (*Horns* and *Fern*). Our proposed GeoNeRF more accurately preserves the details of the scenes while it generates fewer artifacts than IBRNet [53] and MVSNeRF [6] (*e.g.* the leaves in *Fern* or the cymbal in *Drums*). After fine-tuning our model only for 10k iterations on each individual scene (GeoNeRF$_{10k}$), the results are competitive with per-scene optimized vanilla NeRF [36]. Compared with NeRF, GeoNeRF models produce smoother surfaces in *Drums* and higher quality textures for the water in *Ship* and for the floor in *Horns*.

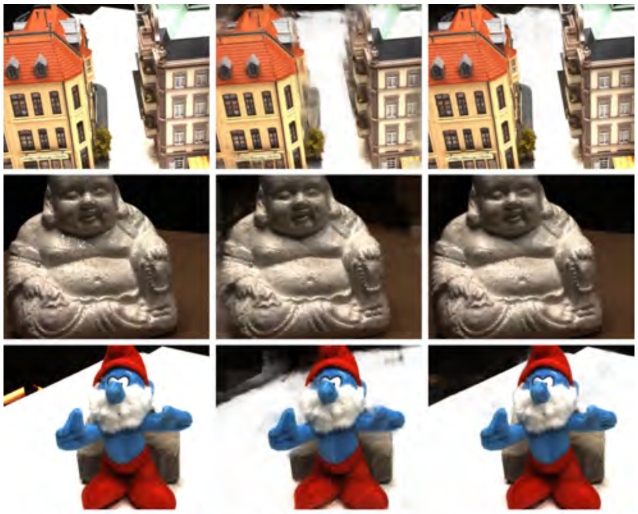


Figure 4. Qualitative comparison of our generalizable GeoNeRF model with MVSNeRF [6], the state-of-the-art model on the DTU dataset [22]. Images are from DTU test scenes. Our method renders sharper images with fewer artifacts.

after 1k iterations (approximately one hour on a single V100 GPU), GeoNeRF$_{1k}$'s results are comparable with NeRF's.

The qualitative comparisons of our model with existing methods on different datasets are provided in Figures 3 and 4. The images produced by our GeoNeRF model better preserve the details of the scene and contain fewer artifacts. For further qualitative analysis as well as demo videos synthesized by our model, refer to the supplementary.

4.1. Ablation Study

An ablation study of our generalizable model on the real forward-facing dataset is presented in Table 2, contrasting the effectiveness of individual components of our proposed model. We evaluated GeoNeRF in the cases where (1) no self-supervision loss is used, (2) no positional encoding is employed, (3) points on a ray are merely sampled uniformly, (4) occluded views are not excluded, (5) view-independent tokens are not regularized with the AE network before predicting volume densities, and (6) only a single cost volume is constructed per-view instead of cascaded multi-level cost volumes.

We also evaluate the impact of the number of source views on our model in Table 3. The results demonstrate the robustness of our model to the sparsity of source views and suggest that GeoNeRF produces high-quality images even with a lower number of source images.

4.2. Results with RGBD Images

To evaluate our RGBD compatible model, GeoNeRF$_{+D}$, we use the DTU dataset [22] to mimic the real scenario where incomplete, low-resolution depth images accompany RGB images. We feed our model the DTU images with a resolution of 600 × 800, while we resize their incomplete depths to 150 × 200. The comparison of the performance of GeoNeRF, with and without depth inputs is presented in Table 4. The results confirm that our GeoNeRF$_{+D}$ model adapts to RGBD images and renders higher quality outputs.

Experiment	PSNR↑	SSIM↑	LPIPS↓
1. W/o self-supervision	25.37	0.836	0.184
2. W/o positional enc.	25.02	0.836	0.189
3. Uniform sampling	25.31	0.835	0.184
4. W/o occlusion masks	25.22	0.834	0.185
5. W/o the AE network	24.92	0.821	0.199
6. Single cost volume	24.60	0.814	0.211
7. Full GeoNeRF	**25.44**	**0.839**	**0.180**

Table 2. Ablation study of the key components of GeoNeRF. The evaluation is performed on the real forward-facing LLFF [35] test scenes. For the details of these experiments, see Section 4.1.

Number of source views	PSNR↑	SSIM↑	LPIPS↓
3	24.33	0.794	0.212
4	25.05	0.823	0.183
5	25.25	0.832	0.178
6	25.37	0.837	**0.176**
9	**25.44**	**0.839**	0.180
IBRNet [53] (10 views)	25.13	0.817	0.205

Table 3. Quantitative analysis of the robustness of our GeoNeRF to the number of input source views on the LLFF [35] test scenes, besides a comparison with IBRNet [53], which uses 10 source views.

Model	PSNR↑	SSIM↑	LPIPS↓
GeoNeRF	31.34	0.959	0.060
GeoNeRF$_{+D}$	**31.58**	**0.961**	**0.057**

Table 4. A comparison of the performance of our RGBD compatible GeoNeRF$_{+D}$ and original GeoNeRF on DTU [22] test scenes. For the details of this experiment, see Section 4.2.

5. Conclusion

We proposed GeoNeRF, a generalizable learning-based novel view synthesis method that renders state-of-the-art quality images for complex scenes without per-scene optimization. Our method leverages the recent architectures in the multi-view stereo field to understand the scene's geometry and occlusions by constructing cascaded cost volumes for source views. The data coming from the source views are then aggregated through an attention-based network, and images for novel poses are synthesized conditioned on these data. An advanced algorithm to select a proper set of nearby views or an adaptive approximation of the optimal number of required cost volumes for a scene could be promising extensions to our method.

References

[1] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, pages 696–712. Springer, 2020. 2

[2] Relja Arandjelović and Andrew Zisserman. Nerf in detail: Learning to sample for view synthesis. *arXiv preprint arXiv:2106.05264*, 2021. 4

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

[4] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 425–432, 2001. 2

[5] Dan Casas, Christian Richardt, John Collomosse, Christian Theobalt, and Adrian Hilton. 4d model flow: Precomputed appearance alignment for real-time 4d video interpolation. In *Computer Graphics Forum*, volume 34, pages 173–182. Wiley Online Library, 2015. 2

[6] 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. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14124–14133, October 2021. 1, 2, 6, 7, 8

[7] Julian Chibane, Aayush Bansal, Verica Lazova, and Gerard Pons-Moll. Stereo radiance fields (srf): Learning view synthesis for sparse views of novel scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7911–7920, 2021. 1

[8] Jeremy S De Bonet and Paul Viola. Poxels: Probabilistic voxelized volume reconstruction. In *Proceedings of International Conference on Computer Vision (ICCV)*, pages 418–425, 1999. 2

[9] Paul Debevec, Yizhou Yu, and George Borshukov. Efficient view-dependent image-based rendering with projective texture-mapping. In *Eurographics Workshop on Rendering Techniques*, pages 105–116. Springer, 1998. 2

[10] Paul E Debevec, Camillo J Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 11–20, 1996. 2

[11] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. *arXiv preprint arXiv:2107.02791*, 2021. 5

[12] Terrance DeVries, Miguel Angel Bautista, Nitish Srivastava, Graham W. Taylor, and Joshua M. Susskind. Unconstrained scene generation with locally conditioned radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14304–14313, October 2021. 2

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020. 4

[14] Ruofei Du, Ming Chuang, Wayne Chang, Hugues Hoppe, and Amitabh Varshney. Montage4d: interactive seamless fusion of multiview video textures. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, pages 1–11, 2018. 2

[15] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. Deepview: View synthesis with learned gradient descent. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2367–2376, 2019. 2

[16] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5515–5524, 2016. 2

[17] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2009. 2

[18] Xiaodong Gu, Zhiwen Fan, Siyu Zhu, Zuozhuo Dai, Feitong Tan, and Ping Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2495–2504, 2020. 2

[19] Philipp Henzler, Niloy J Mitra, and Tobias Ritschel. Learning a neural 3d texture space from 2d exemplars. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8356–8364, 2020. 2

[20] 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. 2

[21] Michal Jancosek and Tomás Pajdla. Multi-view reconstruction preserving weakly-supported surfaces. In *CVPR 2011*, pages 3121–3128. IEEE, 2011. 2

[22] 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. 5, 6, 8

[23] 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

[24] Berk Kaya, Suryansh Kumar, Francesco Sarno, Vittorio Ferrari, and Luc Van Gool. Neural radiance fields approach to deep multi-view photometric stereo. *arXiv preprint arXiv:2110.05594*, 2021. 5

[25] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014. 6

[26] Vladimir Kolmogorov and Ramin Zabih. Multi-camera scene reconstruction via graph cuts. In *European conference on computer vision*, pages 82–96. Springer, 2002. 2

[27] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 31–42, 1996. 2

[28] Jiaxin Li, Zijian Feng, Qi She, Henghui Ding, Changhu Wang, and Gim Hee Lee. Mine: Towards continuous depth mpi with nerf for novel view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12578–12588, 2021. 1, 2

[29] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021. 2

[30] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 2, 3

[31] Yuan Liu, Sida Peng, Lingjie Liu, Qianqian Wang, Peng Wang, Christian Theobalt, Xiaowei Zhou, and Wenping Wang. Neural rays for occlusion-aware image-based rendering. *arXiv preprint arXiv:2107.13421*, 2021. 2

[32] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *International Conference on Learning Representations (ICLR)*, 2017. 6

[33] 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. 2, 7

[34] 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. 2

[35] 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. 5, 6, 7, 8

[36] 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. 1, 2, 4, 6, 7

[37] Phong Nguyen, Animesh Karnewar, Lam Huynh, Esa Rahtu, Jiri Matas, and Janne Heikkila. Rgbd-net: Predicting color and depth images for novel views synthesis. *arXiv preprint arXiv:2011.14398*, 2020. 2

[38] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *International Conference on Computer Vision (ICCV)*, 2021. 5

[39] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 2

[40] Sida Peng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9054–9063, 2021. 2

[41] Eric Penner and Li Zhang. Soft 3d reconstruction for view synthesis. *ACM Transactions on Graphics (TOG)*, 36(6):1–11, 2017. 2

[42] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10901–10911, 2021. 1, 4, 6

[43] Chris Rockwell, David F Fouhey, and Justin Johnson. Pixelsynth: Generating a 3d-consistent experience from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14104–14113, 2021. 2

[44] Radu Alexandru Rosu and Sven Behnke. Neuralmvs: Bridging multi-view stereo and novel view synthesis. *arXiv preprint arXiv:2108.03880*, 2021. 2

[45] 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. 2

[46] Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision*, pages 501–518. Springer, 2016. 2

[47] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. *Advances in Neural Information Processing Systems*, 33, 2020. 2

[48] 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. 2

[49] Pratul P Srinivasan, Richard Tucker, Jonathan T Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 175–184, 2019. 2

[50] Alex Trevithick and Bo Yang. Grf: Learning a general radiance field for 3d representation and rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15182–15192, 2021. 1, 2

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia

Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 4

[52] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *NeurIPS*, 2021. 5

[53] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2021. 1, 2, 4, 5, 6, 7, 8

[54] 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

[55] Qingshan Xu and Wenbing Tao. Learning inverse depth regression for multi-view stereo with correlation cost volume. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34 (07), pages 12508–12515, 2020. 2, 4

[56] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 767–783, 2018. 1, 2, 6

[57] Yao Yao, Zixin Luo, Shiwei Li, Tianwei Shen, Tian Fang, and Long Quan. Recurrent mvsnet for high-resolution multi-view stereo depth inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5525–5534, 2019. 2

[58] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *arXiv preprint arXiv:2106.12052*, 2021. 5

[59] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021. 1, 2, 4, 6

[60] 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

[61] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: learning view synthesis using multiplane images. *ACM Transactions on Graphics (TOG)*, 37(4):1–12, 2018. 2

arXiv:2111.13539v1 [cs.CV] 26 Nov 2021

Supplementary Materials for GeoNeRF: Generalizing NeRF with Geometry Priors

Mohammad Mahdi Johari
Idiap Reserach Institute, EPFL
mohammad.johari@idiap.ch

Yann Lepoittevin
ams OSRAM
yann.lepoittevin@ams.com

François Fleuret
University of Geneva, EPFL
francois.fleuret@unige.ch

1. Additional Technical Details

As stated in the main article, we borrow the architecture of our geometry reasoner from CasMVSNet [2]. We construct $D^{(2)} = 48$ depth planes for the coarsest cost volume, $D^{(1)} = 32$ for the intermediate one, and $D^{(0)} = 8$ for the finest full-resolution cost volume. We use channel size $C = 8$ in group-wise correlation similarity calculations. When training the generalizable model, we create a set of 3–5 nearby source views for constructing each cost volume, whereas for fine-tuning and evaluating, we always use a set of 5 nearby views. Also, we scale input images with a factor uniformly sampled from $\{1.0, 0.75, 0.5\}$ when we train our generalizable model.

The network architectures of Feature Pyramid Network (FPN), 3D regularizer ($R^{(l)}_{3D}$), and the auto-encoder (AE) are provided in Tables 1, 2, and 3 respectively.

2. Additional Qualitative Analysis

Full-size examples of rendered images for novel views by our GeoNeRF model are presented in Figures 1 and 2. Figure 1 includes samples from the real forward-facing LLFF dataset [3], and Figure 2 contains samples from the NeRF realistic synthetic dataset [4]. In addition to the rendered images, we also show the rendered depth maps for each novel view. Images

Input	Layer	Output
Input	ConvBnReLU(3, 8, 3, 1)	conv0_0
conv0_0	ConvBnReLU(8, 8, 3, 1)	conv0
conv0	ConvBnReLU(8, 16, 5, 2)	conv1_0
conv1_0	ConvBnReLU(16, 16, 3, 1)	conv1_1
conv1_1	ConvBnReLU(16, 16, 3, 1)	conv1
conv1	ConvBnReLU(16, 32, 5, 2)	conv2_0
conv2_0	ConvBnReLU(32, 32, 3, 1)	conv2_1
conv2_1	ConvBnReLU(32, 32, 3, 1)	conv2
conv2	Conv(32, 32, 1, 1)	feat2
conv1	Conv(16, 32, 1, 1)	f1_0
conv0	Conv(8, 32, 1, 1)	f0_0
(feat2, f1_0)	Upsample_and_Add(x, y)	f1_1
(f1_1, f0_0)	Upsample_and_Add(x, y)	f0_1
f1_1	Conv(32, 16, 3, 1)	feat1
f0_1	Conv(32, 8, 3, 1)	feat0

Table 1. Network architecture of Feature Pyramid Network (FPN), where *feat2*, *feat1*, and *feat0* are output feature pyramids. Conv(c_{in}, c_{out}, k, s) stands for a 2D convolution with input channels c_{in}, output channels c_{out}, kernel size of k, and stride of s. ConvBnReLU represents a Conv layer followed by Batch Normalization and ReLU nonlinearity. Upsample_and_Add(x, y) adds y to the bilinearly upsampled of x.

Input	Layer	Output
Input	ConvBnReLU(8, 8, 3, 1)	conv0
conv0	ConvBnReLU(8, 16, 3, 2)	conv1
conv1	ConvBnReLU(16, 16, 3, 1)	conv2
conv2	ConvBnReLU(16, 32, 3, 2)	conv3
conv3	ConvBnReLU(32, 32, 3, 1)	conv4
conv4	ConvBnReLU(32, 64, 3, 2)	conv5
conv5	ConvBnReLU(64, 64, 3, 1)	conv6
conv6	TrpsConvBnReLU(64, 32, 3, 2)	x_0
(conv4, x_0)	Add(x, y)	x_1
x_1	TrpsConvBnReLU(32, 16, 3, 2)	x_2
(conv2, x_2)	Add(x, y)	x_3
x_3	TrpsConvBnReLU(16, 8, 3, 2)	x_4
(conv0, x_4)	Add(x, y)	x_5
x_5	ConvBnReLU(8, 8, 3, 1)	prob_0
prob_0	Conv(8, 1, 3, 1)	prob
x_5	ConvBnReLU(8, 8, 3, 1)	feat

Table 2. Network architecture of the 3D regularizer ($R_{3D}^{(l)}$), where *feat* is the output 3D feature map $\mathbf{\Phi}^{(l)}$ and *prob* is the output probablity which is used to regress the depth map $\hat{\boldsymbol{D}}^{(l)}$. Conv($c_{in}$, c_{out}, k, s) stands for a 3D convolution with input channels c_{in}, output channels c_{out}, kernel size of k, and stride of s. ConvBnReLU represents a Conv layer followed by Batch Normalization and ReLU nonlinearity, and TrpsConv stands for transposed 3D convolution. Add(x, y) simply adds y to x.

Input	Layer	Output
Input	ConvLnELU(32, 64, 3, 1)	conv1_0
conv1_0	MaxPool	conv1
conv1	ConvLnELU(64, 128, 3, 1)	conv2_0
conv2_0	MaxPool	conv2
conv2	ConvLnELU(128, 128, 3, 1)	conv3_0
conv3_0	MaxPool	conv3
conv3	TrpsConvLnELU(128, 128, 4, 2)	x_0
[conv2 ; x_0]	TrpsConvLnELU(256, 64, 4, 2)	x_1
[conv1 ; x_1]	TrpsConvLnELU(128, 32, 4, 2)	x_2
[Input ; x_2]	ConvLnELU(64, 32, 3, 1)	Output

Table 3. Network architecture of the auto-encoder network (AE). Conv(c_{in}, c_{out}, k, s) stands for a 1D convolution with input channels c_{in}, output channels c_{out}, kernel size of k, and stride of s. ConvLnELU represents a Conv layer followed by Layer Normalization and ELU nonlinearity, and TrpsConv stands for transposed 1D convolution. MaxPool is a 1D max pooling layer with a stride of 2, and [$\cdot$; $\cdot$] denotes concatenation.

indicated by GeoNeRF are rendered by our generalizable model, while images indicated by GeoNeRF$_{\text{10k}}$ are rendered after fine-tuning our model on each scene for 10k iterations.

3. Per-Scene Breakdown

Tables 4, 5, 6, and 7 break down the quantitative results presented in the main paper into per-scene metrics. The results are consistent with the aggregate results in the main paper. Tables 4 and 5 include the scenes from the real forward-facing LLFF dataset [3], and Tables 6 and 7 contain the scenes from NeRF realistic synthetic dataset [4]. As it was already shown in the main paper, our generalizable GeoNeRF model outperforms all existing generalizable methods on average, and after fine-tuning, it is on par with per-scene optimized vanilla NeRF [4].

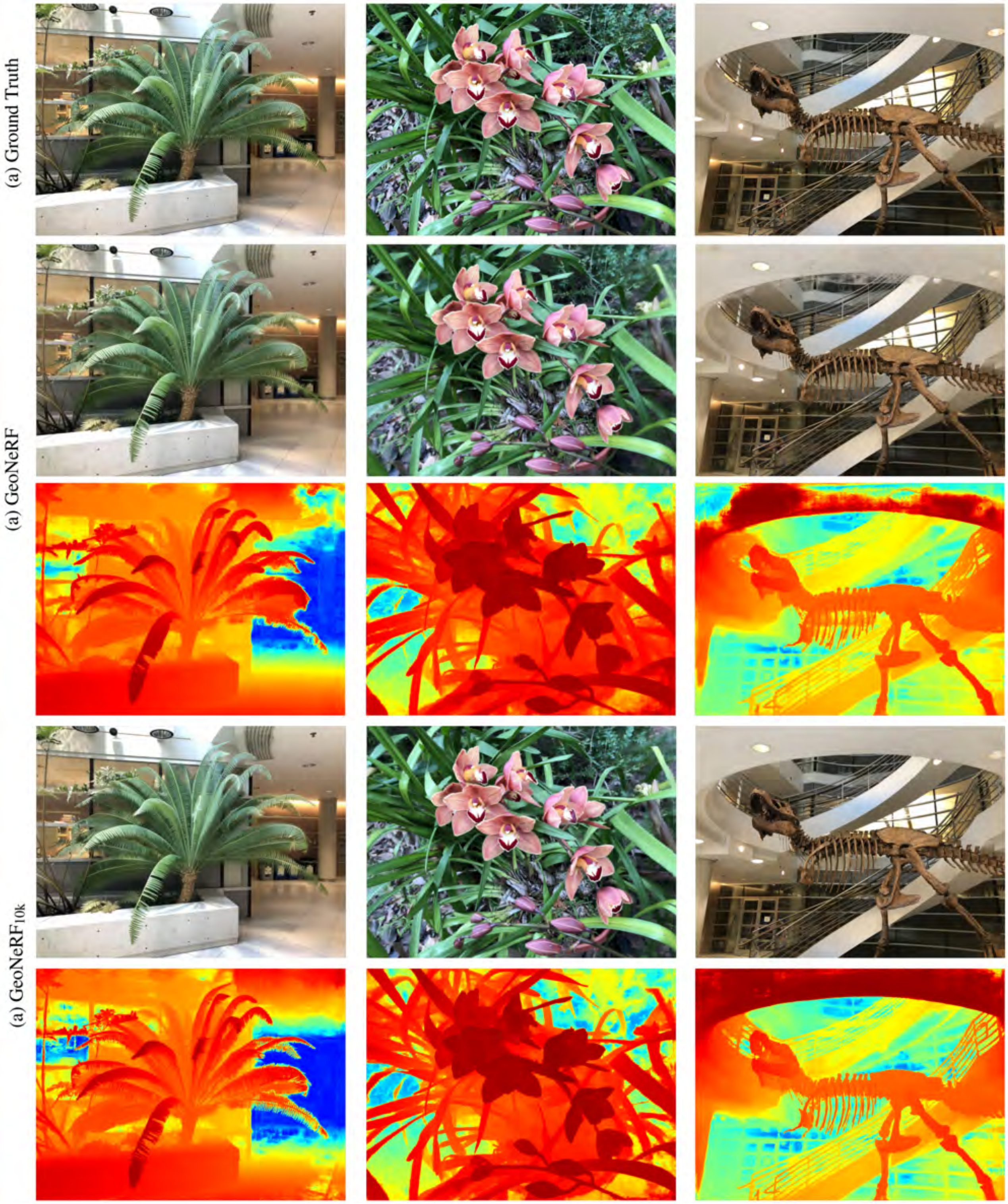


Figure 1. Full-size examples of novel images and their depth map rendered by our generalizable (GeoNeRF) and fine-tuned (GeoNeRF$_{10k}$) models. The images are from test scenes of the real forward-facing LLFF dataset [3].

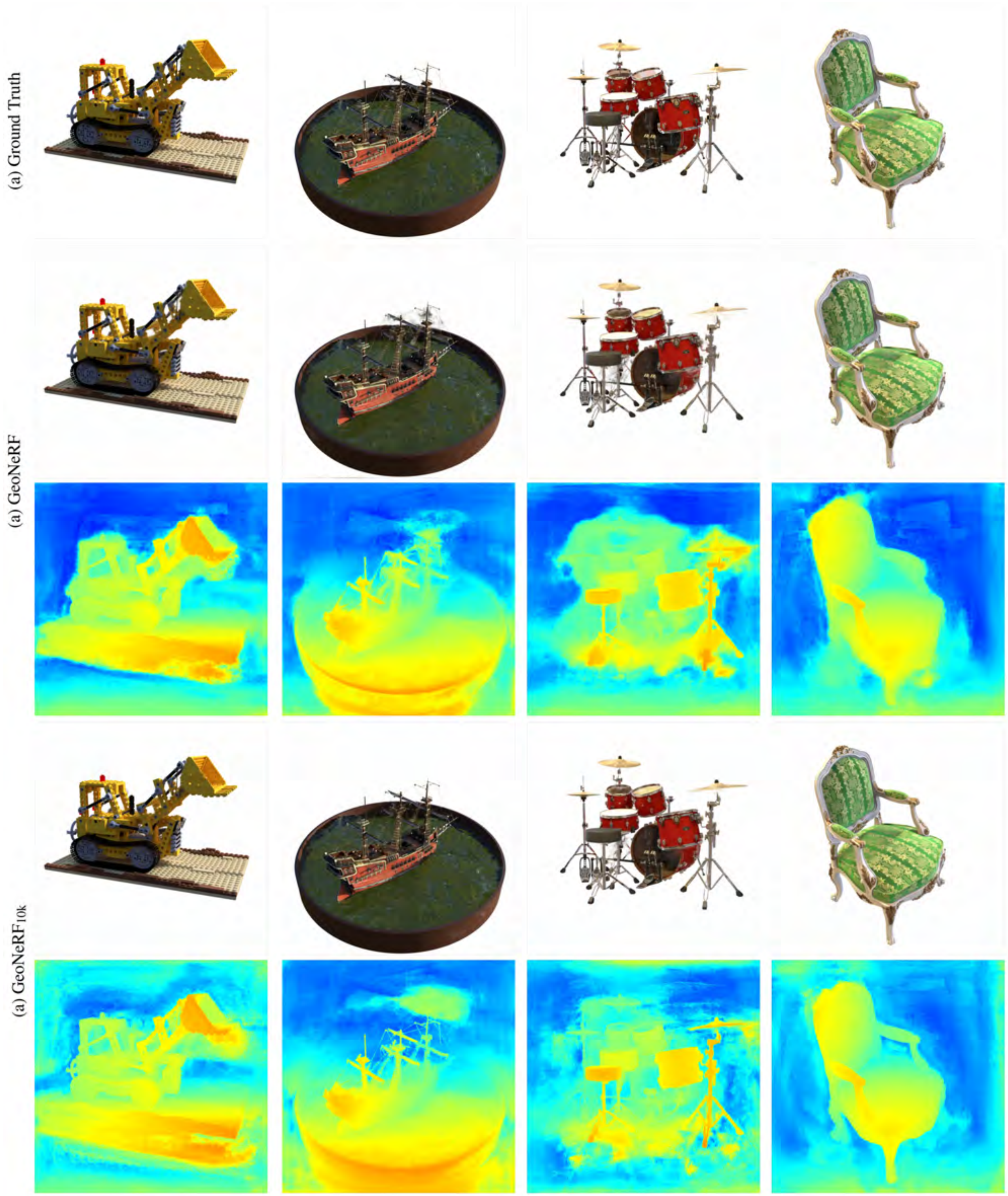


Figure 2. Full-size examples of novel images and their depth map rendered by our generalizable (GeoNeRF) and fine-tuned (GeoNeRF$_{10k}$) models. The images are from test scenes of the NeRF realistic synthetic dataset [4].

	PSNR↑							
	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	T-Rex
pixelNeRF [7]	12.40	10.00	14.07	11.07	9.85	9.62	11.75	10.55
IBRNet [5]	23.84	26.67	30.00	26.48	20.19	19.34	**29.94**	**24.57**
MVSNeRF [1]	21.15	24.74	26.03	23.57	17.51	17.85	26.95	23.20
GeoNeRF	**24.61**	**28.12**	**30.49**	**26.96**	**20.58**	**20.24**	28.74	23.75

	SSIM↑							
	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	T-Rex
pixelNeRF [7]	0.531	0.433	0.674	0.516	0.268	0.317	0.691	0.458
IBRNet [5]	0.772	0.856	0.883	0.869	0.719	0.633	0.946	0.861
MVSNeRF [1]	0.638	**0.888**	0.872	0.868	0.667	0.657	**0.951**	0.868
GeoNeRF	**0.811**	0.885	**0.898**	**0.901**	**0.741**	**0.666**	0.935	**0.877**

	LPIPS↓							
	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	T-Rex
pixelNeRF [7]	0.650	0.708	0.608	0.705	0.695	0.721	0.611	0.667
IBRNet [5]	0.246	0.164	0.153	0.177	0.230	0.287	0.153	0.230
MVSNeRF [1]	0.238	0.196	0.208	0.237	0.313	0.274	0.172	**0.184**
GeoNeRF	**0.202**	**0.133**	**0.123**	**0.140**	**0.222**	**0.256**	**0.150**	0.212

Table 4. Per-scene Quantitative comparison of our proposed GeoNeRF with existing generalizable NeRF models on real forward-facing LLFF dataset [3] in terms of PSNR (higher is better), SSIM [6] (higher is better), and LPIPS [8] (lower is better) metrics.

	PSNR↑							
	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	T-Rex
NeRF [4]	25.17	27.40	**31.16**	27.45	20.92	20.36	**32.70**	**26.80**
GeoNeRF$_{\text{10k}}$	**25.24**	28.57	30.75	**28.12**	**21.40**	20.39	31.51	26.63
GeoNeRF$_{\text{1k}}$	25.08	**28.74**	30.83	27.66	21.16	**20.41**	30.52	26.07

	SSIM↑							
	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	T-Rex
NeRF [4]	0.792	0.827	0.881	0.828	0.690	0.641	0.948	0.880
GeoNeRF$_{\text{10k}}$	**0.829**	0.890	0.900	**0.912**	**0.781**	**0.674**	**0.956**	**0.910**
GeoNeRF$_{\text{1k}}$	0.824	**0.892**	**0.905**	0.908	0.769	0.673	0.946	0.901

	LPIPS↓							
	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	T-Rex
NeRF [4]	0.280	0.219	0.171	0.268	0.316	0.321	0.178	0.249
GeoNeRF$_{\text{10k}}$	**0.185**	0.120	0.125	**0.126**	**0.183**	**0.247**	**0.126**	**0.181**
GeoNeRF$_{\text{1k}}$	0.189	**0.114**	**0.117**	0.130	0.198	0.248	0.135	0.188

Table 5. Per-scene Quantitative comparison of our fine-tuned GeoNeRF with per-scene optimized vanilla NeRF [4] on real forward-facing LLFF dataset [3] in terms of PSNR (higher is better), SSIM [6] (higher is better), and LPIPS [8] (lower is better) metrics. Our model is fine-tuned on each scene for 10k iterations (GeoNeRF$_{\text{10k}}$) and 1k iterations (GeoNeRF$_{\text{1k}}$), and NeRF [4] is optimized for 200k iterations.

	PSNR↑							
	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
pixelNeRF [7]	7.18	8.15	6.61	6.80	7.74	7.61	7.71	7.30
IBRNet [5]	28.54	21.22	24.23	31.72	24.59	22.20	27.97	23.64
MVSNeRF [1]	23.35	20.71	21.98	28.44	23.18	20.05	22.62	23.35
GeoNeRF	**31.84**	**24.00**	**25.28**	**34.33**	**28.80**	**26.16**	**31.15**	**25.08**

	SSIM↑							
	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
pixelNeRF [7]	0.624	0.670	0.669	0.669	0.671	0.644	0.729	0.584
IBRNet [5]	0.948	0.896	0.915	0.952	0.918	0.905	0.962	0.834
MVSNeRF [1]	0.876	0.886	0.898	0.962	0.902	0.893	0.923	**0.886**
GeoNeRF	**0.973**	**0.921**	**0.931**	**0.975**	**0.956**	**0.926**	**0.978**	0.844

	LPIPS↓							
	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
pixelNeRF [7]	0.386	0.421	0.335	0.433	0.427	0.432	0.329	0.526
IBRNet [5]	0.066	**0.091**	0.097	0.067	0.095	**0.115**	0.051	0.219
MVSNeRF [1]	0.282	0.187	0.211	0.173	0.204	0.216	0.177	0.244
GeoNeRF	**0.040**	0.098	**0.092**	**0.056**	**0.059**	0.116	**0.037**	**0.200**

Table 6. Per-scene Quantitative comparison of our proposed GeoNeRF with existing generalizable NeRF models on NeRF realistic syntethic dataset [4] in terms of PSNR (higher is better), SSIM [6] (higher is better), and LPIPS [8] (lower is better) metrics.

	PSNR↑							
	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
NeRF [4]	33.00	25.01	**30.13**	36.18	**32.54**	**29.62**	32.91	28.65
GeoNeRF$_{\text{10k}}$	**33.54**	**25.13**	27.79	**36.26**	30.32	28.19	**33.41**	**28.76**
GeoNeRF$_{\text{1k}}$	32.76	24.74	27.06	35.71	29.79	27.69	32.83	28.11

	SSIM↑							
	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
NeRF [4]	0.967	0.925	**0.964**	0.974	0.961	0.949	0.980	0.856
GeoNeRF$_{\text{10k}}$	**0.980**	**0.935**	0.955	**0.983**	**0.965**	**0.953**	**0.987**	**0.890**
GeoNeRF$_{\text{1k}}$	0.977	0.930	0.948	0.982	0.961	0.948	0.985	0.883

	LPIPS↓							
	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
NeRF [4]	0.046	0.091	**0.044**	0.121	0.050	0.063	0.028	0.206
GeoNeRF$_{\text{10k}}$	**0.024**	**0.073**	0.061	**0.032**	**0.041**	**0.058**	**0.016**	**0.137**
GeoNeRF$_{\text{1k}}$	0.030	0.081	0.069	0.034	0.046	0.069	0.020	0.145

Table 7. Per-scene Quantitative comparison of our fine-tuned GeoNeRF with per-scene optimized vanilla NeRF [4] on NeRF realistic synthetic dataset [4] in terms of PSNR (higher is better), SSIM [6] (higher is better), and LPIPS [8] (lower is better) metrics. Our model is fine-tuned on each scene for 10k iterations (GeoNeRF$_{\text{10k}}$) and 1k iterations (GeoNeRF$_{\text{1k}}$), and NeRF [4] is optimized for 500k iterations.

References

[1] 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. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14124–14133, October 2021. 5, 6

[2] Xiaodong Gu, Zhiwen Fan, Siyu Zhu, Zuozhuo Dai, Feitong Tan, and Ping Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2495–2504, 2020. 1

[3] 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. 1, 2, 3, 5

[4] 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. 1, 2, 4, 5, 6

[5] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2021. 5, 6

[6] 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. 5, 6

[7] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021. 5, 6

[8] 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. 5, 6