NeRFusion: Fusing Radiance Fields for Large-Scale Scene Reconstruction

Xiaoshuai Zhang[1*] Sai Bi[2] Kalyan Sunkavalli[2] Hao Su[1] Zexiang Xu[2]
[1] University of California, San Diego [2] Adobe Research
{xiz040,haosu}@eng.ucsd.edu {sbi,sunkaval,zexu}@adobe.com

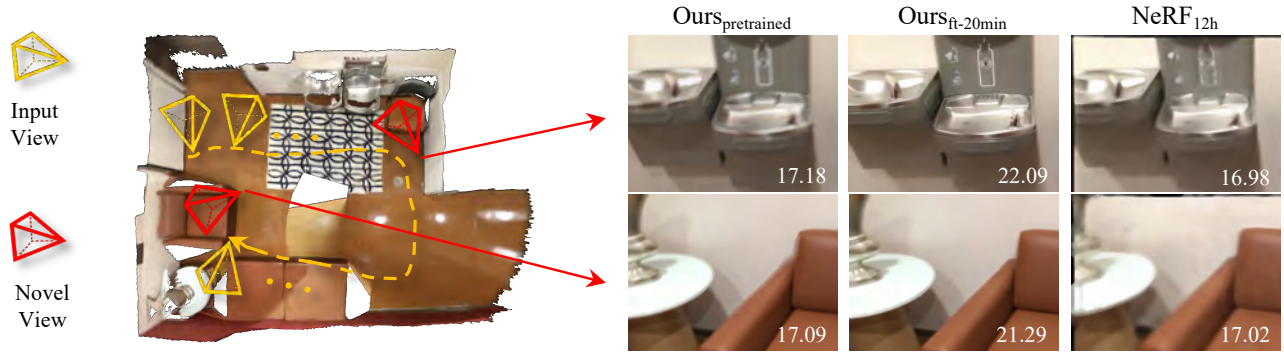

Figure 1. We propose a novel method to enable *fast* reconstruction of volumetric radiance fields of *large-scale* scenes. Our method uses a novel recurrent network – that is *generalizable* and trained across scenes – to sequentially reconstruct a radiance field of a large indoor scene in ScanNet [11] from an input image sequence (marked in yellow) via direct inference. The predicted field can be directly used to render realistic images at novel viewpoints (marked in red), achieving comparable quality to NeRF [28] that takes 12 hours per-scene optimization. This radiance field can be further fine-tuned for a short period of 20 min, leading to boosted quality that significantly outperforms NeRF.

Abstract

While NeRF [28] has shown great success for neural reconstruction and rendering, its limited MLP capacity and long per-scene optimization times make it challenging to model large-scale indoor scenes. In contrast, classical 3D reconstruction methods can handle large-scale scenes but do not produce realistic renderings. We propose NeRFusion, a method that combines the advantages of NeRF and TSDF-based fusion techniques to achieve efficient large-scale reconstruction and photo-realistic rendering. We process the input image sequence to predict per-frame local radiance fields via direct network inference. These are then fused using a novel recurrent neural network that incrementally reconstructs a global, sparse scene representation in real-time at 22 fps. This global volume can be further fine-tuned to boost rendering quality. We demonstrate that NeRFusionachieves state-of-the-art quality on both large-scale indoor and small-scale object scenes, with substantially faster reconstruction than NeRF and other recent methods.[1]

*Research partially done during Xiaoshuai's internship at Adobe Research

[1] https://jetd1.github.io/NeRFusion-Web/

1. Introduction

Reconstructing and rendering large-scale indoor scenes from RGB images is challenging but crucial for various applications in computer vision and graphics, including AR/VR, e-commerce, and robotics. While truncated signed distance function (TSDF) fusion techniques [29, 44] can achieve efficient reconstruction, these methods often use depth sensors and focus on geometric reconstruction only, and cannot synthesize realistic images. Recently, NeRF [28] proposed optimizing scene radiance fields, represented using global MLPs, from RGB images to achieve photo-realistic novel view synthesis. However, NeRF cannot handle large-scale scenes well due to its limited MLP network capacity and impractical slow per-scene optimization.

In this work, we aim to achieve *fast, large-scale* scene-level radiance field reconstruction to make neural scene reconstruction and rendering more practical. As opposed to small-scale object-centric scenes, we use "large-scale" to refer to full-size indoor scenes, like ScanNet scenes [11]), with multiple rooms and objects with complex scene geometry and appearance. To achieve fast radiance field reconstruction on such challenging scenes, we propose a novel neural framework that uses recurrent neural modules to in-

crementally reconstruct a large sparse radiance field from a long RGB image sequence. Unlike NeRF [28], that requires per-scene optimization, our network is generalizable, pre-trained across scenes, and able to efficiently reconstruct large-scale radiance fields via direct network inference. As shown in Fig. 1, our framework can successfully reconstruct a large indoor scene from from an input monocular RGB video from ScanNet [11], to create a high-quality radiance field with photo-realistic novel view synthesis results.

Our reconstructed radiance field is represented by a sparse volume grid with per-voxel neural features; these voxel features are tri-linearly interpolated at any scene location, and used to regress volume density and view-dependent radiance through an MLP decoder for differentiable volume rendering. In contrast to previous methods [14, 24] that reconstruct similar representations using slow per-scene optimization, we present a novel deep neural network that can be trained across scenes and generalize on unseen novel scenes to achieve fast radiance field reconstruction, bypassing per-scene fitting.

Given an input sequence of RGB images with known camera poses (that can be registered by SLAM or SfM techniques), our framework reconstructs a radiance field as a sparse neural volume. Our pipeline is inspired by the classical TSDF fusion workflow [29, 30, 44] that starts from per-view geometry (depth) and fuses the per-view reconstruction across key frames to obtain a global sparse TSDF volume. This workflow is widely used to reconstruct large-scale scenes, but only focuses on geometric reconstruction. Instead, we propose novel neural modules to reconstruct radiance fields as sparse voxels for photo-realistic rendering.

We first reconstruct local radiance fields for each input key frame. We leverage deep MVS techniques and apply sparse 3D convolutions on a world-space cost-volume built from unprojected 2D images features (regressed from a deep 2D CNN) of neighboring key frames. This reconstructs sparse neural voxels that represent a local radiance field. Once estimated, this field can already be used to render realistic images locally, though only for partial scene content seen by the local frames. We propose a recurrent neural fusion module to sequentially fuse multiple local fields across frames. Our fusion module recurrently takes a newly estimated local field as input and learns to incorporate the local voxels to progressively reconstruct a global radiance field modeling the entire scene, by adding new voxels and updating existing voxels. Our full model is trained from end to end, learning to reconstruct radiance fields with arbitrary scene scales from an arbitrary number of input images. We show that our direct network output can already render high-quality images; moreover, our neural field can be effectively fine-tuned by optimizing the predicted voxel features per scene in a short period to achieve better rendering quality (see Fig. 1 and 4).

We train our full framework from end to end with only rendering losses on a combination of scenes from the ScanNet [11], DTU [16], and Google Scanned Object [34] datasets. These datasets contain a large variety of different objects and scenes, allowing for our method to work properly with any scene scale. We demonstrate, on various datasets, that our approach performs better than prior arts, including IBRNet [43] that also designs networks that generalize across scenes. Especially on large-scale indoor scenes, our results from real-time direct network inference can even be on par with NeRF's results from long per-scene optimization. Moreover, after only one hour of per-scene fine-tuning, our quality can be further boosted to the state-of-the-art, outperforming NeRF [27] and NVSF [24] that require much longer per-scene optimization times. Our approach significantly improves the efficiency and scalability of radiance field reconstruction. We believe this is an important step towards making neural scene reconstruction and rendering practical.

2. Related Work

Multi-view scene reconstruction. Abundant research has been conducted on reproducing the appearance of of 3D scens from multi-view data. To reconstruct the geometry, previous methods apply multi-view stereo [37, 38] or depth sensors [29] to acquire the depth information of the scene. Recently, learning-based multi-view stereo methods [8, 9, 12, 15, 26, 47, 48] based on plane-swept cost volumes are also introduced for depth estimation. Given the depth, one category of methods represent scenes with colored point clouds [1, 21, 22], and utilize point splatting to render images of the scene. Another category of methods [29, 30, 44] fuse multi-view depth and reconstruct surface meshes using techniques such as TSDF fusion or Poisson reconstruction, and further generate textures [2, 51] from multi-view images. However, both kinds of methods are sensitive to potential inaccuracies in point clouds and meshes resulting from corrupted depth, especially when there are thin structures and textureless regions, thus suffering from holes and blurry artifacts in the final renderings. While some works apply neural networks [1, 13, 35] such as a 2D CNN in screen-space to mitigate potential errors in the geometry, their models are per-scene optimized for a specific scene (similar to NeRF), requiring a long optimization time. Moreover, the screen-space neural networks typically produce temporally unstable results with flickering artifacts. Instead of estimating and fusing per-view depth, previous methods [6, 19, 40] introduce learning-based methods to aggregate per-view features and predict opacity volumes or signed distance volumes. These methods only focus on geometry reconstruction and cannot produce realistic renderings. In contrast, our approach models scenes as neural volumetric radiance fields and can reproduce the faithful

scene appearance, producing photo-realistic novel views. Our pipeline is pretrained on multi-view image datasets and can generalize to novel scenes at arbitrary scales and enable efficient large-scale neural reconstruction.

Neural Radiance Fields. Volumetric representations [24, 25, 28] have been widely adopted to reconstruct the appearance of the scene. NeRF [28] uses a global MLP to regress the volume density and view-dependent radiance at any arbitrary point in the space, and applies volume rendering to synthesize images at novel viewpoints. Following works extend the framework for different tasks such as relighting [3–5, 39], scene editing [46] and dynamic scene modeling [23, 31, 32]. Similar to NeRF, most of these works train MLP networks, specific for each scene from scratch, which can take hours and even days to optimize, heavily time-consuming. On the other hand, the limited network capacity of MLPs makes these methods hard to scale up to large scene reconstruction. NVSF [24] improves the scalability by building sparse voxel grids with per-voxel features. However their networks and features are still optimized per scene from scratch and it can still take days for large-scale scenes. In contrast, while we use a similar sparse volume, our volume is generated from the direct inference of a pre-trained network, leading to fast large-scale scene reconstruction. The direct network output can be further fine-tuned in a short period to achieve better rendering quality than NeRF and NSVF.

Some previous papers also extend NeRF for generalization. PixelNeRF [49] uses 2D CNNs to extract image features on each sampled point of each ray used in ray marching for regressing the point's volume properties. However, their network is designed for object rendering with few images and is trained specifically for each dataset. IBRNet [43] uses a similar network but has better designs that enable rendering on any scene scales. However, it leverages image features from neighboring views as input, varying across novel viewpoints, which often lead to blurry or flickering artifacts from sparse inputs. Our network instead reconstructs a neural volume with per-voxel features in 3D, modeling scene geometry and appearance in a more consistent way, leading to much better rendering than IBRNet. MVSNeRF [7] also reconstructs 3D volumes; however, it focuses on reconstructing a local volume from a fixed number of three nearby views. In contrast, our network can fuse per-view local reconstruction into a global volume from an arbitrary number of images, leading to highly efficient large-scale scene reconstruction and rendering.

3. Method

We now present our approach for neural scene reconstruction and rendering. Given an input sequence of images I_1,...,I_N of a large-scale scene with their known camera parameters Φ_1,...,Φ_N, our approach reconstructs a radiance field modeling the entire scene for realistic rendering.

Our final output radiance field is represented by a sparse neural volume $\mathcal{V}^g$ with per-voxel neural features (Sec. 3.1). Unlike per-scene optimization methods [24,28], we propose a deep neural network that sequentially takes the images I_t frame by frame as input and convert the sequence to the final sparse reconstruction via direct network inference. Our pipeline first learns to reconstruct a sparse volume $\mathcal{V}_t$ per input image frame, expressing a local radiance field covered by local frames (Sec. 3.2). We then leverage a recurrent fusion module that learns to fuse the per-frame volumes $\mathcal{V}_t$ online, incrementally reconstructing the global large-scale field $\mathcal{V}^g$ (Sec. 3.3). We train our full pipeline from end to end with pure rendering losses. Our model can reconstruct high-quality radiance fields from direct network inference; the estimated field can also be further fine-tuned to boost its quality (Sec. 3.4).

3.1. Sparse Volumes for Radiance Fields

Our output radiance field is modeled by a sparse neural volume $\mathcal{V}$ that has per-voxel neural features in voxels that approximately cover the actual scene surface. We regress volume density σ and view-dependent radiance c at any given 3D location x from this volume using an MLP network, in which we first tri-linearly sample a feature vector and then use the MLP to convert the feature to volume properties, expressed by

$$\sigma, c = R(x, d, \mathcal{V}(x)). \tag{1}$$

Here $\mathcal{V}(x)$ represents the trilinearly interpolated feature at x, R is the MLP, and d is the viewing direction in rendering. The output volume properties regressed from the volume can be directly used to synthesize images at novel target viewpoints via differentiable ray marching as is done in NeRF [28]. This radiance field representation is similar to the one in Neural Sparse Voxel Fields [24] that purely relies on per-scene optimization for the reconstruction. We instead propose to leverage neural networks trained across scenes to predict the neural volumes from image sequences.

In our pipeline, such sparse volumes are reconstructed locally as $\mathcal{V}_t$ per frame t and also globally as $\mathcal{V}_g$ for the entire sequence. The MLP network R is shared across all volumes in the training process. We model the local volumes and the global volume both in the canonical world space.

3.2. Reconstructing Local Volumes

We propose a deep neural network to regress a local neural volume for each input frame t, using its image I_t and $K-1$ images from neighboring views. Usually, given a monocular video, these neighboring views correspond to temporal neighboring frames. Using multiple nearby images for per-frame reconstruction allows the network to

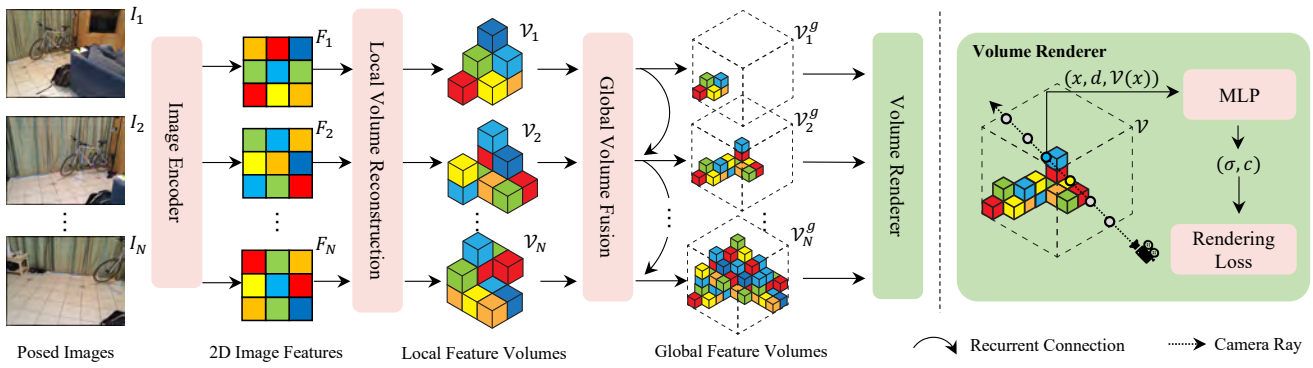

Figure 2. Overview of our framework. Given a sequence of images, 1) we first extract their image features ($F_1 \dots F_N$) using a 2D CNN. 2) Then, at each frame, we reconstruct a local sparse neural volume $\mathcal{V}_1 \dots \mathcal{V}_N$ in the canonical world space by fetching and aggregating 2D features across its neighboring views at visible voxels using a sparse 3D CNN. 3) We further fuse the local sparse volumes across frames using a recurrent neural network and sequentially build global feature volumes $\mathcal{V}_1^g \dots \mathcal{V}_N^g$ to model a radiance field of the entire scene. We regress volume density and view-dependent radiance from the sparse neural volumes to render images with differentiable ray marching.

leverage multi-view correspondence to recover better scene geometry, which a single image cannot provide.

To make the local reconstruction per frame well generalized across scenes, we leverage deep MVS techniques [18, 47], which are known to be generalizable. We extract 2D image features, build a cost volume from the features, and regress a neural feature volume from the cost volume. However, unlike MVSNeRF [7] and other MVS techniques [9, 47] that built frustum volumes in view's perspective coordinate, we construct volumes in the canonical world coordinate frame to align it with the final global volume output $\mathcal{V}_g$, facilitating the following fusion process.

Image feature extraction. We use a deep 2D convolutional neural network to extract 2D image features for each input image. This network maps the input image I_t into a 2D feature map F_t, encoding the scene content from each view.

Local sparse volume. We consider the bounding box that covers the frustums of all K neighboring viewpoints in the world coordinate frame, containing of a set of voxels in the canonical space. The bounding volume is axis-aligned with the world frame; each voxel inside it can be visible to a different number of neighboring views. We mask out all the voxels invisible to all view, leading to a sparse set of voxels in the bounding box. We then unproject the image features into this volume for our local reconstruction.

3D feature volume. For each neighboring viewpoint i and its feature map F_i, we build a 3D feature volume $\mathcal{U}_i$. In particular, for each visible voxel centered at v, we fetch the 2D image feature at its 2D projection from each neighboring view at frame t. In addition to pure image features, we leverage the corresponding viewing direction d_i at v from each viewpoint and compute additional features using an MLP G. The per-view 3D volume $\mathcal{U}_i$ is expressed by

$$\mathcal{U}_i(v) = [F_i(u_i), G(d_i)], \tag{2}$$

where $\mathcal{U}_i(v)$ is the feature at a voxel centered at v, u_i is the center's 2D projection in view i, $[\cdot, \cdot]$ represents feature concatenation. Note that, we encode the additional information of input viewing directions in the reconstruction process; this crucial information makes our following fusion module effectively account for the view-dependent effects captured across frames.

Neural reconstruction. We then aggregate the features across multiple neighboring viewpoints to regress a local volume $\mathcal{V}_t$ at frame t, expressing a local radiance field. We propose to leverage the mean and variance of the per-voxel features in $\mathcal{U}_i$ computed across neighboring viewpoints; such operations have been widely used in building cost volumes in MVS-based techniques [7, 47], where the mean can fuse per-view appearance information and the variance provides rich correspondence cues for geometry reasoning. These two operation are also invariant to the number/order of input; in our case, this naturally handles voxels that have different numbers of visible viewpoints. We use a deep neural network J to process the mean and variance features per voxel to regress the per-view reconstruction by

$$\mathcal{V}_t = J([\text{Mean}_{i \in \mathcal{N}_t} \mathcal{U}_i,\ \text{Var}_{i \in \mathcal{N}_t} \mathcal{U}_i]), \tag{3}$$

Here $\mathcal{N}_t$ represents all K neighboring viewpoints used at frame t; Mean and Var represent element-wise average and variance operation, respectively.

Essentially, we regress the local radiance field from the features across neighboring views. This is similar to MVSNeRF [7]. However, unlike MVSNeRF that considers only local reconstruction and builds perspective frustrum volumes for small-baseline rendering, we leverage these local volumes for global large-scale reconstruction and rendering. We build volumes directly in canonical space, naturally providing per-frame voxel inputs for our fusion module.

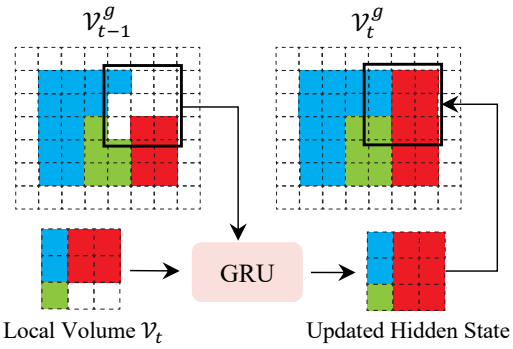

Figure 3. A 2D example illustrating the GRU fusion step. The hidden state, which is also the global feature volume $\mathcal{V}^g_{t-1}$, is adaptively updated by aggregating new information in the incoming local feature volume $\mathcal{V}_t$.

3.3. Fusing Volumes for Global Reconstruction

In order to create a consistent, efficient, and extensible scene reconstruction, we propose to use a global neural volume fusion network to incrementally fuse local feature volumes $\{\mathcal{V}_t\}$ per frame into a global volume $\mathcal{V}^g$.

Fusion. At each frame t, we consider its local sparse volume reconstruction $\mathcal{V}_t$ and the global reconstruction $\mathcal{V}^g_{t-1}$ from the previous frame as recurrent input. We leverage GRUs (Gated Recurrent Unit) [10] with sparse 3D CNNs in our fusion module, allowing our network to learn to recurrently fuse the per-frame local reconstruction and output high-quality global radiance fields. This is expressed by

$$z_t = M_z([\mathcal{V}^g_{t-1}, \mathcal{V}_t]), \tag{4}$$
$$r_t = M_r([\mathcal{V}^g_{t-1}, \mathcal{V}_t]), \tag{5}$$
$$\tilde{\mathcal{V}}^g_t = M_t([r_t * \mathcal{V}^g_{t-1}, \mathcal{V}_t]), \tag{6}$$
$$\mathcal{V}^g_t = (1 - z_t) * \mathcal{V}^g_{t-1} + z_t * \tilde{\mathcal{V}}^g_t, \tag{7}$$

where $*$ is the element-wise multiplication, z_t and r_t are the update gate and the reset gate, M_z, M_r and M_t all deep neural networks with sparse 3D convolution layers. As in standard GRU, M_z and M_r are designed with sigmoid activation in the end, while M_t uses tanh, allowing for the entire model sequentially updating the global reconstruction $\mathcal{V}^g_t$ (seen as the hidden state in a GRU) for every input frame. In this process, we only apply the networks on the voxels covered by the local volume $\mathcal{V}_t$; all other voxels in the global volume are kept unchanged. A 2D illustration of this GRU fusion process is shown in Fig. 3.

Intuitively, the update gate z_t and reset gate r_t in the GRU determine how much information from the previous global volume $\mathcal{V}^g_{t-1}$ as well as how much information from the current local volume $\mathcal{V}_t$ should be incorporated into the new global features. In this way, our module can adaptively improve the global scene reconstruction by filling up holes and refining features while keeping the representation consistent. This fusion process is similar to previous 3D reconstruction pipelines [20, 33, 40] that focus on geometry reconstruction; in contrast, we instead reconstruct neural feature volumes to represent neural radiance fields for volume rendering, leading to photo-realistic novel view synthesis.

Voxel pruning. To maximize the memory and rendering efficiency, we adaptively prune the global volume reconstruction $\mathcal{V}^g_t$ for every frame by removing the non-essential voxels that do not have any scene content inside. We naturally leverage the volume density in each voxel regressed by our radiance field (Eqn. 1), which models the scene geometry. In particular, we prune voxels V if:

$$\min_{i=1\ldots k} \exp(-\sigma(v_i)) > \gamma, v_i \in V, \tag{8}$$

where $\{v_i\}_{i=1}^k$ are k uniformly sampled points inside the voxel V, $\sigma(v_i)$ is the predicted density at location v_i, and γ is a pruning threshold. This pruning step is performed in both later training phase and inference phase once we get a global feature volume $\mathcal{V}^g_t$. By doing so, we make our global volume sparser, leading to more efficient reconstruction and rendering.

3.4. Training and optimization

Once a radiance field (that is represented by a sparse neural volume as described in Sec. 3.1) is reconstructed, our final rendering is achieved via differentiable ray marching using the regressed volume density and view-dependent radiance at any sampled ray points, as is done in NeRF and any other radiance field methods [24, 28]. In this work, our full pipeline is trained, completely depending on the rendering supervision with the ground truth images, without any extra geometry supervision.

In particular, we first train our local reconstruction network and the radiance field decoder (R) with a loss

$$\mathcal{L}_{\text{local}} = \|C_t - \hat{C}\|_2^2, \tag{9}$$

where $\hat{C}$ is the ground truth pixel color and C_t represents the rendered pixel color using the local volume $\mathcal{V}_t$ reconstructed at frame t. This makes the network learn to predict reasonable local neural volumes, which are already renderable and able to produce realistic images locally; it also initializes the radiance field decoder MLP to a reasonable state, which is later shared across local and global volumes. This pre-training allows the local reconstruction module to provide meaningful volume features for the fusion module to utilize in the end-to-end training, effectively facilitating the fusion task. We then train our full pipeline with the local reconstruction network, fusion network, and the radiance field decoder network all together from end to end,

using a rendering loss:

$$\mathcal{L}_{\text{fuse}} = \sum_t \|C_t - \hat{C}\|_2^2 + \|C_t^g - \hat{C}\|_2^2, \qquad (10)$$

where C_t is the pixel color rendered from the local reconstruction $\mathcal{V}_t$ (as is in Eqn. 9) and C_t^g is the color rendered from the global volume $\mathcal{V}_t^g$ after fusing frame t. Basically, we take every intermediate global and local volume ($\mathcal{V}_t$ and $\mathcal{V}_t^g$) at every frame to render novel view images and supervise them with the ground truth. The fusion module thus reasonably learns to fuse local volumes from an arbitrary number of input frames.

After trained, our full network is able to output a high-quality radiance field from direct network inference and produce realistic rendering results (as shown in Fig. 1). In addition, the reconstructed radiance field as a sparse neural volume can also be easily optimized (fine-tuned) per scene further to boost the rendering quality.

Fine-tuning. To fine-tune the estimated radiance field, we optimize the per-voxel neural features in the sparse volume reconstruction $\mathcal{V}^g$ and the MLP decoder per scene with the captured images, leading to better rendering results. Since our initial reconstruction is already very good, a short period of optimization with less than 25k iterations can usually lead to very high quality, which takes less than 1 hour. This is substantially less optimization time than NeRF and other pure per-scene optimization methods.

In this per-scene optimization stage, we also do a coarse-to-fine reconstruction, similar to NSVF [24]. Basically, after every 10k optimization iterations, we further prune unnecessary voxels (using Eqn. 8) and also subdivide each voxel into 8 sub voxels. This prune and subdivision step progressively increases the spatial resolution of the neural volume, further improving our final rendering quality.

4. Implementation Details.

Training datasets. Our training data consists of both large-scale indoor scenes from ScanNet [11] and small objects from DTU [17] and Google Scanned Objects [34]. We randomly sample 100 scenes from ScanNet for training. For DTU, We adopt the training split from PixelNeRF [49], which includes 88 training scenes. In addition, we use the object-centric synthetic renderings of 1,023 models from the Google Scanned Objects [34] generated by [43]. Our training data includes various camera setups and scene types, enabling our model to generalize to all kinds of scenarios. We demonstrate that our model can effectively work on large-scale indoor scenes, as well as scenes of objects.

Training details. For each input image sequence, we uniformly sample key frames from the full sequences for the input frames in network training. For object-centric datasets, we sample 16 views for each scene, and for ScanNet, we sample $2\% - 5\%$ of the full sequence as key frames. All other frames are used for supervision. We use $K = 3$ neighboring views for each input frame, for local volume reconstruction. For video sequence captured by a monocular camera, such as the scenes in ScanNet, we directly take the 3 neighboring key frames temporally. For other datasets, we select the 3 spatially closest viewpoints, in terms of both viewing location and direction.

The sparse volumes and networks are implemented with torchsparse [42]. We train our model using Adam optimizer with an initial learning rate of 0.003. We train our network with 2 NVIDIA 2080Ti GPUs for 3 days. During inference, our network processes frames from ScanNet sequences in real-time at 22 FPS. The final model takes 38 seconds on average to render a 640×480 image on ScanNet .

5. Results

In this section, we evaluate the our model on various datasets. For all results, we denote our results from direct network inference as Ours and our results after per-scene fine-tuning as Ours$_{\text{ft}}$ in all figure and tables. Similar labels are applied to IBRNet and other generalizing methods.

Baselines. We compare our method against the state-of-the-art NeRF methods on novel view synthesis including per-scene optimization methods, such as NeRF [28], NVSF [24], and NerfingMVS [45], and methods that can generalize to new scenes, such as PixelNeRF [49], IBRNet [43], and MVSNeRF [7].

To achieve fair and accurate comparisons, we run our method on the same experiment settings in previous papers, and we try our best to directly use the reported official quantitative results in previous papers or use the official code to run the experiments. We find that the official NeRF and IBRNet code can easily run and work on different datasets, producing corresponding images. We demonstrate visual comparison with these two methods across the three testing sets in Fig. 4. On the other hand, NSVF is very hard to run without enough GPU memory; their official models are optimized on a V100 GPU that has 32G memory; we found it impractical to generate their corresponding results with our resources. We therefore only include NSVF's quantitative results whenever they are reported previously. Besides, the recent MVSNeRF [7] is a very relevant technique, but it is designed to take a fixed number of three nearby views as its network input; as a result, it cannot support large-baseline rendering or arbitrary number of input images for inference. We therefore only compare with MVSNeRF on the DTU dataset in the same experiment setting used in their paper.

Large-scale scenes in ScanNet. We follow the same training and evaluation scheme as described in NerfingMVS [45] for the comparison on ScanNet. We tested our model on the 8 testing scenes used in their paper. From Table 1, we can see that our recurrent neural reconstruction network gener-

Method	Settings	PSNR↑	SSIM↑	LPIPS↓
IBRNet	No per-scene optimization	21.19	0.786	0.358
Ours		**22.99**	**0.838**	**0.335**
NeRF	Per-scene optimization	24.04	0.860	0.334
NSVF		26.01	0.881	-
NeRFingMVS		26.37	0.903	0.245
$\text{IBRNet}_{\text{ft-1.5h}}$		25.14	0.871	0.266
$\text{Ours}_{\text{ft-1h}}$		**26.49**	**0.915**	**0.209**

Table 1. Quantitative comparisons on the ScanNet dataset [11]. We follow the same experiment settings as in NeRFingMVS [45] and report the error metrics including PSNR (higher is better), SSIM (higher is better) and LPIPS (lower is better). Note that, our diret inference results are better than IBRNet [43]. Our fine-tuning results achieve the best numbers in all three metrics.

Method	Settings	PSNR↑	SSIM↑	LPIPS↓
IBRNet	No per-scene optimization	**25.51**	0.916	0.100
Ours		**25.47**	**0.922**	**0.093**
NeRF	Per-scene optimization	31.01	0.947	0.081
NSVF		**31.75**	**0.954**	**0.048**
$\text{IBRNet}_{\text{ft-1.5h}}$		28.19	0.943	0.072
$\text{Ours}_{\text{ft-1h}}$		31.25	**0.953**	0.069

Table 2. Quantitative comparison on the NeRF Synthetic dataset [28]. Our model is able to generate better results than IBRNet in both direct inference and fine-tuning settings. Our model after 1 hour fine-tuning achieves comparable performance to the state-of-the-art per-scene overfitting methods such as NeRF [28] and NVSF [24].

ates significantly better results than IBRNet via direct network inference. After fine-tuning for only a short period of 1 hour, the quality of our results is further boosted significantly, leading to the best PSNR, SSIM and LIPIPS in all compared methods. Note that, the per-scene optimization methods like NeRF, NeRFing MVS and NSVF require substantially longer per-scene optimization time but are still outperformed by our method. Our approach is impressively better than NSVF in this case though both methods have similar final radiance field representation; this indicates that the data priors learned by our recurrent neural network can effectively help the reconstruction and lead to reasonable initial radiance fields, even benefiting the per-scene fine-tuning process.

As shown in Fig. 4, our results on these large-scale scenes are of very high visual quality. Our results are visually much better than the IBRNet's results from both direct inference and per-scene fine-tuning. IBRNet generates tearing artifacts since it performs image-based rendering and can only aggregate a small set of local neighboring views due to limited GPU memory. In contrast, our model learns a unified 3D representation in the canonical space with a recurrent module that is able to efficiently aggregate per-view information across all input views, leading to significantly better rendering quality with better across-view consistency. Note that, even our direct inference renderings are already very realistic and contain few noticeable artifacts; they are arguably comparable to the rendering results of NeRF which require long per-scene optimization. Our approach achieves highly efficient and highly accurate large-scale radiance field reconstruction.

Method	Settings	PSNR↑	SSIM↑	LPIPS↓
PixelNeRF	No per-scene optimization	19.31	0.789	0.382
IBRNet		26.04	0.917	0.190
MVSNeRF		**26.63**	**0.931**	**0.168**
Ours		26.19	0.922	0.177
NeRF	Per-scene optimization	27.01	0.902	0.263
$\text{IBRNet}_{\text{ft-1.5h}}$		31.35	0.956	0.131
$\text{MVSNeRF}_{\text{ft-15min}}$		28.50	0.933	0.179
$\text{Ours}_{\text{ft-1h}}$		**31.79**	**0.962**	**0.119**

Table 3. Quantitative comparisons on the DTU dataset [16]. Our model is able to generate good results under this difficult setting where only 3 input views are given for the direct inference. Our fine-tuning results outperforms other methods in all three metrics.

NeRF Synthetic. Our method also works well on small-scale scenes. We conduct experiments on the NeRF Synthetic 360° dataset, and apply the same evaluation setting as in [28]. As shown in Table 2, without per-scene fine-tuning, our model generates results that are comparable to IBRNet; however, fine-tuning significantly boosts the performance of our model, leading to high accuracy that is much superior to the fine-tuned IBRNet. In practice, IBRNet suffers from the sparsely distributed input views of the dataset with large baselines, where interpolating neighboring views are not effective to synthesize realistic novel view images. Our fine-tuned model also achieves similar performance when compared to per-scene optimization methods [24,28], while ours is optimized for only 1 hour, substantially less than the time other methods require.

DTU. To show that our method works with a small number of input views with small baselines. We also evaluate our model on the DTU dataset, following the experiment settings in MVSNeRF [7], where only 3 views are provided for the setting without per-scene optimization and 16 more views are provided in the per-scene optimization setting. Here we also compare with PixelNeRF [49], which is specifically trained for the DTU dataset in their paper. As demonstrated in Tab. 3 and Fig. 4, similar to previous results, our model generalizes well to the testing scenes and can be efficiently fine-tuned to outperform NeRF and other generalizable methods including IBRNet [43] and MVSNeRF [7].

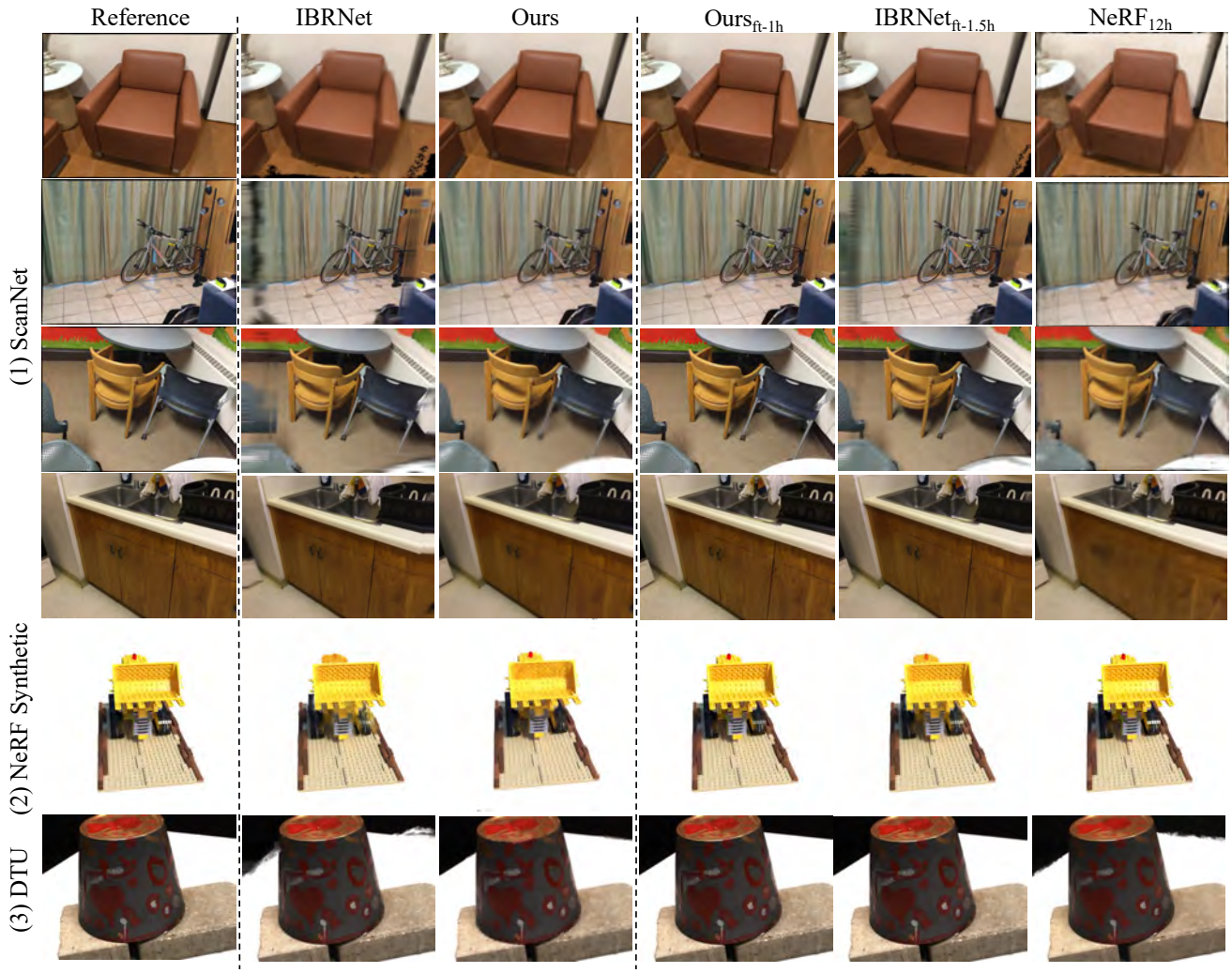

Figure 4. Qualitative comparisons of rendering quality on diverse scenes between our method and state-of-the-art method. Our method achieves better performance than state-of-the-art generalizable method IBRNet [43] in both the direct-inference and the fine-tuned settings, where IBRNet generates results with obvious blurry and tearing artifacts. Our model fine-tuned for 1 hour can generate even better results than NeRF [28] that requires 12 hours of training, especially on large-scale scenes from ScanNet [11].

6. Limitations.

Our approach currently focuses on handling large-scale indoor scenes, but might not be efficient on handling scenes that have foreground objects with distant background, which might appear in unbounded outdoor scenes. This is because we consider a uniform grid for the entire scene, similar to [24]. This can be potentially addressed in the future by doing per-view reconstruction in disparity space or applying spherical coordinates for regions at long distances (similar to [50]). Our method relies on multi-view correspondence; hence, extreme camera poses without enough parallax could lead to problems, which cannot be addressed by any MVS-based techniques. For our current pipeline, we simply sample input frames uniformly, because the camera motion in ScanNet has enough translation. However, a more careful input view selection technique that accounts for relative camera poses may be necessary in practice to address various types of camera motions.

7. Conclusion

In this work, we present a novel neural approach that can achieve fast, large-scale, and high-quality scene reconstruction for photo-realistic rendering. In contrast to traditional TSDF-based reconstruction, we reconstruct scenes as volumetric radiance fields, leading to photo-realistic view synthesis results. Our approach leverages a novel recurrent neural network to process the input image sequence and incrementally reconstruct a global large-scale radiance field by reconstructing and fusing per-frame local radiance fields. We demonstrate that our approach can achieve the state-of-the-art rendering quality for large-scale indoor scenes from ScanNet while taking substantially less reconstruction time.

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] Sai Bi, Nima Khademi Kalantari, and Ravi Ramamoorthi. Patch-based optimization for image-based texture mapping. *ACM Transaction on Graphics*, 36(4):106–1, 2017. 2

[3] 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

[4] 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. In *Proc. ECCV*, 2020. 3

[5] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. Nerd: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12684–12694, 2021. 3

[6] Aljaž Božič, Pablo Palafox, Justus Thies, Angela Dai, and Matthias Nießner. Transformerfusion: Monocular rgb scene reconstruction using transformers. *Proc. Neural Information Processing Systems (NeurIPS)*, 2021. 2

[7] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. *arXiv preprint arXiv:2103.15595*, 2021. 3, 4, 6, 7, 11

[8] Rui Chen, Songfang Han, Jing Xu, and Hao Su. Point-based multi-view stereo network. In *Proc. ICCV*, 2019. 2

[9] 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

[10] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. 5

[11] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 1, 2, 6, 7, 8, 11

[12] 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

[13] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics*, 37(6):1–15, 2018. 2

[14] Peter Hedman, Pratul P Srinivasan, Ben Mildenhall, Jonathan T Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis. *arXiv preprint arXiv:2103.14645*, 2021. 2

[15] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2821–2830, 2018. 2

[16] 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. 2, 7

[17] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 406–413, 2014. 6, 11

[18] Mengqi Ji, Juergen Gall, Haitian Zheng, Yebin Liu, and Lu Fang. SurfaceNet: An end-to-end 3D neural network for multiview stereopsis. In *Proc. ICCV*, 2017. 4

[19] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. In *NeurIPS*. 2017. 2

[20] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 364–375, 2017. 5

[21] Georgios Kopanas, Julien Philip, Thomas Leimkühler, and George Drettakis. Point-based neural rendering with per-view optimization. In *Computer Graphics Forum*, volume 40, pages 29–43. Wiley Online Library, 2021. 2

[22] Christoph Lassner and Michael Zollhofer. Pulsar: Efficient sphere-based neural rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1440–1449, 2021. 2

[23] 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. 3

[24] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *arXiv preprint arXiv:2007.11571*, 2020. 2, 3, 5, 6, 7, 8

[25] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *arXiv preprint arXiv:1906.07751*, 2019. 3

[26] 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

[27] 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

[28] 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

[29] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality*, pages 127–136. IEEE, 2011. 1, 2

[30] Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (ToG)*, 32(6):1–11, 2013. 2

[31] 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. 3

[32] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *arXiv preprint arXiv:2106.13228*, 2021. 3

[33] Matia Pizzoli, Christian Forster, and Davide Scaramuzza. Remode: Probabilistic, monocular dense reconstruction in real time. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2609–2616. IEEE, 2014. 5

[34] Google Research. Google scanned objects. 2, 6, 11

[35] Darius Rückert, Linus Franke, and Marc Stamminger. Adop: Approximate differentiable one-pixel point rendering. *arXiv preprint arXiv:2110.06635*, 2021. 2

[36] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 11

[37] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise View Selection for Unstructured Multi-View Stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 2

[38] 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

[39] 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 *CVPR*, 2021. 3

[40] Jiaming Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. NeuralRecon: Real-time coherent 3D reconstruction from monocular video. *CVPR*, 2021. 2, 5, 11

[41] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019. 11

[42] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching Efficient 3D Architectures with Sparse Point-Voxel Convolution. In *European Conference on Computer Vision (ECCV)*, 2020. 6, 11

[43] 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, 11

[44] Silvan Weder, Johannes Schonberger, Marc Pollefeys, and Martin R Oswald. Routedfusion: Learning real-time depth map fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4887–4897, 2020. 1, 2

[45] Yi Wei, Shaohui Liu, Yongming Rao, Wang Zhao, Jiwen Lu, and Jie Zhou. Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo. In *ICCV*, 2021. 6, 7

[46] Fanbo Xiang, Zexiang Xu, Milos Hasan, Yannick Hold-Geoffroy, Kalyan Sunkavalli, and Hao Su. Neutex: Neural texture mapping for volumetric neural rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7119–7128, 2021. 3

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

[48] 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

[49] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *CVPR*, 2021. 3, 6, 7, 11

[50] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 8

[51] Qian-Yi Zhou and Vladlen Koltun. Color map optimization for 3D reconstruction with consumer depth cameras. *ACM Transactions on Graphics*, 33(4):155, 2014. 2

Figure 5. Effect of fine-tuning. We show the results with our model with different fine-tuning duration. The first column is our results without fine-tuning. Note that our direct inference result have outperformed NeRF and the results from fine-tuning contain significantly more details. PSNRs are shown at top right of each image.

A. Additional Details

We follow [40] to use a modified MnasNet [41] pretrained from ImageNet [36] as the 2D encoder. The output feature channel number from the 2D encoder is 64. The direction encoder G is a 5 layer MLP with 16-channel output. The SparseConvNet J has 5 SparseConv layers implemented with torchsparse [42]. Each of M_z, M_r, M_t has 3 SparseConv layers. All networks use ReLU as activation layers. All features in the feature volumes (global $\mathcal{V}_t^g$ or local $\mathcal{V}_t$) have 16 channels. We apply positional encoding to the volume feature with maximum frequency $L = 5$ before feeding them into the volume renderer. The initial voxel size is set manually for different datasets, specifically $40mm$ for large-scale datasets *e.g.* ScanNet [11], and $4mm$ for object-centric datasets *e.g.* NeRF Synthetic [28] and Google Scanned Objects [34]. When unprojecting 2D features into local feature volumes, we build view frustums with max depth $d_{\max} = 3m$. The pruning threshold γ is set to 0.6 for all experiments. Nearest-neighbor interpolation is used for all coarse-to-fine fine-tuning experiments. The code will be released after the review period.

B. Additional Results

Effect of input view numbers. Figure 6 shows the comparison when sample different numbers of neighboring views to build the feature volume. Our learned global feature fusion module is able to effectively fuse information from different views. Note that, when using more input views, our method could produce sharper details in the rendered images, which are very close to the reference.

Effect of fine-tuning. Figure 5 shows the visual quality of our method when fine-tuning for different time period. Note that our direct inference result have outperformed NeRF, and the results from fine-tuning contain significantly more details. PSNRs are shown at top right of each image.

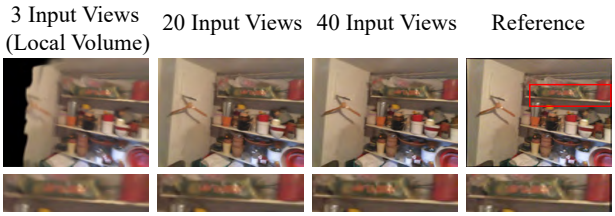

Figure 6. Effect of input view numbers. We show the results when sample different numbers of neighboring views to build the feature volume. All results are from the pretrained model without any further fine-tuning.

Method	Abs Err↓	Acc (8mm)↑
PixelNeRF [49]	0.205/0.211	0.096/0.089
IBRNet [43]	1.123/1.324	0.000/0.000
Ours	**0.034**/**0.036**	**0.722**/**0.709**

Table 4. **Geometry reconstruction.** We evaluate depth reconstruction on the DTU testing set and compare with other two neural rendering methods PixelNeRF [49] and IBRNet [43]. Our method significantly outperforms other neural rendering methods and achieves high depth accuracy. The two numbers of each item refers to the depth at *input* / *novel* views.

Geometry reconstruction. Following [7, 28], we evaluate our geometry reconstruction quality on the DTU dataset [17] 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. We use 16 input views and compare the depth quality on both input views and novel views with 2 common metrics. The results are shown in Table 4. Thanks to the explicit geometry modeling in our pipeline, our approach achieves significantly more accurate geometry than other neural rendering methods.