Depth-supervised NeRF: Fewer Views and Faster Training for Free

Kangle Deng[1] Andrew Liu[2] Jun-Yan Zhu[1] Deva Ramanan[1,3]
[1]Carnegie Mellon University [2]Google [3]Argo AI

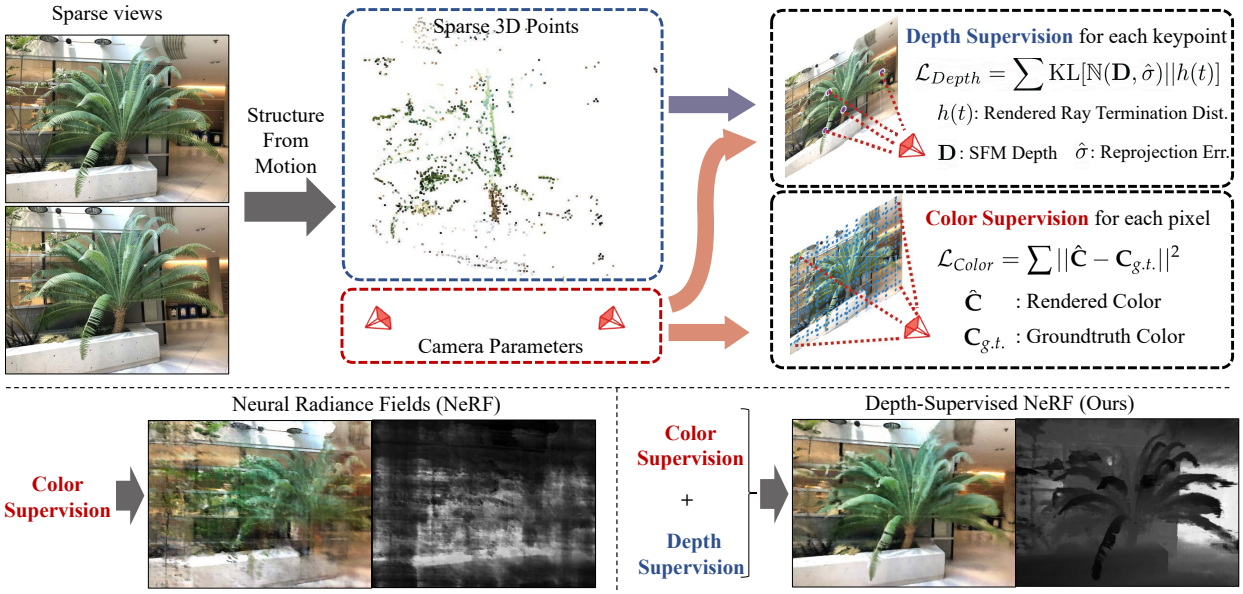

Figure 1. Training NeRFs can be difficult when given insufficient input images. We utilize additional supervision from depth recovered from 3D point clouds estimated from running structure-from-motion and impose a loss to ensure the rendered ray's termination distribution respects the surface priors given by the each keypoint. Because our supervision is complementary to NeRF, it can be combined with any such approach to reduce overfitting and speed up training.

Abstract

A commonly observed failure mode of Neural Radiance Field (NeRF) is fitting incorrect geometries when given an insufficient number of input views. One potential reason is that standard volumetric rendering does not enforce the constraint that most of a scene's geometry consist of empty space and opaque surfaces. We formalize the above assumption through DS-NeRF (Depth-supervised Neural Radiance Fields), a loss for learning radiance fields that takes advantage of readily-available depth supervision. We leverage the fact that current NeRF pipelines require images with known camera poses that are typically estimated by running structure-from-motion (SFM). Crucially, SFM also produces sparse 3D points that can be used as "free" depth supervision during training: we add a loss to encourage the distribution of a ray's terminating depth matches a given 3D keypoint, incorporating depth uncertainty. DS-NeRF can render better images given fewer training views while training 2-3x faster. Further, we show that our loss is compatible with other recently proposed NeRF methods, demonstrating that depth is a cheap and easily digestible supervisory signal. And finally, we find that DS-NeRF can support other types of depth supervision such as scanned depth sensors and RGB-D reconstruction outputs.

1. Introduction

Neural rendering with implicit representations has become a widely-used technique for solving many vision and graphics tasks ranging from view synthesis [5, 15, 25], to re-lighting [12, 13], to pose and shape estimation [17, 21, 31], to 3D-aware image synthesis and editing [3, 11, 23], to modeling dynamic scenes [9, 18, 19]. The seminal work of Neural Radiance Fields (NeRF) [15] demonstrated impressive view synthesis results by using implicit functions to encode volumetric density and color observations.

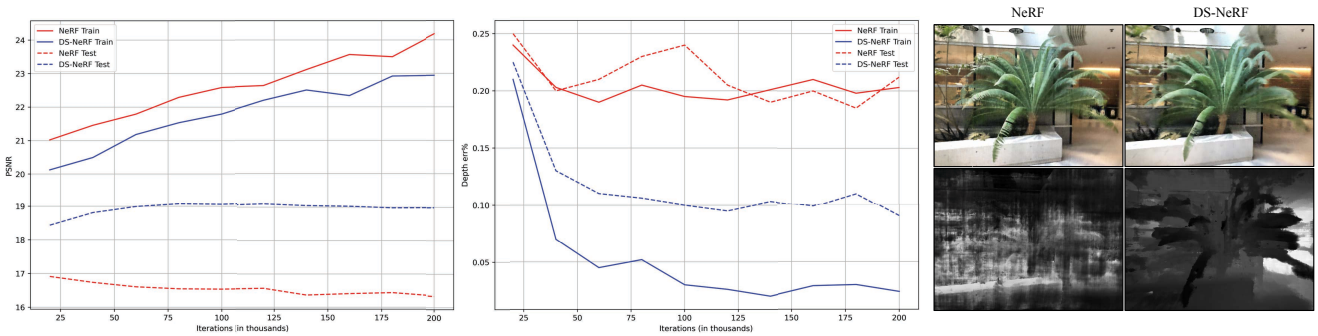

Figure 2. **Few view NeRF.** NeRF is susceptible to overfitting when given few training views. As seen by the PSNR gap between train and test renders (left), NeRF has overfit and fails at synthesizing novel views. Further, the depth map (right) and depth error (middle) for NeRF suggest that its density function has failed to extract the surface geometry and can only reconstruct the training views' colors. Our depth-supervised NeRF model is able to render plausible geometry with consistently lower depth errors.

In spite of this, NeRF has several limitations. Reconstructing both the scene appearance and geometry can be ill-posed given a small number of input views. Figure 2 shows that NeRF can learn wildly inaccurate scene geometries that still accurately render train-views. However, such models produce poor renderings of novel test-views, essentially overfitting to the train set. Furthermore, even given a large number of input views, NeRF can still be time-consuming to train; it often takes between ten hours to several days to model a single scene at moderate resolutions on a single GPU. The training is slow due to both the expensive ray-casting operations and lengthy optimization process.

In this work, we explore depth as an additional, cheap source of supervision to guide the geometry learned by NeRF. Typical NeRF pipelines require images and camera poses, where the latter are estimated from structure-from-motion (SFM) solvers such as COLMAP [22]. In addition to returning cameras, COLMAP also outputs sparse 3D point clouds as well as their reprojection errors. We impose a loss to encourage the distribution of a ray's termination to match the 3D keypoint, incorporating reprojection error as an uncertainty measure. This is a significantly stronger signal than reconstructing only RGB. Without depth supervision, NeRF is implicitly solving a 3D correspondence problem between multiple views. However, the sparse version of this exact problem has already been solved by SFM, whose solution is given by the sparse 3D keypoints. Therefore depth supervision improves NeRF by (softly) anchoring its search over implicit correspondences with sparse explicit ones.

Our experiments show that this simple idea translates to massive improvements in training NeRFs and its variations, regarding both the training speed and the amount of training data needed. We observe that depth-supervised NeRF can accelerate model training by 2-3x while producing results with the same quality. For sparse view settings, experiments show that our method synthesizes better results compared to the original NeRF and recent sparse-views NeRF models [26, 33] on both NeRF Real [15] and Redwood-3dscan [6] We also show that our depth supervision loss works well with depth derived from other sources such as a depth camera. Our code and more results are available at `https://www.cs.cmu.edu/~dsnerf/`. Check the full version of the paper at `https://arxiv.org/abs/2107.02791`.

2. Related Work

NeRF from few views. NeRF [15] was originally shown to work on a large number of images with the LLFF NeRF Real dataset [14] consisting of nearly 50 images per scene. This is because fitting the NeRF volume often requires a large number of views to avoid arriving at degenerate representations. Recent works have sought to decrease the data-hungriness of NeRF in a variety of different ways. PixelNeRF [33] and metaNeRF [26] use data-driven priors recovered from a domain of training scenes to fill in missing information from test scenes. Such an approach works well when given sufficient training scenes and limited gap between the training and test distribution, but such assumptions are not particularly flexible. Another approach is to leverage priors recovered from a different task like semantic consistency [7] or depth prediction [30].

Similar to our insight that the primary difficulty in fitting few view NeRF is correctly modeling 3D geometry, MVS-NeRF [4] combines both 3D knowledge with scene priors by constructing a plane sweep volume before using a pretrained network with generalizable priors to render scenes. One appeal of an approach that utilizes 3D information is the lack of assumption it makes on the problem statement. Unlike the aforementioned approaches which depend on the availability of training data or the applicability of prior assumptions, our approach only requires the existence of 3D keypoints. This gives depth supervision the flexibility to be used not only as a standalone method, but one that can be freely incorporated into existing NeRF methods easily.

Faster NeRF. Another drawback of NeRF is the lengthy optimization time required to fit the volumetric representation. Indeed Mildenhall *et al*. [15] trained a single scene's NeRF model for twelve hours of GPU compute. Many works [20, 32] have found that the limiting factor is not learning the radiance itself, but rather oversampling the empty space during training. Indeed this is a similar intuition to the fact that the majority of the volume is actually empty, but NeRF's initialization is a median uniform density. Our insight is to apply a supervisory signal directly to the NeRF density to increase the convergence of the geometry and to encourage NeRF's density function to mimic the behavior of real world surface geometries.

Depth and NeRF. Several prior works have explored ways to leverage depth information for view synthesis [24, 27] and NeRF training [9, 10, 16, 18, 30]. For instance, 3D keypoints have been demonstrated to be helpful when extending NeRFs with relaxed assumptions like deformable surfaces [18] or dynamic scene flows [9]. Other works like DONeRF [16] proposed training a depth oracle to improve rendering speed by directly smartly sampling the surface of a NeRF density function. Similar to DONeRF, NerfingMVS [30] shows how a monocular depth network can be used to induce depth priors to do smarter sampling during training and inference.

Our work attempts to improve NeRF-based methods by directly supervising the NeRF density function. As depth becomes a more accessible source of data, being able to apply depth supervision becomes increasingly more powerful. For example, recent works have demonstrated how depth extracted from sensors like time-of-flight cameras [1] or RGB-D Kinect sensor [2] can be applied to fit implicit functions. Building upon their insights, we provide a probabilistic formulation of the depth supervision, and show this results in meaningful improvements to NeRF and its variants.

3. Depth-Supervised Ray Termination

We now present our proposed depth-supervised loss for training NeRFs. We first revisit volumetric rendering and then analyze the termination distribution for rays. We conclude with our depth-supervised distribution loss.

3.1. Volumetric rendering revisited

A Neural Radiance Field takes a set of posed images and encodes a scene as a volume density and emitted radiance. More specifically, for a given 3D point $\mathbf{x} \in \mathbb{R}^3$ and a particular viewing direction $\mathbf{d} \in \mathbb{R}^3$, NeRF learns an implicit function f that estimates the differential density σ and RGB color $\mathbf{c}$ like so: $f(\mathbf{x}, \mathbf{d}) = (\sigma, \mathbf{c})$.

To render a 2D image given a pose $\mathbf{P}$, we cast rays $\mathbf{r}$ originating from the $\mathbf{P}$'s center of projection $\mathbf{o}$ in direction $\mathbf{d}$ derived from its intrinsics. We integrate the implicit radiance field along this ray to compute the incoming radiance from any object that lies along $\mathbf{d}$:

$$\hat{\mathbf{C}} = \int_0^\infty T(t)\sigma(t)\mathbf{c}(t)dt, \tag{1}$$

where t parameterizes the aforementioned ray as $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$ and $T(t) = \exp(-\int_0^t \sigma(s)ds)$ checks for occlusions by integrating the differential density between 0 to t. Because the density and radiance are the outputs of neural networks, NeRF methods approximate this integral using a sampling-based Riemann sum instead. The final NeRF rendering loss is given by a reconstruction loss over colors returned from rendering the set of rays $\mathcal{R}(\mathbf{P})$ produced by a particular camera parameter $\mathbf{P}$.

$$\mathcal{L}_{\text{Color}} = \mathbb{E}_{\mathbf{r}\in\mathcal{R}(\mathbf{P})} \left\| \hat{\mathbf{C}}(\mathbf{r}) - \mathbf{C}(\mathbf{r}) \right\|_2^2. \tag{2}$$

Ray distribution. Let us write $h(t) = T(t)\sigma(t)$. In the arXiv version, we show that it is a continuous probability distribution over ray distance t that describes the likelihood of a ray terminating at t. Due to practical constraints, NeRFs assume that the scene lies between a near and far bound (t_n, t_f). To ensure $h(t)$ sums to one, NeRF implementations often treat t_f as an opaque wall. With this definition, the rendered color can be written as an expectation:

$$\hat{\mathbf{C}} = \int_0^\infty h(t)\mathbf{c}(t)dt = \mathbb{E}_{h(t)}[\mathbf{c}(t)].$$

Idealized distribution. The distribution $h(t)$ describes the weighed contribution of sampled radiances along a ray to the final rendered value. Most scenes consist of empty spaces and opaque surfaces that restrict the weighted contribution to stem from the closest surface. This implies that the ideal ray distribution of image point with a closest-surface depth of $\mathbf{D}$ should be $\delta(t - \mathbf{D})$. Figure 3(c) shows that the empirical variance of NeRF termination distributions decreases with more training views, suggesting that high quality NeRFs (trained with many views) tend to have ray distributions that approach the δ-function. This insight motivates our depth-supervised ray termination loss.

3.2. Deriving depth-supervision

Recall that most NeRF pipelines require images with associated camera matrices $(\mathbf{P}_1, \mathbf{P}_2, \ldots)$, often estimated with SFM packages such as COLMAP [22]. Importantly, SFM makes use of bundle adjustment, which also returns 3D keypoints $\{\mathbf{X} : \mathbf{x}_1, \mathbf{x}_2, \ldots \in \mathbb{R}^3\}$ and visibility flags for which keypoints are seen from camera j: $\mathbf{X}_j \subset \mathbf{X}$. Given image I_j and its camera $\mathbf{P}_j$, we estimate the depth of visible keypoints $\mathbf{x}_i \in \mathbf{X}_j$ by simply projecting $\mathbf{x}_i$ with $\mathbf{P}_j$, taking the re-projected z value as the keypoint's depth $\mathbf{D}_{ij}$.

Depth uncertainty. Unsurprisingly $\mathbf{D}_{ij}$ are inherently noisy estimates due to spurious correspondences, noisy camera

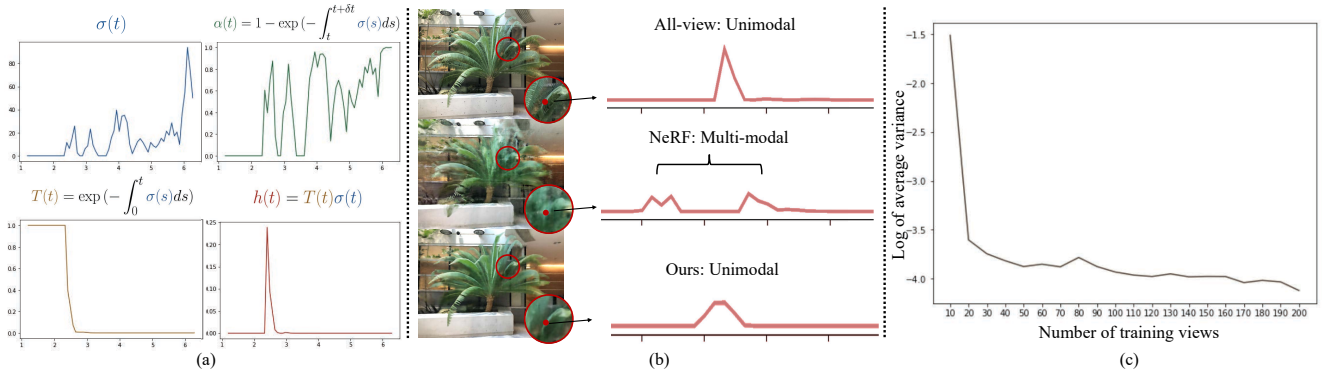

Figure 3. **Ray Termination Distribution.** (a) We plot various NeRF components over the distance traveled by the ray. Even if a ray traverses through multiple objects (as indicated by the multiple peaks of density $\sigma(t)$), we find that the ray termination distribution $h(t)$ is still unimodal. We find that NeRF models trained with sufficient supervision tend to have peaky, unimodal ray termination distributions as seen by the decreasing variance with more views in (c). We posit that the ideal ray termination distribution approaches a δ impulse function.

parameters, or poor COLMAP optimization. The reliability of a particular keypoint $\mathbf{x}_i$ can be measured using the average reprojection error $\hat{\sigma}_i$ across views over which the keypoint was detected. Specifically, we model the location of the first surface encountered by a ray as a random variable $\mathbb{D}_{ij}$ that is normally distributed around the COLMAP-estimated depth $\mathbf{D}_{ij}$ with variance $\hat{\sigma}_i$: $\mathbb{D}_{ij} \sim \mathbb{N}(\mathbf{D}_{ij}, \hat{\sigma}_i)$. Combining the intuition regarding behavior of ideal termination distribution, our objective is to minimize the KL divergence between the rendered ray distribution $h_{ij}(t)$ of $\mathbf{x}_i$'s image coordinates and the noisy depth distribution:

$$\mathbb{E}_{\mathbb{D}_{ij}} \mathrm{KL}[\delta(t - \mathbb{D}_{ij}) || h_{ij}(t)] = \mathrm{KL}[\mathbb{N}(\mathbf{D}_{ij}, \hat{\sigma}_i) || h_{ij}(t)] + const.$$

Ray distribution loss. The above equivalence (see our arXiv version for proof) allows the termination distribution $h(t)$ to be trained with probabilisitic COLMAP depth supervision:

$$\begin{aligned} \mathcal{L}_{Depth} &= \mathbb{E}_{x_i \in X_j} \int \log h(t) \exp\left(-\frac{(t - \mathbf{D}_{ij})^2}{2\hat{\sigma}_i^2}\right) dt \\ &\approx \mathbb{E}_{x_i \in X_j} \sum_k \log h_k \exp\left(-\frac{(t_k - \mathbf{D}_{ij})^2}{2\hat{\sigma}_i^2}\right) \Delta t_k. \end{aligned}$$

Our overall training loss for NeRF is $\mathcal{L} = \mathcal{L}_{\text{Color}} + \lambda_D \mathcal{L}_{\text{Depth}}$ where λ_D is a hyper-parameter balancing color and depth supervision.

4. Experiments

We first evaluate the input data efficiency on view synthesis over several datasets in Section 4.3. For relevant NeRF-related methods, we also evaluate the error of rendered depth maps in Section 4.4. Finally, we analyze training speed improvements in Section 4.5.

4.1. Datasets

DTU MVS Dataset (DTU) [8] captures various objects from multiple viewpoints. Following Yu *et al.*'s setup in PixelNeRF [33], we evaluated on the same test scenes and views. For each scene, we used their subsets of size 3, 6, 9 training views. We run COLMAP with the ground truth calibrated camera poses to get keypoints. Images are down-sampled to a resolution of 400×300 for training and evaluation.

NeRF Real-world Data (NeRF Real) [14, 15] contains 8 real world scenes captured from many forward-facing views. We create subsets of training images for each scene of sizes 2, 5, and 10 views. For every subset, we run COLMAP [22] over its training images to estimate cameras and collect sparse keypoints for depth supervision.

Redwood-3dscan (Redwood) [6] contains RGB-D videos of various objects. We select 5 RGB-D sequences and create subsets of 2, 5, and 10 training frames for each object. We run COLMAP to get their camera poses and sparse point clouds. To connect the scale of COLMAP's pose with the scanned depth, we solve a least-squares that best fits detected keypoints to the scanned depth value. Please see our arXiv version for full details.

4.2. Comparisons

First we consider Local Lightfield Fusion (*LLFF*) [14], an MPI-based representation that learns from multiple view points. Next we consider a set of NeRF baselines.

PixelNeRF [33] expands upon NeRF by using an encoder to train a general model across multiple scenes. *pixelNeRF-DTU* is evaluated using the released DTU checkpoint. For cases where the train and test domain are different, we fine-tune using RGB supervision for additional iterations on each test scene to get *pixelNeRF finetuned*.

MetaNeRF [26] finds a better NeRF initialization over a domain of training scenes before running test-time optimization on new scenes. Because DTU is the only dataset large enough for meta-learning, we only consider the *metaNeRF-DTU* baseline which learns an initialization over DTU for 40K meta-iterations and then finetunes for 1000 steps on

NeRF Real [14]	PSNR↑ 2-view	PSNR↑ 5-view	PSNR↑ 10-view	SSIM↑ 2-view	SSIM↑ 5-view	SSIM↑ 10-view	LPIPS↓ 2-view	LPIPS↓ 5-view	LPIPS↓ 10-view
LLFF	14.3	17.6	22.3	0.48	0.49	0.53	0.55	0.51	0.53
NeRF	13.5	18.2	22.5	0.39	0.57	0.67	0.56	0.50	0.52
metaNeRF-DTU	13.1	13.8	14.3	0.43	0.45	0.46	0.89	0.88	0.87
pixelNeRF-DTU	9.6	9.5	9.7	0.39	0.40	0.40	0.82	0.87	0.81
finetuned	18.2	22.0	24.1	0.56	0.59	0.63	0.53	0.53	0.41
finetuned w/ DS	18.9	22.1	24.4	0.54	0.61	0.66	0.55	0.47	0.42
IBRNet	14.4	21.8	24.3	0.50	0.51	0.54	0.53	0.54	0.51
finetuned w/ DS	19.3	22.3	24.5	0.63	0.66	0.68	**0.39**	0.36	0.38
MVSNeRF	-	17.2	17.2	-	0.61	0.60	-	0.37	0.36
fintuned	-	21.8	22.9	-	**0.70**	0.74	-	**0.27**	**0.23**
fintuned w/ DS	-	22.0	22.9	-	**0.70**	**0.75**	-	0.27	0.25
DS-NeRF									
MSE	19.5	22.2	24.7	0.65	0.69	0.71	0.43	0.40	0.37
KL divergence	**20.2**	**22.6**	**24.9**	**0.67**	0.69	0.72	**0.39**	0.35	0.34

Table 1. **View Synthesis on NeRF Real.** We evaluate view synthesis quality for various methods when given 2, 5, 10 views from NeRF Real. We find that metaNeRF-DTU and pixelNeRF-DTU struggle to learn on NeRF Real due to its domain gap to DTU. PixelNeRF, IBRNet and MVSNeRF can benefit from incorporating the depth supervision loss to achieve their best performance. We find that our DS-NeRF outperforms these methods on a variety of metrics, but especially for the few view settings like 2 and 5 views.

DTU [8]	PSNR↑ 3-view	PSNR↑ 6-view	PSNR↑ 9-view	SSIM↑ 3-view	SSIM↑ 6-view	SSIM↑ 9-view	LPIPS↓ 3-view	LPIPS↓ 6-view	LPIPS↓ 9-view
NeRF	9.9	18.6	22.1	0.37	0.72	**0.82**	0.62	0.35	0.26
metaNeRF-DTU	18.2	18.8	20.2	0.60	0.61	0.67	0.40	0.41	0.35
pixelNeRF-DTU	**19.3**	20.4	21.1	**0.70**	0.73	0.76	**0.39**	0.36	0.34
DS-NeRF									
MSE	16.5	20.5	22.2	0.54	0.73	0.77	0.48	0.31	0.26
KL divergence	16.9	**20.6**	**22.3**	0.57	**0.75**	0.81	0.45	**0.29**	**0.24**

Table 2. **View Synthesis on DTU.** We evaluate on 3, 6, and 9 views respectively for 15 test scenes from the DTU dataset. pixelNeRF-DTU and metaNeRF-DTU perform well given that the domain overlap between training and testing. This is especially true for the few view setting as the lack of information is supplemented by exploiting dataset priors. In spite of this, DS-NeRF is still competitive on view synthesis for 6 and 9 views.

Redwood-3dscan [6]	PSNR↑ 2-view	PSNR↑ 5-view	PSNR↑ 10-view	SSIM↑ 2-view	SSIM↑ 5-view	SSIM↑ 10-view	LPIPS↓ 2-view	LPIPS↓ 5-view	LPIPS↓ 10-view
NeRF	10.5	22.4	23.4	0.38	0.75	0.82	0.51	0.45	0.45
metaNeRF-DTU	14.3	14.6	15.1	0.37	0.39	0.40	0.76	0.76	0.75
pixelNeRF-DTU	12.7	12.9	12.8	0.43	0.47	0.50	0.76	0.75	0.70
MVSNeRF-DTU	-	17.1	17.1	-	0.54	0.53	-	0.63	0.63
finetuned	-	22.7	23.1	-	0.78	0.78	-	0.36	0.34
DS-NeRF	18.1	22.9	23.8	0.62	**0.78**	0.81	0.40	**0.34**	0.42
DS-NeRF w/ RGB-D	**20.3**	**23.4**	**23.9**	**0.73**	0.77	**0.84**	**0.36**	0.35	**0.28**

Table 3. **View Synthesis on Redwood.** We evaluate view synthesis on 2, 5, and 10 input views on the Redwood dataset. DS-NeRF (with COLMAP [22] inputs) outperforms baselines on various metrics across varying numbers of views. Learning DS-NeRF with the RGB-D reconstruction output [34] further improves performance, highlighting the potential of applying our method alongside other sources of depth.

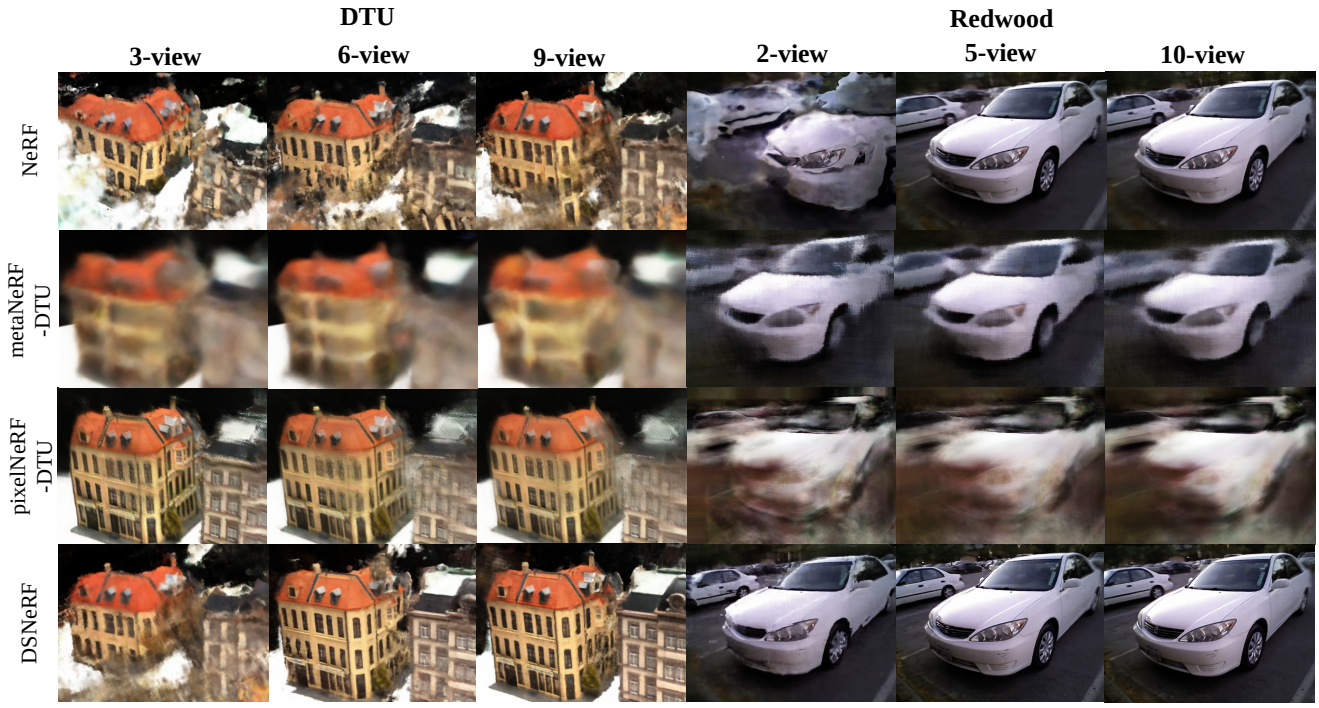

Figure 4. **View Synthesis on DTU and Redwood.** PixelNeRF, which is pre-trained on DTU, performs the best when given 3-views, although we find DS-NeRF to be visually competitive when more views are available. On Redwood, DS-NeRF is the only baseline to perform well on the 2-views setting.

new scenes. We follow metaNeRF's ShapeNet experiments to demonstrate its susceptibility to differences between training and testing domains.

IBRNet [28] extends NeRF by using a MLP and ray transformer to estimate radiance and volume density.

MVSNeRF [4] initializes a plane sweep volume from 3 views before converting it to a NeRF by a pretrained network. MVSNeRF can be further optimized using RGB supervision.

DS-NeRF (Ours). To illustrate the effectiveness of KL divergence, we include a variant of DS-NeRF with an MSE loss between the SFM-estimated and the rendered depth. Figure 6 qualitatively shows that KL divergence penalty produces views with less artifacts on NeRF Real sequences.

DS with existing methods. As our DS loss does not require additional annotation or assumptions, our loss can be inserted into many NeRF-based methods. Here, we also incorporate our loss when finetuning pixelNeRF and IBRNet.

4.3. Few-input view synthesis

We start by comparing each method on rendering test views from few inputs. For view synthesis, we report three metrics (PSNR, SSIM [29], and LPIPS [35]) that evaluate the quality of rendered views against a ground truth.

DTU. We show evaluations on DTU in Table 2 and qualitative results in Figure 4. We find that DS-NeRF renders images from 6 and 9 input views that are competitive with pixelNeRF-DTU, however metaNeRF-DTU and pixelNeRF-DTU are able to outperform DS-NeRF on 3-views. This is not particularly surprising as both methods are trained on DTU scenes and therefore can fully leverage dataset priors.

NeRF Real. As seen in Table 1, our approach renders images with better scores than than NeRF and LLFF, especially when only two and fives input views are available. We also find that metaNeRF-DTU and pixelNeRF struggle which highlights their apparent weakness. These DTU-pretrained models struggle to perform well outside of DTU. Our full approach is capable of achieving good rendering results because we do not utilize assumptions on the test scene's structure. We also add our depth supervision loss to other methods like pixelNeRF and IBRNet and find their performances improve, showing that many methods can benefit from adding depth supervision. MVSNeRF has an existing geometry prior handled by its PSV-initialization, thus we did not see an improvement from adding depth supervision.

Redwood. Like NeRF Real, we find similar improvements in performance across the Redwood dataset in Table 3. Because Redwood includes depth measurements collected with a sensor, we also consider how alternative sources of depth supervision can improve results. We train DS-NeRF, replacing COLMAP supervision with the scaled Redwood depth measurements and find that the denser depth helps even

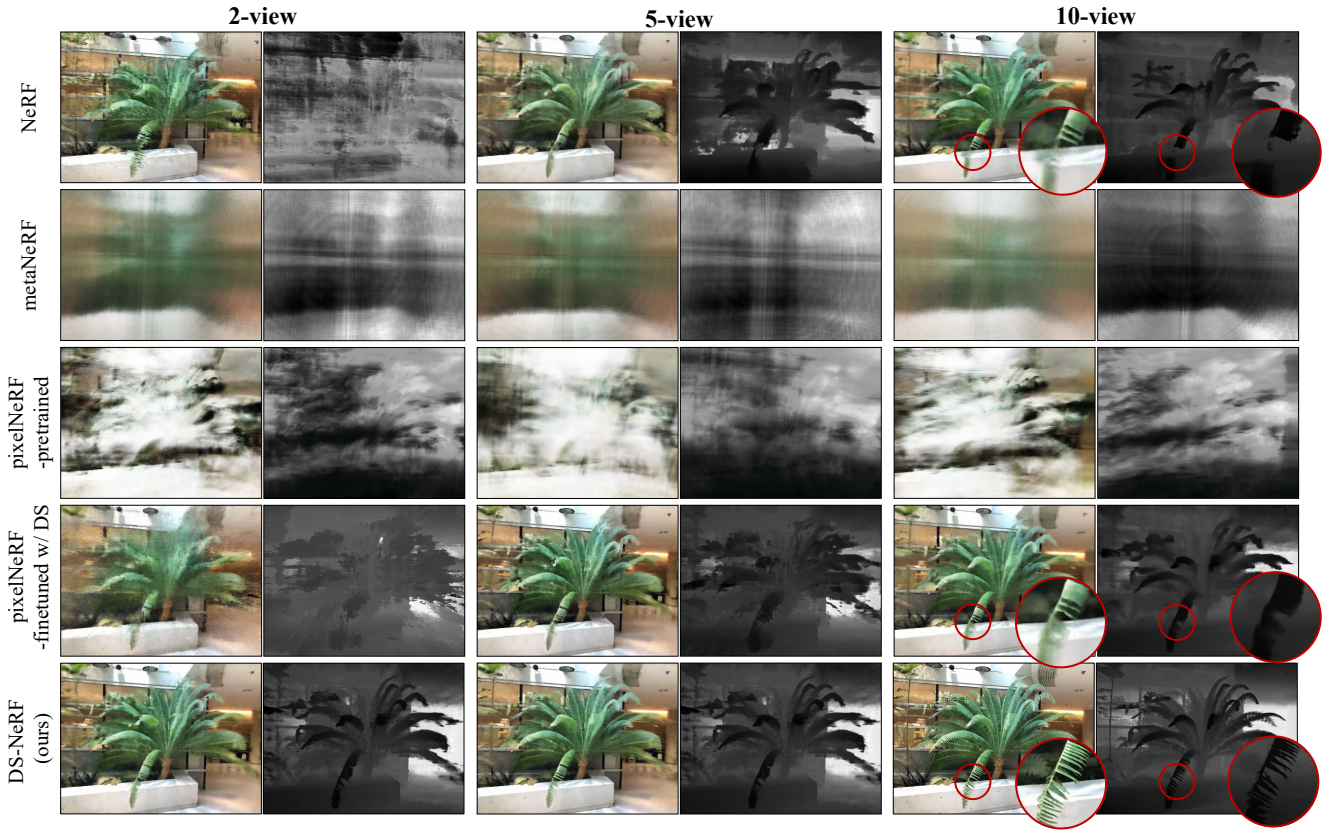

Figure 5. **Qualitative Comparison on NeRF Real.** We render novel views and depth for various NeRF models trained on 2, 5, and 10 views. We find that methods trained with DTU struggle on NeRF Real while methods that use depth-supervision are able to render test views with realistic depth maps, even when only 2 views are provided. Please refer to Table 1 for quantitative comparisons.

Depth err%↓	NeRF real-world			Redwood-3dscan		
	2-view	5-view	10-view	2-view	5-view	10-view
NeRF	20.32	15.00	12.41	25.32	24.34	21.34
metaNeRF-DTU	22.23	22.07	22.30	20.84	21.12	20.96
pixelNeRF-DTU	22.12	22.09	22.06	19.46	19.87	19.54
DS-NeRF	**10.41**	**8.61**	**8.15**	11.42	10.43	9.43
DS-NeRF w/ RGBD	-	-	-	**5.81**	**5.31**	**4.22**

Table 4. **Depth Error.** We compare rendered depth to reference "ground-truth" depth obtained from NeRF Real and Redwood RGB-D. DS-NeRF is able to extract better geometry as indicated by the lower depth errors from test views. We also show DS-NeRF trained with Redwood's dense supervision can significantly improve NeRF's ability to model the underlying geometry.

more, achieving a PSNR of 20.3 on 2-views.

4.4. Depth error

We evaluate NeRF's rendered depth by comparing them to reference "ground truth" depth measurements. For NeRF Real, we use reference depth of test keypoints recovered from running an all-view dense stereo reconstruction. For Redwood [6], we align their released 3D models with our cameras by running 3dMatch [34] and generate reference depths for each test view. Please refer to our arXiv version for more details regarding depth error evaluation. As shown in Table 4, DS-NeRF, trained with supervision obtained only from depth in training views, is able to estimate depth more accurately than all the other NeRF models. While this is not particularly surprising, it does highlight the weakness of training NeRFs only using RGB supervision. For example, in Figure 5, NeRF tends to ignore geometry and fails to produce any coherent depth map.

RBG-D inputs. We consider a variant of depth supervision using RGB-D input from Redwood. We derive dense depth map for each training view using 3DMatch [34] with RGB-D input. With dense depth supervision, we can render rays for any pixel in the valid region, and apply our KL depth-supervision loss. As shown in Table 3 and Table 4, dense depth supervision produces even better-quality images and significantly lower depth errors.

4.5. Analysis

Overfitting. Figure 2 shows that NeRF can overfit to a small number of input views by learning degenerate 3D geometries. Adding depth supervision can assist NeRF to disambiguate geometry and render better novel views.

Faster Training. To quantify speed improvements in NeRF

Figure 6. **Depth Supervision Ablations.** We render novel views for NeRF and DS-NeRF trained on 2 views and 5 views. NeRF fails to render novel views as evident by the many artifacts. Using MSE between rendered and sparse depth improves results slightly, but with KL Divergence, DS-NeRF is able to render images with the fewest artifacts.

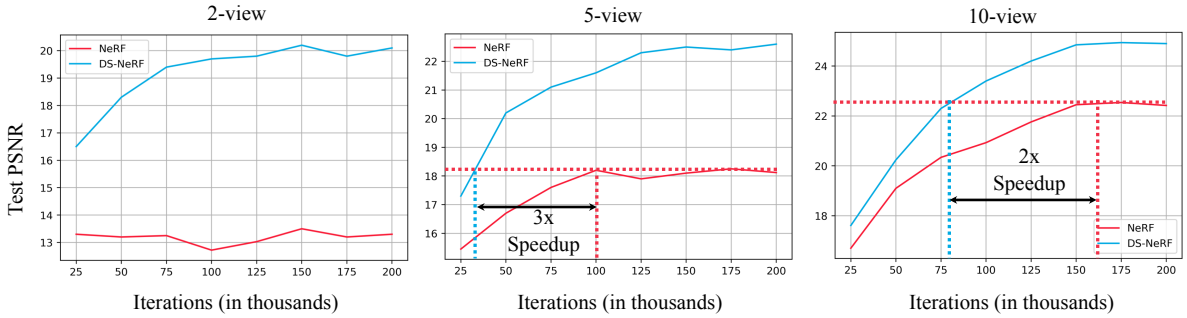

Figure 7. **Faster Training.** We train DS-NeRF and NeRF under identical conditions and observe that DS-NeRF is able to reach NeRF's peak PSNR quality in a fraction of the number of iterations across. For 2 views, we find that NeRF is unable to match DS-NeRF's performance.

training, we compare training DS-NeRF and NeRF under identical settings. Like in Section 4.3, we evaluate view synthesis quality on test views under various number of input views from NeRF Real using PSNR. We can compare training speed performance by plotting PSNR on test views versus training iterations in Figure 7.

DS-NeRF achieves a particular test PSNR threshold using 2-3x less training iterations than NeRF. These benefits are significantly magnified when given fewer views. In the extreme case of only 2-views, NeRF is completely unable to match DS-NeRF's performance. While these results are given in terms of training iteration, we can translate them into wall time improvements. On a single RTX A5000, a training loop of DS-NeRF takes $\sim$ 362.4 ms/iter while NeRF needs $\sim$ 359.8 ms/iter. Thus in the 5-view case, DS-NeRF achieves NeRF's peak test PSNR around 13 hours faster, a massive improvement considering the negligible cost.

Discussion. We introduce Depth-supervised NeRF, a model for learning neural radiance fields that takes advantage of depth supervision. Our model uses "free" supervision provided by sparse 3D point clouds computed during standard SFM pre-processing steps. This additional supervision has a significant impact; DS-NeRF trains 2-3x faster and produces better results from fewer training views (improving PSNR from 13.5 to 20.2). While recent research has sought to improve NeRF by exploiting priors learned from category-specific training data, our approach requires no training and thus generalizes (in principle) to any scenes on which SFM succeeds. This allows us to integrate depth supervision to many NeRF-based methods and observe significant benefits. Finally, we provide cursory experiments that explore alternate forms of depth supervision such as active depth sensors. Please see our arXiv version for a discussion on limitations and societal impact of our paper.

Acknowledgments. We thank Takuya Narihira, Akio Hayakawa, Sheng-Yu Wang, Richard Tucker, Konstantinos Rematas, and Michaelu Zollhöfer for helpful discussion. We are grateful for the support from Sony Corporation, Singapore DSTA, and the CMU Argo AI Center for Autonomous Vehicle Research.

References

[1] Benjamin Attal, Eliot Laidlaw, Aaron Gokaslan, Changil Kim, Christian Richardt, James Tompkin, and Matthew O'Toole. Törf: Time-of-flight radiance fields for dynamic scene view synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 3

[2] Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural rgb-d surface reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[3] 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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1

[4] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 2, 6

[5] Julian Chibane, Aayush Bansal, Verica Lazova, and Gerard Pons-Moll. Stereo radiance fields (srf): Learning view synthesis from sparse views of novel scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2021. 1

[6] Sungjoon Choi, Qian-Yi Zhou, Stephen Miller, and Vladlen Koltun. A large dataset of object scans. *arXiv:1602.02481*, 2016. 2, 4, 5, 7

[7] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 2

[8] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 4, 5

[9] 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 (CVPR)*, 2021. 1, 3

[10] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3

[11] Steven Liu, Xiuming Zhang, Zhoutong Zhang, Richard Zhang, Jun-Yan Zhu, and Bryan Russell. Editing conditional radiance fields. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 1

[12] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1

[13] Moustafa Meshry, Dan B. Goldman, Sameh Khamis, Hugues Hoppe, Rohit Pandey, Noah Snavely, and Ricardo Martin-Brualla. Neural rerendering in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[14] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 2019. 2, 4, 5

[15] 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 (ECCV)*, 2020. 1, 2, 3, 4

[16] Thomas Neff, Pascal Stadlbauer, Mathias Parger, Andreas Kurz, Joerg H. Mueller, Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, and Markus Steinberger. DONeRF: Towards Real-Time Rendering of Compact Neural Radiance Fields using Depth Oracle Networks. *Computer Graphics Forum*, 40(4), 2021. 3

[17] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[18] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Deformable neural radiance fields. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 1, 3

[19] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1

[20] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 3

[21] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 1

[22] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 3, 4, 5

[23] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 1

[24] Meng-Li Shih, Shih-Yang Su, Johannes Kopf, and Jia-Bin Huang. 3d photography using context-aware layered depth inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[25] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 1

[26] Matthew Tancik, Ben Mildenhall, Terrance Wang, Divi Schmidt, Pratul P. Srinivasan, Jonathan T. Barron, and Ren Ng. Learned initializations for optimizing coordinate-based neural representations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 4

[27] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[28] 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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 6

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

[30] 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 *IEEE International Conference on Computer Vision (ICCV)*, 2021. 2, 3

[31] Lin Yen-Chen, Pete Florence, Jonathan T. Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. iNeRF: Inverting neural radiance fields for pose estimation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021. 1

[32] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. PlenOctrees for real-time rendering of neural radiance fields. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 3

[33] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 4

[34] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 5, 7

[35] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 6