arXiv:2201.08845v6 [cs.CV] 2 Dec 2022

Point-NeRF: Point-based Neural Radiance Fields

Qiangeng Xu[1] † Zexiang Xu[2] Julien Philip [2] Sai Bi[2] Zhixin Shu[2]
Kalyan Sunkavalli[2] Ulrich Neumann[1]
[1]University of Southern California [2]Adobe Research
{qiangenx,uneumann}@usc.edu {zexu,juphilip,sbi,zshu,sunkaval}@adobe.com

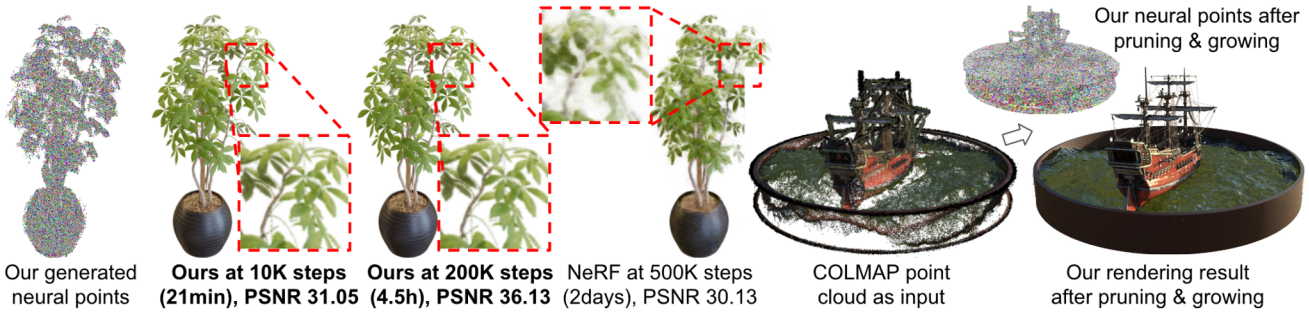


Figure 1. Point-NeRF uses neural 3D points to efficiently represent and render a continuous radiance volume. The point-based radiance field can be predicted via network forward inference from multi-view images. It can then be optimized per scene to achieve reconstruction quality that surpasses NeRF [35] in tens of minutes. Point-NeRF can also leverage off-the-shelf reconstruction methods like COLMAP [44] and is able to perform point pruning and growing that automatically fix the holes and outliers that are common in these approaches.

Abstract

Volumetric neural rendering methods like NeRF [35] generate high-quality view synthesis results but are optimized per-scene leading to prohibitive reconstruction time. On the other hand, deep multi-view stereo methods can quickly reconstruct scene geometry via direct network inference. Point-NeRF combines the advantages of these two approaches by using neural 3D point clouds, with associated neural features, to model a radiance field. Point-NeRF can be rendered efficiently by aggregating neural point features near scene surfaces, in a ray marching-based rendering pipeline. Moreover, Point-NeRF can be initialized via direct inference of a pre-trained deep network to produce a neural point cloud; this point cloud can be finetuned to surpass the visual quality of NeRF with 30× faster training time. Point-NeRF can be combined with other 3D reconstruction methods and handles the errors and outliers in such methods via a novel pruning and growing mechanism. The experiments on the DTU [18], the NeRF Synthetics [35], the ScanNet [11] and the Tanks and Temples [23] datasets demonstrate Point-NeRF can surpass the existing methods and achieve the state-of-the-art results.

1. Introduction

Modeling real scenes from image data and rendering photo-realistic novel views is a central problem in computer vision and graphics. NeRF [35] and its extensions [29, 32, 64] have shown great success on this by modeling neural radiance fields. These methods [35, 38, 64] often reconstruct radiance fields using global MLPs for the entire space through ray marching. This leads to long reconstruction times due to the slow per-scene network fitting and the unnecessary sampling of vast empty space.

We address this issue using Point-NeRF, a novel point-based radiance field representation that uses 3D neural points to model a continuous volumetric radiance field. Unlike NeRF that purely depends on per-scene fitting, Point-NeRF can be effectively initialized via a feed-forward deep neural network, pre-trained across scenes. Moreover, Point-NeRF avoids ray sampling in the empty scene space by leveraging classical point clouds that approximate the actual scene geometry. This advantage of Point-NeRF leads to more efficient reconstruction and more accurate rendering than other neural radiance field models [8, 35, 53, 63].

Our Point-NeRF representation consists of a point cloud

†This work is partially done during the internship at Adobe Research. Code and results: xharlie.github.io/projects/project_sites/pointnerf.

with per-point neural features: each neural point encodes the local 3D scene geometry and appearance around it. Prior point-based rendering techniques [2] use similar neural point clouds but perform rendering with rasterization and 2D CNNs operating in image space. We instead treat these neural points as local neural basis functions in 3D to model a continuous volumetric radiance field which enables high-quality rendering using differentiable ray marching. In particular, for any 3D location, we propose to use an MLP network to aggregate the neural points in its neighborhood to regress the volume density and view-dependent radiance at that location. This expresses a continuous radiance field.

We present a learning-based framework to efficiently initialize and optimize the point-based radiance fields. To generate a initial field, we leverage deep multi-view stereo (MVS) techniques [59], i.e., applying a cost-volume-based network to predict depth which is then unprojected to 3D space. In addition, a deep CNN is trained to extract 2D feature maps from input images, naturally providing the per-point features. These neural points from multiple views are combined as a neural point cloud, which forms a point-based radiance field of the scene. We train this point generation module with the point-based volume rendering networks from end to end, to render novel view images and supervise them with the ground truth. This leads to a *generalizable* model that can directly predict a point-based radiance field at inference time. Once predicted, the initial point-based field is further optimized per scene in a short period to achieve photo-realistic rendering. As shown in Fig. 1 (left), 21 minutes of optimization with Point-NeRF outperforms a NeRF model trained for days.

Besides using the in-built point cloud reconstruction, our approach is *generic* and can also generate a radiance field based on a point cloud of other reconstruction techniques. However, the reconstructed point cloud produced by techniques like COLMAP [44], in practice, contain holes and outliers that adversely affect the final rendering. To address this issue, we introduce *point growing* and *pruning* as part of our optimization process. We leverage the geometric reasoning during volume rendering [13] and grow points near the point cloud boundary in high volume density regions and prune points in low-density regions. The mechanism effectively improves our final reconstruction and rendering quality. We show an example in Fig. 1 (right) where we convert COLMAP points to a radiance field and successfully fill large holes and produce photo-realistic renderings.

We train our model on the DTU dataset [18] and evaluate on DTU testing scenes, NeRF synthetic, Tanks & Temples [23], and ScanNet [11] scenes. The results demonstrate that our approach can achieve state-of-the-art novel view synthesis, outperforming many prior arts including point-based methods [2], NeRF, NSVF [29], and many other generalizable neural methods [8, 53, 63] (see (Tab. 1 and 2)).

2. Related Work

Scene representations. Traditional and neural methods have studied many 3D scene representations, including volumes [19, 25, 41, 46, 56], point clouds [1, 40, 51], meshes [20, 52], depth maps [17, 28], and implicit functions [9, 33, 37, 60], in diverse vision and graphics applications. Recently, various neural scene representations have been presented [4, 30, 47, 67], advancing the state of the art in novel view synthesis and realistic rendering, with volumetric neural radiance fields (NeRFs) [35] producing high fidelity results. NeRFs are often reconstructed as global MLPs [35, 38, 64] that encode the entire scene space; this can be inefficient and expensive when reconstructing complex and large-scale scenes. Instead, Point-NeRF is a localized neural representation, combining volumetric radiance fields with point clouds that are classically used to approximate scene geometry. We distribute fine-grained neural points to model complex local scene geometry and appearance, leading to better rendering quality than NeRF (see Fig. 6, 7).

Voxel grids with per-voxel neural features [8, 16, 29] are also a local neural radiance representation. However, our point-based representation adapts better to actual surfaces, leading to better quality. Also, we directly predict good initial neural point features, bypassing the per-scene optimization that is required by most voxel-based methods [16, 29].

Multi-view reconstruction and rendering. Multi-view 3D reconstruction has been extensively studied and addressed with a number of structure-from-motion [43, 49, 50] and multi-view stereo techniques [10, 14, 25, 44, 59]. Point clouds are often the direct output from MVS or depth sensor, though they are usually converted to meshes [21, 31] for rendering and visualization. Meshing can introduce errors and may require image-based rendering [6, 12, 66] for high-quality rendering. We instead directly use point clouds from deep MVS to achieve realistic rendering.

Point clouds have been widely used in rendering, often via rasterization-based point splatting, and even differentiable rasterization modules [26, 55]. However, reconstructed point clouds often have holes and outliers that lead to artifacts in rendering. Point-based neural rendering methods address this by splatting neural features and using 2D CNNs to render them [2, 24, 34]. In contrast, our point-based approach utilizes 3D volume rendering, leading to significantly better results than previous point-based methods.

Neural radiance fields. NeRFs [35] have demonstrated remarkably high-quality results for novel view synthesis. They have been extended to achieve dynamic scene capture [27, 39], relighting [3, 5], appearance editing [57], fast rendering [16, 62], and generative models [7, 36, 45]. However, most methods [3, 27, 39, 57] still follow the original NeRF framework and train per-scene MLPs to represent radiance fields. We make use of neural points with spatially

varying neural features in a scene to encode its radiance field. This localized representation can model more complex scene content than pure MLPs that have limited network capacity. More importantly, we show that our point-based neural field can be efficiently initialized via a pre-trained deep neural network that generalizes across scenes and leads to highly efficient radiance field reconstruction.

Prior works also present generalizable radiance field-based methods. PixelNeRF [63] and IBRNet [53] aggregate multi-view 2D image features at every sampled ray point to regress volume rendering properties for radiance field rendering. In contrast, we leverage features in 3D neural points around the scene surface to model radiance fields. This avoids sampling points in the vast empty space and leads to higher rendering quality and faster radiance field reconstruction than PixelNeRF and IBRNet. MVSNeRF [8] can achieve very fast voxel-based radiance field reconstruction. However, its prediction network requires a fixed number of three small-baseline images as input and thus can only efficiently reconstruct local radiance fields. Our approach can fuse neural points from an arbitrary number of views and achieve fast reconstruction of complete 360 radiance fields which MVSNeRF cannot support.

3. Point-NeRF Representation

We present our novel point-based radiance field representation, designed for efficient reconstruction and rendering (see Fig. 2 (b)). We start with some preliminaries.

Volume rendering and radiance fields. Physically-based volume rendering can be numerically evaluated via differentiable ray marching. Specifically, a pixel's radiance can be computed by marching a ray through the pixel, sampling M shading points at $\{x_j \mid j = 1, ..., M\}$ along the ray, and accumulating radiance using volume density, as:

$$\begin{aligned} c &= \sum_{M} \tau_j (1 - \exp(-\sigma_j \Delta_j)) r_j, \\ \tau_j &= \exp(-\sum_{t=1}^{j-1} \sigma_t \Delta_t). \end{aligned} \tag{1}$$

Here, τ represents volume transmittance; σ_j and r_j are the volume density and radiance for each shading point j at x_j, Δ_t is the distance between adjacent shading samples.

A radiance field represents the volume density σ and view-dependent radiance r at any 3D location. NeRF [35] proposes to use a multi-layer perceptron (MLP) to regress such radiance fields. We propose Point-NeRF that instead utilizes a neural point cloud to compute the volume properties, allowing for faster and higher-quality rendering.

Point-based radiance field. We denote a neural point cloud by $P = \{(p_i, f_i, \gamma_i) | i = 1, ..., N\}$, where each point i is located at p_i and associated with a neural feature vector f_i that encodes the local scene content. We also assign each point a scale confidence value $\gamma_i \in [0, 1]$ that represents how likely that point is being located near an actual scene surface. We regress the radiance field from this point cloud.

Given any 3D location x, we query K neighboring neural points around it within a certain radius R. Our point-based radiance field can be abstracted as a neural module that regresses volume density σ and view-dependent radiance r (along any viewing direction d) at any shading location x from its neighboring neural points as:

$$(\sigma, r) = \text{Point-NeRF}(x, d, p_1, f_1, \gamma_1, ..., p_K, f_K, \gamma_K). \tag{2}$$

We use a PointNet-like [40] neural network, with multiple sub-MLPs, to do this regression. Overall, we first conduct neural processing for each neural point and then aggregate the multi-point information to obtain the final estimates.

Per-point processing. We use an MLP F to process each neighboring neural point to predict a new feature vector for the shading location x by:

$$f_{i,x} = F(f_i, x - p_i). \tag{3}$$

Essentially, the original feature f_i encodes the local 3D scene content around p_i. This MLP network expresses a local 3D function that outputs the specific neural scene description $f_{i,x}$ at x, modeled by the neural point in its local frame. The usage of relative position $x - p$ makes the network invariant to point translation for better generalization.

View-dependent radiance regression. We use standard inverse distance weighting to aggregate the neural features $f_{i,x}$ regressed from these K neighboring points to obtain a single feature f_x that describes scene appearance at x:

$$f_x = \sum_i \gamma_i \frac{w_i}{\sum w_i} f_{i,x}, \text{where } w_i = \frac{1}{\|p_i - x\|}. \tag{4}$$

Then an MLP, R, regress the view-dependent radiance from this feature given a viewing direction, d:

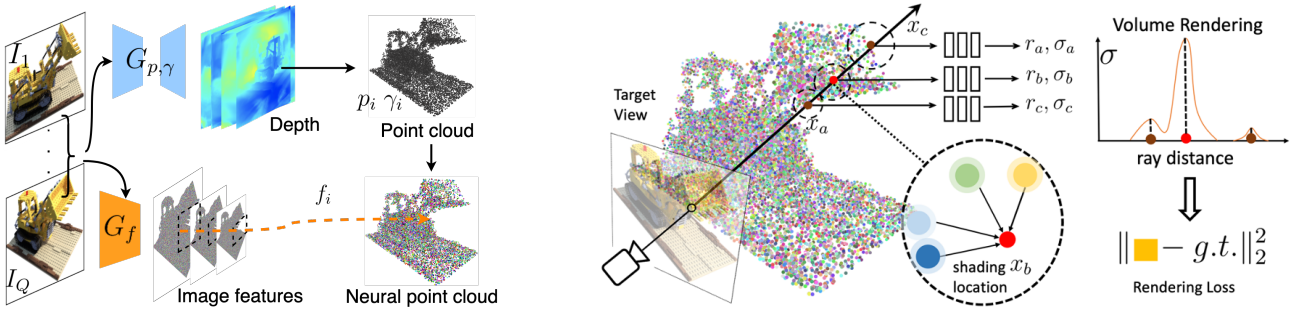
$$r = R(f_x, d). \tag{5}$$

The inverse-distance weight w_i is widely used in scattered data interpolation; we leverage it to aggregate neural features, making closer neural points contribute more to the shading computation. In addition, we use the per-point confidence γ in this process; this is optimized in the final reconstruction with a sparsity loss, giving the network the flexibility of rejecting unnecessary points.

Density regression. To compute volume density σ at x, we follow a similar multi-point aggregation. However, we first regress a density σ_i per point using an MLP T and then do inverse distance-based weighting, given by:

$$\sigma_i = T(f_{i,x}) \tag{6}$$

$$\sigma = \sum_i \sigma_i \gamma_i \frac{w_i}{\sum w_i}, w_i = \frac{1}{\|p_i - x\|}. \tag{7}$$



(a) Neural Point Generation.

(b) Point-NeRF Representation with Volume Rendering.

Figure 2. Overview of Point-NeRF. (a) From multi-view images, our model generates depth for each view by using a cost volume-based 3D CNNs $G_{p,\gamma}$ and extract 2D features from the input images by a 2D CNN G_f. After aggregating the depth map, we obtain a point-based radiance field in which each point has a spatial location p_i, a confidence γ_i and the unprojected image features f_i. (b) To synthesize a novel view, we conduct differentiable ray marching and compute shading only nearby the neural point cloud (e.g., x_a, x_b, x_c). At each shading location, Point-NeRF aggregates features from its K neural point neighbors and compute radiance r and volume density σ then accumulate r using σ. The entire process is end-to-end trainable and the point-based radiance field can be optimized with the rendering loss.

Thus, each neural point directly contributes to the volume density, and point confidence γ_i is explicitly associated with this contribution. We leverage this in our point removal process (see Sec. 4.2).

Discussion. Unlike previous neural point-based methods [2, 34] that rasterize point features and then render them with 2D CNNs, our representation and rendering are entirely in 3D. By using a point cloud that approximates the scene geometry, our representation naturally and efficiently adapts to scene surfaces and avoids sampling shading locations in empty scene space. For shading points along each ray, we implement an efficient algorithm to query neighboring neural points; details are in the supplemental material.

4. Point-NeRF Reconstruction

We now introduce our pipeline for efficiently reconstructing point-based radiance fields. We first leverage a deep neural network, trained across scenes, to generate an initial point-based field via direct network inference (Sec. 4.1). This initial field is further optimized per scene with our point growing and pruning techniques, leading to our final high-quality radiance field reconstruction (Sec. 4.2). Figure. 3 shows this workflow with the corresponding gradient updates for the initial prediction and per-scene optimization.

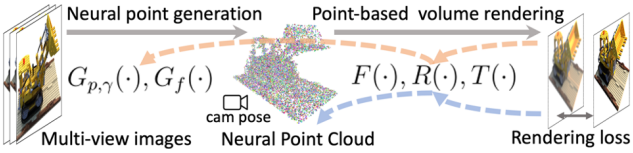


Figure 3. The dash lines indicate gradient updates for radiance field initialization and per-scene optimization.

4.1. Generating initial point-based radiance fields

Given a set of known images I_1,...,I_Q, and a point cloud, our Point-NeRF representation can be reconstructed by optimizing the randomly initialized per-point neural features and the MLPs with a rendering loss (similar to NeRF). However, this pure per-scene optimization depends on an exisiting point cloud, and can be prohibitively slow. Therefore, we propose a neural generation module to predict all neural point properties, including point locations p_i, neural features f_i and point confidence γ_i, via a feed-forward neural network for efficient reconstruction. The direct inference of the network outputs a good initial point-based radiance field. The initial fields can then be fine-tuned to achieve high-quality rendering. In a very short period, the rendering quality is better or on par with NeRF which takes substantially longer time to optimize (see Tab. 1 and 2).

Point location and confidence. We leverage deep MVS methods to generate 3D point locations using cost volume-based 3D CNNs [10, 59]. Such networks produce high-quality dense geometry and generalize well across domains. For each input image I_q with camera parameters Φ_q at viewpoint q, we follow MVSNet [17] to first build a plane-swept cost volume by warping 2D image features from neighboring viewpoints and then regress depth probability volume using deep 3D CNNs. A depth map is computed by linearly combining per-plane depth values weighted by the probabilities. We unprojected the depth map to 3D space to get a point cloud $\{p_1, ..., p_{N_q}\}$ per view q.

Since the depth probabilities describe the likelihood of the point being on the surface, we tri-linearly sample the depth probability volume to obtain the point confidence γ_i at each point p_i. The above process can be expressed by

$$\{p_i, \gamma_i\} = G_{p,\gamma}(I_q, \Phi_q, I_{q_1}, \Phi_{q_1}, I_{q_2}, \Phi_{q_2}, ...), \quad (8)$$

where $G_{p,\gamma}$ is the MVSNet-based network. $I_{q_1}, \Phi_{q_1}, ...$ are additional neighboring views used in the MVS reconstruction; we use two additional views in most cases.

Point features. We use a 2D CNN G_f to extract neural 2D image feature maps from each image I_q. These fea-

ture maps are aligned with the point (depth) prediction from $G_{p,\gamma}$ and are used to directly predict per-point features f_i as:

$$\{f_i\} = G_f(I_q). \tag{9}$$

In particular, we use a VGG network architecture for G_f that has three downsampling layers. We combine intermediate features at different resolutions as f_i, providing a meaningful point description that models multi-scale scene appearance. (See Fig. 2(a))

End-to-end reconstruction. We combine point clouds from multiple viewpoints to obtain our final neural point cloud. We train the point generation networks along with the representation networks, from end to end with a rendering loss (see Fig. 3). This allows our generation modules to produce reasonable initial radiance fields. It also initializes the MLPs in our Point-NeRF representation with reasonable weights, significantly saving the per-scene fitting time.

Moreover, apart from using the full generation module, our pipeline also supports using a point cloud reconstructed from other approaches like COLMAP [44], where our model (excluding the MVS network) can still provide meaningful initial neural features for each point. Please refer to our supplementary material for the details.

4.2. Optimizing point-based radiance fields

The above pipeline can output a reasonable initial point-based radiance field for a novel scene. Through differentiable ray marching, we can further improve the radiance field by optimizing the neural point cloud (point features f_i and point confidence γ_i) and the MLPs in our representation, for that specific scene (see Fig. 3).

The initial point cloud, especially ones from external reconstruction methods (e.g., Metashape or COLMAP in Fig. 1), can often contain holes and outliers that degrade the rendering quality. During per-scene optimization, to solve this problem, we find that directly optimizing the location of the existing points makes the training unstable and cannot fill the large holes (see 1). Instead, we apply novel point pruning and growing techniques that gradually improve both geometry modeling and rendering quality.

Point pruning. As introduced in Sec. 3, we designed point confidence values γ_i that describe whether a neural point is near a scene surface. We utilize these confidence values to prune unnecessary outlier points. Note that the point confidence is directly related to the per-point contribution in volume density regression (Eqn. 7); as a result, low confidence reflects low volume density in a point's local region indicating that it is empty. Therefore, we prune points that have $\gamma_i < 0.1$ every 10K iterations.

We also impose a sparsity loss on point confidence [30]:

$$\mathcal{L}_{\text{sparse}} = \frac{1}{|\gamma|} \sum_{\gamma_i} [log(\gamma_i) + log(1 - \gamma_i)] \tag{10}$$

which forces the confidence value to be close to either zero or one. As shown in Fig. 4, this pruning technique can remove outlier points and reduce the corresponding artifacts.

Point growing. We also propose a novel technique to grow new points to cover missing scene geometry in the original point cloud. Unlike point pruning that directly utilizes information from existing points, growing points requires recovering information in empty regions where no point exists. We achieve this by progressively growing points near the point cloud boundary based on the local scene geometry modeled by our Point-NeRF representation.

In particular, we leverage the per-ray shading locations (x_j in Eqn. 1) sampled in the ray marching to identify new point candidates. Specifically, we identify the shading location x_{j_g} with the highest opacity along the ray:

$$\alpha_j = 1 - \exp(-\sigma_j \Delta_j), \quad j_g = \underset{j}{\operatorname{argmax}}\, \alpha_j. \tag{11}$$

We compute ϵ_{j_g} as x_{j_g}'s distance to its closest neural point.

For a marching ray, we grow a neural point at x_{j_g} if $\alpha_{j_g} > T_{\text{opacity}}$ and $\epsilon_{j_g} > T_{\text{dist}}$. This implies that the location lies near the surface, but is far from other neural points. By repeating this growing strategy, our radiance field can be expanded to cover missing regions in the initial point cloud. Point growing especially benefits point clouds reconstructed by methods like COLMAP that are not dense (see Fig. 4). We show that even on an extreme case with only 1000 initial points, our technique is able to progressively grow new points and reasonably cover the object surface (see Fig. 5).

5. Implementation details

Network details. We apply frequency positional encoding on the relative position and the per-point features for the per-point processing network G_f, and the viewing direction for the network R. We extract multi-scale images features from three layers at different resolutions in network G_f, leading to a vector with 56 (8+16+32) channels. We additionally append the corresponding viewing directions from each input viewpoint, to handle view-dependent effects. Therefore our final per-point neural feature is a 59-channel vector. Please refer to our supplemental material for the details of network architectures and neural point querying during shading.

Training and optimization details. We train our full pipeline on the DTU dataset, using the same training and testing split as PixelNeRF and MVSNeRF. We first pretrain the MVSNet-based depth generation network using the ground truth depth similar to the original MVSNet paper [59]. We then train our full pipeline from end to end purely with a L2 rendering loss $\mathcal{L}_{\text{render}}$, supervising our rendered pixels from ray marching (via Eqn. 1) with the ground truth, to obtain our Point-NeRF reconstruction network. We

train our full pipeline using Adam [22] optimizer with an initial learning rate of $5e^{-4}$. Our feed-forward network takes $0.2s$ to generate a point cloud from three input views.

In the per-scene optimization stage, we adopt a loss function that combines the rendering and the sparsity loss

$$\mathcal{L}_{\text{opt}} = \mathcal{L}_{\text{render}} + a\mathcal{L}_{\text{sparse}}, \tag{12}$$

where we use $a = 2e^{-3}$ for all our experiments. We perform point growing and pruning every 10K iterations to achieve our final high-quality reconstruction.

6. Experiments

6.1. Evaluation on the DTU testing set.

We evaluate our model on the DTU testing set. We produce novel view synthesis from both direct network inference and per-scene fine-tuning optimization, and compare them with the previous state-of-the art methods including PixelNeRF [63], IBRNet [53], MVSNeRF [8], and NeRF [35]. IBRNet and MVSNeRF utilize similar per-scene fine-tuning; we fine-tune all methods with 10k iterations for the comparison. Additionally, we show our results with only 1k iterations to demonstrate the optimization efficiency.

Tab. 1 shows the quantitative results of all methods with PSNR, SSIM, and LPIPS; qualitative rendering results are shown in Fig. 6. We can see that our fine-tuning results after 10k iterations achieve the best SSIM and LPIPS [65], two out of the three metrics. These are significantly better than MVSNeRF and NeRF. While IBRNet produces slightly better PSNRs, our final renderings in fact recover more accurate texture details and highlights as shown Fig. 6. On the other hand, IBRNet is also more expensive to fine-tune, taking 1 hour—5x longer than ours for the same iterations. This is because IBRNet utilizes a large global CNN, whereas Point-NeRF leverages local point features with small MLPs that are easier to optimize. More importantly, our neural points lies near actual scene surfaces, thus avoids sampling ray points in the empty space.

Apart from the optimization results, our initial radiance field estimated from our network is significantly better than PixelNeRF. In this case, our direct inference is worse than IBRNet and MVSNet, mainly because these two methods are using more complex variance-based feature extraction. Our point features are extracted from a simple VGG network. The same design is used in PixelNeRF; we achieve significantly better results than PixelNeRF due to our novel surface-adaptive point-based representation.

While a more complex feature extractor as in IBRNet might improve quality, it will add burden to memory usage and training efficiency. More importantly, our generation network has already provided high-quality initial radiance field to support efficient optimization. We show that with even 2 min / 1K iterations of fine-tuning for our method leading to a very high visual quality comparable to MVS-NeRF's final 10k-iteration results. This clearly demonstrates the high reconstruction efficiency of our approach.

6.2. Evaluation on the NeRF Synthetic dataset.

While our model is purely trained on the DTU dataset, our network generalizes well to novel datasets that have completely different camera distributions. We demonstrate such results on the NeRF synthetic dataset and compare with other methods with qualitative results in Fig. 7 and quantitative results in Tab. 2. We compare with a point-based rendering model (NPBG) [2], a generalizable radiance field method (IBRNet) [53], and per-scene radiance field reconstruction techniques (NeRF and NSVF) [29, 35].

Comparisons with generalizing methods. We compare with IBRNet, to the best of our knowledge, is the previous best NeRF-based generalizable model that can handle free-viewpoint rendering with any arbitrary numbers. Note that, this dataset has a 360° camera distribution, which is much wider than the DTU dataset. In this case, methods like MVSNeRF cannot be applied, since it recovers a local perspective frustum volume from three input images, which cannot cover the entire 360° viewing range. We, therefore, compare with IBRNet and focus on final results after per-scene optimization in this experiment. We use their released model to produce the results. Our results at 20k iterations (Point-NeRF$_{20K}$) have already outperformed IBRNet's converged results with better PSNR, SSIM, and LIPIPS; we also achieve rendering quality with better geometry and texture details as shown in Fig. 7.

Comparisons with pure per-scene methods. Our results after 20K iterations are quantitatively very close to NeRF's results trained with 200K iterations. Visually, our model at $20K$ iterations already has better renderings in some cases, e.g. the Ficus scene (4th row) in Fig. 7. Point-NeRF$_{20K}$ is optimized for only 40 minutes, which is at least $30\times$ faster than the 20+ hours optimization time taken by NeRF. NSVF's [29] results are also from very long per-scene optimization and yet are only slightly better than our 40min results. Optimizing our model for 200K until convergence can lead to significantly better results than NeRF, NSVF, and all other comparison methods. As shown in Fig. 7, our 200K results contain the most geometry and texture details. Attribute to the point growing technique, our method is the only one that can fully recover details like the thin rope structure in the Ship scene (2nd row).

Comparisons with point-based rendering. Our results are significantly better than the previous state-of-the-art point-based rendering methods. We run NPBG [2] using the same point cloud generated by our MVSNet-based network. However, NPBG can only produce blurry rendering results with their rasterization and 2D CNN framework. In contrast, we leverage volumetric rendering technique with neu-

	No Per-scene Optimization				Per-scene Optimization				
	PixelNeRF [63]	MVSNeRF [8]	IBRNet [53]	Ours	Ours_{1K}	Ours_{10K}	MVSNeRF_{10K}	IBRNet_{10K}	NeRF_{200k}
PSNR ↑	19.31	26.63	26.04	23.89	28.43	30.12	28.50	**31.35**	27.01
SSIM ↑	0.789	0.931	0.917	0.874	0.929	**0.957**	0.933	0.956	0.902
LPIPS_{Vgg} ↓	0.382	0.168	0.190	0.203	0.183	**0.117**	0.179	0.131	0.263
Time↓	-	-	-	-	**2min**	20min	24min	1h	10h

Table 1. Comparisons of our Point-NeRF with radiance-based models [29, 32, 53] and a point-based rendering model [2] on the DTU dataset [18] with the novel view synthesis setting introduced in [8]. The subscripts indicate the number of iterations during optimization.

	NPBG [2]	NeRF [32]	IBRNet [53]	NSVF [29]	$\text{Point-NeRF}^{col}_{200K}$	Point-NeRF_{20K}	Point-NeRF_{200K}
PSNR ↑	24.56	31.01	28.14	31.75	31.77	30.71	**33.31**
SSIM ↑	0.923	0.947	0.942	0.964	0.973	0.967	**0.978**
LPIPS_{Vgg} ↓	0.109	0.081	0.072	-	0.062	0.081	**0.049**
LPIPS_{Alex} ↓	0.095	-	-	0.047	0.040	0.050	**0.027**

Table 2. Comparisons of Point-NeRF with radiance-based models [29,32,53] and a point-based rendering model [2] on the Synthetic-NeRF dataset [32]. The subscripts indicate the number of iterations. Our model not only surpasses other methods when converged after $200K$ steps (Point-NeRF_{200K}), but surpasses IBRNet [53] and is on par with NeRF [35] when optimized by only $20K$ steps (Point-NeRF_{20K}). Our methods can also initialize radiance fields based on point clouds reconstructed by methods such as COLMAP ($\text{Point-NeRF}^{col}_{200K}$).

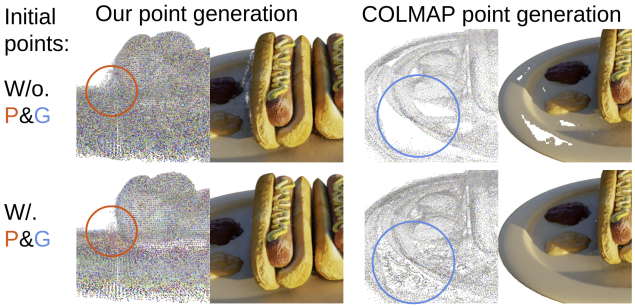


Figure 4. Our neural point clouds and rendered novel views with or without point pruning and growing (P&G). P&G improves both the geometries and rendering results when using the point cloud reconstructed from our model or from COLMAP [44].

ral radiance fields, leading to photo-realistic results.

6.3. Evaluation on the Tanks & Temples and the ScanNet dataset.

We compare Point-NeRF with NSVF on the Tanks & Temples and the ScanNet dataset in Tab. 3. Please refer to the supplemental materials for more comparisons.

	Tanks & Temples [23]	ScanNet [11]
NSVF [29]	28.40 / 0.900 / 0.153	25.48 / 0.688 / 0.301
Point-NeRF	**29.61 / 0.954 / 0.080**	**30.32 / 0.909 / 0.220**

Table 3. The quantitative results (PSNR / SSIM / LPIPS_{Alex}) on the Tanks & Temples and the ScanNet dataset.

6.4. Additional experiments.

Converting COLMAP point clouds to Point-NeRF Apart from using our full pipeline, Point-NeRF can also be used to convert standard point clouds reconstructed by other techniques to point-based radiance fields. We run experiments for this on the full NeRF synthetic dataset, using the point cloud reconstructed by COLMAP [44]. The quantitative results are shown as Point-NeRF_{col} in Tab. 2. Since COLMAP point clouds may contain a lot of holes (as shown in Fig. 1) and noises, we optimize the model for 200K after the initialization to address the point cloud issues with our point growing and pruning techniques. Note that, even from this low-quality point cloud, our final results are still of very high quality with very high SSIM and LPIPS numbers compared to all other methods. This demonstrates that our technique can be potentially combined with any existing point cloud reconstruction techniques, to achieve realistic rendering while improving the point cloud geometry.

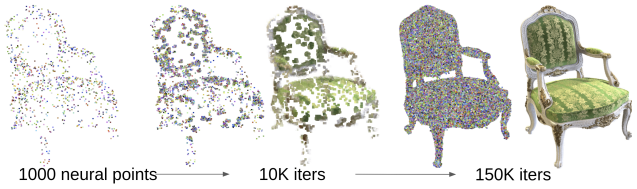


Figure 5. Starting from 1000 randomly sampled COLMAP points of the Chair scene, our point growing mechanism can help complete the geometry and generate high-quality novel views when only being supervised by RGB images.

Point growing and pruning. To further demonstrate the effectiveness of our point growing and pruning modules, we show ablation study results with and without the point growing and pruning in the per-scene optimization. We conduct this experiment on the Hotdog and Ship scenes, using both our full model and our model with COLMAP point clouds. The quantitative results are shown in Tab. 4; our point growing and pruning techniques are very effective, significantly improving the reconstruction results on both cases. We also show the visual results of the Hotdog scene in Fig. 4. We can clearly see that our model is able to prune the point outliers on the left and successfully fill the severe holes on the right in the original COLMAP point cloud.

We also manually create an extreme example to show our point growing technique in Fig. 5, where we start from a very sparse point cloud with only 1000 points sampled from our original point reconstruction. We demonstrate that our approach can progressively grow new point from the point cloud boundary until filling the entire scene surface through iterations. This example further demonstrates the effectiveness of our model, which has high potentials in using image data to recover the accurate scene geometry and appearance from low-quality point clouds.

Please find more results in the supplemental materials.

Method	P&G	Ship	Hotdog
Ours	No	25.50 / 0.878 / 0.182	34.91 / 0.983 / 0.067
Ours	Yes	**30.97 / 0.942 / 0.124**	**37.30 / 0.991 / 0.037**
COLMAP	No	19.35 / 0.905 / 0.167	29.91 / 0.978 / 0.061
COLMAP	Yes	**30.18 / 0.941 / 0.134**	**35.49 / 0.986 / 0.061**

Table 4. The quantitative results (PSNR / SSIM / LPIPS$_{Vgg}$) of the Ship and Hotdog scene with or without point pruning and growing (P&G). The improvements are significant when using either our generated points or the point cloud generated by COLMAP [44].

7. Conclusion

In this paper, we present a novel approach for high-quality neural scene reconstruction and rendering. We propose a novel neural scene representation—Point-NeRF—that models a volumetric radiance field with a neural point cloud. We reconstruct a good initialization of Point-NeRF directly from input images via direct network inference and show that we can efficiently finetune this initialization for a scene. This enables highly efficient Point-NeRF reconstruction with only 20–40 min per-scene optimization, leading to rendering quality comparable to and even surpassing NeRF that requires substantially longer training time (20+ hours). We also present novel effective growing and pruning techniques for our per-scene optimization, significantly improving our results and making our approach robust with different point cloud quality. Our Point-NeRF successfully combines the advantages from both classical point cloud representation and neural radiance field representation, making an important step towards a practical scene reconstruction solution with high efficiency and realism.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3D point clouds. In *ICML*, pages 40–49, 2018. 2

[2] 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, 4, 6, 7, 12, 13, 16

[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. 2

[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. 2

[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. 2

[6] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proc. SIGGRAPH*, pages 425–432, 2001. 2

[7] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5799–5809, 2021. 2

[8] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. *arXiv preprint arXiv:2103.15595*, 2021. 1, 2, 3, 6, 7, 9, 12

[9] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proc. CVPR*, 2019. 2

[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 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2524–2534, 2020. 2, 4

[11] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017. 1, 2, 7, 12, 13

[12] Paul Debevec, Yizhou Yu, and George Borshukov. Efficient view-dependent image-based rendering with projective texture-mapping. In *Rendering Techniques' 98*, pages 105–116. 1998. 2

[13] Robert A Drebin, Loren Carpenter, and Pat Hanrahan. Volume rendering. *ACM Siggraph Computer Graphics*, 22(4):65–74, 1988. 2

[14] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, 2009. 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015. 12

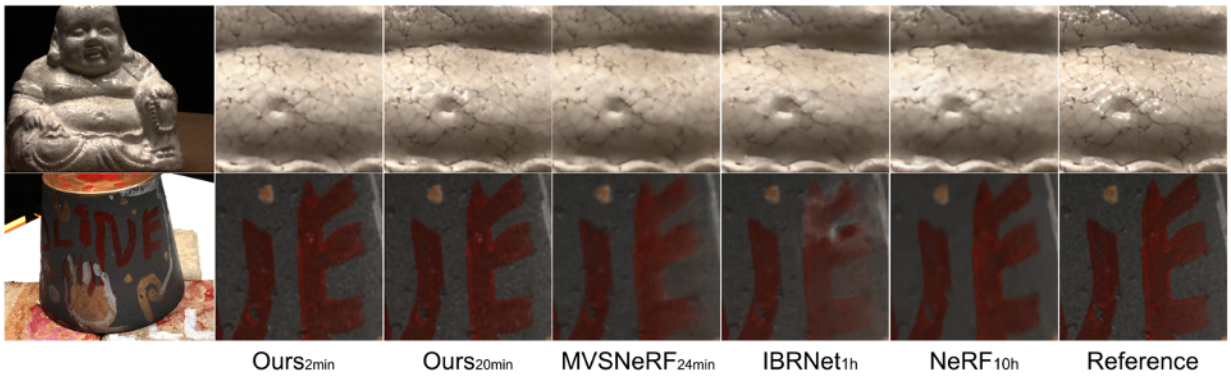


Figure 6. Qualitative comparisons of per-scene optimization on the DTU dataset [18]. Our Point-NeRF can recover texture details and geometrical structures more accurately than other methods. Point-NeRF also demonstrates superior efficiency. Within two mins, our model trained for 1K steps is already on par with the state-of-the-art methods such as MVSNeRF [8] and IBRNet [53]

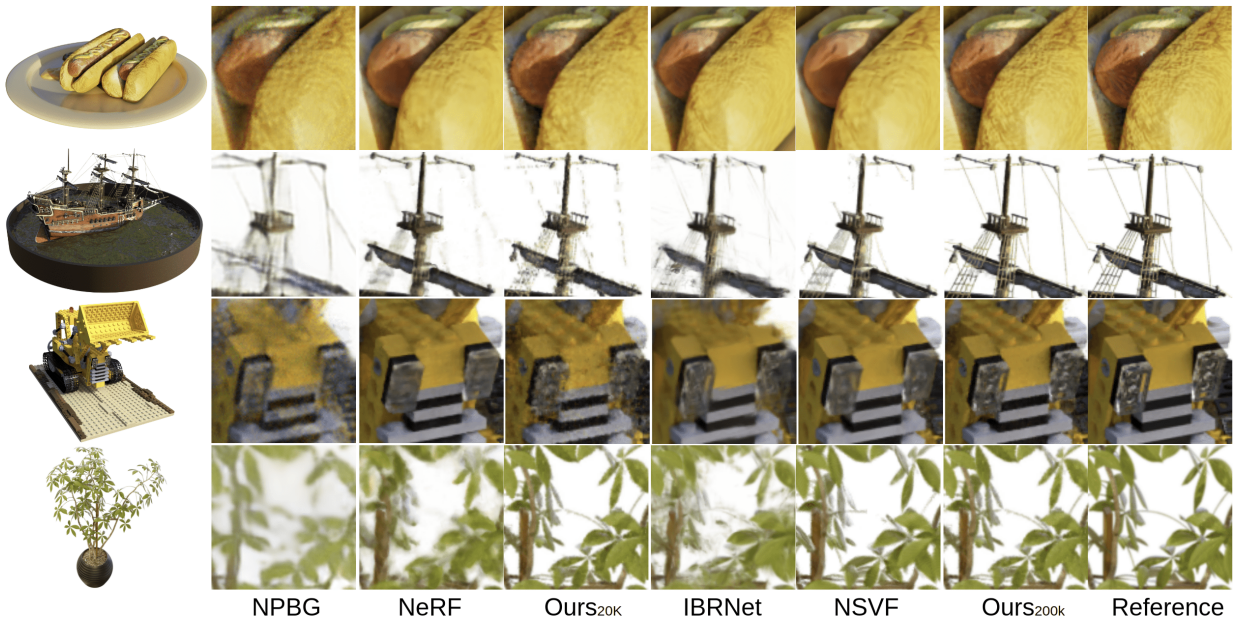


Figure 7. Qualitative comparisons on the NeRF Synthetic dataset [35]. The subscripts indicate the number of iterations. Our Point-NeRF can capture fine details and thin structures (see the rope on row 2). Point-NeRF also demonstrates superior efficiency. Our model trained for 20K steps already on par with NeRF with 30× faster training time.

[16] 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

[17] 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, 4

[18] 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, 2, 7, 9, 12

[19] 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. 2

[20] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proc. ECCV*, 2018. 2

[21] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proc. Eurographics Symposium on Geometry Processing*, volume 7, 2006. 2

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

[23] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 36(4), 2017. 1, 2, 7, 13, 14, 16

[24] Georgios Kopanas, Julien Philip, Thomas Leimkühler, and George Drettakis. Point-based neural rendering with per-view optimization. In *Computer Graphics Forum*, vol-

ume 40, pages 29–43. Wiley Online Library, 2021. 2

[25] 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

[26] 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

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

[28] Fayao Liu, Chunhua Shen, Guosheng Lin, and Ian Reid. Learning depth from single monocular images using deep convolutional neural fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(10):2024–2039, 2016. 2

[29] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *arXiv preprint arXiv:2007.11571*, 2020. 1, 2, 6, 7, 12, 13, 14, 15, 16

[30] 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. 2, 5, 13, 14

[31] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Computer Graphics*, 21(4):163–169, 1987. 2

[32] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7210–7219, 2021. 1, 7, 13, 16

[33] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. *Proc. CVPR*, 2019. 2

[34] Moustafa Meshry, Dan B Goldman, Sameh Khamis, Hugues Hoppe, Rohit Pandey, Noah Snavely, and Ricardo Martin-Brualla. Neural rerendering in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6878–6887, 2019. 2, 4

[35] 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, 6, 7, 9, 12, 13, 14, 15

[36] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11453–11464, 2021. 2

[37] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proc. CVPR*, 2020. 2

[38] 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. 1, 2

[39] 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. 2

[40] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proc. CVPR*, 2017. 2, 3

[41] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proc. CVPR*, 2016. 2

[42] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. *arXiv preprint arXiv:2103.13744*, 2021. 16

[43] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proc. CVPR*, 2016. 2

[44] 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. 1, 2, 5, 7, 8, 13

[45] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. *arXiv preprint arXiv:2007.02442*, 2020. 2

[46] Steven M Seitz and Charles R Dyer. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision*, 35(2):151–173, 1999. 2

[47] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3D feature embeddings. In *Proc. CVPR*, 2019. 2

[48] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *arXiv preprint arXiv:1906.01618*, 2019. 12, 13

[49] Chengzhou Tang and Ping Tan. BA-net: Dense bundle adjustment network. In *Proc. ICLR*, 2019. 2

[50] Sudheendra Vijayanarasimhan, Susanna Ricco, Cordelia Schmid, Rahul Sukthankar, and Katerina Fragkiadaki. Sfm-net: Learning of structure and motion from video. *arXiv preprint arXiv:1704.07804*, 2017. 2

[51] Jinglu Wang, Bo Sun, and Yan Lu. MVPnet: Multi-view point regression networks for 3D object reconstruction from a single image. *Proc. AAAI Conference on Artificial Intelligence*, 2019. 2

[52] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single RGB images. In *Proc. ECCV*, 2018. 2

[53] 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. 1, 2, 3, 6, 7, 9, 12

[54] W Weng and X Zhu. Convolutional networks for biomedical image segmentation. *IEEE Access*, 2015. 14

[55] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7467–7477, 2020. 2

[56] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proc. CVPR*, 2015. 2

[57] 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. 2

[58] Qiangeng Xu, Xudong Sun, Cho-Ying Wu, Panqu Wang, and Ulrich Neumann. Grid-gcn for fast and scalable point cloud learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5661–5670, 2020. 15

[59] 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, 5, 14, 16, 17

[60] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. In *Proc. NeurIPS*, 2020. 2

[61] Alex Yu, Sara Fridovich-Keil, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. *arXiv preprint arXiv:2112.05131*, 2021. 16

[62] 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. 2, 16

[63] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *CVPR*, 2021. 1, 2, 3, 6, 7

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

[65] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 6

[66] 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

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

Appendix

A. Ablation Studies on Point Features Initialization

	Extract_{20k}	Rand_{20k}	Extract_{200k}	Rand_{200k}
PSNR↑	**30.71**	25.44	**33.77**	32.01
SSIM↑	**0.967**	0.932	**0.973**	0.972

Table 5. Comparisons between using the extracted image features to initialize the point features (our full model) or using the random initialized features.

We conduct experiments to demonstrate the importance of our feature initialization. We compare our full model and our model initialized without using the extracted image features on the NeRF Synthetic dataset [35]. Without using the features from images, we randomly initialize the point features by using the popular Kaiming Initialization [15]. As shown in Table 5, the neural points with image features not only achieve better performance after convergence at $200K$ iterations but also converge much faster in the beginning. The randomly initialized neural points even cannot perform as well as our full model, still outperforms state-of-the-art methods such as NeRF and NSVF [29].

B. Per-scene Breakdown Results of the DTU Dataset

We show the per scene detailed quantitative results of the comparisons on the DTU [18] dataset in Table 6 and additional qualitative comparisons in our video. Since our method also faithfully reconstructs the scene geometry, our method has the best SSIM scores in most of the cases. Our model also has the best LPIPS for most of the scenes and therefore, is more visually authentic, as shown in the Figure 6 of the main paper and the video. IBRNet combines the colors from the source views to compute the radiance colors during shading. This image-based approach results in better PSNR. However, as shown in our video, our method is more temporal consistent because the local radiance and geometries are consistently stored at each neural point location.

C. Per-scene Breakdown Results of the NeRF Synthetic Dataset

We show the per scene detailed quantitative results of the comparisons on the NeRF Synthetic [35] dataset in Table 7 and additional qualitative comparisons in our video. Point-NeRF achieves the best PSNRs, SSIMs and LPIPSs on most of the scenes and outperforms state-of-the-art methods [2, 29,35,53] with a big margin. On the other hand, our method initiated with COLMAP points is on par with NeRF. Even starting from the unideal initial points, we still manage to improve the geometry reconstruction and generate a high-quality radiance field with point pruning and growing. The fact that our model at $20K$ iterations matches the results of NeRF at $500K$ iterations clearly demonstrates our ability of fast convergence.

Scan	#1	#8	#21	#103	#114
SSIM↑					
Ours_{1K}	0.935	0.906	0.913	0.944	0.948
Ours_{10K}	0.962	0.949	0.954	0.961	0.960
MVSNeRF_{10K} [8]	0.934	0.900	0.922	0.964	0.945
IBRNET_{10K} [53]	0.955	0.945	0.947	0.968	0.964
NeRF_{200K} [35]	0.902	0.876	0.874	0.944	0.913
$\text{LPIPS}_{Vgg}\downarrow$					
Ours_{1K}	0.151	0.207	0.201	0.208	0.148
Ours_{10K}	0.095	0.130	0.134	0.145	0.096
MVSNeRF_{10K}	0.171	0.261	0.142	0.170	0.153
IBRNET_{10K}	0.129	0.170	0.104	0.156	0.099
NeRF_{200K}	0.265	0.321	0.246	0.256	0.225
PSNR↑					
Ours_{1K}	28.79	28.39	24.78	30.36	29.82
Ours_{10K}	30.85	30.72	26.22	32.08	30.75
MVSNeRF_{10K}	28.05	28.88	24.87	32.23	28.47
IBRNET_{10K}	31.00	32.46	27.88	34.40	31.00
NeRF_{200K}	26.62	28.33	23.24	30.40	26.47

Table 6. Quantity comparison on five sample scenes in the DTU testing set with the view synthesis setting introduced in [8]. The subscripts indicate the number of iterations during optimization.

D. Evaluation on Large-scale 3D Scenes (ScanNet).

While our model is purely trained on a dataset of objects (the DTU dataset), our network generalizes well to large-scale 3D scene datasets. Following [29], we use two 3D scenes, scene 0101_04 and scene 0241_01, from ScanNet [11]. We extract both RGB and depth images from the original videos and from which we sample one out of five frames as training set and use the rest for testing. The RGB images are scaled to 640 x 480. We finetune each scene for 300K steps with point pruning and growing.

We compare with 3 other state-of-the-art methods with quantitative results in Tab. 2. In particular, we compare with a scene representation model (SRN) [48], NeRF [35] and a

	NeRF Synthetic							
	Chair	Drums	Lego	Mic	Materials	Ship	Hotdog	Ficus
	PSNR↑							
NPBG [2]	26.47	21.53	24.84	26.62	21.58	21.83	29.01	24.60
NeRF [35]	33.00	25.01	32.54	32.91	29.62	28.65	36.18	30.13
NSVF [29]	33.19	25.18	32.54	34.27	**32.68**	27.93	37.14	31.23
Point-NeRF$^{col}_{200K}$	35.09	25.01	32.65	35.54	26.97	30.18	35.49	33.24
Point-NeRF$_{20K}$	32.50	25.03	32.40	32.31	28.11	28.13	34.53	32.67
Point-NeRF$_{200K}$	**35.40**	**26.06**	**35.04**	**35.95**	29.61	**30.97**	**37.30**	**36.13**
	SSIM↑							
NPBG	0.939	0.904	0.923	0.959	0.887	0.866	0.964	0.940
NeRF	0.967	0.925	0.961	0.980	0.949	0.856	0.974	0.964
NSVF	0.968	0.931	0.960	0.987	**0.973**	0.854	0.980	0.973
Point-NeRF$^{col}_{200K}$	0.990	0.944	0.983	0.993	0.955	0.941	0.986	0.989
Point-NeRF$_{20K}$	0.981	0.944	0.980	0.986	0.959	0.916	0.983	0.986
Point-NeRF$_{200K}$	**0.991**	**0.954**	**0.988**	**0.994**	0.971	**0.942**	**0.991**	**0.993**
	LPIPS$_{Vgg}$ ↓							
NPBG	0.085	0.112	0.119	0.060	0.134	0.210	0.075	0.078
NeRF	0.046	0.091	0.050	0.028	0.063	0.206	0.121	0.044
Point-NeRF$^{col}_{200K}$	0.026	0.099	0.031	0.019	0.100	0.134	0.061	0.028
Point-NeRF$_{20K}$	0.051	0.103	0.054	0.039	0.102	0.181	0.074	0.043
Point-NeRF$_{200K}$	**0.023**	**0.078**	**0.024**	**0.014**	**0.072**	**0.124**	**0.037**	**0.022**
	LPIPS$_{Alex}$ ↓							
NSVF	0.043	0.069	0.029	0.010	**0.021**	0.162	0.025	0.017
Point-NeRF$^{col}_{200K}$	0.013	0.073	0.016	0.011	0.076	0.087	0.032	0.012
Point-NeRF$_{20K}$	0.027	0.057	0.022	0.024	0.076	0.127	0.044	0.022
Point-NeRF$_{200K}$	**0.010**	**0.055**	**0.011**	**0.007**	0.041	**0.070**	**0.016**	**0.009**

Table 7. Detailed breakdown of quantitative metrics of individual scenes for the NeRF Synthetic [35] for our method and baselines. All scores are averaged over the testing images. The subscripts are the number of iterations of the models and Point-NeRF$^{col}_{200K}$ indicates our method initiates from COLMAP points and optimized for 200 thousand iterations.

	Average over two scenes				**Scene 101**	**Scene 241**
	SRN [48]	NeRF [32]	NSVF [29]	Point-NeRF (Ours)	Point-NeRF (Ours)	
PSNR ↑	18.25	22.99	25.48	**30.32**	30.13	30.51
SSIM ↑	0.592	0.620	0.688	**0.909**	0.912	0.906
RMSE ↓	14.764	0.681	0.079	**0.031**	0.032	0.030
LPIPS$_{Alex}$ ↓	0.586	0.369	0.301	**0.220**	0.203	0.238
LPIPS$_{Vgg}$ ↓	-	-	-	**0.292**	0.286	0.299

Table 8. Quantity comparison on two scenes in the ScanNet dataset [11] selected in NSVF [29]. RMSE is the Root Mean Square Error. Our method Point-NeRF outperforms all state-of-the-art methods in all metrics by substantial margins.

sparse voxel-based neural radiance field, NSVF [29]. The qualitative comparison is shown in Tab. 8 and visual results are shown in Figure 8. Our Point-NeRF outperforms all these previous studies in all metrics by substantial margins. Please find more visual results in our video.

E. The Tanks and Temple Dataset

We also experiment Point-NeRF on the Tanks and Temples dataset [23]. we reconstruct the radiance field of five scenes selected in NSVF [29] and compare our model with three models NV [30], NeRF [35] and NSVF [29]. We show the quantitative comparison in Tab. 9 and visualize quality results in Figure 9. Please find more visual results in our video.

F. Initializing Neural Points from COLMAP Points

Point-NeRF can use the points of any external reconstruction method. For instance, the output of COLMAP [44] is a point cloud $\{(p_i)|i = 1, ..., N\}$. We set γ_i as 0.3

Figure 8. The qualitative results of our Point-NeRF on the ScanNet dataset [23]. The first row shows five generated test frames of scene 101 and the second row shows five generated test frames of scene 241.

	Tanks & Tamples					
	Ignatius	Truck	Barn	Caterpillar	Family	Mean
			PSNR ↑			
NV [30]	26.54	21.71	20.82	20.71	28.72	23.70
NeRF [35]	25.43	25.36	24.05	23.75	30.29	25.78
NSVF [29]	27.91	26.92	27.16	26.44	33.58	28.40
Point-NeRF (Ours)	**28.43**	**28.22**	**29.15**	**27.00**	**35.27**	**29.61**
			SSIM ↑			
NV [30]	0.992	0.793	0.721	0.819	0.916	0.848
NeRF [35]	0.920	0.860	0.750	0.860	0.932	0.864
NSVF [29]	0.930	0.895	0.823	0.900	0.954	0.900
Point-NeRF (Ours)	**0.961**	**0.950**	**0.937**	**0.934**	**0.986**	**0.954**
			LPIPS_{Alex} ↓			
NV [30]	0.117	0.312	0.479	0.280	0.111	0.260
NeRF [35]	0.111	0.192	0.395	0.196	0.098	0.198
NSVF [29]	0.106	0.148	0.307	0.141	0.063	0.153
Point-NeRF (Ours)	**0.069**	**0.077**	**0.120**	**0.111**	**0.024**	**0.080**
			LPIPS_{Vgg} ↓			
Point-NeRF (Ours)	**0.079**	**0.117**	**0.180**	**0.156**	**0.046**	**0.115**

Table 9. Quantity comparison on five scenes in the Tanks and Temples dataset [23] selected in NSVF [29]. Our method Point-NeRF outperforms all state-of-the-art models in all metrics by substantial margins.

in the beginning. The confidence score of valid points will be pushed to 1 during the optimization process. To acquire point features f_i for a point, We first rule out all the views where the point is occluded by other points, then we find the view of which the camera is the closest to the point. Then from that view, we can unproject the point onto the feature maps extracted by G_f (see Figure 2(a) in the main paper) from the selected view and obtain the f_i.

G. Networks Architectures

Cost volume-based CNN $G_{p,\gamma}$. Our cost volume-based CNN adopts the popular architecture of [59], which is simple and efficient. It includes three layers of depth features extraction CNN, while the latter two layers down-samples the spatial dimension by 4 and output a feature map with 32 channels. Then, these features from each view will be warped according to camera pose and the variance will be computed. The variance features will go through a narrow U-Net [54] and output a 1-channel feature to calculate the depth probability.

Image Feature Extraction 2D CNN G_f. The image feature extraction network takes inputs of RGB image and has three down-sampling layers, each output feature with channels of $8, 16, 32$. We extract the point features by unprojecting a 3D point to each layer and taking the multi-scale features.

Point-based Radiance Fields MLP. We visualize the details of the point feature aggregation and radiance computa-

Figure 9. The qualitative results of our Point-NeRF on the Tanks and Temples dataset.

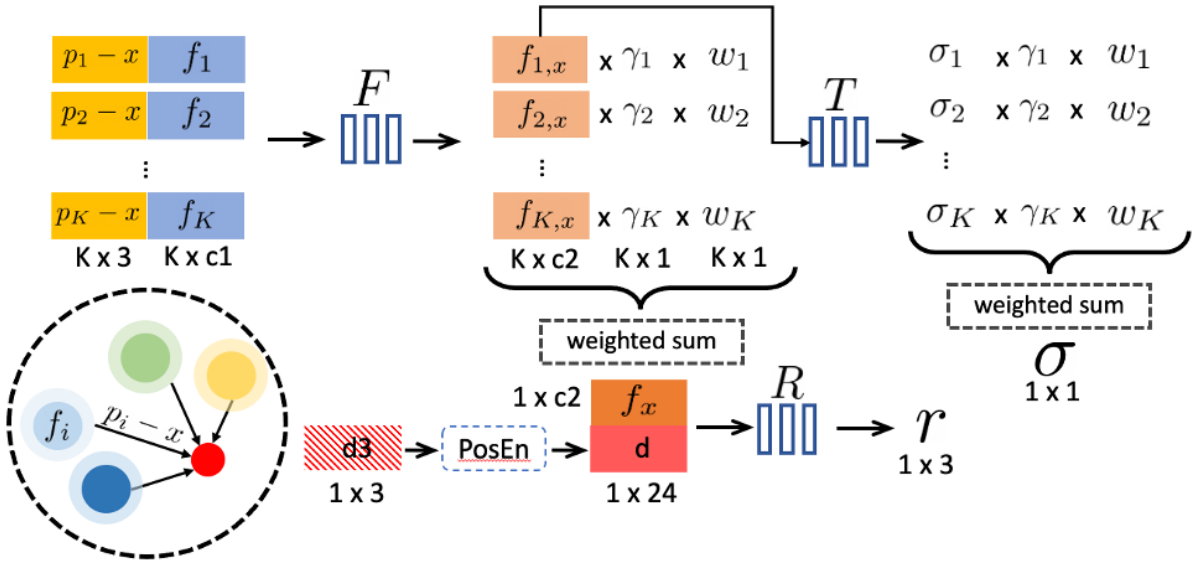


Figure 10. The network pipeline of radiance fields computation at a shading location x from K neural points neighbors. "PosEN" indicates positional encoding [35]. "d3" indicates the 3 channels vector of view directions at x. The final outputs are the radiance color r and density σ. Please also refer to the equations (3-7) in the main paper.

tion in Figure 10. In all of our experiments, we set $c_1 = 56$, $c2 = 128$. The MLPs F, R, T have 2, 3, 2 layers, respectively. The intermediate feature channels of F and T are 256, and 128 channels for R.

H. Neural Point Querying

To efficiently query neural point neighbors for ray marching, inspired by the CAGQ point query introduced in [58], we implement a grid query method. Then we build grid-point indices which register each neural point to evenly spaced 3D grids. Since these grids in the perspective coordinate are cubic, in the world coordinate, they have shapes of spherical voxels.

With the grid-point indices, we can discover grids that have neural points and also their grid neighbors. These grid neighbors are the regions of interest since there should exist neural points within the query radius. If a ray crosses these regions, we can place shading points inside. Finally, we query neural points by directly retrieving the stored neural points according to the grid-point indices.

In all of our experiments, we query 8 nearest neural point neighbors for each shading location. Along each ray, we only search for neural point neighbors and compute radiance for shading locations in a grid that is occupied itself or nearby occupied grids. Therefore, our shading is much more efficient by skipping the empty space, unlike other radiance fields representations. This is one key advantage that enables fast convergence. Even NSVF [29], high-performance local radiance representation, has to probe the empty space in the beginning and gradually prune the voxels along its training process.

The benefit of this strategy is two-fold: First, we only

place shading points in the area that exists neural points, so that we avoid radiance computation in the empty space. Second, the nearby points can be efficiently retrieved according to the indices, which substantially accelerate the point query speed.

I. Limitations

Because we do not focus on the rendering speed and we have not optimized our implementation (point querying and point feature aggregation) for fast rendering. Although, our model is naturally faster than NeRF (3X) due to that we skip the shading in empty space. We believe future works on combining mechanisms introduced in current papers such as [42, 62] with our point-based radiance representation would further benefit the neural rendering technology.

J. Additional Discussion and Issues Need Attention

Processing the points generated by MVSNet We have received constructive feedbacks and hope to make it clear that when Point-NeRF uses MVSNet [59] to reconstruct point cloud, the point fusion after depth estimation by MVSNet will use the alpha channel in the NeRF-Synthetic Dataset (as our published code indicates). It is due to the fact that MVSNet cannot handle background very well and will create too many outlier points in the background areas. Since images in the Tanks and Temples Dataset [23] don't have a alpha channel, we filter out the MVSNet points that appear in the regions of the pure background color. On the NeRF-Synthetic Dataset, the methods we compared with [29, 32], used the inputs: RGB images with the knowledge of the pure color background. Therefore, To improve the fairness, on the NeRF-Synthetic Dataset, we include results of Point-NeRF with MVSNet when using background color for filtering (not the alpha channel anymore). Its results is shown in Table 10 and one can cite which ever setting one thinks is fair.

Please note that, in our experiments, COLMAP doesn't use any filtering. Therefore, there is no impact on COLMAP results. When compare with NPGB [2], we use the same point cloud. Since it is more meaningful to rule out the impact of the point cloud quality, we advocate other point-based rendering works to use the same point cloud if willing to compare with our results. The point clouds are included in the checkpoints we published in the github repo.

Our original intention of using MVSNet is due to its simplicity and the fact that it is one of the earlies deep learning-based MVS model. We, thus, encourage users to try a more advanced MVS model so that no filtering is needed.

ScanNet and Unbounded Scenes We also receive comments about our ScanNet experiments, and we would like to state very clearly that we use the depth images from the ScanNet Dataset to initialize the point cloud. It is because NSVF is our major baseline on this dataset and it uses this setting. In our original paragraph Appendix D we have provided this information, and we hope this could clear the potential false expectation from readers.

Since Point-NeRF is a local radiance representation, without additional components, such as an additional background NeRF (used by Plenoxel [61]), it cannot handle background in Unbounded scenes (also known as inside-out scenes). For ScanNet, there is not much of background since it is a indoor scene with noisy depth images, every parts in the room can be deemed as foreground.

Point-NeRF with MVSNet (background color filtering) on NeRF Synthetic

	Chair	Drums	Lego	Mic	Materials	Ship	Hotdog	Ficus	Mean
PSNR ↑	35.60	26.04	35.27	35.91	29.65	30.61	37.34	35.61	33.25
SSIM ↑	0.991	0.954	0.989	0.994	0.971	0.938	0.991	0.992	0.978
LPIPS$_{Alex}$ ↓	0.023	0.078	0.021	0.014	0.071	0.129	0.036	0.025	0.050
LPIPS$_{Vgg}$ ↓	0.010	0.055	0.010	0.007	0.041	0.076	0.016	0.011	0.028

Table 10. We use MVSNet [59] to reconstruct the points and filter them by using background color, then initialize neural points and optimized our Point-NeRF model for 200 thousand iterations.