MVSNeRF: Fast Generalizable Radiance Field Reconstruction from Multi-View Stereo

Anpei Chen*[1] Zexiang Xu*[2] Fuqiang Zhao[1] Xiaoshuai Zhang[3] Fanbo Xiang[3]
Jingyi Yu[1] Hao Su[3]
[1] ShanghaiTech University [2] Adobe Research [3] University of California, San Diego
{chenap,zhaofq,yujingyi}@shanghaitech.edu.cn zexu@adobe.com {xiz040,fxiang,haosu}@eng.ucsd.edu

Figure 1. We train our MVSNeRF with scenes of objects in the DTU dataset [20]. Our network can effectively *generalize* across diverse scenes; even for a complex indoor scene, our network can reconstruct a neural radiance field from only three input images (a) and synthesize a realistic image from a novel viewpoint (b). While this result contains artifacts, it can be largely improved by fine-tuning our reconstruction on more images for only 6 min ($4k$ iterations) (c), which achieves comparable quality with the NeRF's [34] result (d) from 5.1h per-scene optimization ($100k$ iterations).

Abstract

We present MVSNeRF, a novel neural rendering approach that can efficiently reconstruct neural radiance fields for view synthesis. Unlike prior works on neural radiance fields that consider per-scene optimization on densely captured images, we propose a generic deep neural network that can reconstruct radiance fields from only three nearby input views via fast network inference. Our approach leverages plane-swept cost volumes (widely used in multi-view stereo) for geometry-aware scene reasoning, and combines this with physically based volume rendering for neural radiance field reconstruction. We train our network on real objects in the DTU dataset, and test it on three different datasets to evaluate its effectiveness and generalizability. Our approach can generalize across scenes (even indoor scenes, completely different from our training scenes of objects) and generate realistic view synthesis results using only three input images, significantly outperforming concurrent works on generalizable radiance field reconstruction. Moreover, if dense images are captured, our estimated radiance field representation can be easily fine-tuned; this leads to fast per-scene reconstruction with higher rendering quality and substantially less optimization time than NeRF.

1. Introduction

Novel view synthesis is a long-standing problem in computer vision and graphics. Recently, neural rendering approaches have significantly advanced the progress in this area. Neural radiance fields (NeRF) and its following works [34, 31, 27] can already produce photo-realistic novel view synthesis results. However, one significant drawback of these prior works is that they require a very long per-scene optimization process to obtain high-quality radiance fields, which is expensive and highly limits the practicality.

Our goal is to make neural scene reconstruction and rendering more practical, by enabling *highly efficient* radiance field estimation. We propose MVSNeRF, a novel approach that *generalizes well across scenes* for the task

∗ Equal contribution
Research done when Anpei Chen was in a remote internship with UCSD.

of reconstructing a radiance field from *only several* (as few as three) unstructured multi-view input images. With strong generalizability, we avoid the tedious per-scene optimization and can directly regress realistic images at novel viewpoints via fast network inference. If further optimized on more images with only a short period (5-15 min), our reconstructed radiance fields can even outperform NeRFs [34] with hours of optimization (see Fig. 1).

We take advantage of the recent success on deep multi-view stereo (MVS) [50, 18, 10]. This line of work can train generalizable neural networks for the task of 3D reconstruction by applying 3D convolutions on cost volumes. Similar to [50], we build a cost volume at the input reference view by warping 2D image features (inferred by a 2D CNN) from nearby input views onto sweeping planes in the reference view's frustrum. Unlike MVS methods [50, 10] that merely conduct depth inference on such a cost volume, our network reasons about both scene geometry and appearance, and outputs a neural radiance field (see Fig. 2), enabling view synthesis. Specifically, leveraging 3D CNN, we reconstruct (from the cost volume) a *neural scene encoding volume* that consists of per-voxel neural features that encode information about the local scene geometry and appearance. Then, we make use of a multi-layer perceptron (MLP) to decode the volume density and radiance at arbitrary continuous locations using tri-linearly interpolated neural features inside the encoding volume. In essence, the encoding volume is a localized neural representation of the radiance field; once estimated, this volume can be used directly (dropping the 3D CNN) for final rendering by differentiable ray marching (as in [34]).

Our approach takes the best of the two worlds, learning-based MVS and neural rendering. Compared with existing MVS methods, we enable differentiable neural rendering that allows for training without 3D supervision and inference time optimization for further quality improvement. Compared with existing neural rendering works, our MVS-like architecture is natural to conduct cross-view correspondence reasoning, facilitating the generalization to unseen testing scenes and also leading to better neural scene reconstruction and rendering. Our approach can, therefore, significantly outperform the recent concurrent generalizable NeRF work [54, 46] that mainly considers 2D image features without explicit geometry-aware 3D structures (See Tab. 1 and Fig. 4). We demonstrate that, using only three input images, our network trained from the DTU dataset can synthesize photo-realistic images on testing DTU scenes, and can even generate reasonable results on other datasets that have very different scene distributions. Moreover, our estimated three-image radiance field (the neural encoding volume) can be further easily optimized on novel testing scenes to improve the neural reconstruction if more images are captured, leading to photo-realistic results that are comparable or even better than a per-scene overfit NeRF, despite of ours taking substantially less optimization time than NeRF (see Fig. 1).

These experiments showcase that our technique can be used either as a strong reconstructor that can reconstruct a radiance field for realistic view synthesis when there are only few images captured, or as a strong initializor that significantly facilitates the per-scene radiance field optimization when dense images are available. Our approach takes an important step towards making realistic neural rendering practical. We have released the code at mvsnerf.github.io.

2. Related Work

Multi-view stereo. Multi-view stereo (MVS) is a classical computer vision problem, aiming to achieve dense geometry reconstruction using images captured from multiple viewpoints, and has been extensively explored by various traditional methods [12, 24, 23, 14, 39, 16, 38]. Recently, deep learning techniques have been introduced to address MVS problems [50, 19]. MVSNet [50] applies a 3D CNN on a plane-swept cost volume at the reference view for depth estimation, achieving high-quality 3D reconstruction that outperforms classical traditional methods [16, 38]. Following works have extended this technique with recurrent plane sweeping [51], point-based densification [8], confidence-based aggregation [30], and multiple cost volumes [10, 18], improving the reconstruction quality. We propose to combine the cost-volume based deep MVS technique with differentiable volume rendering, enabling efficient reconstruction of radiance fields for neural rendering. Unlike MVS approaches that use direct depth supervision, we train our network with image loss only for novel view synthesis. This ensures the network to satisfy multi-view consistency, naturally allowing for high-quality geometry reconstruction. As a side product, our MVSNeRF can achieve accurate depth reconstruction (despite of no direct depth supervision) comparable to the MVSNet [50]. This can potentially inspire future work on developing unsupervised geometry reconstruction methods.

View synthesis. View synthesis has been studied for decades with various approaches including light fields [17, 25, 47, 21, 42, 7], image-based rendering [13, 3, 40, 5, 4], and other recent deep learning based methods [56, 55, 49, 15]. Plane sweep volumes have also been used for view synthesis [35, 55, 15, 33, 49]. With deep learning, MPI based methods [55, 11, 33, 41] build plane sweep volumes at reference views, while other methods [15, 49] construct plane sweeps at novel viewpoints; these prior works usually predict colors at the discrete sweeping planes and aggregate per-plane colors using alpha-blending or learned weights.

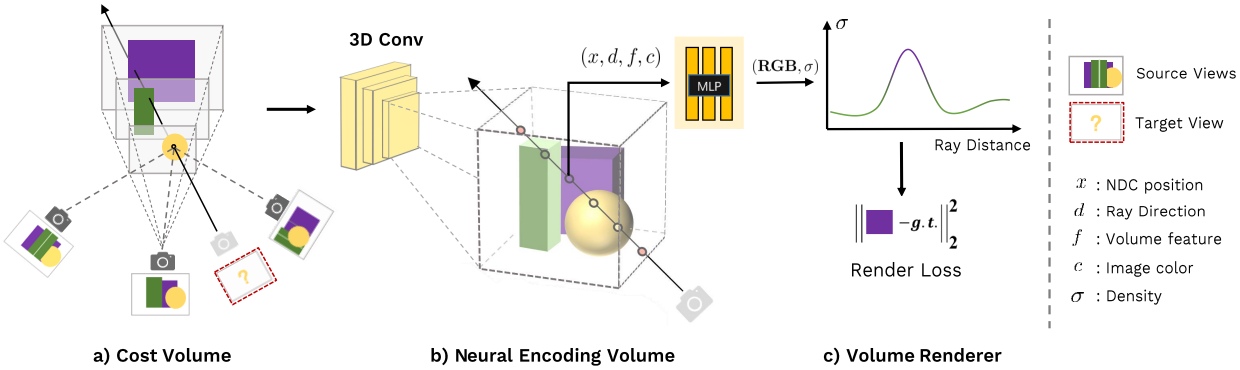

Figure 2. Overview of MVSNeRF. Our framework first constructs a cost volume (a) by warping 2D image features onto a plane sweep. We then apply 3D CNN to reconstruct a neural encoding volume with per-voxel neural features (b). We use an MLP to regress volume density and RGB radiance at an arbitrary location using features interpolated from the encoding volume. These volume properties are used by differentiable ray marching for final rendering (c).

Instead of direct per-plane color prediction, our approach infers per-voxel neural features in the plane sweep as a scene encoding volume and can regress volume rendering properties from it at arbitrary 3D locations. This models a continuous neural radiance field, allowing for physically based volume rendering to achieve realistic view synthesis.

Neural rendering. Recently, various neural scene representations have been presented to achieve view synthesis and geometric reconstruction tasks [55, 45, 28, 2, 34]. In particular, NeRF [34] combines MLPs with differentiable volume rendering and achieves photo-realistic view synthesis. Following works have tried to advance its performance on view synthesis [31, 27]; other relevant works extend it to support other neural rendering tasks like dynamic view synthesis [26, 36, 43], challenge scenes [29, 52], pose estimation [32], real-time rendering [53], relighting [37, 1, 9], and editing [48, 6]. We refer the readers to [44] for a comprehensive review of neural rendering. However, most prior works still follow the original NeRF and require an expensive per-scene optimization process. We instead leverage deep MVS techniques to achieve across-scene neural radiance field estimation for view synthesis using only few images as input. Our approach utilizes a plane swept 3D cost volume for geometric-aware scene understanding, achieving significantly better performance than concurrent works [54, 46] that only consider 2D image features for the generalization of radiance field reconstruction.

3. MVSNeRF

We now present our MVSNeRF. Unlike NeRF [34] that reconstructs a radiance field via a per-scene "network memorization", our MVSNeRF learns a generic network for radiance field reconstruction. Given M input captured images I_i $(i = 1, .., M)$ of a real scene and their known camera parameters Φ_i, we present a novel network that can reconstruct a radiance feild as a neural encoding volume and use it to regress volume rendering properties (density and view-dependent radiance) at arbitrary scene locations for view synthesis. In general, our entire network can be seen as a function of a radiance field, expressed by:

$$\sigma, r = \text{MVSNeRF}(x, d; I_i, \Phi_i) \quad (1)$$

where x represents a 3D location, d is a viewing direction, σ is the volume density at x, and r is the output radiance (RGB color) at x depending on the viewing direction d. The output volume properties from our network can be directly used to synthesize a novel image I_t at a novel target viewpoint Φ_t via differentiable ray marching.

In this paper, we consider a sparse set of nearby input views for efficient radiance field reconstruction. In practice we use $M = 3$ views for our experiments, while our approach handles unstructured views and can easily support other numbers of inputs. The overview of our MVSNeRF is shown in Fig. 2. We first build a cost volume at the reference view (we refer to the view $i = 1$ as the reference view) by warping 2D neural features onto multiple sweeping planes (Sec. 3.1). We then leverage a 3D CNN to reconstruct the neural encoding volume, and use an MLP to regress volume rendering properties, expressing a radiance field (Sec. 3.2). We leverage differentiable ray marching to regress images at novel viewpoints using the radiance field modeled by our network; this enables end-to-end training of our entire framework with a rendering loss (Sec. 3.3). Our framework achieves radiance field reconstruction from few images. On the other hand, when dense images are captured, the reconstructed encoding volume and the MLP decoder can also be fast fine-tuned independently to further improve the rendering quality (Sec. 3.4).

3.1. Cost volume construction.

Inspired by the recent deep MVS methods [50], we build a cost volume P at the reference view ($i = 1$), allowing for geometry-aware scene understanding. This is achieved by warping 2D image features from the m input images to a plane sweep volume on the reference view's frustrum.

Extracting image features. We use a deep 2D CNN T to extract 2D image features at individual input views to effectively extract 2D neural features that represent local image appearance. This sub-network consists of downsampling convolutional layers and convert an input image $I_i \in \mathbb{R}^{H_i \times W_i \times 3}$ into a 2D feature map $F_i \in \mathbb{R}^{H_i/4 \times W_i/4 \times C}$,

$$F_i = T(I_i), \tag{2}$$

where H and W are the image height and width, and C is the number of resulting feature channels.

Warping feature maps. Given the camera intrinsic and extrinsic parameters $\Phi = [K, R, t]$, we consider the homographic warping

$$\mathcal{H}_i(z) = K_i \cdot (R_i \cdot R_1^T + \frac{(t_1 - t_i) \cdot n_1^T}{z}) \cdot K_1^{-1} \tag{3}$$

where $\mathcal{H}_i(z)$ is the matrix warping from the view i to the reference view at depth z, K is the intrinsic matrix, and R and t are the camera rotation and translation. Each feature map F_i can be warped to the reference view by:

$$F_{i,z}(u, v) = F_i(\mathcal{H}_i(z)[u, v, 1]^T), \tag{4}$$

where $F_{i,z}$ is the warped feature map at depth z, and (u, v) represents a pixel location in the reference view. In this work, we parameterize (u, v, z) using the normalized device coordinate (NDC) at the reference view.

Cost volume. The cost volume P is constructed from the warped feature maps on the D sweeping planes. We leverage the variance-based metric to compute the cost, which has been widely used in MVS [50, 10] for geometry reconstruction. In particular, for each voxel in P centered at (u, v, z), its cost feature vector is computed by:

$$P(u, v, z) = \mathrm{Var}(F_{i,z}(u, v)), \tag{5}$$

where Var computes the variance across M views.

This variance-based cost volume encodes the image appearance variations across different input views; this explains the appearance variations caused by both scene geometry and view-dependent shading effects. While MVS work uses such a volume only for geometry reconstruction, we demonstrate that it can be used to also infer complete scene appearance and enable realistic neural rendering.

3.2. Radiance field reconstruction.

We propose to use deep neural networks to effectively convert the built cost volume into a reconstruction of radiance field for realistic view synthesis. We utilize a 3D CNN B to reconstruct a neural encoding volume S from the cost volume P of raw 2D image feature costs; S consists of per-voxel features that encode local scene geometry and appearance. An MLP decoder A is used to regress volume rendering properties from this encoding volume.

Neural encoding volume. Previous MVS works [50, 18, 10] usually predict depth probabilities directly from a cost volume, which express scene geometry only. We aim to achieve high-quality rendering that necessitates inferring more appearance-aware information from the cost volume. Therefore, we train a deep 3D CNN B to transform the built image-feature cost volume into a new C-channel neural feature volume S, where the feature space is learned and discovered by the network itself for the following volume property regression. This process is expressed by:

$$S = B(P). \tag{6}$$

The 3D CNN B is a 3D UNet with downsampling and upsampling convolutional layers and skip connections, which can effectively infer and propagate scene appearance information, leading to a meaningful scene encoding volume S. Note that, this encoding volume is predicted in a unsupervised way and inferred in the end-to-end training with volume rendering (see Sec. 3.3). Our network can learn to encode meaningful scene geometry and appearance in the per-voxel neural features; these features are later continuously interpolated and converted into volume density and view-dependent radiance.

The scene encoding volume is of relative low resolution because of the downsampling of 2D feature extraction; it is challenging to regress high-frequency appearance from this information alone. We thus also incorporate the original image pixel data for the following volume regression stage, though we later show that this high-frequency can be also recovered in an augmented volume via a fast per-scene fine-tuning optimization (Sec. 3.4).

Regressing volume properties. Given an arbitrary 3D location x and a viewing direction d, we use an MLP A to regress the corresponding volume density σ and view-dependent radiance r from the neural encoding volume S. As mentioned, we also consider pixel colors $c = [I(u_i, v_i)]$ from the original images I_i as additional input; here (u_i, v_i) is the pixel location when projecting the 3D point x onto view i, and c concatenates the colors $I(u_i, v_i)$ from all views as a $3M$-channel vector. The MLP is expressed by:

$$\sigma, r = A(x, d, f, c), \quad f = S(x), \tag{7}$$

where $f = S(x)$ is the neural feature trilinearly interpolated from the volume S at the location x. In particular, x is parameterized in the reference view's NDC space and d is represented by a unit vector at the reference view's coordinate. Using NDC space can effectively normalize the scene scales across different data sources, contributing to the good generalizability of our method. In addition, inspired by NeRF [34], we also apply positional encoding on the position and direction vectors (x and d), which further enhance the high-frequency details in our results.

Radiance field. As a result, our entire framework models a neural radiance field, regressing volume density and view-dependent radiance in the scene from few (three) input images. In addition, once the scene encoding volume S is reconstructed, this volume combined with the MLP decoder A can be used independently without the prepending 2D and 3D CNNs. They can be seen as a standalone neural representation of the radiance field, outputting volume properties and thus supporting volume rendering.

3.3. Volume rendering and end-to-end training.

Our MVSNeRF reconstructs a neural encoding volume and regresses volume density and view-dependent radiance at arbitrary points in a scene. This enables applying differentiable volume rendering to regress images colors.

Volume rendering. The physically based volume rendering equation can be numerically evaluated via differentiable ray marching (as is in NeRF [34]) for view synthesis. In particular, a pixel's radiance value (color) is computed by marching a ray through the pixel and accumulating radiance at sampled shading points on the ray, given by:

$$\begin{aligned} c_t &= \sum_k \tau_k (1 - \exp(-\sigma_k)) r_k, \\ \tau_k &= \exp(-\sum_{j=1}^{k-1} \sigma_j), \end{aligned} \tag{8}$$

where c_t is the final pixel color output, and τ represents the volume transmittance. Our MVSNeRF as a radiance field function essentially provides the volume rendering properties σ_k and r_k for the ray marching.

End-to-end training. This ray marching rendering is fully differentiable; it thus allows our framework to regress final pixel colors at novel viewpoints using the three input views from end to end. We supervise our entire framework with the groundtruth pixel colors, using an L2 rendering loss:

$$L = \|c_t - \tilde{c}_t\|_2^2, \tag{9}$$

where $\tilde{c}_t$ is the groundtruth pixel color sampled from the target image I_t at a novel viewpoint. This is the only loss we use to supervise our entire system. Thanks to the physically based volume rendering and end-to-end training, the rendering supervision can propagate the scene appearance and correspondence information through every network components and regularize them to make sense for final view synthesis. Unlike previous NeRF works [34, 31, 27] that mainly focus on per-scene training, we train our entire network across different scenes on the DTU dataset. Our MVSNeRF benefits from the geometric-aware scene reasoning in cost volume processing and can effectively learn a generic function that can reconstruct radiance fields as neural encoding volumes on novel testing scenes enabling high-quality view synthesis.

3.4. Optimizing the neural encoding volume.

When training across scenes, our MVSNeRF can already learn a powerful generalizable function, reconstructing radiance fields across scenes from only three input images. However, because of the limited input and the high diversity across different scenes and datasets, it is highly challenging to achieve perfect results on different scenes using such a generic solution. On the other hand, NeRF avoids this hard generalization problem by performing per-scene optimization on dense input images; this leads to photo-realistic results but is extremely expensive. In contrast, we propose to fine-tune our neural encoding volume – that can be instantly reconstructed by our network from only few images – to achieve fast per-scene optimization when dense images are captured.

Appending colors. As mentioned, our neural encoding volume is combined with pixel colors when sent to the MLP decoder (Eqn. 7). Retaining this design for fine-tuning still works but leads to a reconstruction that always depends on the three inputs. We instead achieve an independent neural reconstruction by appending the per-view colors of voxel centers as additional channels to the encoding volume; these colors as features are also trainable in the per-scene optimization. This simple appending initially introduces blurriness in the rendering, which however is addressed very quickly in the fine-tuning process.

Optimization. After appended with colors, the neural encoding volume with the MLP is a decent initial radiance field that can already synthesize reasonable images. We propose to further fine-tune the voxel features along with the MLP decoder to perform fast per-scene optimization when dense images are available. Note that, we optimize only the encoding volume and the MLP, instead of our entire network. This grants more flexibility to the neural optimization to adjust the per-voxel local neural features independently upon optimization; this is an easier task than trying to optimize shared convolutional operations across voxels. In addition, this fine-tuning avoids the expensive

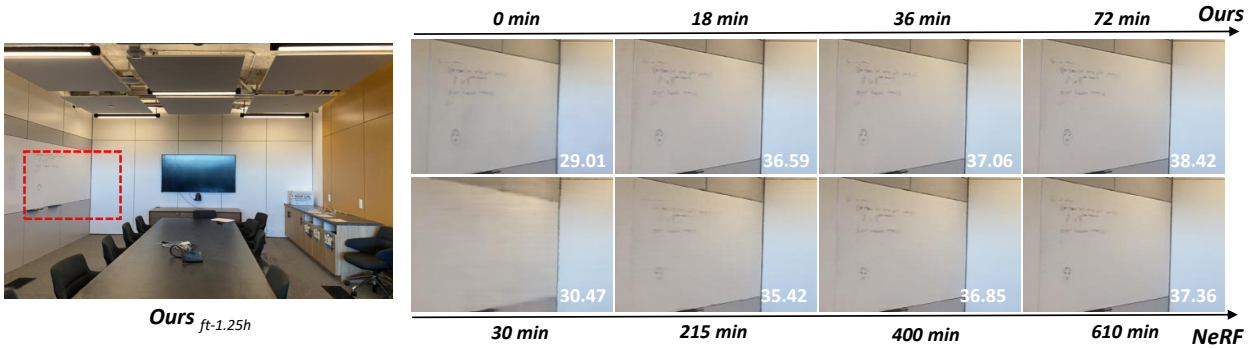

Figure 3. Optimization progress. We show results of our fine-tuning (top) and optimizing a NeRF [34] (bottom) with different time periods. Our 0-min result refers to the initial output from our network inference. Note that our 18-min results are already much better than the 215-min NeRF results. PSNRs of the image crops are shown in the figure.

network processing of the 2D CNN, plane-sweep warping, and 3D CNN. As a result, our optimization can therefore be very fast, taking substantially less time than optimizing a NeRF from scratch, as shown in Fig. 3.

Our per-scene optimization leads to a clean neural reconstruction, independent of any input image data (thanks to appending color channels), similar to [34, 27]; the dense input images can be therefore dropped after optimization. In contrast, the concurrent works [54, 46] require retaining the input images for rendering. Our encoding volume is also similar to Sparse Voxel fields [27]; however ours is initially predicted by our network via fast inference, instead of the pure per-scene optimization in [27]. On the other hand, we can (as future work) potentially subdivide our volume grid in the fine-tuning for better performance as is done in [27].

4. Implementation details

Dataset. We train our framework on the DTU [20] dataset to learn a generalizable network. We follow PixelNeRF [54] to partition the data to 88 training scenes and 16 testing scenes, and use an image resolution of 512×640. We also test our model (merely trained on DTU) on the Realistic Synthetic NeRF data [34] and the Forward-Facing data [33], which have different scene and view distributions from our training set. For each testing scene, we select 20 nearby views; we then select 3 center views as input, 13 as additional input for per-scene fine-tuning, and take the remaining 4 as testing views.

Network details. We use $f = 32$ channels for feature extraction, which is also the number of feature channels in the cost volume and neural encoding volume (before appending color channels). We adopt $D = 128$ depth hypotheses uniformly sampled from near to far to specify the plane sweep volume. Our MLP decoder is similar to the MLP of NeRF [34], but more compact, consisting of 6 layers. Unlike NeRF reconstructing two (coarse and fine) radiance fields as separate networks, we only reconstruct one single radiance field and can already achieve good results; an extension to coarse-to-fine radiance fields can be potentially achieved at fine-tuning, by optimizing two separate encoding volumes with the same initialization. For ray marching, we sample 128 shading points on each marching ray. We show detailed network structure in the supplementary materials.

We train our network using one *RTX 2080 Ti* GPU. For the across-scene training on DTU, we randomly sample 1024 pixels from one novel viewpoints as a batch, and use Adam [22] optimizer with an initial learning rate of $5e-4$.

5. Experiments

We now evaluate our method and show our results.

Comparisons on results with three-image input. We compare with two recent concurrent works, PixelNeRF[54] and IBRNet [46] that also aim to achieve the generalization of radiance field reconstruction. We use the released code and trained model of PixelNeRF and retrain IBRNet on the DTU data (see Sec. 4); we train and test these methods using 3 input views as used in our paper. We compare all methods on three datesets [34, 20, 33] with the same input views and use 4 additional images to test each scene. We show the quantitative results in Tab. 1 and visual comparisons in Fig. 4.

As shown in Fig. 4, our approach can achieve realistic view synthesis results using only three images as input across different datasets. While our model is trained only on DTU, it can generalize well to the other two datesets that have highly different scene and view distributions. In contrast, PixelNeRF [54] tends to overfit the training setting on DTU. Although it works reasonably on the DTU testing scenes, it contains obvious artifacts on the Realistic Synthetic scenes and even completely fails on the Forward-Facing scenes. IBRNet [46] can do a better job

Method	Settings	Synthetic Data (NeRF [31])			Real Data (DTU [20] / Forward-Facing [33])		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
PixelNeRF [54]	No per-scene optimization	7.39	0.658	0.411	19.31/11.24	0.789/0.486	0.382/0.671
IBRNet [46]		22.44	0.874	0.195	26.04/21.79	0.917/0.786	0.190/0.279
Ours		**23.62**	**0.897**	**0.176**	**26.63/21.93**	**0.931/0.795**	**0.168/0.252**
NeRF$_{10.2h}$ [34]	Per-scene optimization	**30.63**	**0.962**	**0.093**	27.01/**25.97**	0.902/0.870	0.263/ 0.236
IBRNet$_{ft-1.0h}$ [46]		25.62	0.939	0.110	**31.35**/24.88	**0.956**/0.861	**0.131/0.189**
Ours$_{ft-15min}$		27.07	0.931	0.168	28.50/25.45	0.933/**0.877**	0.179/0.192

Table 1. **Quantitative results of novel view synthesis.** We show averaged results of PSNRs, SSIMs and LPISs on three different datasets. On the top, we compare our method with concurrent neural rendering methods [54, 46] with direct network inference. On the bottom, we show our fine-tuning results with only 15min optimization ($10k$ iterations), IBRNet 1.0h optimization ($10k$ iterations) and compare with NeRF's [34] results with 10.2h optimization ($200k$ iterations).

Method	Abs err↓	Acc (0.01)↑	Acc (0.05)↑
MVSNet	**0.018**/ –	0.603/ –	**0.955**/ –
PixelNeRF	0.245/0.239	0.037/0.039	0.176/0.187
IBRNet	1.69/1.62	0.000/0.000	0.000/0.001
Ours	0.023/**0.035**	**0.746**/**0.717**	0.913/**0.866**

Table 2. **Depth reconstruction.** We evaluate our unsupervised depth reconstruction on the DTU testing set and compare with other two neural rendering methods (also without depth supervision) PixelNeRF [54] and IBRNet [46], and a learning based MVS method MVSNet [50] that is trained with groundtruth depth. Our method significantly outperforms other neural rendering methods (PixelNeRF and IBRNet) and achieve high depth accuracy comparable to MVSNet. The two numbers of each item refers to the depth at reference/novel views; we mark with "-" when one does not have a reference/novel view.

than PixelNeRF when testing on other datasets, but flicker artifacts can still be observed and much more obvious than ours as shown in the appendix video.

These visual results clearly reflect the quantitative results shown in Tab. 1. The three methods can all obtain reasonable PSNRs, SSIMs and LPIPs on the DTU testing set. However, our approach consistently outperforms PixelNeRF and IBRNet with the same input for all three metrics. More impressively, our results on the other two testing datasets are significantly better than the comparison methods, clearly demonstrating the good generalizability of our technique. In general, the two comparison methods both directly aggregate across-view 2D image features at ray marching points for radiance field inference. Our approach instead leverages MVS techniques for geometry-aware scene reasoning in plane-swept cost volumes, and reconstructs a localized radiance field representation as a neural encoding volume with explicit 3D structures. This leads to the best generalizablity and the highest rendering quality of our results across different testing scenes.

Per-scene fine-tuning results. We also show our per-scene optimization results using 16 additional input images in Tab. 1 and Fig. 4, generated by fine-tuning the neural encoding volume (with the MLP) predicted by our network (Sec. 3.4). Because of the strong initialization obtained from our network, we only fine-tune our neural reconstruction for a short period of 15 minutes (10k iterations), which can already lead to photo-realistic results. We compare our fast fine-tuning results with NeRF's [34] results generated with substantially longer optimization time (as long as 10.2 hours). Note that, our initial rendering results can be significantly boosted with even only 15min fine-tuning; this leads to high-quality results that are on par (Realistic Synthetic) or better (DTU and Forward-Facing) than NeRF's results with 30 times longer optimization time. We also show results on one example scene that compare the optimization progresses of our method and NeRF with different optimization times in Fig. 3, which clearly demonstrates the significantly faster convergence of our technique. By taking our generic network to achieve strong initial radiance field, our approach enables highly practical per-scene radiance field reconstruction when dense images are available.

Depth reconstruction. Our approach reconstructs a radiance field that represents scene geometry as volume density. We evaluate our geometry reconstruction quality by comparing depth reconstruction results, generated from the volume density by a weighted sum of the depth values of the sampled points on marched rays (as is done in [34]). We compare our approach with the two comparison radiance field methods [54, 46] and also the classic deep MVS method MVSNet [50] on the DTU testing set. Thanks to our cost-volume based reconstruction, our approach achieves significantly more accurate depth than the other neural rendering methods [54, 46]. Note that, although our network is trained with only rendering supervision and no depth supervision, our approach can achieve high reconstruction accuracy comparable to the MVS method [50] that has direct depth supervision. This demonstrates

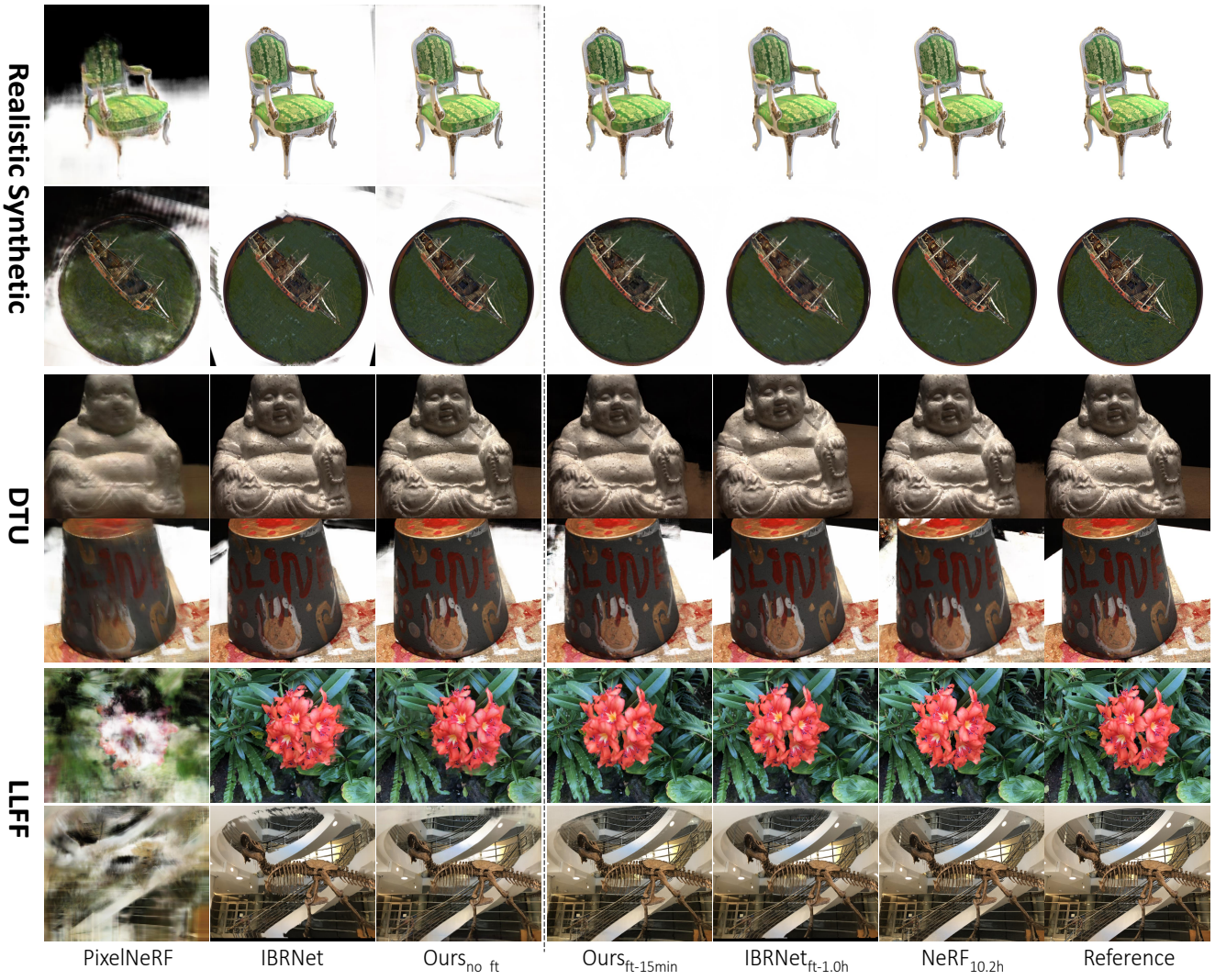

Figure 4. Rendering quality comparison. On the left, we show rendering results of our method and concurrent neural rendering methods [54, 46] by directly running the networks. We show our 15-min fine-tuning results and NeRF's [34] 10.2h-optimization results on the right.

the high quality of our geometry reconstruction, which is one critical factor that leads to our realistic rendering.

6. Conclusion

We present a novel generalizable approach for high-quality radiance field reconstruction and realistic neural rendering. Our approach combines the main advantages of deep MVS and neural rendering, successfully incorporating cost-volume based scene reasoning into physically based neural volumetric rendering. Our approach enables high-quality radiance field reconstruction from only three input views and can achieve realistic view synthesis results from the reconstruction. Our method generalizes well across diverse testing datasets and can significantly outperform concurrent works [54, 46] on generalizable radiance field reconstruction. Our neural reconstruction can also be fine-tined easily for per-scene optimization, when dense input images are available, allowing us to achieve photo-realistic renderings that are better than NeRF while using substantially less optimization time. Our work offers practical neural rendering techniques using either few or dense images as input.

7. Acknowledgements

This work was supported by NSFC programs (61976138, 61977047); the National Key Research and Development Program (2018YFB2100500); STCSM (2015F0203-000-06) and SHMEC (2019-01-07-00-01-E00003); NSF grant IIS-1764078 and gift money from VIVO.

References

[1] Sai Bi, Zexiang Xu, Pratul Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. *arXiv preprint arXiv:2008.03824*, 2020. 3

[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. 3

[3] 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. ACM, 2001. 2

[4] Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Transactions on Graphics (TOG)*, 32(3):30, 2013. 2

[5] Gaurav Chaurasia, Olga Sorkine, and George Drettakis. Silhouette-aware warping for image-based rendering. In *Computer Graphics Forum*, volume 30, pages 1223–1232. Wiley Online Library, 2011. 2

[6] Anpei Chen, Ruiyang Liu, Ling Xie, Zhang Chen, Hao Su, and Jingyi Yu. Sofgan: A portrait image generator with dynamic styling, 2021. 3

[7] Anpei Chen, Minye Wu, Yingliang Zhang, Nianyi Li, Jie Lu, Shenghua Gao, and Jingyi Yu. Deep surface light fields. *Proc. ACM Comput. Graph. Interact. Tech.*, 1(1):14:1–14:17, July 2018. 2

[8] Rui Chen, Songfang Han, Jing Xu, and Hao Su. Point-based multi-view stereo network. In *Proceedings of the ICCV*, pages 1538–1547, 2019. 2

[9] Zhang Chen, Anpei Chen, Guli Zhang, Chengyuan Wang, Yu Ji, Kiriakos N Kutulakos, and Jingyi Yu. A neural rendering framework for free-viewpoint relighting. In *Proceedings of the CVPR*, pages 5599–5610, 2020. 3

[10] Shuo Cheng, Zexiang Xu, Shilin Zhu, Zhuwen Li, Li Erran Li, Ravi Ramamoorthi, and Hao Su. Deep stereo using adaptive thin volume representation with uncertainty awareness. In *Proceedings of the CVPR*, pages 2524–2534, 2020. 2, 4

[11] Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H Kim, and Jan Kautz. Extreme view synthesis. In *Proceedings of the ICCV*, pages 7781–7790, 2019. 2

[12] Jeremy S De Bonet and Paul Viola. Poxels: Probabilistic voxelized volume reconstruction. In *Proceedings of ICCV*, pages 418–425, 1999. 2

[13] 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. ACM, 1996. 2

[14] Carlos Hernández Esteban and Francis Schmitt. Silhouette and stereo fusion for 3d object modeling. *Computer Vision and Image Understanding*, 96(3):367–392, 2004. 2

[15] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *Proceedings of the CVPR*, pages 5515–5524, 2016. 2

[16] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2010. 2

[17] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 43–54. ACM, 1996. 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 CVPR*, pages 2495–2504, 2020. 2, 4

[19] Sunghoon Im, Hae-Gon Jeon, Stephen Lin, and In-So Kweon. Dpsnet: End-to-end deep plane sweep stereo. In *7th International Conference on Learning Representations, ICLR 2019*. International Conference on Learning Representations, ICLR, 2019. 2

[20] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engil Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *2014 CVPR*, pages 406–413. IEEE, 2014. 1, 6, 7

[21] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM Transactions on Graphics (TOG)*, 35(6):193, 2016. 2

[22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

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

[24] Kiriakos N Kutulakos and Steven M Seitz. A theory of shape by space carving. *International journal of computer vision*, 38(3):199–218, 2000. 2

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

[26] 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 CVPR*, 2021. 3

[27] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *arXiv preprint arXiv:2007.11571*, 2020. 1, 3, 5, 6, 12

[28] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Transactions on Graphics*, 38(4):65, 2019. 3

[29] Haimin Luo, Anpei Chen, Qixuan Zhang, Bai Pang, Minye Wu, Lan Xu, and Jingyi Yu. Convolutional neural opacity radiance fields. In *IEEE International Conference on Computational Photography, ICCP 2021, Haifa, Israel, May 23-25, 2021*, pages 1–12. IEEE, 2021. 3

[30] Keyang Luo, Tao Guan, Lili Ju, Haipeng Huang, and Yawei Luo. P-mvsnet: Learning patch-wise matching confidence aggregation for multi-view stereo. In *Proceedings of the ICCV*, pages 10452–10461, 2019. 2

[31] 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. *arXiv preprint arXiv:2008.02268*, 2020. 1, 3, 5, 7

[32] 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. 3

[33] 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)*, 38(4):1–14, 2019. 2, 6, 7

[34] 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, 3, 5, 6, 7, 8, 11, 12

[35] Eric Penner and Li Zhang. Soft 3d reconstruction for view synthesis. *ACM Transactions on Graphics (TOG)*, 36(6):235, 2017. 2

[36] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the CVPR*, pages 10318–10327, 2021. 3

[37] Daniel Rebain, Wei Jiang, Soroosh Yazdani, Ke Li, Kwang Moo Yi, and Andrea Tagliasacchi. Derf: Decomposed radiance fields. *arXiv preprint arXiv:2011.12490*, 2020. 3

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

[39] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE computer society conference on CVPR*, volume 1, pages 519–528. IEEE, 2006. 2

[40] Sudipta Sinha, Drew Steedly, and Rick Szeliski. Piecewise planar stereo for image-based rendering. 2009. 2

[41] 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 *CVPR*, pages 175–184, 2019. 2

[42] Pratul P Srinivasan, Tongzhou Wang, Ashwin Sreelal, Ravi Ramamoorthi, and Ren Ng. Learning to synthesize a 4d rgbd light field from a single image. In *ICCV*, pages 2262–2270, 2017. 2

[43] Guoxing Sun, Xin Chen, Yizhang Chen, Anqi Pang, Pei Lin, Yuheng Jiang, Lan Xu, Jingya Wang, and Jingyi Yu. Neural free-viewpoint performance rendering under complexhuman-object interactions. In *Proceedings of the 29th ACM International Conference on Multimedia*, 2021. 3

[44] 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. 3

[45] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019. 3

[46] 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. In *CVPR*, 2021. 2, 3, 6, 7, 8, 12

[47] Daniel N Wood, Daniel I Azuma, Ken Aldinger, Brian Curless, Tom Duchamp, David H Salesin, and Werner Stuetzle. Surface light fields for 3d photography. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 287–296. ACM Press/Addison-Wesley Publishing Co., 2000. 2

[48] Fanbo Xiang, Zexiang Xu, Miloš Hašan, Yannick Hold-Geoffroy, Kalyan Sunkavalli, and Hao Su. NeuTex: Neural Texture Mapping for Volumetric Neural Rendering. In *The CVPR*, 2021. 3

[49] Zexiang Xu, Sai Bi, Kalyan Sunkavalli, Sunil Hadap, Hao Su, and Ravi Ramamoorthi. Deep view synthesis from sparse photometric images. *ACM Transactions on Graphics*, 38(4):76, 2019. 2

[50] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. MVSNet: Depth inference for unstructured multi-view stereo. In *ECCV*, pages 767–783, 2018. 2, 4, 7

[51] 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 CVPR*, pages 5525–5534, 2019. 2

[52] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *arXiv preprint arXiv:2106.12052*, 2021. 3

[53] 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. 3

[54] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *CVPR*, 2021. 2, 3, 6, 7, 8, 12

[55] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018. 2, 3

[56] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *European conference on computer vision*, pages 286–301. Springer, 2016. 2

A. Per-scene optimization.

More details. As described in the paper Sec. 3.4, we optimize the predicted neural encoding volume with the MLP decoder for each scene to achieve final high-quality fine-tuning results. The neural encoding volume with the MLP is an effective neural representation of a radiance field. All fine-tuning results (and also NeRF's optimization comparison results) are generated using a single NVIDIA RTX 2080Ti GPU. On this hardware, our 15min fine-tuning corresponds to 10k training iterations and NeRF's 10.2h optimization corresponds to 200k training iterations.

Our neural encoding volume is reconstructed in the frustrum of the reference view; it thus only covers the scene content in the frustrum. As a result, for a large scene, artifacts may appear when some parts that are not located in the frustrum show up in the novel view. Therefore, we extend the neural encoding volume by padding its boundary voxels when fine-tuning on some large scenes. This can address the out-of-frustrum artifacts, though the padding voxels are not well reconstructed initially by the network and may require longer fine-tuning to achieve high quality.

As described in the paper, we do not apply two-stage (coarse and fine) ray sampling as done in NeRF [34]. We uniformly sample points (with per-step small randomness) along each marching ray. We find that 128 points are enough for most scenes and keep using 128 point for our across-scene training on the DTU dataset. When fine-tuning, we increase the number of points to 256 for some challenging scenes.

Optimization progress. We have demonstrated in the paper that our 15min fine-tuning results of DTU and LLFF dataset are comparable with the 10.2h optimization results of NeRF [34]. We now show comparisons of the per-scene optimization progress between our fine-tuning and NeRF's from-scratch optimization. Note that, thanks to the strong initial reconstruction predicted by our network, our fine-tuning is consistently better than NeRF's optimization through 200k training iterations. As mentioned, our each 18min result corresponds to the result at the 12k-th training iteration, which is at the very early stage in the curves; however, as demonstrated, it can be already better than the NeRF's result after 48k iterations, corresponding to the 10.2h optimization result shown in the paper. Moreover, while our 15min results are already very good, our results can be further improved over more iterations, if continuing optimizing the radiance fields.

B. Network Architectures

We show detailed network architecture specifications of our 2D CNN (that extracts 2D image features), 3D CNN (that outputs a neural encoding volume), and MLP decoder (that regresses volume properties) in Tab 3.

Layer	**k**	**s**	**d**	**chns**	input
CBR2D_0	3	1	1	$3/8$	I
CBR2D_1	3	1	1	$8/8$	CBR2D_0
CBR2D_2	5	2	2	$8/16$	CBR2D_1
CBR2D_3	3	1	1	$16/16$	CBR2D_2
CBR2D_4	3	1	1	$16/16$	CBR2D_3
CBR2D_5	5	2	2	$16/32$	CBR2D_4
CBR2D_6	3	1	1	$32/32$	CBR2D_5
T	3	1	1	$32/32$	CBR2D_6
CBR3D_0	3	1	1	$32+9/8$	T, I
CBR3D_1	3	2	1	$8/16$	CBR3D_0
CBR3D_2	3	1	1	$16/16$	CBR3D_1
CBR3D_3	3	2	1	$16/32$	CBR3D_2
CBR3D_4	3	1	1	$32/32$	CBR3D_3
CBR3D_5	3	2	1	$32/64$	CBR3D_4
CBR3D_6	3	1	1	$64/64$	CBR3D_5
CTB3D_0	3	2	1	$64/32$	CTB3D_0 + CBR3D_4
CTB3D_1	3	2	1	32/16	CTB3D_1 + CBR3D_2
CTB3D_2	3	2	1	$16/8$	CTB3D_2 + CBR3D_0
PE_0	-	-	-	3/63	x
LR_0	-	-	-	8+12/256	f, c
LR_1	-	-	-	63/256	PE
LR_{i+1}	-	-	-	256/256	$\text{LR}_i \cdot \text{LR}_0$
σ	-	-	-	256/1	LR_6
PE_1	-	-	-	3/27	d
LR_7	-	-	-	27+256/256	PE_1, LR_6
c	-	-	-	256/3	LR_7

Table 3. From top to bottom: 2D CNN based feature extraction model, 3D CNN based neural encoding volume prediction model and MLP based volume properties regression model ($i \in [1, ..., 5]$). **k** is the kernel size, **s** is the stride, **d** is the kernel dilation, and **chns** shows the number of input and output channels for each layer. We denote CBR2D/CBR3D/CTB3D/LR to be ConvBnReLU2D, ConvBnReLU3D, ConvTransposeBn3D and LinearRelu layer structure respectively. PE refers to the positional encoding as used in [34].

C. Limitations.

Our approach generally achieves fast radiance field reconstruction for view synthesis on diverse real scenes. However, for highly challenging scenes with high glossiness/specularities, the strong view-dependent shading effects can be hard to directly recovered via network inference and a longer fine-tuning process can be required to fully reconstruct such effects. Our radiance field representation is reconstructed within the frustrum of the reference view. As a result, only the scene content seen by the reference view is well reconstructed and initialized for the following fine-tuning stage. Padding the volume (as discussed earlier) can incorporate content out of the original frustrum; however, the unseen parts (including those that are in the frustrucm but are occluded and invisible in the

DTU Dataset					
Scan	#1	#8	#21	#103	#114
PSNR↑					
PixelNeRF	21.64	23.70	16.04	16.76	18.40
IBRNet	25.97	**27.45**	20.94	27.91	27.91
Ours	**26.96**	27.43	**21.55**	**29.25**	**27.99**
$\text{NeRF}_{10.2h}$	26.62	28.33	23.24	30.40	26.47
IBRNet_{ft-1h}	**31.00**	**32.46**	**27.88**	**34.40**	**31.00**
$\text{Ours}_{ft-15min}$	28.05	28.88	24.87	32.23	28.47
SSIM↑					
PixelNeRF	0.827	0.829	0.691	0.836	0.763
IBRNet	0.918	0.903	0.873	0.950	0.943
Ours	**0.937**	**0.922**	**0.890**	**0.962**	**0.949**
$\text{NeRF}_{10.2h}$	0.902	0.876	0.874	0.944	0.913
IBRNet_{ft-1h}	**0.955**	0.945	**0.947**	0.968	**0.964**
$\text{Ours}_{ft-15min}$	**0.934**	0.900	**0.922**	**0.964**	0.945
LPIPS ↓					
PixelNeRF	0.373	0.384	0.407	0.376	0.372
IBRNet	0.190	0.252	0.179	0.195	0.136
Ours	**0.155**	**0.220**	**0.166**	**0.165**	**0.135**
$\text{NeRF}_{10.2h}$	0.265	0.321	0.246	0.256	0.225
IBRNet_{ft-1h}	**0.129**	**0.170**	**0.104**	**0.156**	**0.099**
$\text{Ours}_{ft-15min}$	0.171	0.261	0.142	0.170	0.153

Table 4. **Quantity comparison on five sample scenes in the DTU testing set**.

view) are not directly recovered by the network. Therefore, it is challenging to use a single neural encoding volume to achieve rendering in a wide viewing range around a scene (like 360° rendering). Note that, a long per-scene fine-tuning process with dense images covering around the scene can still achieve 360° rendering, though it can be as slow as training a standard NeRF [34] (or Sparse Voxel Fields [27] that is similar to our representation) to recover those uninitialized regions in the encoding volume. Combining multiple neural encoding volumes at multiple views can be an interesting future direction to achieve fast radiance field reconstruction with larger viewing ranges.

D. Per-scene breakdown.

We show the pre-scene breakdown analysis of the quantitative results presented in the main paper for the three dataset (*Realistic Synthetic*, *DTU* and *LLFF*).

These results are consistent with the averaged results shown in the paper. In general, since the training set consists of DTU scenes, all three methods can work reasonably well on the DTU testing set. Our approach can outperform PixelNeRF [54], when using the same three-image input, and achieve higher PSNR and SSIM and lower LPIPS. Note that, as mentioned in the paper, the implementation of IBRNet [46] is trained and tested with 10 input images to achieve its best performance as used in their paper. Nonetheless, our results with three input images are still quantitatively comparable to the results of IBRNet with 10 input images on the DTU testing set; IBRNet often achieves better PSNRs while we often achieve better SSIMs and LPIPSs.

More importantly, as already shown in paper, when testing on novel datasets, our approach generalizes significantly better than PixelNeRF and IBRNet, leading to much better quantitative results on the Synthetic Data and the Forward-Facing dataset. We also provide detailed per-scene quantitative results for the three testing datasets in Tab. 3-10. Please also refer to the supplementary video for video comparisons.

	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
				PSNR↑				
PixelNeRF	7.18	8.15	6.61	6.80	7.74	7.61	7.71	7.30
IBRNet	**24.20**	18.63	21.59	27.70	22.01	**20.91**	22.10	22.36
Ours	23.35	**20.71**	**21.98**	**28.44**	**23.18**	20.05	**22.62**	**23.35**
NeRF	**31.07**	**25.46**	**29.73**	**34.63**	**32.66**	**30.22**	**31.81**	29.49
IBRNet$_{ft-1h}$	28.18	21.93	25.01	31.48	25.34	24.27	27.29	21.48
Ours$_{ft-15min}$	26.80	22.48	26.24	32.65	26.62	25.28	29.78	26.73
				SSIM↑				
PixelNeRF	0.624	0.670	0.669	0.669	0.671	0.644	0.729	0.584
IBRNet	0.888	0.836	0.881	0.923	0.874	0.872	0.927	0.794
Ours	**0.876**	**0.886**	**0.898**	**0.962**	**0.902**	**0.893**	**0.923**	**0.886**
NeRF	**0.971**	**0.943**	**0.969**	**0.980**	**0.975**	**0.968**	**0.981**	**0.908**
IBRNet$_{ft-1h}$	0.955	0.913	0.940	0.978	0.940	0.937	0.974	0.877
Ours$_{ft-15min}$	0.934	0.898	0.944	0.971	0.924	0.927	0.970	0.879
				LPIPS ↓				
PixelNeRF	0.386	0.421	0.335	0.433	0.427	0.432	0.329	0.526
IBRNet	**0.144**	0.241	**0.159**	0.175	**0.202**	**0.164**	**0.103**	0.369
Ours	0.282	**0.187**	0.211	**0.173**	0.204	0.216	0.177	**0.244**
NeRF	**0.055**	**0.101**	**0.047**	**0.089**	**0.054**	0.105	**0.033**	0.263
IBRNet$_{ft-1h}$	0.079	0.133	0.082	0.093	0.105	**0.093**	0.040	**0.257**
Ours$_{ft-15min}$	0.129	0.197	0.171	0.094	0.176	0.167	0.117	0.294

Table 5. **Quantity comparison on the Realistic Synthetic dataset**.

	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	Trex
				PSNR↑				
PixelNeRF	12.40	10.00	14.07	11.07	9.85	9.62	11.75	10.55
IBRNet	20.83	22.38	**27.67**	22.06	**18.75**	15.29	**27.26**	20.06
Ours	**21.15**	**24.74**	26.03	**23.57**	17.51	**17.85**	26.95	**23.20**
NeRF$_{10.2h}$	**23.87**	26.84	**31.37**	25.96	21.21	19.81	**33.54**	**25.19**
IBRNet$_{ft-1h}$	22.64	26.55	30.34	25.01	**22.07**	19.01	31.05	22.34
Ours$_{ft-15min}$	23.10	**27.23**	30.43	**26.35**	21.54	**20.51**	30.12	24.32
				SSIM↑				
PixelNeRF	0.531	0.433	0.674	0.516	0.268	0.317	0.691	0.458
IBRNet	**0.710**	0.854	**0.894**	0.840	**0.705**	0.571	0.950	0.768
Ours	0.638	**0.888**	0.872	**0.868**	0.667	**0.657**	**0.951**	**0.868**
NeRF$_{10.2h}$	**0.828**	0.897	**0.945**	0.900	0.792	0.721	**0.978**	**0.899**
IBRNet$_{ft-1h}$	0.774	0.909	0.937	0.904	**0.843**	0.705	0.972	0.842
Ours$_{ft-15min}$	0.795	**0.912**	0.943	**0.917**	0.826	**0.732**	0.966	0.895
				LPIPS ↓				
	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	Trex
PixelNeRF	0.650	0.708	0.608	0.705	0.695	0.721	0.611	0.667
IBRNet	0.349	0.224	**0.196**	0.285	**0.292**	0.413	**0.161**	0.314
Ours	**0.238**	**0.196**	0.208	**0.237**	0.313	**0.274**	0.172	**0.184**
NeRF$_{10.2h}$	0.291	0.176	0.147	0.247	0.301	0.321	0.157	0.245
IBRNet$_{ft-1h}$	0.266	0.146	**0.133**	0.190	**0.180**	0.286	**0.089**	0.222
Ours$_{ft-15min}$	**0.253**	**0.143**	0.134	**0.188**	0.222	**0.258**	0.149	**0.187**

Table 6. **Quantity comparison on the Forward Facing dataset**.