

Pre-NeRF 360: Enriching Unbounded Appearances for Neural Radiance Fields

Ahmad AlMughrabi

Umair Haroon

Ricardo Marques*

Petia Radeva*

Universitat de Barcelona

Abstract

Neural radiance fields (NeRF) appeared recently as a powerful tool to generate realistic views of objects and confined areas. Still, they face serious challenges with open scenes, where the camera has unrestricted movement and content can appear at any distance. In such scenarios, current NeRF-inspired models frequently yield hazy or pixelated outputs, suffer slow training times, and might display irregularities, because of the challenging task of reconstructing an extensive scene from a limited number of images. We propose a new framework to boost the performance of NeRF-based architectures yielding significantly superior outcomes compared to the prior work. Our solution overcomes several obstacles that plagued earlier versions of NeRF, including handling multiple video inputs, selecting keyframes, and extracting poses from real-world frames that are ambiguous and symmetrical. Furthermore, we applied our framework, dubbed as "Pre-NeRF 360", to enable the use of the Nutrition5k dataset in NeRF and introduce an updated version of this dataset, known as the N5k360 dataset. The source code, the dataset, and pre-trained weights for Pre-NeRF are publicly available at¹.

1 Introduction

Addressing the persistent issue of view synthesis in 3D scene reconstruction and rendering using 2D images has received a lot of attention in the literature [4, 5, 24, 25,

¹<https://amughrabi.github.io/prenerf>

*These authors contributed equally to this work.



Figure 1: Visual results of (a) Mip-NeRF 360 and (b) Pre-NeRF 360: our framework captures finer details in certain areas of the image, such as tree leaves, ground textures, etc and achieved more improved and better performance in just few number of iterations.

39]. Recently, neural volumetric representations, specifically Neural Radiance Fields (NeRF) [25], emerged as a promising approach for learning to represent 3D objects and scenes from an input set of 2D images. After the training phase, NeRF is able to map 5D input coordinates (consisting of a position in 3D scene-space and a 2D spherical direction representing the radiance flow direction) to scene properties such as color and density. This radiance field information is then used to synthesize novel views of the learned 3D scene by resorting to direct volume rendering techniques.

Several methods based on NeRF were proposed to improve view synthesis. For instance, these NeRF variations methods differ in many aspects, such as positional encoding [4], sampling strategy [44], network architecture [5], etc. Yet, these NeRF variations methods aggregate colors and densities of discrete points along viewing rays via differentiable volumetric rendering to synthesize novel views termed NeRF-based methods [25].

The NeRF technique has led to significant improvements in producing realistic visual representations [25] thanks to an MLP-based approximation of the scene radiance field. However, the point-based sampling of the radiance field approximation employed by NeRFs often leads to rendering artefacts at different resolutions. To address this issue, Mip-NeRF [4] expanded upon NeRF by utilizing volumetric frustums along a cone or cylinder to reason instead. Despite the quality enhancement, NeRF and mip-NeRF encounter important challenges in unbounded scenes, where the camera can face any direction and the scene content can be at varying distances.

Rendering extensive unbounded scenes using NeRF architectures raises three crucial challenges: *parameterization*, *efficiency* and *ambiguity* [5].

Parameterization: refers to defining a set of parameters or constraints that can represent the space of a 3D scene. Allocating more capacity to nearby objects and less to distant ones is crucial for accurate rendering unbounded scenes. NeRFs are successful in pairing specific scene types with appropriate 3D parameterizations. Scenes unbounded in all directions require different parameterizations, as explored by NeRF++ [46] and DONeRF [26], which shrink distant points towards the origin. Mip-NeRF 360 extends this idea to Mip-NeRF by presenting a way to apply smooth parameterization to volumes and introducing their parameterization for unbounded scenes.

Efficiency: In terms of efficiency, more extensive and detailed scenes require more network capacity. Still, the process of densely querying a large MLP along each ray during training can be expensive in terms of time and resources. Therefore, NeRF and Mip-NeRF-like architectures struggle with complex and large scenes and require costly training time. Mip-NeRF 360 trained two Multi-Layer Perceptrons (MLPs) (Proposal MLP and NeRF MLP) instead of a single NeRF MLP supervised at multiple scales. The proposal MLP predicts volumetric density, which is used to resample intervals for a NeRF MLP to render the image. The Proposal MLP is not supervised by the input image, but rather by the histogram weights generated by the NeRF MLP. This approach enables us to leverage the advantages of a small proposal MLP frequently evaluated alongside a large NeRF MLP evaluated relatively few times. Consequently, the total capacity of Mip-NeRF 360 is significantly greater than that of Mip-

NeRF ($\sim 15\times$), whereas the training time also slightly increases ($\sim 2\times$) [5]. It also accelerates the training by 300% with greatly improved rendering quality. Numerous attempts have been made to speed up the rendering of a trained NeRF by distilling or compressing it [15, 30, 43], but these methods do not affect the training process. Even though an octree acceleration structure [32] is constructed while optimizing a NeRF-like model in the Neural Sparse Voxel Fields [22] method, it does not result in a significant reduction in training time [5].

Ambiguity: One of the main limitations of NeRF-like architectures arises from the difficulty of creating a 3D model from multiple 2D images and generating high-quality views from novel camera angles. There are many NeRFs that explain away the input images, but only a few approaches succeed in producing satisfactory outcomes for novel views [15, 25, 28, 48]. However, these solutions tackle different problems than Mip-NeRF 360, such as non-smooth surfaces and slow rendering [5]. Moreover, these constraints are created explicitly for the point samples that NeRF utilizes. In contrast, Mip-NeRF 360 is intended to operate with the continuous weights specified along every Mip-NeRF ray. Unfortunately, several NeRF-based techniques are even incapable of processing unbounded 360-degree scenes. For instance, RegNeRF [27] cannot render such scenes correctly, resulting in empty or dark-brown frames since these NeRF-like architectures are not designed to handle unbounded spherical scenes.

To eliminate ambiguity in our proposed data model, we introduce a *Reduction Detection layer* that filters out all the input images that are blurry, noisy or redundant. Choosing to remove the corrupted frames instead of recovering them is related to handling real-world scenes, where our methodology aims to determine the high amount of information as lowest as the number of frames. However, incorporating methods like Deblurring [37, 7], Image-Super Resolution [40] or Image Restoration [20, 19] may have adverse effects on the resilience of our approach. When reconstructing the frame, various obstacles may arise, such as evident rectangular artifacts in the resulting frames, the omission of critical small details in the scene, the accumulation of error percentages in these methods into the final NeRF-like model, memory limitations, and most significantly, a negative impact on the system's overall speed. Additionally, these meth-

ods may rely heavily on previous knowledge to enhance images, which imposes further constraints on NeRF-like approaches.

Our Reduction Detector layer uses two filters to detect Defocus Blur and near-Image Similarity in 2D input images. Fast Fourier Transformation (FFT) [2] is used with the Laplacian method to handle Defocus Blur. Whereas for near-Image Similarity, the Perceptual Hashing (P-Hash) [16] algorithm is used with Hamming distance thresholding. These techniques help to improve the input data’s accuracy and reliability by identifying and removing redundant, blurry, noisy images. These filtered and clean input images are then passed to Camera Pose Estimation (CPE) to extract more accurate and precise poses and features of the selected unbounded scene.

Camera Pose Estimation is essential to feed NeRF with the viewing direction (θ, Φ) . Camera Poses Estimation is a computer vision technique that determines the camera’s position and orientation in 3D space relative to a scene. It is essential for various applications such as augmented reality [3], 3D reconstruction [12], and robotics [36]. Accurate camera pose estimation is critical for generating photo-realistic rendering of 3D scenes from NeRF. Several techniques are available, including Colmap [35], SuperGlue [34], Hierarchical localization (Hloc) [33], and Pixel-Perfect Structure-from-Motion (PixSfM) [21], which can accurately estimate the camera poses even in challenging scenarios. All NeRF-like methods used Colmap [35] as a preprocessing step for estimating the viewing direction (θ, Φ) . Colmap is a modern Structure from Motion (SfM) technique that uses various advanced strategies to improve reconstruction. These strategies include geometric verification, next-best view selection, robust triangulation, iterative outlier filtering, and efficient bundle adjustment parameterization via redundant view mining. Colmap is more powerful and complete than current SfM techniques [35] while maintaining efficiency. Generally, Structure from Motion (SfM) fails when the scene exhibits symmetries and duplicated structures [8, 42], where it also applied on Colmap, as shown in Fig. 5a. However, prior works [21, 33, 34] aim to fix this problem where they still use Colmap as a backbone. In our proposed framework, we propose to integrate into the NeRF framework the Pixel-Perfect Structure-from-Motion (PixSfM) [21] as it addresses the complex scene appearance issue with high precision and

shows a reasonable speed and adaptive solution to cope with memory limitation.

PixSfM is an advanced computer vision framework that uses innovative refinement techniques to improve the accuracy of SfM by aligning low-level image information from multiple views. It leverages the capabilities of several robust networks and is built upon the solid foundation of Hloc [33] and SuperGlue [34] architectures, with Colmap as a base. PixSfM improves the accuracy of camera poses and scene geometry by enhancing the initial key-point locations and refining points, and camera poses as post-processing. It is highly robust, even in challenging scenarios, and is a versatile tool for enhancing the accuracy of SfM. Therefore, in our proposed data model, we replaced Colmap with the more advanced and versatile CPE technique, PixSfM.

In summary, in this paper, we designed a framework to address the challenges faced by NeRF-like methods in generating realistic renderings of unbounded scenes, as shown in Fig. 1. The main **contributions** of this article are:

1. We introduce a new Reduction deduction layer in the Pre-NeRF 360 framework to address the ambiguity issue that arises in unbounded scenes. This layer incorporates two optimized filters, Defocus Blur and near-Image Similarity eliminating noisy, blurry and redundant input images.
2. We adopt an advanced and innovative camera pose estimation technique called PixSfm [21] instead of the usually used Colmap tool for Camera Pose Estimation [35]. PixSfm enables us to obtain more intricate and accurate camera pose and features, which helps any NeRF-like architectures to generate highly detailed and photo realistic renderings of unbounded scenes. We show that our framework based on PixSfm is able to extract dynamic and high-level features, and differentiate between extracted camera positions that correspond to common features, reducing significantly the ambiguity in camera pose estimation.
3. We offer an enhanced and upgraded version of the public Nutrition5k dataset called N5k360. We enabled it to be accepted for all NeRF-like architecture in order to create 3D view and rendering of the

dishes. The N5k360 dataset contains all Nutrition5k dishes, and will be done publicly available after publishing the paper to be utilized with any NeRF-like architecture.

4. We conducted a comprehensive set of experiments using the Mip-NeRF 360 and our proposed N5k360 dataset. In comparison with Mip-NeRF 360 dataset, our proposed framework achieved slightly improved and better performance with very fewer iterations. We validated the results using three evaluation metrics (PSNR, SSIM, LPIPS) improving the state of the art performance.

2 Proposed Methodology

This section presents a detailed description of our framework. Our study focuses on how to improve the NeRF input data (blurred, noisy, and redundant input images and pose estimations) in order to alleviate the ambiguity problem of NeRF-based models.

2.1 Overview

Fig. 2 shows a diagram representing the general overview of our proposed approach. The input of our model is a set of videos (or a collection of images) that is used to extract a set of key frames represented by a subsampled set of data (e.g. every k^{th} frame) from each video. Our framework is composed of three main parts: reduction detector, camera pose estimation and converter. Within the reduction detector part, **Defocus Blur reduction** module removes all the blurry frames from the list of sampled frames; followed by a **near-Image Similarity reduction** module removing the duplicate frames from the blurry-free sampled frames. Consequently, a **Camera Poses Estimation** module extracts the low detailed features and match them, resulting in the cameras' locations (i.e. poses). Lastly, a **Converter** module reformats the Camera poses information and the frames into parsable NeRF-based formats such as LLFF or Blender.

2.2 Preliminaries: NeRF for View Synthesis

NeRFs [25] can be formalized as a function F_Θ which takes as input a continuous 5D coordinate and yields a color and a density at that input location, such that:

$$F_\Theta : (\mathbf{x}, d) \rightarrow (c, \sigma),$$

where $\mathbf{x} = (x, y, z)$ is a 3D spatial location, $d = (\theta, \Phi)$ represents a spherical direction, $c = (r, g, b)$ is the color at the 5D input coordinate and σ is the corresponding volume density.

NeRFs are typically parameterized as an MLP, which captures the 5D radiance field of a given 3D scene. This inherently volumetric information is then projected onto the image plane defined by a virtual camera to synthesize a new (unseen) photo-realistic image. The typical approach uses direct volume rendering, where virtual rays r with origin o at the virtual camera position are cast towards the scene. The number of rays equals the number of pixels in the synthetic image. Their direction d is set so that each ray $r(t) = o + td$ passes through the center of each image pixel, with $t \geq 0$ being the ray parameter determining the current position along the given ray. The color of each pixel is then computed by marching along its corresponding ray and collecting the color c and volume density σ values at a set of discrete locations. Finally, the collected values are used to compose the final pixel color. This process can be formalized as [25]:

$$\hat{C}(r) = \sum_{i=1}^N T_i (1 - \exp(-\sigma_i \delta_i)) c_i,$$

with $T_i = \exp \left(- \sum_{j=1}^{i-1} \sigma_j \delta_j \right),$

where N refers to the number of sampled points across the ray, $\delta_i = t_{i+1} - t_i$ refers to the distance between two adjacent samples, c_i and σ_i refer to the per-point radiance and density, and T_i refers to the accumulated transmittance.

Since NeRF is differentiable, it is possible to define a loss function that allows NeRF training to minimize the mean-squared error (MSE) between the predicted renderings and the corresponding ground-truth colors:

$$\mathcal{L}_{MSE} = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \left\| \hat{C}(r) - C(r) \right\|^2$$

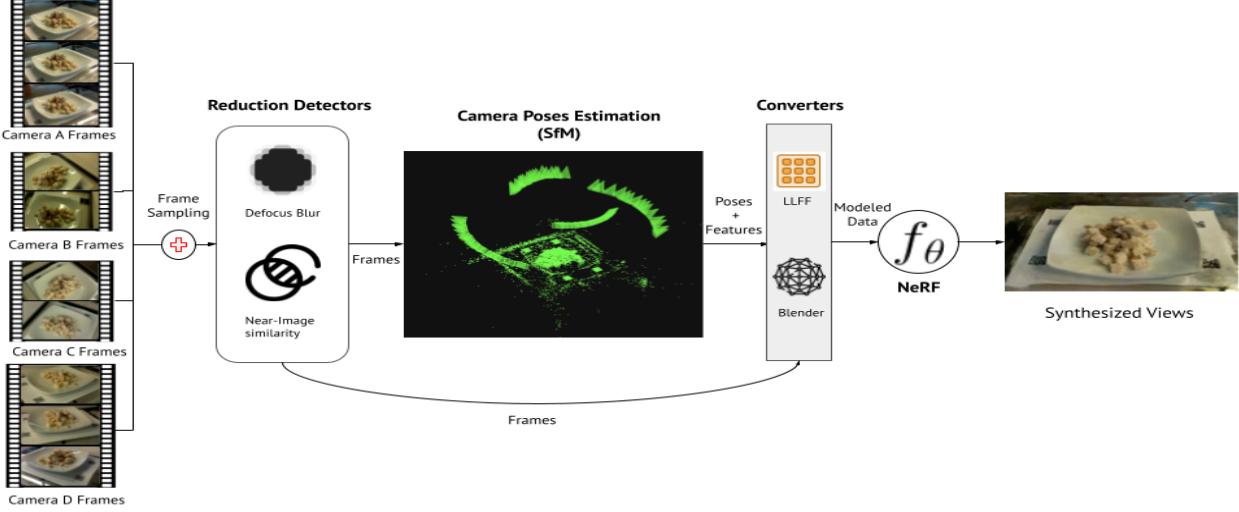


Figure 2: Our proposed framework diagram, which outlines the entire workflow from a set of videos to any NeRF-like application. The data representation used in the diagram consists of four input videos that were taken from the Nutrition5k dataset.

where \mathcal{R} refers to the batch of the randomly sampled rays that belong to one or all training images, while $\hat{C}(r)$ and $C(r)$ refer to the ground truth and output color of ray r . Notably, this per-pixel optimization approach lacks holistic spatial understanding and makes NeRF sensitive to disturbance in pixel intensity.

Camera Pose Estimation: Since the input of NeRF is a continuous 5D coordinate containing a point in the space and the camera pose, CPE is used to compute the camera pose from a set of videos or frames. Assume CPE is a function called E that accept a set of frames (X):

$$E(X) = I(x_i, \theta, \Phi)$$

and returns $I(x_i, \theta, \Phi)$, which is the estimated view direction (θ, Φ) with the associated frame, $x_i \in X$.

PixSfM [21] computes features by using deep CNNs. PixSfM provides direct alignment of low-level image information from multiple views: it first adjusts initial keypoint locations prior to any geometric estimation, and subsequently refine points and camera poses as a post-processing. In particular, PixSfM adjusts both keypoints and bundles, before and after reconstruction, by direct

image alignment in a learned feature space. Exploiting this locally-dense information is significantly more accurate than geometric optimization, while deep, high-dimensional features extracted by a CNN ensure wider convergence in challenging conditions. The approach first refines the 2D keypoints only from tentative matches by optimizing a direct cost over dense feature maps. The second stage operates after SfM and refines 3D points and poses with a similar feature metric cost. Thus, the formulation elegantly combines globally-discriminative sparse matching with locally-accurate dense details. The refinement is robust to large detection noise and appearance changes, due to the optimization of the feature metric error based on dense features predicted by a neural network.

2.3 Pre-NeRF 360

Our framework aims to improve the input of NeRF framework in order to overcome mentioned above artefacts. Let us consider as input multiple videos X from different cameras or the same camera, as shown in Fig. 2:

$$X = \{X_i | i \in (1, n)\}$$

where X is a set of videos, X_i is the i -th video, where $i \in [0, n]$, and n is the number of videos in X .

Key frame extraction: In order to minimise the redundant frames and speed up the process we first subsample the videos namely, from each video we select every k^{th} frame obtaining $X' = S(X_i, k)$. Since duplicates in frames causes instability and less robustness in the model training [49], the subsampling process reduces the likelihood of duplicates in the input data.

Defocus blur reduction D_{Red} : When the inputs are corrupted, such as compressed in JPEG format or blurred due to motion, the reconstructed scenes may show apparent artefacts. Corruptions often occur during the real-world capture and preprocessing stages. It potentially seems to lead to inaccurate reconstruction. Still, the more essential and interesting question —*how different corruption severity and types impact the robustness of NeRF*— is still an open field to explore.

In order to tackle this problem, we propose to measure image sharpness in the frequency domain using the Fast Fourier Transform (FFT) with a Blur degree threshold h_b [10]:

$$X'' = B(\forall x_i \in X') = \begin{cases} \text{remove, if } FFT(x_i) \leq h_b \\ \text{keep, otherwise.} \end{cases}$$

Near-Image Similarity Reduction: The frame sampling approach of the key frame generates a list of images $X_{red} \subseteq X$ that still could have frames redundancy. This can occur when the camera takes longer than k^{th} frames to move while recording. We apply a **near-image similarity** [38] method to detect the duplicates among X' , especially to handle too smooth or slow camera movement behaviour. To detect near-image similarity, we apply Perceptual hashing function (P_{Hash}) [45] that generates a unique fingerprint for each frame based on its content. P_{Hash} examines the features of a frame and generates a 64-bit number fingerprint. Then, all hashes are constructed in BKTree [6] data structure to find "the closest" hashes. After that, we use Hamming Distance (HD) to find the P closest hashes corresponding to similar image frames.

Assume H denotes a hash function which takes one frame as a given input and return a binary string of length l . Assume x indicates a particular frame in X' , and \hat{x} denotes a modified version of this frame which is "per-

ceptually similar" to x . Assume y denotes a frame that is "perceptually different" from x in X' . Assume x' and y' denote hash values. $0/1^l$ represents binary strings of length l . Then the four wanted properties of a perceptual hash are identified as follows:

(i) Equal distribution (unpredictability) of hash values:

$$P(H(x) = x') \approx \frac{1}{2^l}, \forall x' \in 0/1^l$$

where P stands for probability and l for the length of hash code.

(ii) Perceptually different frames x and y exhibit pairwise independence:

$$P(H(x) = x' | H(y) = y') \approx P(H(x) = x'), \forall x', y' \in 0/1^l$$

(iii) Perceptually similar frames x and \hat{x} exhibit invariance:

$$P(H(x) = H(\hat{x})) \approx 1$$

(iv) Perceptually different frames x and y exhibit distinction:

$$P(H(x) = H(y)) \approx 0$$

Assume the HD ($D(u, v)$) is counting the number of index (i) differences between u and v , where u and v are binary strings of length l :

$$D(u | H(x), v | H(y)) = |\{i : u_i \neq v_i, i = 1, \dots, l\}|$$

Assume $X_{red} = N(X', h_s)$ [16] is a function that finds all near-similarity frames within a threshold, h_s on the HD:

$$\begin{aligned} X_{red} &= N(\forall x_i \in X', \forall y_i \in X') \\ &= \{y_i | P(H(x_i), H(y_i)) \approx 1, x_i \neq y_i\} \end{aligned}$$

where it returns a set of frames duplicates forming X_{red} , $X_{red} \subseteq X'$, and for each $x_i \subseteq X_{red}$ there is an $y_i \subseteq X'$ that is a duplicate of x_i .

In order to achieve the efficient NeRF performance, we remove X_{red} from X'' as follows:

$$X''' = N_{Red}(\forall x_i \in X'') = \begin{cases} \text{remove} & \text{if } x_i \in X_{red} \\ \text{keep} & \text{otherwise} \end{cases}$$

where the frames in X''' are obtained removing all frames with a similar hash code controlled by the HD threshold h_s (frames with a smaller Hash distance than h_s are

considered redundant and removed). Obtaining X''' with a distance lower than h_s leads to high frames overlapping. Note that ensuring the scene has balanced (high enough, but not redundant) data is still an open question for NeRFs.

Note that obtaining X''' is exponentially impacted by the number of frames in X'' ; cleaning up the de-focus blurred images before entering the near-Image Similarity Reduction step leads to significantly lower computation and better reduction speed.

2.4 N5k360

In this article, we propose a new and advanced 360 food dataset named N5k360. N5k360 is an improved and enhanced version of the Nutrition5k dataset that is compatible with all NeRF architectures. Our aim was to construct a 360° viewed 5,000 realistic food dishes that are created from Nutrition5k dataset, as mentioned in fig 4.

The Nutrition5k dataset contains visual and nutritional information for 5,000 realistic food dishes. These dishes were captured from Google cafeterias using a custom scanning rig. The scanning rig is equipped with four Raspberry Pi cameras (A, B, C, and D) strategically positioned to capture a 360-degree view of the dishes. Cameras A and D are mounted opposite each other at the top of the rig, while cameras B and C are positioned on the sides to cover the remaining areas of the dish.

Being Nutrition5K a real dataset referring to a complex real domain i.e. food domain, we encountered numerous issues and challenges when adapting the Nutrition5k dataset for NeRF, including low-resolution videos, errors caused by humans and cameras, and blurry and redundant images, as shown in Fig.3. Since the dishes were recorded using four Raspberry Pi cameras, the resulting low-resolution videos harmed the PSNR value. Furthermore, the footage contained numerous blurry and redundant images due to the camera's brief recording pauses. Moreover, the videos also included some human errors, such as hands, mobile phones, and lagging in the recorded videos. Therefore, firstly we filtered these video images from the human errors and then passed them to our proposed data model where we applied the reduction detection layer, which filters the blurry, noisy and redundant images followed by the CPE and NeRF.

In order to generate the N5k360 dataset, we extend our

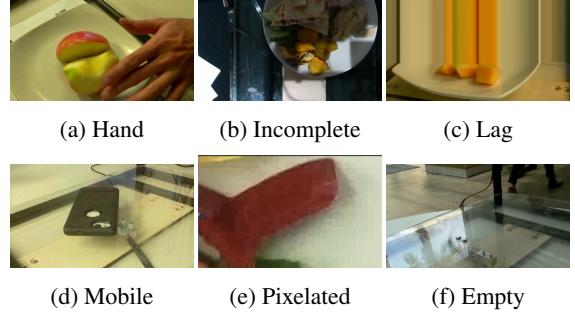


Figure 3: Examples of problematic and challenging issues occurred while dealing with Nutrition5k dataset.

framework by including an image-rotation pipeline designed to rotate inverted frames by 180° degrees. The purpose of this pipeline is to ensure proper synchronization of all frames captured by the recording cameras, as shown in Figure 4. In some cases, it was necessary to repeat the workflow with a higher threshold h_b as the CPE process led to undesired outcomes, such as blurry frames. Insufficient features extracted from the frames caused the CPE to fail in estimating camera poses for all frames in a scene, leading to exclusion from the feature matching by the SfM.

3 Experimental Results

In this section we comment the Mip-NeRF 360 dataset, the implementation setting, the results and the comparison of PixSfM to Colmap CPE. Our proposed framework was evaluated on both multifaceted datasets, Mip-NeRF 360 and N5k360. As usual for NeRF evaluation, we provide the results for three error metrics, namely PSNR, SSIM and LPIPS. The Training Time (hrs), Model Parameters (M), number of Iterations and the Equivalent number of Iterations to TPU are also presented for all the scenes in the datasets.

3.1 Dataset

The Mip-NeRF 360 dataset [5] comprises 9 different scenes, including 5 outdoor and 4 indoor scenes. Each

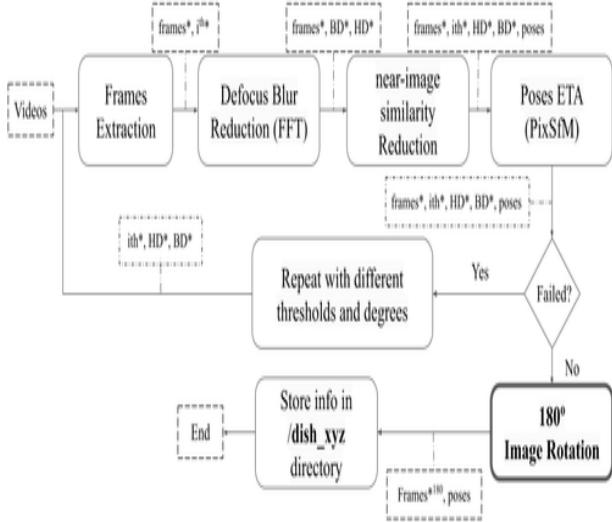


Figure 4: Our framework with an additional step of data rotation 180° , to apply it on the Nutrition5k dish videos.

scene features a sophisticated central area and intricate background details. To capture these scenes, specific measures were taken to minimize photometric variations by fixing camera exposure settings, minimizing lighting changes and avoiding moving objects. The examples of these scenes are illustrated in Fig.6.

3.2 Implementation setting

We used TPU v2 with 32 cores [17], and linear scaling rule [13] to fit our model configurations with NVIDIA GeForce RTX 3090/24G. Our model has 4096 as a batch size, a learning rate that is annealed log-linearly from 5×10^{-4} to 5×10^{-6} , and 1×10^6 iterations. Simultaneously, we allocated 2 TPU v2 with 7 cores for each node without using the linear scaling rule for 3 days.

3.3 Pre-NeRF 360 Results

We evaluate our model on the two datasets: Mip-NeRF 360 dataset and the N5k360 dataset. In Table 1, we present the PSNR, SSIM [41], LPIPS [47], Time (hours), and Iterations mean values for all NeRF-like methods for the nine scenes. One can note that our model outperforms

	PSNR↑	SSIM↑	LPIPS↓	Time(hrs)	Params	Iters
[11, 25]	23.85	0.605	0.451	4.16	1.5M	250k
[26]	24.03	0.607	0.455	4.59	1.4M	250k
[4]	24.04	0.616	0.441	3.17	0.7M	250k
[46]	25.11	0.676	0.375	9.45	2.4M	250k
[14]	23.70	0.666	0.318	-	-	250k
[18]	23.71	0.735	0.252	-	-	250k
[31]	25.33	0.771	0.211	-	-	250k
[4]*	26.19	0.748	0.285	22.71	9.0M	250k
[46]*	26.39	0.750	0.293	19.88	9.0M	250k
[5]	27.69	0.792	0.237	6.89	9.9M	250k
[5]+	26.26	0.786	0.237	6.90	9.9M	250k
Ours	26.96	0.776	0.201	15.80	9.0M	78,75k

Table 1: A quantitative comparison of our model with the SOTA on the Mip-NeRF 360 dataset. (*) denotes the NeRF-like method with Bigger MLP, while (+) denotes to NeRF-like method with Generative Latent Optimization (GLO) [23].

all NeRF-like methods by 1.18x in LPIPS. Note that our model needs 3.17x less number of iterations. Moreover, our model outperforms Mip-NeRF 360 [5] w/GLO by 2.59%, while Mip-NeRF 2.64% in PSNR. In contrast, our model outperforms all NeRF-like methods, except Mip-NeRF 360 and Mip-NeRF 360 w/GLO in SSIM outperforming our model by 2.02% and 1.27%, accordingly.

To validate our framework on the N5k360 dataset, we randomly selected the whole set of videos for 8 dishes from the Nutrition5k dataset and evaluated them on our proposed framework. These 8 dishes are shown in Fig.7, whereas the evaluations are shown in Table 2. We present the PSNR, SSIM [41], LPIPS [47], Time (hours), and Iterations values for our model for randomly selected 8 scenes from N5k360 dataset. Our model shows in average of 24.25 PSNR, 0.81 SSIM, and 0.13 LPIPS, while the training time is 55 minutes for 50k iterations.

3.4 PixSfM vs Colmap CPE Comparison

Based on our experiments, we found that the classical frame-matching NeRF paradigm for CPE, i.e. Colmap, detects key points per frame once and for all, which

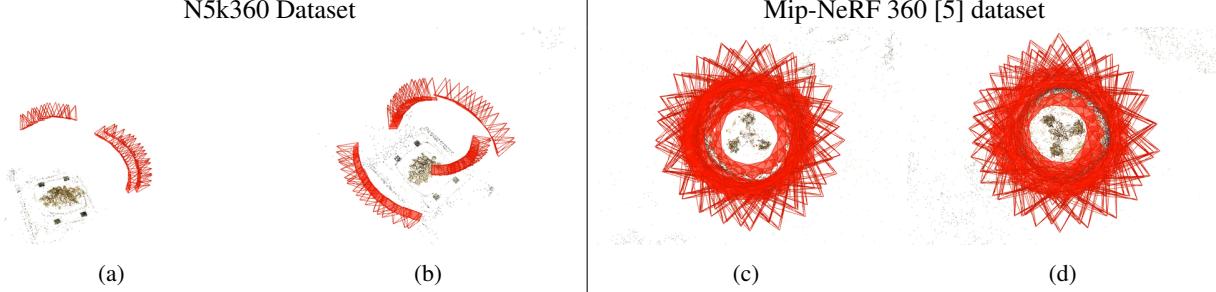


Figure 5: Comparison between Colmap & PixSfM feature extractions: (a) The extracted features of Colmap using the N5k360 dataset. (b) The extracted features of PixSfM using N5k360 dataset. (c) The extracted features of Colmap using Mip-NeRF 360 dataset. (d) The extracted features of PixSfM using Mip-NeRF 360 dataset.

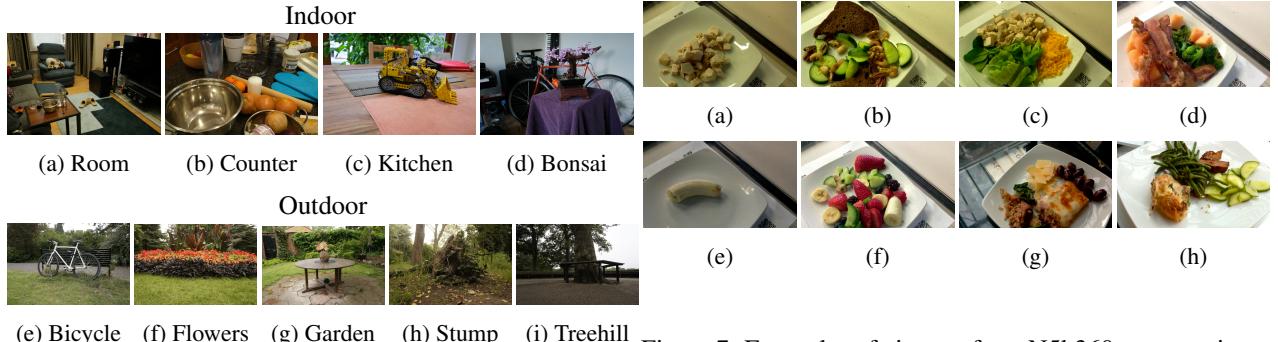


Figure 6: Examples of pictures from the Mip-NeRF 360 [5] dataset, pictures Figs. (6e- 6i) are representing Outdoor scenes showing much more challenging scenes for the NeRF. In contrast, Figs. (6a- 6d) are representing the Indoor scenes, which show lower degree challenges than Outdoor scenes.

can yield poorly-localized features and propagate significant errors to the final geometry. Furthermore, Colmap fails to detect low-level frame information, more precisely, the keypoints from multiple views, which makes Colmap’s matching process failing to find any proper matches. Moreover, it gives view directions (θ, Φ) with a different range of error margins. For instance, when the camera symmetrically captured a low-level information indoor scene such as an N5k360 dish, the estimated camera poses are overlapped as shown in Fig. 5.a, while

PixSfM CPE can detect them precisely as shown in Fig. 5.b. For instance, considering the amount of localized features, our experiments show that Colmap found fewer features as shown in 5.c, while PixSfM CPE found much richer features.

4 Conclusions and Future Work

In this paper, we presented a stable data framework for volume rendering that enhances NeRF-like frameworks. We found that a well-preparing complex and challenging data for a NeRF-like models is error-prone and time-consuming task. Using a Reduction detection layer with Defocus blur and near image similarity detection allows to filter most NeRF challenges faced by unbounded scenes. Furthermore, we adopted a more advanced and efficient

Dish ID	PSNR↑	SSIM↑	LPIPS↓	Time(hrs)	Params	Iters
50704750	25.55	0.85	0.133	0.55	9.0M	50k
50710793	24.53	0.80	0.12	0.55	9.0M	50k
50712459	25.57	0.82	0.11	0.55	9.0M	50k
50772617	26.86	0.86	0.11	0.55	9.0M	50k
50775219	26.86	26.86	0.11	0.55	9.0M	50k
50777256	24.99	0.82	0.11	0.55	9.0M	50k
61575996	17.54	0.60	0.22	0.55	9.0M	50k
61664061	22.10	0.81	0.11	0.55	9.0M	50k
Average	24.25	0.81	0.13	0.55	9.0M	50k

Table 2: A quantitative results of our model using randomly selected scenes from N5k360.

camera pose estimation technique represented by PixSfM, enabling us to extract more detailed and precise poses from the filtered images, that is critical for the NeRF in order to assure feasible and robust results. Our data model outperformed the performance of the majority of NeRF-like frameworks after just a few training iterations. Additionally, we also utilized our data model to create N5k360 dataset that is the first volumetric food representation employing the proposed NeRF-based framework.

Our findings shed important light on the stability of NeRF-based models. Still, the more essential and interesting question —*how balanced and imbalanced camera locations for a 360 scene impact the robustness of NeRF*— is still open. Moreover, more corruption reduction methods can be integrated to extend our Reduction Detector module. We hope this work can help providing a backbone and stimulating additional studies to create more reliable NeRF-like systems for real-world applications.

References

- [1] Lameck Mbangula Amugongo, Alexander Kriebitz, Auxane Boch, and Christoph Lütge. Mobile computer vision-based applications for food recognition and volume and calorific estimation: A systematic review. In *Healthcare*, volume 11, page 59. Multidisciplinary Digital Publishing Institute, 2023.
- [2] SS Aravindh, A Gopi, Gummadi Likhith Chowdary, Kvr Bhagavath, and Desaboina Rahul Srinivas. Implementation of blur image to sharp image conversion using lapla-
- cian approach. In *2022 3rd International Conference on Smart Electronics and Communication (ICOSEC)*, pages 339–345. IEEE, 2022.
- [3] Lewis Baker, Jonathan Ventura, Tobias Langlotz, Shazia Gul, Steven Mills, and Stefanie Zollmann. Localization and tracking of stationary users for augmented reality. *The Visual Computer*, pages 1–18, 2023.
- [4] 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 IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021.
- [5] 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/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022.
- [6] Walter A. Burkhard and Robert M. Keller. Some approaches to best-match file searching. *Communications of the ACM*, 16(4):230–236, 1973.
- [7] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part VII*, pages 17–33. Springer, 2022.
- [8] Zhaopeng Cui and Ping Tan. Global structure-from-motion by similarity averaging. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 864–872, 2015.
- [9] Radhamadhab Dalai, Nibedita Dalai, and Kishore Kumar Senapati. An accurate volume estimation on single view object images by deep learning based depth map analysis and 3d reconstruction. *Multimedia Tools and Applications*, pages 1–24, 2023.
- [10] Kanjar De and V Masilamani. Image sharpness measure for blurred images in frequency domain. *Procedia Engineering*, 64:149–158, 2013.
- [11] Boyang Deng, Jonathan T Barron, and Pratul P Srinivasan. Jaxnerf: an efficient jax implementation of nerf, 2020. URL <https://github.com/google-research/google-research/tree/master/jaxnerf>.
- [12] Chen Feng, Haojia Li, Fei Gao, Boyu Zhou, and Shaojie Shen. Predrecon: A prediction-boosted planning framework for fast and high-quality autonomous aerial reconstruction. *arXiv preprint arXiv:2302.04488*, 2023.
- [13] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large mini-

- batch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.
- [14] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (TOG)*, 37(6):1–15, 2018.
- [15] Peter Hedman, Pratul P Srinivasan, Ben Mildenhall, Jonathan T Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5875–5884, 2021.
- [16] Tanuj Jain, Christopher Lennan, Zubin John, and Dat Tran. Imagededup. <https://github.com/idealo/imagededup>, 2019.
- [17] Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th annual international symposium on computer architecture*, pages 1–12, 2017.
- [18] Georgios Kopanas, Julien Philip, Thomas Leimkühler, and George Drettakis. Point-based neural rendering with per-view optimization. In *Computer Graphics Forum*, volume 40, pages 29–43. Wiley Online Library, 2021.
- [19] Jingyun Liang, Jiezheng Cao, Yuchen Fan, Kai Zhang, Rakesh Ranjan, Yawei Li, Radu Timofte, and Luc Van Gool. Vrt: A video restoration transformer. *arXiv preprint arXiv:2201.12288*, 2022.
- [20] Jingyun Liang, Yuchen Fan, Xiaoyu Xiang, Rakesh Ranjan, Eddy Ilg, Simon Green, Jiezheng Cao, Kai Zhang, Radu Timofte, and Luc Van Gool. Recurrent video restoration transformer with guided deformable attention. *arXiv preprint arXiv:2206.02146*, 2022.
- [21] Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. Pixel-perfect structure-from-motion with featuremetric refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5987–5997, 2021.
- [22] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020.
- [23] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7210–7219, 2021.
- [24] Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul P Srinivasan, and Jonathan T Barron. Nerf in the dark: High dynamic range view synthesis from noisy raw images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16190–16199, 2022.
- [25] 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. *Communications of the ACM*, 65(1):99–106, 2021.
- [26] Thomas Neff, Pascal Stadlbauer, Mathias Parger, Andreas Kurz, Joerg H Mueller, Chakravarty R Alla Chaitanya, Anton Kaplanyan, and Markus Steinberger. Donerf: Towards real-time rendering of compact neural radiance fields using depth oracle networks. In *Computer Graphics Forum*, volume 40, pages 45–59. Wiley Online Library, 2021.
- [27] Michael Niemeyer, Jonathan T Barron, Ben Mildenhall, Mehdi SM Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5480–5490, 2022.
- [28] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5589–5599, 2021.
- [29] Andrea Pieroni and Lisa Price. *Eating and healing: traditional food as medicine*. Crc Press, 2006.
- [30] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14335–14345, 2021.
- [31] Gernot Riegler and Vladlen Koltun. Stable view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12216–12225, 2021.
- [32] Hanan Samet. *The design and analysis of spatial data structures*, volume 85. Addison-wesley Reading, MA, 1990.
- [33] 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/CVF Conference on Computer Vision and Pattern Recognition*, pages 12716–12725, 2019.
- [34] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020.

- [35] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016.
- [36] Seungbo Shim, Seong-Won Lee, Gye-Chun Cho, Jin Kim, and Sung-Mo Kang. Remote robotic system for 3d measurement of concrete damage in tunnel with ground vehicle and manipulator. *Computer-Aided Civil and Infrastructure Engineering*, 2023.
- [37] Lei Sun, Christos Sakaridis, Jingyun Liang, Qi Jiang, Kailun Yang, Peng Sun, Yaozu Ye, Kaiwei Wang, and Luc Van Gool. Mefnet: Multi-scale event fusion network for motion deblurring. *arXiv preprint arXiv:2112.00167*, 2021.
- [38] KK Thyagarajan and G Kalaiarasi. A review on near-duplicate detection of images using computer vision techniques. *Archives of Computational Methods in Engineering*, 28:897–916, 2021.
- [39] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5481–5490. IEEE, 2022.
- [40] Xintao Wang, Liangbin Xie, Chao Dong, and Ying Shan. Real-esrgan: Training real-world blind super-resolution with pure synthetic data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1905–1914, 2021.
- [41] 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.
- [42] Qingen Yan, Long Yang, Ling Zhang, and Chunxia Xiao. Distinguishing the indistinguishable: Exploring structural ambiguities via geodesic context. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3836–3844, 2017.
- [43] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5752–5761, 2021.
- [44] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021.
- [45] Christoph Zauner. Implementation and benchmarking of perceptual image hash functions. 2010.
- [46] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020.
- [47] 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*, pages 586–595, 2018.
- [48] Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, Paul Debevec, William T Freeman, and Jonathan T Barron. Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics (TOG)*, 40(6):1–18, 2021.
- [49] Stephan Zheng, Yang Song, Thomas Leung, and Ian Goodfellow. Improving the robustness of deep neural networks via stability training. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 4480–4488, 2016.

Appendix A Additional Results

In this appendix, we present an extra set of results and discussion which could not be included in the main paper due to space constraints. These extra results further support the superior performance of our approach with respect to the state of the art. Finally, we would like to recall that, as mentioned in Sec. 3.2 of the main paper, the presented results were generated with a much smaller number of training iterations, compared to the state-of-the-art.

A.1 Datasets

As discussed in the main paper, we validated the performance of our framework on both, Mip-NeRF 360 [5] and N5k360 datasets. Our model specializes in unbounded scenes, i.e., scenes that are not restricted by lighting and temporary obstructions constraints, as illustrated in Fig. 3 in the main paper. Therefore, the datasets we considered provide complex unbounded scenes that represent a strong test for our proposal. This is in contrast with the dataset used in the “NeRF in the Wild” [23] approach that targets unconstrained scenes where the scene is captured under uncontrolled settings, which causes limitations that prevent it from accurately representing numerous common, real-life occurrences in unregulated images, such as changes in lighting or temporary obstructions.

The Mip-NeRF 360 dataset is a collection of images that capture a wide range of complex scenes from various vantage points, offering unparalleled levels of detail and contextual information. On the other hand, the N5k360 dataset is food-centric, presenting a 360 view of food from four different cameras with two distinct angles. Rendering such datasets can be intricate and perplexing for NeRF-like architectures, as their camera pose estimation approach (Colmap [35]) fails to extract rich and correct poses from multiple videos of different camera angles, as shown in Fig 5 in the main paper. Our proposed framework (Pre-NeRF 360) successfully handled both datasets and utilized their respective complexities and flexibilities to generate highly detailed and precise 360-degree renderings and depth maps for all scenes, as shown in Fig 8. Furthermore, Pre-NeRF 360 effectively handles a large number of input images, as demonstrated by its performance with the Mip-NeRF 360 dataset, as well as with

Classes	# Images	Resolution
Bicycle	194	4946×3286
Bonsai	291	3118×2078
Counter	238	3115×2076
Flowers	173	5025×3312
Garden	184	5187×3361
Kitchen	270	3115×2078
Room	307	3114×2075
Stump	124	4978×3300
Treehill	141	5068×3326

Table 3: mip-NeRF 360 Dataset Description

Classes	# Images	Resolution
Dish (50704750)	65	1920×1080
Dish (50710793)	61	1920×1080
Dish (50712459)	62	1920×1080
Dish (50772617)	59	1920×1080
Dish (50775219)	62	1920×1080
Dish (50777256)	59	1920×1080
Dish (61575996)	20	1920×1080
Dish (61664061)	22	1920×1080

Table 4: N5k360 dataset description.

a smaller number of input images, as exemplified by the N5k360 dataset.

The detailed information of both datasets regarding scene types, number of images and resolution, is presented in Table 3 and Table 4. Table 3 shows that the Mip-NeRF 360 dataset contains an average of 214 images per scene, with a maximum of 307 images for the *room* indoor scene and a minimum of 124 images for the *stump* outdoor scene. In contrast, Table 4 reveals that our N5k360 dataset contains an average of 52 images per scene, with a maximum of 65 images for the *Dish (50704750)* scene and a minimum of 20 images for the *Dish (61575996)* scene. Interestingly, the average number of images in the Mip-NeRF 360 dataset is 214% larger than that of our N5k360 dataset. Furthermore, the average resolution of a scene in the Mip-NeRF 360 dataset is 4185×2766, while our N5k360 dataset has an average resolution of 1920×1080, meaning that the Mip-NeRF 360 dataset has

a resolution that is $5.88\times$ higher than that of our eight scenes in the N5k360 dataset.

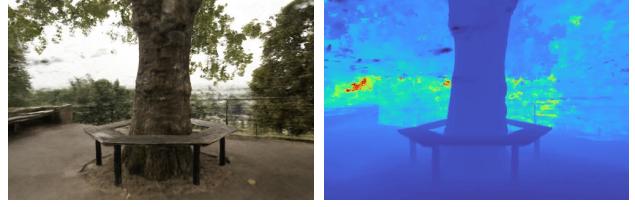
The main objective of creating the N5k360 dataset is to establish a comprehensive food dataset with a 360° view, that can maximize the utility of NeRF-based techniques in confronting a variety of food-related challenges, such as food volume estimation [1], Food 3D Reconstruction [9], and Food as medicine [29] etc. To demonstrate the effectiveness of our approach, we randomly selected eight 360° nutrition dishes and used them to train our Mip-NeRF 360 model. Detailed quantitative and qualitative evaluations are presented in Sec. 3.

A.2 Qualitative and quantitative results

In order to emphasize the qualitative and quantitative results of both datasets, Fig 9 presents visual renderings of all 9 classes from the Mip-NeRF 360 dataset, including their evaluation metrics such as PSNR, SSIM, and LPIPS. Our framework shows a blurry rendering for the highly textured areas like the green grass and intensely detailed ground or leaves, as seen in Fig. 9d and Fig 9d. On the other hand, in indoor scenes, our framework outperforms all NeRF-like methods for room and counter scenes (as shown in Table 5), and illustrated as in Fig. 9d and Fig. 9c, while for Kitchen (Fig. 9d) and Bonsai (Fig. 9b) it competes with the NeRF state-of-the-arts.

Similarly, Fig 10 highlights the rendered frames of the 8 evaluated dishes from the N5k360 dataset, along with their corresponding evaluation metrics. We trained our framework on the N5k360 dataset for 50,000 iterations per scene. Despite the relatively low number of iterations, our model produces high-quality renderings for scenes such as those depicted in Fig. 10a, Fig. 10b, Fig. 10c, Fig. 10d, and Fig. 10f. Interestingly, for some scenes with a small number of frames, our model shows lower rendering performance, as shown in Fig. 10e, Fig. 10g, and Fig. 10h. In addition, our framework is capable of recognizing foreground and background objects, resulting in well-defined depth maps, as shown in Fig 8.

Table 5 provides a comprehensive comparison of the Mip-NeRF 360 dataset with state-of-the-art methods, categorized by class. This table also highlights the top three performers for each evaluation metric. Upon reviewing Table 5, we observed that our framework, despite having a low number of iterations, is capable of competing with



(a) Treehill rendered frame (b) Treehill depth map

Figure 8: An example of the qualitative results produced by Pre-NeRF 360: a rendered frame accompanied by its corresponding depth map.

the current state of the art. Additionally, it has also exhibited a higher PSNR in 5 of the 9 scenes (3 outdoor and 2 indoor). Moreover, it is worth noting that our model experiences a degradation in performance when handling the flowers and treehill scenes that contain intricate details in the background, such as grass, tree leaves, and highly textured grounds. Furthermore, our analysis of the SSIM reveals that in 5 out of the 9 scenes (3 outdoor and 2 indoor), our framework outperforms all NeRF-like approaches, while it shows comparable performance to the NeRF-like methods in the kitchen (marked in orange) and bicycle (marked in yellow) scenes. For LPIPS, we outperform all NeRF-like methods in five scenes (1 outdoor and 4 indoor) and compete with the Stable View Synthesis (SVS) [31] and Mip-NeRF 360 [5] in the other 4 scenes. Our model utilizes a per-pixel reconstruction loss during the training process, whereas SVS employs a self-supervised approach that directly minimizes a perceptual loss, similar to LPIPS. This fundamental difference in the training methodology could explain the observed difference in the results.

For more detailed information, we have provided rendering and depth videos in the supplemental materials.

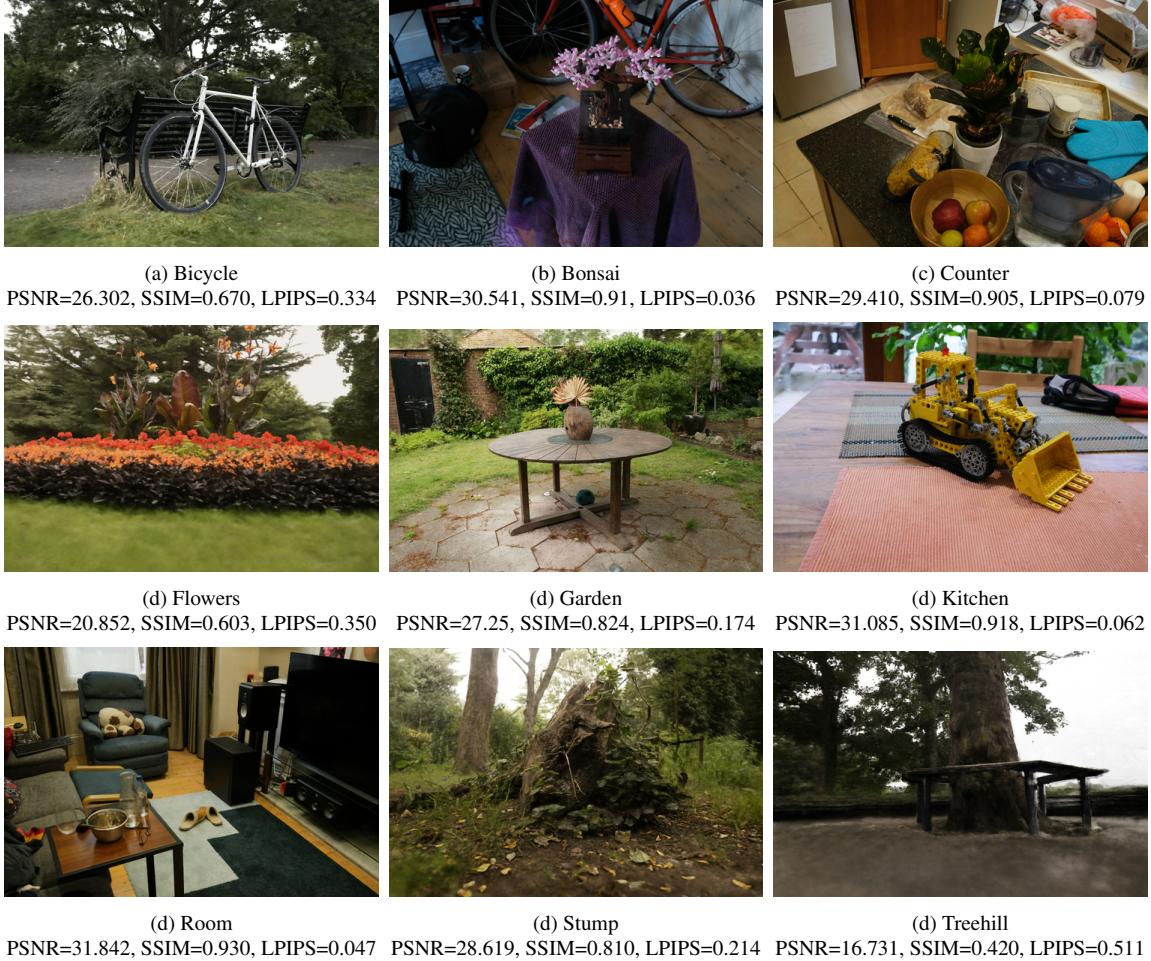


Figure 9: The qualitative results of our framework, including each scene's associated mean PSNR, SSIM, and LPIPS metrics. Our findings reveal that the treehill and flowers scenes were the most challenging of all the scenes.



Figure 10: The qualitative results produced by our framework on the proposed N5k360 dataset. Each scene is accompanied by its associated mean PSNR, SSIM, and LPIPS metrics. Our findings demonstrate that Fig. 10g and Fig. 10h were the most challenging among the eight scenes. This is because their backgrounds contain more information compared to the other scenes, as they feature a paper underneath the dish. Additionally, the N5k360 dataset shares similar challenges with "the wild" [23].

	PSNR ↑								
	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai
NeRF [11, 25]	21.76	19.40	23.11	21.73	21.28	28.56	25.67	26.31	26.81
NeRF w/ DONeRF [26] param.	21.67	19.48	23.29	23.38	21.70	28.28	25.74	25.42	27.32
mip-NeRF [4]	21.69	19.31	23.16	23.10	21.21	28.73	25.59	26.47	27.13
NeRF++ [46]	22.64	20.31	24.32	24.34	22.20	28.87	26.38	27.80	29.15
Deep Blending [14]	21.09	18.13	23.61	24.08	20.80	27.20	26.28	25.02	27.08
Point-Based Neural Rendering [18]	21.64	19.28	22.50	23.90	20.98	26.99	25.23	24.47	28.42
Stable View Synthesis [31]	22.79	20.15	25.99	24.39	21.72	28.93	26.40	28.49	29.07
mip-NeRF [4] w/bigger MLP	22.90	20.79	25.85	23.64	21.71	30.67	28.61	29.95	31.59
NeRF++ [46] w/bigger MLPs	23.75	21.11	25.91	25.48	22.77	30.13	27.79	29.85	30.68
Mip-NeRF 360 [5]	24.37	21.73	26.98	26.40	22.87	31.63	29.55	32.23	33.46
Mip-NeRF 360 [5] w/GLO	23.95	21.60	25.09	25.98	21.99	28.24	28.40	30.81	30.27
Our Model	26.30	20.85	27.24	28.61	16.73	31.84	29.40	31.086	30.54
	SSIM ↑								
	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai
NeRF [11, 25]	0.455	0.376	0.546	0.453	0.459	0.843	0.775	0.749	0.792
NeRF w/ DONeRF [26] param.	0.454	0.379	0.542	0.522	0.461	0.841	0.776	0.678	0.813
mip-NeRF [4]	0.454	0.373	0.543	0.517	0.466	0.851	0.779	0.745	0.818
NeRF++ [46]	0.526	0.453	0.635	0.594	0.530	0.852	0.802	0.816	0.876
Deep Blending [14]	0.466	0.320	0.675	0.634	0.523	0.868	0.856	0.768	0.883
Point-Based Neural Rendering [18]	0.608	0.487	0.735	0.651	0.579	0.887	0.868	0.876	0.919
Stable View Synthesis [31]	0.663	0.541	0.818	0.683	0.606	0.905	0.886	0.910	0.925
mip-NeRF [4] w/bigger MLP	0.612	0.514	0.777	0.643	0.577	0.903	0.877	0.902	0.928
NeRF++ [46] w/bigger MLPs	0.630	0.533	0.761	0.687	0.597	0.883	0.857	0.888	0.913
Mip-NeRF 360 [5]	0.685	0.583	0.813	0.744	0.632	0.913	0.894	0.920	0.941
Mip-NeRF 360 [5] w/GLO	0.687	0.582	0.800	0.745	0.619	0.907	0.890	0.916	0.932
Our Model	0.670	0.603	0.824	0.810	0.420	0.931	0.904	0.918	0.910
	LPIPS ↓								
	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai
NeRF [11, 25]	0.536	0.529	0.415	0.551	0.546	0.353	0.394	0.335	0.398
NeRF w/ DONeRF [26] param.	0.542	0.539	0.436	0.492	0.545	0.368	0.394	0.410	0.368
mip-NeRF [4]	0.541	0.535	0.422	0.490	0.538	0.346	0.390	0.336	0.370
NeRF++ [46]	0.455	0.466	0.331	0.416	0.466	0.335	0.351	0.260	0.291
Deep Blending [14]	0.377	0.476	0.231	0.351	0.383	0.266	0.258	0.246	0.275
Point-Based Neural Rendering [18]	0.313	0.372	0.197	0.303	0.325	0.216	0.209	0.160	0.178
Stable View Synthesis [31]	0.243	0.317	0.137	0.281	0.286	0.182	0.168	0.125	0.164
mip-NeRF [4] w/bigger MLP	0.372	0.407	0.205	0.357	0.401	0.229	0.239	0.152	0.204
NeRF++ [46] w/bigger MLPs	0.356	0.395	0.223	0.328	0.386	0.270	0.270	0.177	0.230
Mip-NeRF 360 [5]	0.301	0.344	0.170	0.261	0.339	0.211	0.204	0.127	0.176
Mip-NeRF 360 [5] w/GLO	0.296	0.343	0.173	0.258	0.338	0.208	0.206	0.129	0.182
Our Model	0.334	0.350	0.174	0.214	0.511	0.047	0.078	0.062	0.036

Table 5: A detailed version of Table 1 in the main paper which provides metrics for individual scenes and assesses the performance of our framework alongside various NeRF and non-NeRF baselines. Even though some scenes are more challenging than others, the relative rankings of all techniques across each scene tend to align with the average metrics ranking. The top result is denoted by the color red, the second by orange, and the third by yellow.