NeRFuser: Large-Scale Scene Representation by NeRF Fusion

Jiading Fang*,1, Shengjie Lin*,1, Igor Vasiljevic[2], Vitor Guizilini[2],
Rares Ambrus[2], Adrien Gaidon[2], Gregory Shakhnarovich[1], Matthew R. Walter[1]

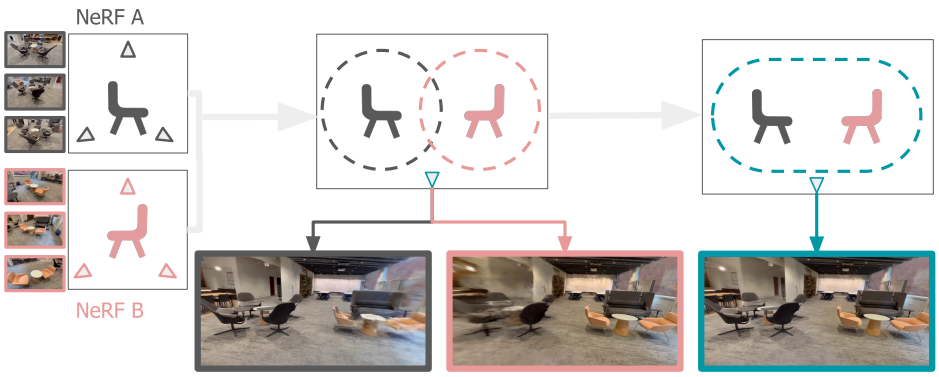

Figure 1: **NeRFuser.** Starting from separately constructed input NeRFs, NeRF A and NeRF B, we render images at novel viewpoints, including those not well-covered by either input NeRF. Our proposed *registration through re-rendering* followed by a novel blending procedure leads to a higher quality result (lower right) than renders from the original NeRFs (A or B).

Abstract

A practical benefit of implicit visual representations like Neural Radiance Fields (NeRFs) is their memory efficiency: large scenes can be efficiently stored and shared as small neural nets instead of collections of images. However, operating on these implicit visual data structures requires extending classical image-based vision techniques (e.g., registration, blending) from image sets to neural fields. Towards this goal, we propose NeRFuser, a novel architecture for NeRF registration and blending that assumes only access to pre-generated NeRFs, and not the potentially large sets of images used to generate them. We propose registration from re-rendering, a technique to infer the transformation between NeRFs based on images synthesized from individual NeRFs. For blending, we propose sample-based inverse distance weighting to blend visual information at the ray-sample level. We evaluate NeRFuser on public benchmarks and a self-collected object-centric indoor dataset, showing the robustness of our method, including to views that are challenging to render from the individual source NeRFs.

Code available at https://github.com/ripl/nerfuser
*Equal contribution.
[1]Toyota Technological Institute at Chicago, Chicago, IL.
[2]Toyota Research Institute, Los Altos, CA.

1. Introduction

The transmission of visual data relies on efficient video and image encoding and decoding, technologies with

decades of development behind them. Most computer vision methods and tools are specialized to this type of data—methods that align and blend images are ubiquitous [32] and fundamentally designed to work on explicit 2D representations of images.

However, a new learning-based representation for visual data has emerged in recent years: *neural fields* [39]. Pioneered by neural radiance fields (NeRFs) [16], these implicit representations allow for efficient visual compression and impressive view synthesis, generating a potentially infinite set of possible views from a fixed set of training images. Despite the promise of this representation as a storage and communication format, there is a lack of tools that *treat NeRFs as data*, much like common image processing tools treat images.

Towards expanding the utility of NeRFs as a data representation, we propose NeRFuser (Fig. 1), a NeRF fusion framework for the registration and blending of pre-trained NeRFs. Treating input NeRFs as black boxes (i.e., raw data), without access to the images that generate them, our method can register NeRFs (in both pose and scale) and render images from blended NeRFs. Removing the need for source images also greatly reduces memory consumption.

A typical scene may be captured by 100 images, each about 1 MB in size. In contrast, NeRF, which acts as a compression of the individual images, provides an implicit representation of the scene that takes up approximately 5 MB, a 20× reduction from the set of original images. Directly transferring this implicit representation makes it possible to build real-time 3D capturing applications (e.g. NeRF streaming).

NeRFuser fuses NeRFs in two steps: *registration* and *blending*. For the first step, we propose registration from re-rendering. It takes advantage of the ability of modern NeRFs to synthesize high-quality views, which enables us to make use of a 2D image matching pipeline for the 3D NeRF registration task. For the second step, inspired by BlockNeRF [33], we propose a fine-grained sample-based blending method, including a novel compatible weighting method. In summary, we propose (i) a **novel registration from re-rendering method**, that aligns uncalibrated implicit representations and solves for relative scale and pose; and (ii) a **new blending method to composite predictions from multiple NeRFs**, resulting in blended images that are better than images rendered by any individual NeRF.

2. Related Work

2.1. Neural Radiance Fields

A Neural Radiance Field (NeRF) [33] is a parametric representation of a 3D scene. It optimizes a neural network composed of MLPs to encode the scene as density and radiance fields, which can be used to synthesize novel views through volumetric rendering. Since its introduction, many follow-up works [2, 17, 3, 41, 31, 34] have improved over the original implementation.

One line of improvement involves the reconstruction of large-scale NeRFs [38, 43, 33, 30, 46, 45, 35, 21, 33]. However, most of these works focus on reconstructing the entire scene with a single model. While progressive training [38, 30] and carefully designed data structures [43, 46, 45] have helped to expand the expressivity of a single model, other works [21, 33] have shown that a collection of many small models can perform better, while maintaining the same number of parameters. Our method provides a novel way to reason over many small models, combining them to improve performance.

2.2. NeRF Registration

NeRFs are optimized from posed images, with poses usually obtained using a structure-from-motion (SfM) method [28, 29, 26, 27, 8, 22, 9]. Because these methods are scale-agnostic, the resulting coordinate system will have an arbitrary scale specific to each NeRF. Jointly using multiple NeRFs requires *NeRF Registration*, i.e.,solving for the relative transformation between their coordinate systems.

Note that the setting is different from "NeRF Inversion" [40, 14] that estimates the 6-DoF camera pose relative to the pre-trained NeRF given an image, a technique that has been used for NeRF-based visual navigation and localization [1, 15, 6]. However, these tasks can potentially be handled by NeRFuser if formulated as NeRF-to-NeRF pose estimation. Also relevant are works that jointly optimize NeRF representations along with the poses and intrinsics [13, 37, 12]. However, NeRFuser only uses SfM on re-rendered images, and does not modify the pre-trained NeRFs themselves.

While there is a large body of work on registration for explicit representations (e.g., point-clouds) [20], there are few works on NeRF registration. To the best of our knowledge, there are two works related to ours: nerf2nerf [10] and Zero-NeRF [19]. Both approaches perform registration in a purely geometric way—extracting surfaces from the NeRF representations, and thus do not take full advantage of the rich encoded radiance information and its rendering capability. Further, nerf2nerf is only capable of *local* registration under a known scale, and even then requires a reasonable initialization in the form of human annotations. On the other hand, our method performs scaled *global* registration and does not require any human annotations.

2.3. NeRF Blending

Image blending is a highly researched topic in computational photography [5, 4], but few works have discussed blending in terms of NeRFs. Relevant is Block-NeRF [33], which blends NeRFs in both image- and pixel-wise manner. It introduces two ways to measure the blending weights of NeRFs: inverse distance weighting (IDW) and visibility prediction. IDW computes the contribution of each NeRF according to:

$$w_i \propto d_i^{-\gamma}, \tag{1}$$

where d_i is some notion of distance between the camera and elements of NeRF i, $\gamma \in \mathcal{R}$ is a positive hyper-parameter that modulates the blending rate.

Building on this work, we propose a blending approach that operates on ray samples, as well as a new IDW method for sample-wise blending. NeRFuser provides a principally more refined way of blending, which we show leads to sharper images.

3. Methodology

In this section we will first describe our NeRF registration method *registration from re-rendering* and then our blending technique *IDW-Sample*.

3.1. Background

In this section we include background information on volumetric rendering and inverse distance weighing

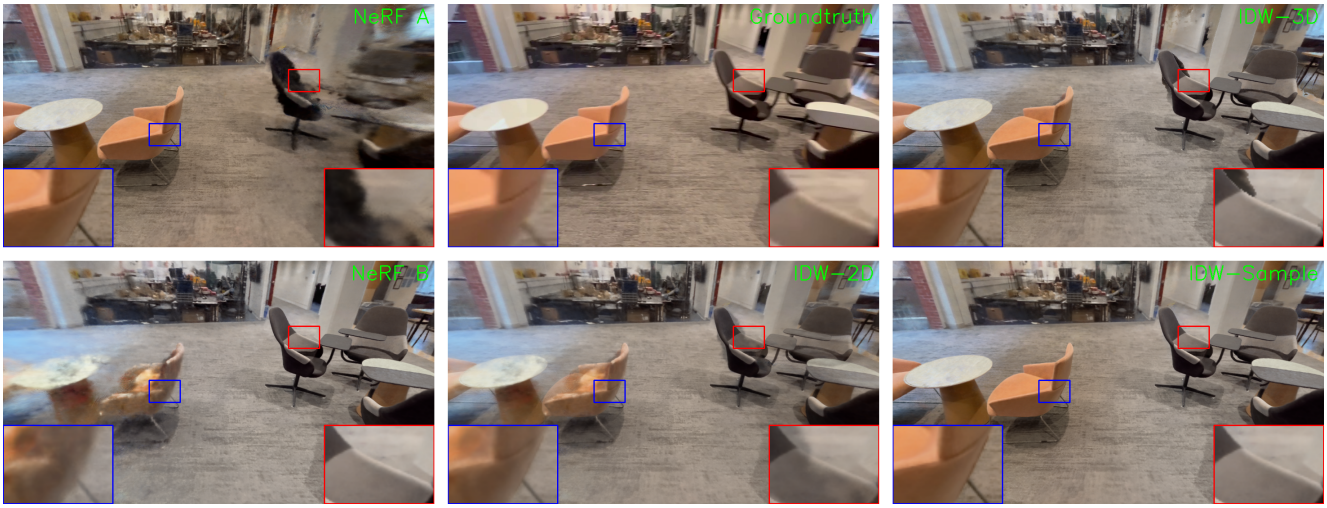

Figure 2: Qualitative comparison of blending methods. Our proposed *IDW-Sample* produces high-quality blending for both chairs, while baseline methods fail on at least one chair. Notice that the blended results (e.g., *IDW-Sample*) are even sharper than the real test image, which exhibits motion blur, demonstrating an advantage of fusing information from multiple NeRFs.

(IDW) [33] methods for neural radiance field (NeRF) [16] blending.

3.1.1 Volumetric Rendering

Given a 3D point $\boldsymbol{p}$ and a viewing direction $\boldsymbol{d}$, NeRF $\mathcal{R}$ predicts the scene density σ at that point and its color $\boldsymbol{c}$ when viewed along direction $\boldsymbol{d}$

$$[\sigma(\boldsymbol{p}), \boldsymbol{c}(\boldsymbol{p}, \boldsymbol{d})] = \mathcal{R}(\boldsymbol{p}, \boldsymbol{d}). \tag{2}$$

To render novel views, consider the image pixel corresponding to camera ray $\boldsymbol{r} = (\boldsymbol{o}, \boldsymbol{d})$, where $\boldsymbol{o}$ is the camera's optical center and $\boldsymbol{d}$ is the ray direction.

In practice, non-overlapping samples are proposed along the ray at locations with significant predicted density. Assuming K intervals of length δ_k sampled along ray $\boldsymbol{r}$ at a distance of t_k from $\boldsymbol{o}$, a NeRF predicts the density and color for each sample as

$$\sigma_k = \sigma(\boldsymbol{o} + t_k \boldsymbol{d}), \tag{3a}$$
$$\boldsymbol{c}_k = \boldsymbol{c}(\boldsymbol{o} + t_k \boldsymbol{d}, \boldsymbol{d}). \tag{3b}$$

The accumulated color is

$$\boldsymbol{C}(\boldsymbol{r}) = \sum_{k=1}^{n} T_k \alpha_k \boldsymbol{c}_k, \tag{4}$$

where $\alpha_k = 1 - \exp(-\sigma_k \delta_k)$ is the probability that light is blocked in this sampled interval of length δ_k along the ray at location t_k. The probability of light reaching this interval (i.e., not being blocked along the way) is then

$$T_k = \prod_{l=1}^{k-1} (1 - \alpha_l) = \exp\left(- \sum_{l=1}^{k-1} \sigma_l \delta_l \right). \tag{5}$$

Additionally, termination probability is defined as

$$p_k = T_k \alpha_k, \tag{6}$$

which is the probability of light traveling along $\boldsymbol{r}$ and getting blocked at sample k. Equation 4 thus becomes

$$\boldsymbol{C}(\boldsymbol{r}) = \sum_{k=1}^{n} p_k \boldsymbol{c}_k. \tag{7}$$

3.1.2 NeRF Blending with IDW

When using multiple NeRFs to render an image, the contribution of each NeRF can be determined using inverse distance weighting (IDW)

$$w_i \propto d_i^{-\gamma}, \tag{8}$$

where d_i is the Euclidean distance of the blending element (e.g. ray sample in *IDW-Sample*) to NeRF i's origin, $\gamma \in \mathbb{R}^+$ is a hyper-parameter that modulates the blending rate. Block-NeRF [33] proposes two variants of IDW, namely *IDW-2D* and *IDW-3D*, which we discuss below.

IDW-2D blends images using an image-wise weighting

$$I = \sum_i w_i I_i, \tag{9}$$

where we use the distance between the query camera center and NeRF i's origin as d_i to compute w_i. While *IDW-2D* works well when the query camera is much closer to one of the NeRFs than the others, it suffers when the query camera is roughly of the same distance from all source NeRFs. In

the later case, the blended image will be a blurry mixture affected by noisy regions existing in source images, resulting in poor visual quality.

IDW-3D is a pixel-wise weighting strategy that considers the distance between the origin $\boldsymbol{x}_i$ of each NeRF i and the 3D coordinates $\boldsymbol{p}_i^{(j)}$ of pixel j determined using the expected depth predicted by NeRF i,

$$d_i^{(j)} = \|\boldsymbol{x}_i - \boldsymbol{p}_i^{(j)}\|_2. \tag{10}$$

Each pixel j is then rendered by substituting $d_i^{(j)}$ into Equation 8 as

$$I^{(j)} = \sum_i w_i^{(j)} I_i^{(j)}. \tag{11}$$

The major problem with *IDW-3D* is that, to accurately obtain the expected point of ray termination $\boldsymbol{p}_i^{(j)}$, it requires NeRF i to faithfully predict the depth for pixel j. This is not always fulfilled since (i) NeRFs are not known to accurately reconstruct the scene geometry; and moreover, (ii) source NeRFs will be focusing on different portions of the scene by design, leading to invalid blending weights. Empirically, *IDW-3D* usually performs the worst among all blending methods.

3.2. Registration from Re-Rendering

The first step of our pipeline is to estimate the relative transformations between two or more input NeRFs.

We assume that each NeRF is trained on a separate set of images, capturing different, yet overlapping, portions of the same scene (i.e., each input NeRF has at least one neighbor). We do not assume a specific type of training data, e.g., the poses used to train the NeRFs may have come from KinectFusion [18] and have metric scale, or they may be the result of a SfM pipeline (e.g., COLMAP [29]) and thus have arbitrary scale.

As a result, each NeRF may have a unique coordinate system that is inconsistent with the others, in terms of translation and rotation as well as scale. Without loss of generality, the following discussion focuses on the registration of two NeRFs, A and B, as the extension to more than two NeRFs is straightforward.

Our goal is to find the transformation $T_{BA} \in \mathrm{SIM}(3)$ that transforms a 3D point p_B in NeRF B to its corresponding point p_A in NeRF A as $p_A = T_{BA} p_B$. Note that $T_{BA} = \left[\begin{smallmatrix} R_{BA} & \boldsymbol{t}_{BA} \\ \mathbf{0} & 1 \end{smallmatrix}\right] S_{BA}$ can be decomposed into a rotation R_{BA}, translation $\boldsymbol{t}_{BA}$, and uniform scaling of factor s_{BA}, where S_{BA} is a diagonal matrix $\mathrm{diag}(s_{BA}, s_{BA}, s_{BA}, 1)$.

First, we assume that the NeRFs are produced with sufficient training views, so that they can generate high quality novel views. We then sample a set of poses (e.g. uniformly on the upper hemisphere), which we use as local poses to query both NeRFs to get re-renderings.

We re-purpose off-the-shelf structure-from-motion methods (SuperPoint as the feature [8] and SuperGlue as the matcher [27]) on the union of re-rendered images from the two NeRFs in order to recover their poses in the same coordinate system, which we then use for registration, as discussed next.

Procedure and Notation Given the trained model of NeRF A, we query it with sampled camera poses $\left\{G^A_{A_i}\right\}_i$ to synthesize images $\{I_{A_i}\}_i$. $G^A_{A_i} \in \mathrm{SE}(3)$ is specified as a pose matrix that transforms a point from the coordinate system of camera A_i to that of NeRF A. I_{A_i} is the image synthesized from NeRF A using the query pose $G^A_{A_i}$. Likewise, we query NeRF B with $\left\{G^B_{B_i}\right\}_i$ to synthesize images $\{I_{B_i}\}_i$. We then feed images $\{I_{A_i}\}_i \cup \{I_{B_i}\}_i$ as input to SfM, and obtain poses $\left\{G^C_{A_i}\right\}_i$ and $\left\{G^C_{B_i}\right\}_i$ as output. $G^C_{A_i} \in \mathrm{SE}(3)$ is the recovered pose of image I_{A_i} from SfM. It is specified as a pose matrix that transforms a point from the coordinate frame of camera A_i to C, where C is the coordinate system determined by this SfM execution. Note that an $\mathrm{SE}(3)$ pose matrix does not involve scale, so the induced camera-to-world transformation always assumes a specific camera instance that shares the same scale as the world.

Recovering Scale Let $S_{AC} = \mathrm{diag}(s_{AC}, s_{AC}, s_{AC}, 1)$ be the scale matrix from NeRF A to C, meaning that one unit length in NeRF A equals s_A units in C. Considering $G_{ij} \in \mathrm{SE}(3)$ as the pose of camera A_i relative to camera A_j when specified in C's scale, we have

$$\begin{aligned} G_{ij} &= {G^C_{A_j}}^{-1} G^C_{A_i} && \text{using } C \text{ as bridge} \\ &= S_{AC} {G^A_{A_j}}^{-1} G^A_{A_i} {S_{AC}}^{-1} && \text{using } A \text{ as bridge} \end{aligned} \tag{12}$$

If we further dissect $G^C_{A_i}$ as $\left[\begin{smallmatrix} R^C_{A_i} & \boldsymbol{t}^C_{A_i} \\ \mathbf{0} & 1 \end{smallmatrix}\right]$ and repeat this for $G^C_{A_j}, G^A_{A_i}, G^A_{A_j}$, Equation 12 becomes

$$\begin{aligned} &\begin{bmatrix} {R^C_{A_j}}^\top R^C_{A_i} & {R^C_{A_j}}^\top (\boldsymbol{t}^C_{A_i} - \boldsymbol{t}^C_{A_j}) \\ \mathbf{0} & 1 \end{bmatrix} \\ &= \begin{bmatrix} {R^A_{A_j}}^\top R^A_{A_i} & s_{AC} {R^A_{A_j}}^\top (\boldsymbol{t}^A_{A_i} - \boldsymbol{t}^A_{A_j}) \\ \mathbf{0} & 1 \end{bmatrix}. \end{aligned} \tag{13}$$

Note that all components involved in Equation 13 are either determined when sampling camera poses or given by SfM with the exception of s_{AC}, which is what we want to recover. Specifically, equating the L2 norm of the translation part from both sides of Equation 13 gives

$$s_{AC} = \frac{\|\boldsymbol{t}^C_{A_i} - \boldsymbol{t}^C_{A_j}\|_2}{\|\boldsymbol{t}^A_{A_i} - \boldsymbol{t}^A_{A_j}\|_2} \tag{14}$$

In practice, we use the median over all i, j pairs to construct S_{AC}. S_{BC} is recovered similarly.

Recovering Transformations Let $T_{AC} \in \mathrm{SIM}(3)$ be the transformation from NeRF A to C. Using camera A_i as bridge, we have $T_{AC} = G^C_{A_i} S_{AC} {G^A_{A_i}}^{-1}$. In practice, we compute T_{AC} over all instances of camera A_i, and choose the closest valid $\mathrm{SIM}(3)$ transformation to the median result. T_{BC} is recovered similarly. We then compute NeRF B to NeRF A transformation as $T_{BA} = {T_{AC}}^{-1} T_{BC}$.

Robustness to pose estimation errors While our proposed registration method works better with more accurate SfM results on NeRF-synthesized images, it is also robust to SfM's errors. When computing the relative scale, we only need to recover at least two poses (so that at least one pair is formed to be used in Equation 14) from each NeRF's re-renderings. This is easily achievable with a reasonably sampled set of query poses. Moreover, since we consider the median result as the final estimation, the impact of erroneous poses will be minimal. A similar analysis also holds for transformation recovery, except that only a single pose is needed for the estimation.

3.3. NeRF Blending

Given two or more registered NeRFs and a query camera pose, NeRF blending [33] aims to combine predictions from the individual NeRFs with the goal of high-quality novel view synthesis. Without loss of generality, we consider again the two-NeRF setting: A and B with relative transformation $T_{BA} \in \mathrm{SIM}(3)$. Let $G_B \in \mathrm{SE}(3)$ be a pose defined in NeRF B's coordinate system that can be used to query NeRF B. To get the corresponding pose G_A to query NeRF A, we first decompose $T_{BA} = G_{BA} S_{BA}$, and compute $G_A = T_{BA} G_B {S_{BA}}^{-1}$.

For blending, there are three key concepts to consider: (i) **when to blend**: in what case should it be used; (ii) **what to blend**: at what granularity should it happen; and (iii) **how to blend**: in which way should we compute blending weights? Block-NeRF [33] answers (i) with *visibility thresholding*, where if the mean visibility of a frame (predicted by a visibility network) is above a certain threshold then blending is activated. Afterwards, it answers (ii) by introducing image- and pixel-wise blending. Finally, it handles (iii) by inverse-distance-weighting (IDW) and predicted visibility weighting. Importantly, to achieve any of these results, a visibility prediction network has to be trained jointly with the NeRF and used during inference. In our setting, we do not assume access to a visibility network, since we are dealing with black-box uncalibrated NeRFs not generated for this particular purpose.

In this paper, we answer (i) by proposing a simpler threshold that is solely based on distance. We answer (ii) by proposing a novel sample-based blending method recognizing the fact that the color of a pixel is computed using samples along the ray in NeRF during volumetric rendering. We answer (iii) by proposing an IDW method for our sample-based blending. Since we use IDW with sample-based blending, we coin our method *IDW-Sample*.

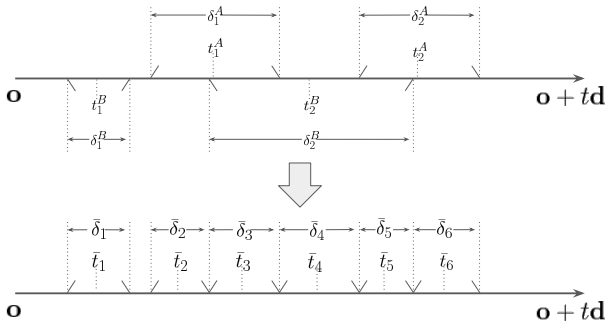

Figure 3: An illustration of how ray samples proposed by two NeRFs are merged based on their locations and lengths. Top: two sets of ray samples proposed by NeRF A and NeRF B; Bottom: the single set of merged ray samples.

Without loss of generality, we discuss the blending of two registered NeRFs, A and B. Our findings easily extend to an arbitrary number of NeRFs and also to any volumetric representation.

3.3.1 Distance Test for NeRF Selection

The decision of when to render using blended NeRFs, rather than just one NeRF, is an important question, because NeRFs can only render with high-quality within their effective range. Rendering using distant NeRFs, whose rendering quality is poor, can only be harmful. Hence, we introduce a test based on the distance between the origin of the query camera and the NeRF centers. Denoting the distances from NeRF A and NeRF B as d_A and d_B, the test value is $\tau = \max\left(\frac{d_A}{d_B}, \frac{d_B}{d_A}\right)$. If τ is greater than a threshold, it means that the second-closest NeRF is sufficiently far, in which case it is better to simply use the rendering of the closest NeRF to the query camera; otherwise, IDW-based blending is enabled.

3.3.2 IDW-Sample for NeRF Blending

During NeRF's volumetric rendering stage, a pixel's color is computed using samples along the ray. Recognizing this fact, we propose a sample-wise blending method that calculates the blending weights for each ray sample using IDW. We show that the original volumetric rendering methodology can be easily extended to take advantage of these new sample-wise blending weights, resulting in our proposed *IDW-Sample* strategy. **Merge Ray Samples** Consider a pixel to be rendered, which gets unprojected into a ray. Since ray samples are separately proposed according to the density field of each source NeRF, we need to merge them

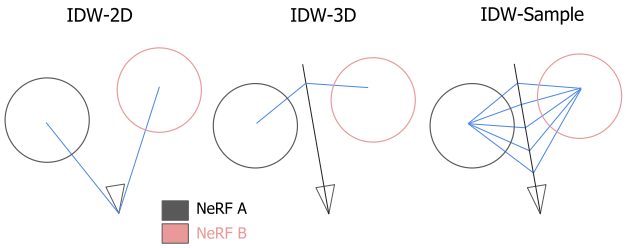

Figure 4: Illustration of IDW-based blending methods. IDW-2D depends on the distance between camera center and NeRF centers. IDW-3D depends on the distance between estimated ray depth and NeRF centers. IDW-Sample depends on the NeRF centers and sample positions that are irrespective to depth quality, which is major downside of IDW-3D.

into a single set. As illustrated in Figure 3, given samples $\{(t_k^A, \delta_k^A)\}_k$ and $\{(t_k^B, \delta_k^B)\}_k$ proposed from NeRF A and NeRF B, respectively, we merge them into a single set of ray samples $\{(\bar{t}_k, \bar{\delta}_k)\}_k$ by taking the sample location t and length δ. We update the termination probability and color of each new sample in the merged set for each source NeRF. Given a ray sample proposed by a NeRF, we assume its termination probability mass is uniformly distributed over its length, while its color is the same for any point within coverage.

Blending Process We use IDW to compute the blending weight for each sample. Specifically, let $\boldsymbol{x}_i$ be the origin of NeRF$_i$ for $i \in \{A, B\}$, $\boldsymbol{o}$ be the camera's optical center, $\boldsymbol{r} = (\boldsymbol{o}, \boldsymbol{d})$ be the ray corresponding to pixel j to be rendered, and $(\bar{t}_k, \bar{\delta}_k)$ be a ray sample from the merged samples set. We compute its blending weight as $w_{i,k} \propto {d_{i,k}}^{-\gamma}$, where $d_{i,k} = \|\boldsymbol{x}_i - (\boldsymbol{o} + \bar{t}_k \boldsymbol{d})\|_2$.

The blended pixel j is

$$I^{(j)} = \sum_k \sum_i w_{i,k} \bar{p}_{i,k} \bar{\boldsymbol{c}}_{i,k} \tag{15}$$

Weights $w_{i,k}$ are normalized following two steps: (i) $\sum_i w_{i,k} = 1, \ \forall k$; and (ii) $\sum_k \sum_i w_{i,k} \bar{p}_{i,k} = 1$. Step (i) indicates that our method does not change the relative weighting of samples along a given ray, which is already dictated by the termination probability. Step (ii) ensures that the rendered pixel has a valid color. Figure 4 provides an illustration.

4. Experiments

In this section, we describe our registration and blending experiments on Scannet, Block-NeRF, and an object-centric scene dataset we collect, which we will make available.

4.1. Datasets

Object-Centric Indoor Scenes We created a dataset consisting of three indoor scenes, using an iPhone 13 mini in video mode. Each scene consists of three video clips – we choose two objects in each scene, and collect two (overlapping) video sequences that focus on each object. Then, we collect a third (test) sequence that observes the entire scene.

We extract images at 3.75 fps from all three video clips and feed them jointly to an structure-from-motion (SfM) tool (we use the hloc toolbox [26, 27]) to recover their poses. These poses are defined in a shared coordinate system, which we denote as C. We then center and normalize the poses for each training set, so that the processed poses are located within the bounding box $[-1, 1]^3$. Note that this step induces a local coordinate system for each NeRF, which we denote as A and B respectively. We record the resulting transformations $\{T_{CA}, T_{CB}\} \subset \mathrm{SIM}(3)$ and treat them as ground-truth. NeRFs A and B are then trained separately from the corresponding training set of images. We test NeRFuser on this dataset and report results of both registration and blending in Section 4.2.

ScanNet Dataset Since the "ground-truth" poses of our dataset are estimated using SfM based only on RGB images, they are potentially not as accurate as what could be obtained using RGB-D data. Hence, we use the ScanNet dataset to further test registration performance. The ScanNet dataset provides a total of 1513 RGB-D scenes with annotated camera poses, from which we use the first 218 scenes. We downsample the frames so that roughly 200 posed RGB-D images are kept from each scene. We then split the images into three sets: two for training NeRFs and one for testing. Specifically, images are split according to their temporal order. We first randomly select 10% of all images as the test set. Of the remaining images, we label the first 25% as training set A, the last 25% as training set B, and randomly label the middle 50% as either A or B. This splitting strategy creates a moderate spatial overlap among A, B and the test sets. Once we have the splits, we center and normalize the training poses the same way as in Section 4.1. The resulting transformations T_{CA}, T_{CB} are recorded as ground-truth. After generaring the NeRFs, we check their quality according to validation PSNR and keep the best 25 scenes. We test NeRF registration on this dataset and compare with point-cloud registration in Section 4.3.

Mission Bay Dataset To further test the rendering quality of different blending methods, we run experiments on the Mission Bay dataset from Block-NeRF [33], which features a street scene from a single capture. The dataset is collected using 12 cameras that capture the surround of a car that drives along a straight street with a quarter turn

Blending	Ground-truth T_{BA}			Estimated $\hat{T}_{BA}$		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF	20.92	0.716	0.369	20.90	0.714	0.370
Nearest	23.81	0.779	0.283	23.68	0.774	0.287
IDW-2D	24.70	0.795	0.267	24.64	0.792	0.267
IDW-3D	23.48	0.776	0.279	23.45	0.772	0.280
IDW-Sample (Ours)	**24.91**	**0.813**	**0.228**	**24.83**	**0.810**	**0.229**

Table 1: Blending results on Object-Centric Indoor Scenes. *IDW-Sample* works the best for all metrics with both ground-truth and estimated transformations. Results with estimated $\hat{T}_{BA}$ are only marginally worse than those with ground-truth T_{BA}, which demonstrates that our proposed NeRF registration is accurate enough for the downstream blending task.

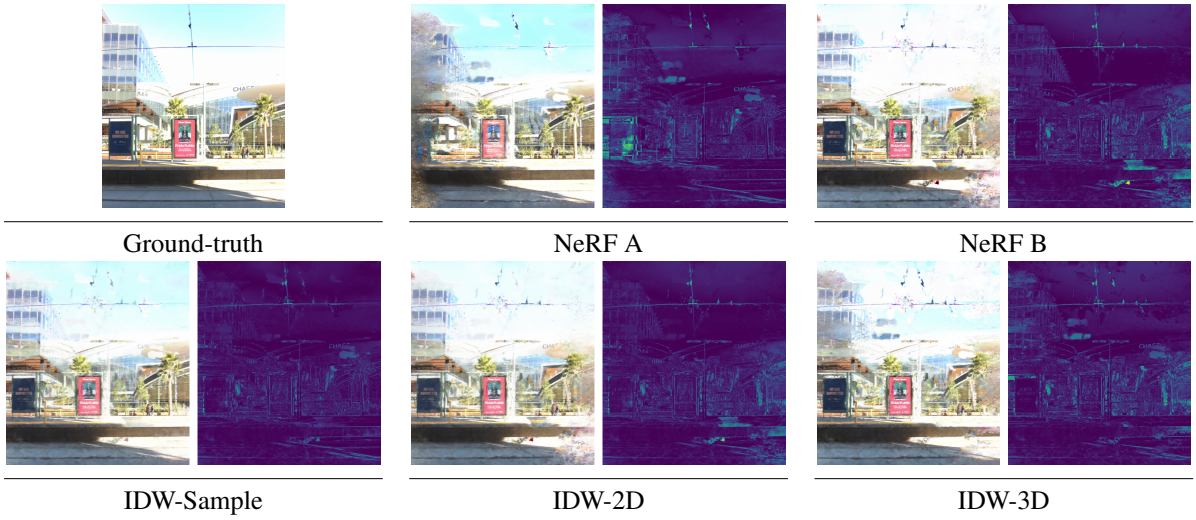

Figure 5: NeRF blending with IDW-based methods on the Mission Bay dataset. Per-pixel errors are visualized as heat maps. Individual NeRFs renderings have large artifacts on either side, which are best resolved by *IDW-Sample* blending.

at the end. The dataset has approximately 12500 images, with poses recovered by odometry. We split the images into training and test sets as follows: for every 30 time-stamped images, we use the first 25 to construct a block for training and reserve the remaining 5 for testing. This results in a total of 31 training sets that we use to construct NeRFs, and 30 test sets between every 2 neighboring blocks. We test the blending of two neighboring NeRFs using only the side-view camera (cam73) from the test sets, for which the difference of NeRF renderings are the most visible. We report results in Section 4.4.

4.2. NeRF Fusion

We test NeRFuser including both NeRF registration and NeRF blending on Object-Centric Indoor Scenes. For registration, we generate 32 poses that are roughly uniformly placed on the upper hemisphere of radius 1, with elevation from 0 to 30°. We use them as local poses to query NeRFs A and B, and feed the 64 synthesized images jointly to SfM. NeRF B to NeRF A transformation $\hat{T}_{BA}$ is recovered using our proposed procedure. To evaluate its accuracy, given ground-truth and estimated transformations $\{T, \hat{T}\} \subset \mathrm{SIM}(3)$, we first compute $\Delta T = \hat{T}T^{-1}$. It is then decomposed into $\Delta G \in \mathrm{SE}(3)$ and $\Delta S \in \mathrm{SIM}(3)$ as $\Delta T = \Delta G \Delta S$. Rotation error r_{err} is computed as the angle (in degrees) of ΔG's rotation matrix. Translation error t_{err} is computed as the L2 norm of ΔG's translation vector. Note that by definition, t_{err} is measured in NeRF A's unit. For scale error, we extract $\Delta s = |\Delta S|^{1/3}$ and compute $s_{\text{err}} = |\log \Delta s|$. We report T_{BA} errors against ground-truth: $r_{\text{err}} = 0.031°, t_{\text{err}} = 0.0013, s_{\text{err}} = 0.0045$.

For blending, we set distance test ratio $\tau = 1.8$ and blending rate $\gamma = 5$. Since it depends on the NeRF B to NeRF A transformation, we report results in two settings: (i) using ground-truth T_{BA} and (ii) using estimated $\hat{T}_{BA}$. The second setting is specifically used to showcase the compound performance of the full NeRFuser framework. In addition to IDW-based methods, we also include

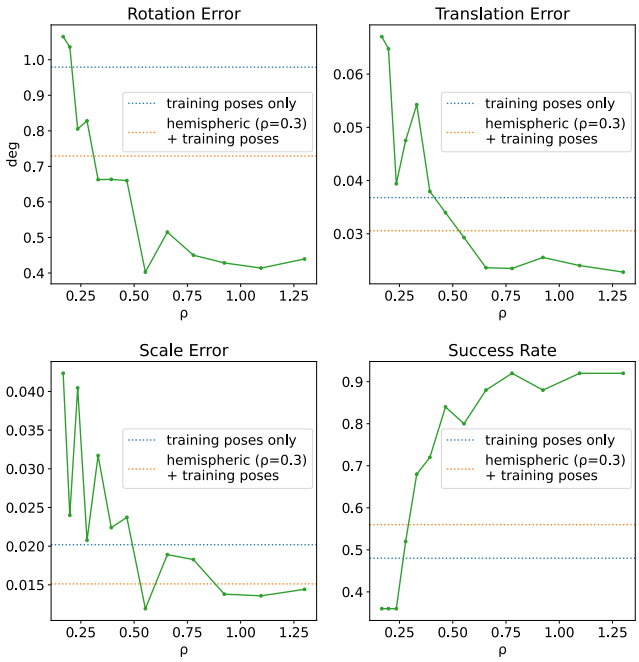

Figure 6: Effect of re-rendering poses on NeRF registration. With more sampled poses, the registration errors go down while success rate improves (green curve). Using additional hemispheric poses besides the training ones also proves helpful (orange line vs. blue line). More interestingly, with a large enough ratio ρ, registration with hemispherically sampled poses outperforms training poses when using the same number or fewer poses in total. It shows that it is beneficial to have a larger spatial span of re-rendering poses for registration as illustreated in Figure 8.

NeRF and *Nearest* as baselines. *NeRF* directly uses NeRF-synthesized images, while *Nearest* uses the rendering from the closer NeRF to the query pose. We evaluate blending results against ground-truth images on three metrics: PSNR [11], SSIM [36] and LPIPS [42]. We report numbers averaged over test images of all three scenes from our dataset in Table 1.

4.3. NeRF Registration

To further test the registration performance on a large-scale dataset, we use the ScanNet dataset [7] as prepared according to Section 4.1. We repeat the same registration procedure as detailed in Section 4.2, except that 60 hemispheric poses are sampled instead of 32. During experiments, we notice failure cases due to NaN or outlier values. To report more meaningful numbers, we treat cases that meet any of the following conditions as failure: (i) is NaN or (ii) $r_{\text{err}} > 5^\circ$ or (iii) $t_{\text{err}} > 0.2$ or (iv) $s_{\text{err}} > 0.1$.

We also compare our method against various point-cloud registration (PCR) baselines using both (i) point–clouds extracted from NeRFs and (ii) point-clouds fused from ground-truth posed RGB-D images. We describe the point-cloud data preparation for each scene as below. While there are scale-adaptive methods [25] for PCR, most available and well-tested implementations presume that the two point-clouds to be registered are measured in the same unit. Since T_{BA} is measured in NeRF A's unit, we make sure that the measuring unit of both point-clouds is the same as NeRF A's. For (i) NeRF-extracted point-clouds, the unit conversion is achieved by applying S_{BA} to point-cloud B. For (ii) RGB-D fused point-clouds, the unit conversion is achieved by applying S_{CA} to both point-clouds. Additionally, note that in this case all ground-truth poses are defined in the same world coordinate system. To enable a fair comparison, we further transform RGB-D fused point-cloud A by G_{BA} after unit conversion. After these processing steps, we get point-clouds ready for registration, whose ground-truth solution is G_{BA} for both (i) and (ii). We report in Table 2 the results of our registration method and various PCR baselines averaged over all successfully registered scenes, as well as the success rate.

Registration	r_{err} (°)	t_{err}	s_{err}	Success
NeRF-extracted point-cloud				
ICP [23]	3.027	0.1151	N/A	0.13
FGR [44]	4.549	0.1844	N/A	0.04
FPFH [24]	2.805	0.0381	N/A	0.17
RGB-D-fused point-cloud				
ICP [23]	1.598	0.0816	N/A	0.17
FGR [44]	1.330	0.0372	N/A	0.71
FPFH [24]	**0.049**	**0.0205**	N/A	*0.79*
NeRFuser	*0.588*	*0.0315*	**0.0211**	**0.84**

Table 2: **Registration results on ScanNet.** We compare to point-cloud registration methods on both NeRF-extracted point-cloud and ground-truth RGB-D-fusion point-cloud. Due to the noisy geometry of NeRF reconstructions, registration performance on NeRF-extracted point-clouds is inferior. However, NeRFuser is comparable to the registration performance on RGB-D-fused methods in terms of r_{err} and t_{err}, while having the highest success rate. Our method also recovers the relative scale, while point-cloud baselines work on the relaxed problem of $\mathrm{SE}(3)$ pose recovery. **Bold** numbers are the best, *italic* numbers are second best.

4.4. NeRF Blending

To further test our blending performance, we use the outdoor Mission Bay dataset as described in Section 4.1, with ground-truth transformations. We set the distance test ratio $\tau = 1.2$, and the blending rate $\gamma = 10$. Quantitative results averaged over test images of all scenes are reported in Table 3. Qualitative results are visualized in Figure 5.

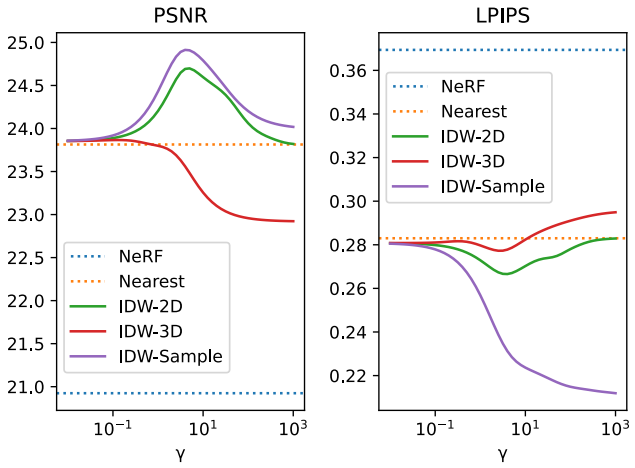

Figure 7: Effect of blending rate γ in IDW-based blending ranging from $[0.01, 1000]$. For all blending methods, quality initially increases with γ, but then decreases as γ increases further.

Blending	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF	17.306	0.571	0.502
Nearest	19.070	0.657	0.398
IDW-2D	19.692	0.659	0.413
IDW-3D	18.806	0.636	0.433
IDW-Sample (Ours)	**19.986**	**0.678**	**0.388**

Table 3: Blending results on Mission Bay dataset. *IDW-Sample* performs the best for all metrics.

4.5. Ablation Studies

Ablation on re-rendering poses for NeRF registration We study the registration performance on ScanNet dataset w.r.t. the number of sampled poses. To account for the fact that each scene may be of a different scale, we introduce ρ as the ratio of the number of sampled poses over the number of training views. We geometrically sample $\rho \in [0.167, 1.3]$, and generate the hemispheric poses accordingly. We evaluate the performance of NeRF registration w.r.t. ρ averaged over all ScanNet scenes. In addition, we include 2 more settings. (i) *training poses only*: instead of hemispheric poses, use NeRF's training poses for re-rendering; (ii) *hemispheric + training poses*: use NeRF's training poses together with hemispherically sampled ones ($\rho = 0.3$) for re-rendering. Results are reported in Figure 6. We want to additionally highlight that, since registration using a relatively small number of sampled poses for re-rendering can still work well, it implies that the registration procedure will not take long. In practice, it typically only takes minutes to finish.

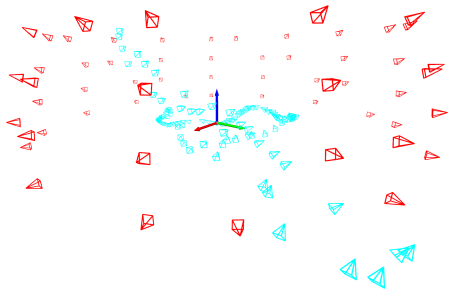

Figure 8: Distributions of training poses and sampled poses on a ScanNet scene used for NeRF registration. Sample poses are less cluttered than training ones that come from a handheld camera trajectory, which results in a wider baseline for easier SfM.

Ablation of γ in IDW-based blending We study the effect of blending rate γ in IDW-based blending on Object-Centric Indoor Scenes. Specifically, we use ground-truth transformations and set distance test ratio $\tau = 1.8$. We geometrically sample γ in $[10^{-2}, 10^{3}]$. For each sampled γ, we blend NeRFs with all IDW-based methods and report the results averaged over test images of all 3 scenes from Object-Centric Indoor Scenes. Since *Nearest* and *NeRF* are not affected by γ, we draw dotted horizontal lines for comparison. The results are shown in Figure 7. In $\gamma \to 0$ case, all IDW-based methods become the same as using the mean image. In $\gamma \to \infty$ case, *IDW-2D* becomes the same as *Nearest*, while *IDW-Sample* becomes analogous to KiloNeRF [21] (more details in appendix 6.2). We find the optimal γ in between the extremes for any IDW-based method. Moreover, our proposed *IDW-Sample* almost always performs the best for any given γ.

Ablation on Blending Performance over Query Poses We provide a qualitative ablation study to showcase the performance of our proposed blending method *IDW-Sample* against baselines for different test poses with respect to two NeRF centers in Figure 9. The study is supposed to provide a geometric sense of where the blending method gives the most benefits.

5. Conclusion

We have introduced NeRFuser, a NeRF fusion pipeline that registeres and blends arbitrary many NeRFs treated as input data. To address the problem of registration, we propose *registration from re-rendering*, taking advantage of NeRF's ability to synthesize high quality novel views. To address the problem of blending, we propose *IDW-Sample*, leveraging the ray sampling nature of NeRF rendering. While we have demonstrated NeRFuser's robust performance in multiple scenarios, it inherits any of the failure cases of the input NeRFs. Variants that use structured priors can easily be integrated into our framework, since it

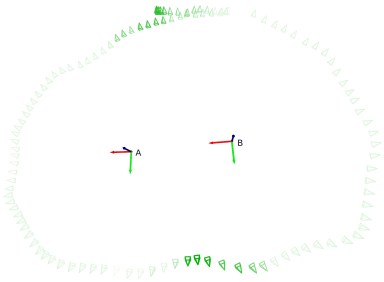

Figure 9: A visualization of how *IDW-Sample* performs against *Nearest*. The axes denote the reference frames for two input NeRFs, while the green camera frusta denote the test camera poses. The darker the camera frusta, the better *IDW-Sample* performs compared to *Nearest*. The best-performing poses are those that evenly observe the scene.

is agnostic to the source NeRFs. We believe this tool will help enable the increased proliferation of implicit representations as raw data for future 3D vision applications.

References

[1] Michal Adamkiewicz, Timothy Chen, Adam Caccavale, Rachel Gardner, Preston Culbertson, Jeannette Bohg, and Mac Schwager. Vision-only robot navigation in a neural radiance world. *IEEE Robotics and Automation Letters*, 7(2):4606–4613, 2021.

[2] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 5835–5844, 2021.

[3] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5460–5469, 2021.

[4] Matthew A. Brown and David G. Lowe. Recognising panoramas. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 1218–1225, 2003.

[5] Peter J. Burt and Edward H. Adelson. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31:532–540, 1983.

[6] Timothy Chen, Preston Culbertson, and Mac Schwager. Catnips: Collision avoidance through neural implicit probabilistic scenes. *arXiv preprint arXiv:2302.12931*, 2023.

[7] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas A. Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2432–2443, 2017.

[8] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperPoint: Self-supervised interest point detection and description. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 224–236, 2018.

[9] Mihai Dusmanu, Ignacio Rocco, Tomás Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-Net: A trainable CNN for joint description and detection of local features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8084–8093, 2019.

[10] Lily Goli, Daniel Rebain, Sara Sabour, Animesh Garg, and Andrea Tagliasacchi. nerf2nerf: Pairwise registration of neural radiance fields. *arXiv preprint arXiv:2211.01600*, 2022.

[11] Alain Horé and Djemel Ziou. Image quality metrics: PSNR vs. SSIM. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, pages 2366–2369, 2010.

[12] Yoonwoo Jeong, Seokjun Ahn, Christopher Choy, Anima Anandkumar, Minsu Cho, and Jaesik Park. Self-calibrating neural radiance fields. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 5846–5854, 2021.

[13] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. BARF: Bundle-adjusting neural radiance fields. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 5741–5751, 2021.

[14] Yunzhi Lin, Thomas Müller, Jonathan Tremblay, Bowen Wen, Stephen Tyree, Alex Evans, Patricio A Vela, and Stan Birchfield. Parallel inversion of neural radiance fields for robust pose estimation. *arXiv preprint arXiv:2210.10108*, 2022.

[15] Dominic Maggio, Marcus Abate, J. Shi, Courtney Mario, and Luca Carlone. Loc-NeRF: Monte Carlo localization using neural radiance fields. *arXiv preprint arXiv:2209.09050*, 2022.

[16] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[17] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (TOG)*, 41(4), 2022.

[18] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew William Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 127–136, 2011.

[19] Casey Peat, Oliver Batchelor, Richard Green, and James Atlas. Zero NeRF: Registration with zero overlap. *arXiv preprint arXiv:2211.12544*, 2022.

[20] F. Pomerleau, Francis Colas, and Roland Y. Siegwart. A review of point cloud registration algorithms for mobile robotics. *Foundations and Trends in Robotics*, 4, 2015.

[21] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. KiloNeRF: Speeding up neural radiance fields with thousands of tiny MLPs. In *Proceedings of the In-*

ternational Conference on Computer Vision (ICCV), pages 14315–14325, 2021.

[22] Jérôme Revaud, Philippe Weinzaepfel, César Roberto de Souza, No'e Pion, Gabriela Csurka, Yohann Cabon, and M. Humenberger. R2D2: Repeatable and reliable detector and descriptor. *arXiv preprint arXiv:1906.06195*, 2019.

[23] Szymon M. Rusinkiewicz and Marc Levoy. Efficient variants of the ICP algorithm. In *Proceedings of the International Conference on 3-D Digital Imaging and Modeling*, pages 145–152, 2001.

[24] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (FPFH) for 3D registration. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3212–3217, 2009.

[25] Yusuf Sahillioglu and Ladislav Kavan. Scale-adaptive ICP. *Graphical Models*, 2021.

[26] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12716–12725, 2019.

[27] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4938–4947, 2020.

[28] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4104–4113, 2016.

[29] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 501–518, 2016.

[30] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J. Davison. iMAP: Implicit mapping and positioning in real-time. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 6209–6218, 2021.

[31] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5449–5459, 2021.

[32] Richard Szeliski. Image alignment and stitching: A tutorial. *Foundations and Trends in Computer Graphics and Vision*, 2(1):1–104, 2007.

[33] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P. Srinivasan, Jonathan T. Barron, and Henrik Kretzschmar. Block-NeRF: Scalable large scene neural view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8238–8248, 2022.

[34] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, and Angjoo Kanazawa. Nerfstudio: A modular framework for neural radiance field development. *arXiv preprint arXiv:2302.04264*, 2023.

[35] Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-NeRF: Scalable construction of large-scale NeRFs for virtual fly-throughs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12912–12921, 2021.

[36] Zhou Wang, Alan Conrad Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13:600–612, 2004.

[37] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. NeRF–: Neural radiance fields without known camera parameters. *arXiv preprint arXiv:2102.07064*, 2021.

[38] Yuanbo Xiangli, Linning Xu, Xingang Pan, Nanxuan Zhao, Anyi Rao, Christian Theobalt, Bo Dai, and Dahua Lin. CityNeRF: Building NeRF at city scale. *arXiv preprint arXiv:2112.05504*, 2021.

[39] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. *Computer Graphics Forum*, 41(2):641–676, May 2022.

[40] Lin Yen-Chen, Pete Florence, Jonathan T Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. iMeRR: Inverting neural radiance fields for pose estimation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1323–1330, 2021.

[41] Alex Yu, Sara Fridovich-Keil, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5491–5500, 2021.

[42] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 586–595, 2018.

[43] Xiaoshuai Zhang, Sai Bi, Kalyan Sunkavalli, Hao Su, and Zexiang Xu. NeRFusion: Fusing radiance fields for large-scale scene reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5439–5448, 2022.

[44] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Fast global registration. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 766–782, 2016.

[45] Zihan Zhu, Songyou Peng, Viktor Larsson, Zhaopeng Cui, Martin R. Oswald, Andreas Geiger, and Marc Pollefeys. NICER-SLAM: Neural implicit scene encoding for RGB SLAM. *arXiv preprint arXiv:2302.03594*, 2023.

[46] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. NICE-SLAM: Neural implicit scalable encoding for SLAM. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12786–12796, 2022.

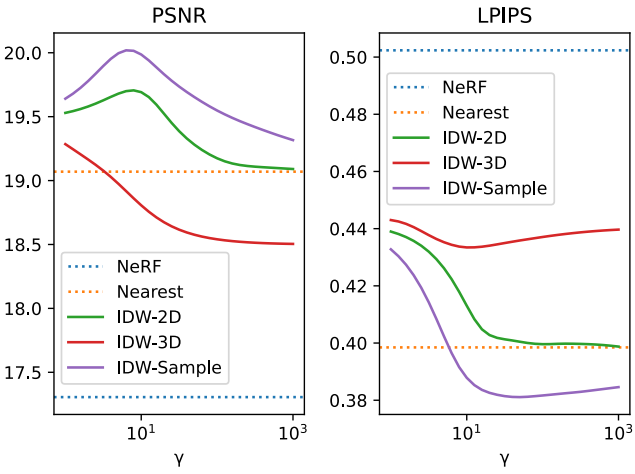

Figure 10: Effect of the blending rate γ ranging from $[1, 10^3]$. Blending quality first increases with γ, then starts to decrease as γ increases further. For any given γ, our *IDW-Sample* works the best.

6. Appendix

6.1. Ablation of γ in IDW-based Blending on the Mission Bay Dataset

We additionally present the blending rate γ ablation on Mission Bay Dataset. Specifically, we use ground-truth transformations and set distance test ratio $\tau = 1.2$. We geometrically sample γ in $[1, 10^3]$. For each sampled γ, we blend NeRFs with all IDW-based methods and report the results averaged over test images of all scenes from the Mission Bay Dataset. Since *Nearest* and *NeRF* are not affected by γ, we draw dotted horizontal lines for comparison. The results are shown in Figure 10.

6.2. Connection to KiloNeRF

In this section we establish a relationship between IDW-Sample and KiloNeRF [21]. Within the framework of our *IDW-Sample* method, KiloNeRF is a special case when the blending rate $\gamma \to \infty$. Intuitively, KiloNeRF employs a grid of small NeRFs within the axis-aligned bounding box of the scene, where each small NeRF is only responsible for the spatial cube it occupies. Specifically, given a sample (δ, t) on ray $(\boldsymbol{o}, \boldsymbol{d})$, KiloNeRF first determines which grid it falls into based on the ray sample location $\boldsymbol{o} + t\boldsymbol{d}$. The NeRF corresponding to this grid will be given a weight of 1, while all other NeRFs will be weighted 0. In *IDW-Sample*, if we set the power γ to infinity, it means that, for each ray sample, it will only take the field information from the closest NeRF, but not a weighted sum of information from all NeRFs. Thus, only the closest NeRF becomes responsible for that sample, which resembles the case for KiloNeRF. Our method generalizes this approach, since we can freely choose a γ smaller than infinity to tune the range that each NeRF is responsible for, which results in better rendering quality (see how *IDW-Sample* in Figure 10 degrades as $\gamma \to \infty$). An ablation study of γ is provided on both the Object-centric Indoor Scenes (in the main document) and the Mission Bay Dataset (in subsection 6.1).