Re-Nerfing: Improving Novel Views Synthesis through Novel Views Synthesis

Felix Tristram[*,1]   Stefano Gasperini[*,1,2]   Nassir Navab[1]   Federico Tombari[1,3]

[1] Technical University of Munich   [2] VisualAIs   [3] Google

Abstract. Neural Radiance Fields (NeRFs) have shown remarkable novel view synthesis capabilities even in large-scale, unbounded scenes, albeit requiring hundreds of views or introducing artifacts in sparser settings. Their optimization suffers from shape-radiance ambiguities wherever only a small visual overlap is available. This leads to erroneous scene geometry and artifacts. In this paper, we propose Re-Nerfing, a simple and general multi-stage data augmentation approach that leverages NeRF's own view synthesis ability to address these limitations. With Re-Nerfing, we enhance the geometric consistency of novel views as follows: First, we train a NeRF with the available views. Then, we use the optimized NeRF to synthesize pseudo-views around the original ones with a view selection strategy to improve coverage and preserve view quality. Finally, we train a second NeRF with both the original images and the pseudo views masking out uncertain regions. Extensive experiments applying Re-Nerfing on various pipelines on the mip-NeRF 360 dataset, including Gaussian Splatting, provide valuable insights into the improvements achievable without external data or supervision, on denser and sparser input scenarios. Project page: https://renerfing.github.io.

Keywords: Novel view synthesis · Neural rendering · Data augmentation

1   Introduction

Neural Radiance Fields (NeRFs) have revolutionized 3D scene representation and rendering, enabling unprecedented quality in synthesizing novel views from a set of images [33]. NeRF optimizes a continuous volumetric scene function in the weights of two multi-layer perceptrons (MLPs), which, when queried with rays sampled from posed cameras, generate corresponding RGB values and volume densities [33]. These principles have been applied across various tasks, simplifying content creation, rendering, and reconstruction workflows.

Despite its remarkable success, NeRFs [33] have several limitations, such as slow training [34], failures when noisy or no camera poses are available [5, 30], and computationally intense rendering mostly incompatible with mobile applications [12]. While many works have been proposed to address these issues,

---
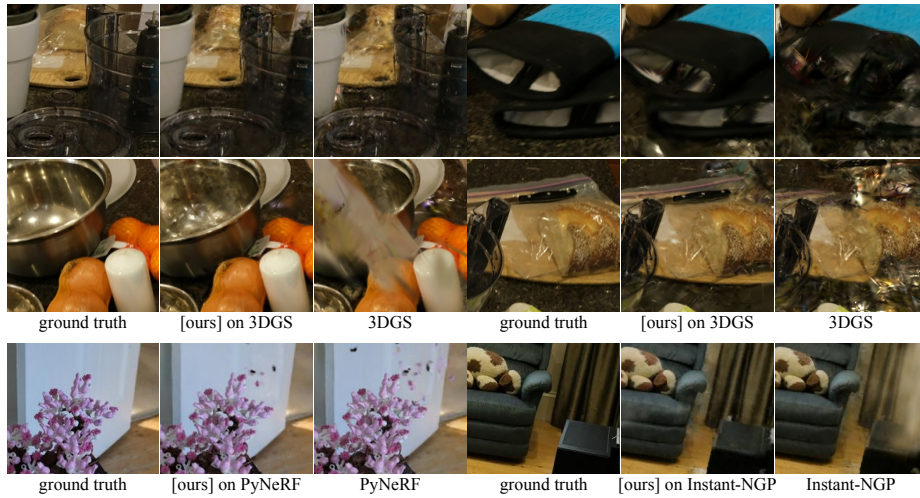
Fig. 1: Examples of synthesized views by 3D Gaussian Splatting [26], PyNeRF [52], Instant-NGP [34] with and without the proposed Re-Nerfing. Our approach improves the image quality thanks to the enforcement of geometric constraints on synthesized pseudo views. Applied to the baselines, the proposed method enables the reconstruction of finer details and improves upon the shape-radiance ambiguity. Image crops show test views from the mip-NeRF 360 [3] dataset. The models were trained on the sparser setting (30 views).

collecting enough high-quality images with sufficient overlap to ensure successful model convergence remains a challenging task. Towards this end, researchers have explored the application of NeRFs in sparse view settings, with only a handful of images available [51, 55].

Moreover, complex geometries and large-scale environments often lead to artifacts due to the inability to consistently encode structures [2, 15, 43]. This is particularly severe in sparse settings, where the limited views lead to hallucinations manifesting, e.g., as floaters in the free space due to the ambiguity of shape and radiance [15, 31, 51]. To mitigate these issues, depth and structural priors have been inserted into the optimization to leverage additional geometric information [15, 43, 51]. However, reliable dense depth estimates are also hard to obtain, and using wrongly estimated depth can further hurt performance [15].

In this work, we address these open issues with a simple and effective solution exploiting the inherent novel view synthesis capability of NeRF to improve its own performance. We achieve this with Re-Nerfing, a multi-stage, general pipeline that introduces structural constraints from synthetic views generated by a previously trained NeRF on the same scene. As shown in Figure 1, thanks to its ability to improve the view coverage while preserving quality discarding uncertain regions, Re-Nerfing enables improvements in both sparse and dense settings. The main contributions of this paper can be summarized as follows:

- With Re-Nerfing, we show how a NeRF's and Gaussian Splatting's novel view synthesis is beneficial to enhance their own rendering quality.
- We propose a simple and effective iterative augmentation technique to enhance any NeRF's outputs by training a new NeRF with the addition of synthesized views selected to improve the scene coverage and masked to discard uncertain areas.
- We provide insights on how the views should be selected and show how masking their uncertain regions can help mitigating a sub-optimal selection.
- We show the wide applicability of Re-Nerfing upon various novel view synthesis pipelines [26, 34, 52] improving in both denser and sparser settings.

2   Related Work

Given a dense set of images, traditional interpolation techniques based on light field sampling can synthesize photorealistic novel views [29]. In relatively sparser settings, neural networks have been used for view synthesis by exploiting the correlation of the visual appearance of an instance across different views [17, 57]. More recently, 3DGS [26] has shown great promise in view synthesis tasks by building on a previously reconstructed geometry and optimizing the scene as 3D Gaussians with spherical harmonics.

Neural implicit representations have emerged as a shift from traditional explicit representations like point clouds, meshes, and voxels [40, 58], by encoding 3D shapes implicitly in the weights of a neural network. The pioneering works of Occupancy Networks [32] and DeepSDF [36] initiated this line of work by encoding complex shapes in a continuous function space through a network. Unlike traditional representations, here, no space discretization is needed, enabling finer details [46].

NeRF Mildenhall et al. [33] introduced NeRF by combining neural implicit representations with differentiable rendering, encoding both volumetric properties and view-dependent appearance and leading to high-quality novel views. NeRF has catalyzed a myriad of works tackling various aspects of its limitations and further improving novel view synthesis [54]. Among these, Instant-NGP [34], FastNeRF [20] and others [8, 19, 47] enable significantly faster training speeds, BARF [30], SPARF [51] and Nope-NeRF [5] cope with imperfect or absent camera poses, mip-NeRF 360 [3] deals with unbounded scenes and Nerfies [38] among others [7, 18, 39] deals with dynamic scenes. Additionally, the work of Rössle et al. [43] delivers more consistent geometry, CoNeRF [25] enables altering the output, Panoptic Neural Fields [27] incorporates scene understanding information, Block-NeRF [48] renders city-scale scenes, and CamP [37] optimizes the camera parameters. Zip-NeRF [4] combines mip-NeRF 360 and grid-based models such as Instant-NGP to reach state-of-the-art results with fast training speeds.

Sparse settings Nevertheless, highly sparse settings are particularly challenging as the standard multi-view constraints are not sufficient [13, 51]. Strong geometric information is required to reconstruct the scene and enable satisfactory novel views. Towards this end, researchers have explored the support of additional
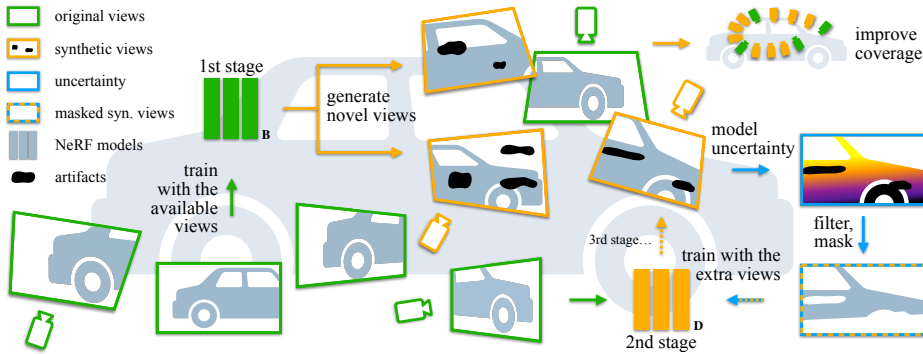
Fig. 2: Re-Nerfing is a multi-stage framework. Compatible with any NeRF and 3DGS pipeline, it operates by first training a model with the available views (green, 1st). This model is then used to generate novel views from camera poses to improve the scene coverage (orange). Then, we compute the NeRF's model uncertainty on such novel views (blue) and discard the uncertain regions (orange-blue). Finally, we use these masked views and the original ones to train a second model (orange, 2nd). The process can be repeated iteratively by generating the novel views with the second model (3rd stage).

data and external models to provide geometric supervision [43, 51] or semantic knowledge [41, 55]. Such techniques enable novel view synthesis even from a single image. SPARF [51], for example, relies on a pixel correspondence network to enforce a geometrically accurate solution. PixelNeRF [55] learns a prior from multiple scenes and conditions the novel views on the few inputs available. However, by depending on external models and large amounts of training data, they eliminate the advantage of NeRF of not requiring extensive training data but just that for the instance or scene of interest.

Geometric constraints Enforcing geometric consistency plays a crucial role in NeRF as it directly impacts the quality of the generated views [43]. DS-NeRF [15] supervises NeRF's optimization with sparse depth obtained from COLMAP [44], thereby reducing overfitting and accelerating training. Rössle et al. [43] went a step further by feeding the sparse COLMAP point cloud to a depth completion network and constraining NeRF's optimization according to the estimated depth and its associated uncertainty. Urban Radiance Fields [42] focuses on outdoor scenes where LiDAR data is available, which is used for losses that benefit surface estimation. SPARF [51] exploits multi-view geometry constraints and pixel correspondences, minimizing the re-projection error using the depth rendered by the NeRF and the pose estimates, which are optimized jointly.

Data augmentation for NeRF Data augmentation aims to increase the pool of training data with samples that are variations of the existing ones (e.g., image flip), ultimately regularizing the model. As for other computer vision tasks and settings [22, 28, 35], data augmentation techniques have also been proposed for novel view synthesis. In this domain, PANeRF [1], GeoAug [9], and VM-NeRF [6]

generate new views by warping existing ones via homography [1] or exploiting the NeRF's depth estimates [6, 9]. Instead, Aug-NeRF [11] trains more robust NeRFs with worst-case perturbations of the input coordinates, the intermediate features, and the pre-rendering outputs.

Distillation and semi-supervised learning Knowledge distillation involves transferring knowledge from a large, complex model (teacher) to a smaller, more efficient one (student) [24]. This paradigm has been widely adopted and extended to semi-supervised learning, from segmentation [10] to depth estimation [21]. Naive-student [10] predicts pseudo-labels for a set of unlabeled data and later trains a new model using the original data and the newly pseudo-labeled samples. Tosi et al. [50] achieved remarkable depth estimates by training a stereo depth model using data synthesized by NeRF. NeRFmentation [16] trains a NeRF model and uses it to generate augmented data for a monocular depth estimator.

In this work, we enhance novel view synthesis in both denser and sparser settings by exploiting NeRF's inherent view synthesis capabilities. Unlike existing approaches [42, 43, 51, 55], we do so in a data augmentation manner by using only the available views without extra data or additional models. We achieve this by training a baseline NeRF on the available data and using it to generate novel views that we add to the training data of a subsequent NeRF model. Specifically, we sample such views to improve the scene coverage while preserving the rays' quality by masking out uncertain regions. In contrast to Tosi et al. [50] and Feldmann et al. [16], who trained a depth estimator with NeRF data, we train a NeRF with other NeRF data. Then, unlike other augmentation methods [1, 6, 9], our newly introduced views are fully synthesized from a NeRF model. As for standard (e.g., flipping and cropping) and more sophisticated augmentations [22, 28, 35], which can be mixed to further improve the models [28], ours can be combined with other augmentation techniques and is not an alternative to them.

3 Preliminaries

Neural Radiance Fields represent a continuous volumetric function in the weights of a neural representation, typically a Multi-Layer-Perceptron (MLP) [33], hash-grid [34] or (tri-)plane representation [18]. The spatial coordinates of rays originating from a set of cameras with known intrinsic and extrinsic parameters are mapped to their corresponding density and color values and then rendered with traditional volume rendering techniques.

The color $C(\mathbf{r})$ of a ray $\mathbf{r} = \mathbf{o} + \mathbf{d}t$ with origin $\mathbf{o}$ and direction $\mathbf{d}$ is then aggregated as the weighted sum of each color sample between the near and far bounds $[t_n, t_f]$ of the ray as

$$C(\mathbf{r}) = \int_{t_n}^{t_f} w(t) \cdot c(t)\mathbf{d}t, \tag{1}$$

where $w(t)$ are the volumetric rendering weights of each sample t and $c(t)$ the corresponding color. The weight-terms $w(t)$ can be derived from the volumetric

density $\sigma(t)$ with:

$$w(t) = \exp\left(-\int_{t_n}^{t} \sigma(s)ds\right) \cdot \sigma(t). \tag{2}$$

Using these rendering weights the expected termination depth $\hat{D}$ of the ray $\mathbf{r}$ can then be calculated by taking the weighted sum of each sampling location t along the ray between $[t_n, t_f]$ such that:

$$\hat{D} = \int_{t_n}^{t_f} w(t) \cdot t \ \mathbf{d}t \tag{3}$$

The continuous integrals in equations (1), (2), (3) are discretized following [33].

4   Method

As shown in Figure 2, the proposed Re-Nerfing operates in a multi-stage fashion. First, a baseline NeRF is trained with the available views (Section 3), then it is used to generate novel views to improve the scene coverage (Section 4.1), and lastly, a new NeRF is trained on the original and the synthetic views (Section 4.2), discarding uncertain regions of the rendered views to improve the signal's quality. Re-Nerfing is a general framework that is compatible with any pipeline for novel view synthesis. Furthermore, Re-Nerfing follows an iterative process, so further rounds can be executed using the NeRF trained at the previous iteration as a baseline for the next one.

4.1   Augmenting via Synthesized Views

The proposed method builds on the simple, well-known observation that including more views in the training data has a positive impact on the output quality [15,51]. In general, extra views improve the representation of details and textures while reducing ambiguities (e.g., color-radiance) and artifacts. Additional viewpoints bring value due to their contribution of new data from novel perspectives that can help disambiguate the geometric and visual properties of the scene. However, such extra views may not exist. This is not only the case in sparser settings [51], e.g., due to crowd-sourced data, but also in scenarios where all images available have already been used, and it is not possible to capture more [4].

With Re-Nerfing, we take this observation a step further and mitigate the lack of extra views by adding synthesized views to the training data. After training a baseline model $\mathbf{B}$, such as the one described in Section 3, we use it to generate novel views. While the synthesis procedure itself is standard [33], we do not just generate views for rendering or evaluation purposes as commonly done but leverage them as an intermediate data augmentation step. Specifically, we exploit the model's view synthesis purpose to enhance the output quality of another model $\mathbf{D}$ trained at a later stage by adding more views of the same scene (Section 4.2).
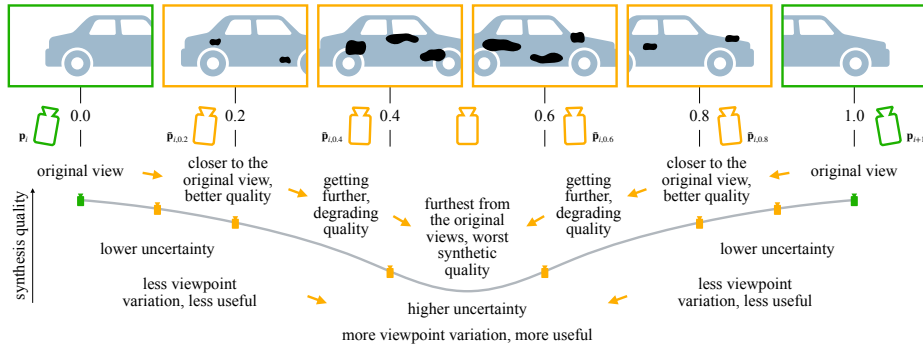
Fig. 3: Illustration of the trade-offs arising when synthesizing novel views (orange) in between existing ones (green). The numbers indicate the interpolation factors. The closer to the original views, the higher the quality of the synthesis and the lower the uncertainty. Moving further from the original views (i.e., toward 0.5), the quality degrades and the uncertainty increases. Yet, an opposite trade-off occurs as views too close to the original ones do not bring any extra information. We balance this by using many views and removing artifacts thanks to the uncertainty estimates.

Nevertheless, such synthesized views are not as good as real images, otherwise, the novel view synthesis problem would be already solved. To exploit the advantage of having more viewpoints, we not only synthesize them in the vicinity of the original views, where the displacement is small and the quality is better, but also relatively further, where the quality degrades. We illustrate such trade-offs in Figure 3. Additionally, we mitigate the quality degradation by masking out uncertain regions of the synthesized views (Section 4.2).

View coverage We seek to improve the view coverage of the captured scene or object through our augmentations. This is particularly relevant in sparser settings. Toward this end, we generate multiple novel views from camera poses interpolated across existing neighboring views. We define the hyperparameters N and Ω, being the augmentation factor and the set of $N - 1$ interpolation factors α_j, respectively. Specifically, for the simple case of equally spaced novel views:

$$\Omega = \left\{ \alpha_j \mid \alpha_j = \frac{j}{N}, \ j = 1, 2, \ldots, N - 1 \right\} \tag{4}$$

However, as depicted in Figure 3, equally spaced synthesized views are not ideal as the quality would degrade around $\alpha = 0.5$. This is particularly relevant in sparser settings where the larger camera displacement between existing poses would lead to worse synthetic views.

Considering the existing camera poses $\mathbf{P}_i$ and $\mathbf{P}_{i+1}$ with $\mathbf{P} = [\mathbf{p}, \mathbf{q}]$, such that $\mathbf{p}$ is the position vector in 3D space and $\mathbf{q}$ is the quaternion representation of the rotation,

$$\bar{\mathbf{p}}_{i,j} = (1 - \alpha_j)\mathbf{p}_i + \alpha_j \mathbf{p}_{i+1} \tag{5}$$

is the interpolation of the position component given an interpolation factor α_j. For the rotation, we compute the interpolated $\bar{\mathbf{q}}_{i,j}$ from the original pose quaternions $\mathbf{q}_i$ and $\mathbf{q}_{i+1}$ using SLERP [45] as follows:

$$\bar{\mathbf{q}}_{i,j} = \frac{\sin((1 - \alpha_j)\theta)}{\sin(\theta)}\mathbf{q}_i + \frac{\sin(\alpha_j\theta)}{\sin(\theta)}\mathbf{q}_{i+1} \tag{6}$$

Instead of interpolating across existing views, other strategies could generate novel views around a half dome centered at the scene or object center, or following a specific pattern. However, in real-world scenes, such techniques may raise issues with occlusions, as the sampled poses might collide with parts of the scene (e.g., wall) or end up inside an object. Interpolating is more conservative and does not require extra information about the scene, such as the scene occupancy, to avoid collisions and unrealistic viewpoints. Nevertheless, multiple pose sampling strategies could be combined to further enhance the view synthesis.

The formulation above is based on two consecutive poses $\mathbf{P}_i$ and $\mathbf{P}_{i+1}$. In settings where sequential information is not available, a sequence can be simulated by considering consecutive poses from neighboring positions in 3D space [14]. This can be achieved with a nearest neighbor on the vectors $\mathbf{p}$ using the Euclidean distance or the angular distance between the quaternions $\mathbf{q}$.

4.2   Re-Nerfing: Re-Training NeRF

After training the first model $\mathbf{B}$ and using it to generate novel views (Section 4.1), we train another NeRF $\mathbf{D}$ from scratch with the addition of the synthesized views. To improve the signal's quality, we mask out uncertain regions of the generated views with Bayes' Rays [23].

Uncertainty masks We seek to add a high-quality training signal to the new model $\mathbf{D}$ by means of the synthesized views. Since the quality of such views is not on par with the originals, they may display artifacts that can impact the renderings. Therefore, we want to remove such artifacts and improve the training signal. We do so by estimating the model uncertainty for each synthesized view with Bayes' Rays [23] and masking out the uncertain regions. The method involves simulating spatially parametrized perturbations of the radiance field and using a Bayesian Laplace approximation to create a volumetric uncertainty field. This field can be rendered similarly to an additional color channel. Specifically, for each ray $\mathbf{r}$ of a novel pose $\bar{\mathbf{P}}$, we synthesize the color $C_{\mathbf{B}}(\mathbf{r})$ and estimate the corresponding uncertainty $U(\mathbf{r})$. Then, we filter out the rays with an uncertainty $U(\mathbf{r})$ higher than the threshold μ.

To optimize the second model $\mathbf{D}$, we use the masked generated views as part of the training images and query their rays during the course of optimization. The intuition behind using these synthesized images is that they provide a more diverse yet reliable set of views, which allows the NeRF representation to better triangulate the underlying scene geometry.

This is especially important at the beginning of the training, where the density of the scene still varies significantly during optimization and where the

core geometrical structure of the scene is being learned. As the optimization is close to convergence, the discrepancy in image quality between the pseudo-views and the original images increases, with the former becoming detrimental to the overall reconstruction, so we remove the synthesized ones from the sampling pool.

Third and later stages The proposed method uses a base model $\mathbf{B}$ to augment the training data for a later model $\mathbf{D}$. This introduces a general paradigm that can be repeated iteratively until the gain saturates. Therefore, new augmented views can be synthesized with $\mathbf{D}$ and masked to train a new model $\mathbf{D}_2$, and so on. As our synthesized data augmentation regularizes the later models $\mathbf{D}_n$, these are less prone to overfitting on the initial views, which would allow us to increase the model size and reach higher quality. However, this is out of the scope of this work, as we use identical architectures across the iterations.

5   Experiments and Results

Since having a more diverse set of views during the start of optimization benefits reconstruction, we observe faster convergence on the test views when using Re-Nerfing. Interestingly, not only does using Re-Nerfing improve convergence speed and test-view quality but it can also boost PSNR values on the training views, further reinforcing our intuition.

5.1   Experimental Setup

Dataset and metrics We test our method on the challenging 7 public scenes introduced with mip-NeRF 360 [3], comprising both bounded indoor and large unbounded outdoor settings. The data features a large amount of real images captured all around an object or scene. Given the high amount of views available, this dataset allows us to showcase the novel view quality in denser and sparser settings. For sparser setups, we select the training views by downsampling all images available (including the test ones) to preserve the coverage of the scene. We then select the test views among those test from the original set that were not selected as training images. Toward this end, we selected 30 training images (50 for *stump* and *bicycle* due to the convergence issues of the baselines when using 30 views). The exact data splits will be made available. We evaluate the novel views on the standard metrics and errors, i.e., PSNR, SSIM [53], and LPIPS [56].

Prior Works and Baselines We showcase the effectiveness of our method by applying it on diverse novel view synthesis methods, namely PyNeRF [52], Instant-NGP [34], and 3DGS [26]. 3DGS uses an explicit representation, so there is no model uncertainty to estimate. Therefore, with 3DGS, we show only the impact of the simpler augmentation without masks.

Implementation Details We train our method on an image resolution downsampled by eight from the original image sizes for the available outdoor scenes and down-sampled by four for the indoor scenes, bringing both to a comparable image size. All models are trained using Nerfstudio [49]. For PyNeRF [52], we used the original implementation, which is compatible with Nerfstudio. Instead,

Method	sparser views			denser views		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Instant-NGP [34]	20.76	0.659	0.301	25.87	0.812	0.162
+ Re-Nerfing [ours]*	**21.40**	**0.682**	**0.270**	**25.88**	**0.814**	**0.161**
PyNeRF [52]	22.65	0.746	0.182	27.41	0.874	0.094
+ Re-Nerfing [ours]	**23.51**	**0.772**	**0.160**	**27.43**	**0.875**	**0.093**
3DGS [26]	20.49	0.600	0.259	27.83	0.855	**0.109**
+ Re-Nerfing [ours]*	**21.30**	**0.627**	**0.235**	**28.03**	**0.859**	0.111

Table 1: Rendering performance on the mip-NeRF 360 [3] dataset in denser (full) and sparser settings averaged over all scenes. The proposed Re-Nerfing is applied on the Instant-NGP [34], PyNeRF [52], and 3DGS [26] baselines, using the corresponding baseline to augment the views. * indicates that no uncertainty estimates were used.

for Gaussian Splatting [26] and Instant-NGP [34], we used their implementation available in Nerfstudio. We trained the models for 30k iterations with a batch size of 8192 rays for PyNerf and 1024 rays for Instant-NGP, and inherit all other losses and optimisation schemes of our baseline methods. The influence of the pseudo-views on the RGB loss is removed after 8k iterations for PyNerf and Gaussian-Splatting and after 200 iterations for Instant-NGP. We train all models on a single 24GB NVIDIA RTX 3090 or RTX 4090 GPU.

5.2   Quantitative Results

In Table 1, we report the results on the mip-NeRF 360 dataset [3] applying the proposed Re-Nerfing on Instant-NGP [34], PyNeRF [52], and 3D Gaussian Splatting [26], in both sparser and denser settings.

Re-Nerfing The proposed method brings notable improvements in sparser settings, increasing PSNR by 0.86 dB over PyNeRF and 0.64 dB over Instant-NGP, while improving SSIM and LPIPS as well. In sparser settings, the relatively low amount of views makes it difficult to disambiguate the scene geometry and properly render the novel views. In such settings, increasing the amount of images is particularly beneficial. Re-Nerfing generates such views with the baseline, thereby relying on the exact same original views. Yet, our iterative augmentation strategy brings valuable benefits, providing additional viewpoints on which the methods learn better representations of the scenes. For 3D Gaussian Splatting, our method brings a remarkable PSNR improvement of 0.81 dB over the baseline. Re-Nerfing aids in structuring the scene and reducing ambiguities. All sets of Gaussians were optimized from scratch with the same hyperparameters and settings. Despite their synthesized nature, adding more training views mitigates overfitting and acts as a regularizer. Such results are aligned with those of other iterative frameworks from other domains, where later models outperform earlier ones thanks to an implicit regularization effect occurring in the later stages [10, 21]. In denser settings, the benefit is reduced since the amount of original views is relatively high, with good coverage of the area of interest,

Method	3DGS [26]			PyNeRF [52]		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
baseline	19.31	0.609	0.296	17.57	0.643	0.325
baseline, 2x iterations	19.68	0.633	0.274	17.93	0.653	0.331
baseline + 1 round of [ours]	21.08	0.686	0.241	18.09	0.676	0.299
baseline + 2 rounds of [ours]	21.89	0.720	0.214	18.36	0.693	0.285
baseline + 3 rounds of [ours]	22.47	0.748	0.191	18.41	0.699	0.281
baseline + 4 rounds of [ours]	22.63	0.756	0.184	18.47	0.704	0.275
baseline + 5 rounds of [ours]	**22.76**	**0.761**	**0.178**	**18.64**	**0.708**	**0.272**

Table 2: Rendering performance on the mip-NeRF 360 [3] *counter* scene in the sparser setting (30 views). The proposed Re-Nerfing is applied iteratively multiple times (rounds) and the effect of training the baseline for double the amount of iterations is also shown (2x). All other models in this work were trained for the same amount of iterations (30k).

allowing each method to reconstruct the scene significantly better. Therefore, in such setups, the synthesized views are rather close to the original ones, so the viewpoint advantage and the information gain are small. Nevertheless, Re-Nerfing slightly improves over each of its baselines, demonstrating its efficacy. Thanks to its simplicity, Re-Nerfing is widely applicable to both implicit and explicit methods, such as PyNeRF [52] and 3DGS [26], respectively.

Re-Nerfing as an iterative method In Table 2, we show the impact of our method when applying it iteratively. As described in Section 4.2, Re-Nerfing uses renderings from a first model **B** to augment the training data of a second model **D**. To further improve the output quality, the process can be repeated iteratively, by using **D** to render the augmented views. We show this in Table 2, where at each iteration, the quality improves across the board for both pipelines, namely 3DGS [26] and PyNeRF [52]. This confirms the intuition that led to Re-Nerfing, for which novel view synthesis helps the task of novel view synthesis. While this varies from scene to scene, we notice that the performance saturates beyond 5 rounds. Overall, when applied to 3DGS [26], the proposed Re-Nerfing leads to a substantial boost of **+3.45 PSNR**, without using any extra data or external models, other than what is already available to the baseline.

Re-Nerfing vs. more training iterations Throughout this work, we optimized all models and Gaussians for 30k iterations to ensure fairness among the comparisons. By being multi-stage, applying the proposed Re-Nerfing, increases the total training time, e.g., by 2x considering a single round of Re-Nerfing, albeit resetting the optimization every time. Therefore, in Table 2, we assess the impact of doubling the number of iterations for the baselines. This results in a total number of iterations and training time equivalent to 1 round of Re-Nerfing. While doubling the iteration count improves the output quality of the baseline, this remains significantly below applying Re-Nerfing for 1 round. Such comparison showcases the significant benefits of the proposed method, which introduces synthetic novel views among the original ones.

Method	PSNR	SSIM	LPIPS
PyNeRF [52]	22.42	0.632	0.218
2x: .50	23.35	0.669	0.190
3x: .33-.66	23.40	0.666	0.191
3x: .20-.80	23.61	0.670	0.189
3x: .10-.90	23.47	0.670	0.191
4x: .25-.50-.75	23.51	0.667	0.192
5x: .2-.4-.6-.8	23.63	0.668	0.189
7x: .1-.2-.4-.6-.8-.9	**23.73**	**0.672**	**0.184**
11x	23.48	0.669	0.189

Table 3: View selection strategies on the sparser setting of the *stump* scene of the mip-NeRF 360 [3] dataset (50 views). All models are based on PyNeRF [52]. Augmentation factors N from 2 to 11 are evaluated. 11x corresponds to the α factors .05-.1-.2-.3-.4-.5-.6-.7-.8-.9-.95.

Trade-offs with pseudo-views poses In Table 3, we evaluate different poses for the novel view selection on PyNeRF [52]. All interpolate across pairs of the original views, but each does it at different locations via the interpolation factors α_j (e.g., 0.33 and 0.75). Remarkably, all augmentations are beneficial over PyNeRF, bringing a gain in PSNR between $+0.9$ and $+1.3$ dB. The results show the trade-offs depicted in Figure 3: the furthest from the original views (e.g., .33-.66 is further than 0.20-0.80), the lower the quality of the novel views, and the smaller the performance improvement. However, the opposite trade-off occurs, too: the closer from the initial images (e.g., 0.10-0.90 are closer than 0.20-0.80), the less amount of new information to gain for the multi-view optimization. Considering these trade-offs, it is better to sample more pseudo-views closer to the original ones while still including further viewpoints to add valuable information. We do so with the 7x configuration, where the increments along the interpolation are smaller around the original views and increase further away from 0.1 to 0.2.

Impact of the augmentation factor In Table 3, we also explore different augmentation factors N (e.g., 2x and 5x). The results show that adding more images (i.e., higher factors) leads to improvements despite the introduction of sub-optimal, synthesized views. Doubling the views (2x) brings the largest relative gain with $+0.9$ dB PSNR. Then, the performance further improves when the augmentation factor is increased. However, at high N factors (e.g., 11x), benefits reduce as the influence of the many synthetic views overcomes the few original images. 7x provides a good balance and leads to the best scores.

Ablation study In Table 4, we report an ablation study for both PyNeRF and Instant-NGP in the sparser setting (30 views). Training again with the synthetic views produced by the baseline (with $N = 7$) degrades the performance for all metrics for PyNeRF, while it improves for Instant-NGP. The degradation for PyNeRF is due to the lower quality signal of 6x more views than the reliable signal of the original views. Specifically, the model learns the artifacts of the augmented views and underperforms. Instead, for Instant-NGP, the quality of the augmented views was higher, leading to improvements. Nevertheless, removing the images from the sampling pool once the scene structure has been mainly learned (stop) prevents both models from overfitting to the artifacts and leads to

Method	PyNeRF [52]			Instant-NGP [34]		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
baseline	23.16	0.840	0.119	20.75	0.709	0.232
+ augmented views	23.08	0.829	0.125	21.54	0.740	0.208
+ stop aug. views	23.84	0.862	0.102	21.57	0.743	0.201
+ masks	**24.08**	**0.872**	**0.099**	**21.70**	**0.753**	**0.195**

Table 4: Ablation on the mip-NeRF 360 [3] dataset in the sparser settings of the *kitchen* scene (30 views). The proposed Re-Nerfing is applied on Instant-NGP [34] and PyNeRF [52], using the corresponding baseline to augment the views.

better results on all metrics, with large improvements for PyNeRF. Furthermore, masking out the uncertain regions of such synthetic views improves even further across the board for both methods. This shows once again how synthetic views of the same scene are beneficial aid the novel view synthesis task. In particular, the results confirm the importance of increasing the scene coverage and that of having a good training signal, testifying the need for the uncertainty-based masks and our augmentation strategy.
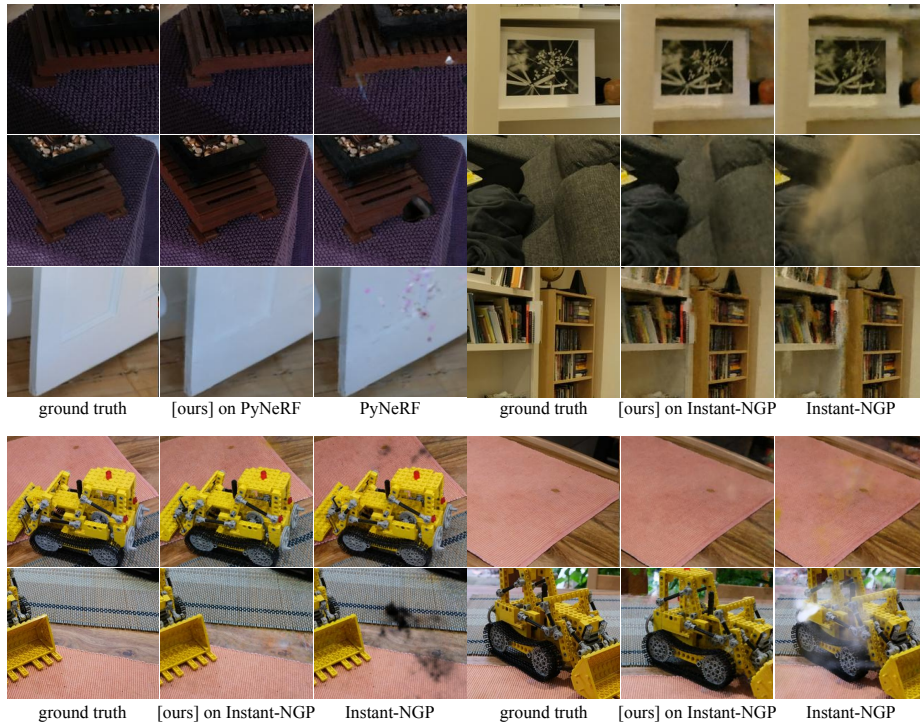
5.3   Qualitative Results

Figure 4 shows examples of synthesized images with PyNeRF [52] and Instant-NGP [34] in the sparser setting, with and without our method. As seen in the quantitative results (Section 5.2), the output quality of the proposed Re-Nerfing improves significantly upon the baselines, especially in terms of the reduction of floating artifacts ("floaters"). In particular, the addition of our method sharpens the scene and better preserves the small details, such as the table cloth in the *bonsai* scene, or the books and picture frame in the *room* scene. Remarkably, the proposed method is able to remove dominant artifacts in the renderings by Instant-NGP [34] and smaller ones from PyNeRF [52], despite using the same original views, architecture, and hyperparameters as the baselines. The augmented views help disambiguate the scene geometry, allowing both models to deliver better results.

In Figure 1, we show additional qualitative results, including examples from 3D Gaussian Splatting [26]. The improvements over the base 3DGS are remarkable. Our augmentation method brings significant improvements, resulting in sharper renderings across the board, with 3D Gaussians better fitting the scene geometry.

The efficacy of our method on both implicit (NeRF-based) and explicit scene representations is particularly notable as it is achieved without using extra data, additional models, or supervision, only using already available information.

The **supplementary material** includes additional details and results, such as the performance on each scene, the ability of Re-Nerfing to improve on specific views, and extra qualitative examples in a video.

Limitations Since our method relies on the first stage to provide reasonable renderings from a good estimate of the underlying scene geometry, it is challenging to enhance poorly rendered scenes. In extremely sparse settings where the

Fig. 4: Qualitative results on cropped images from the test set of the mip-NeRF 360 dataset. The models were trained on the sparser setting. The proposed Re-Nerfing is applied on PyNeRF [52] and Instant-NGP [34].

rendering quality is lower, combining our approach with other methods [15,51] would be an interesting direction. The proposed method also relies on the uncertainty estimates to discard artifacts and low-quality regions. Therefore, better uncertainty estimates would be beneficial in the second and later stages. Uncertainty calibration could help refine the estimates while preserving the signal from higher-quality areas.

6 Conclusion

This work enhances NeRFs and 3DGSs by leveraging their own novel view synthesis capabilities. The introduced Re-Nerfing is a general approach that synthesizes novel views to improve the scene coverage and augment the training data for a new model. To provide a valuable training signal for the second model, Re-Nerfing computes the NeRF's uncertainty for each augmented view to discard the uncertain regions and reduce artifacts. As shown with various pipelines, this simple method yields improvements in both rendering quality and convergence speed of sparser and denser scenes, mitigating the need for extensive captures of the scene.

A   Supplementary Material

In this supplementary material, we complement the main paper with additional results in greater detail. Specifically, in Section A.1 we show how our method can optimize for specific viewpoints, in Section A.2, we show the results for each scene separately, and in Section A.3 we discuss qualitative results from the attached supplementary video.

A.1   Optimizing for Specific Views

Method	sparser views		
	PSNR	SSIM	LPIPS
PyNeRF [52]	22.42	0.632	0.218
adding 4 randomly interpolated synthetic views (1.08x)	22.58	0.653	0.205
adding 4 test synthetic views (1.08x)	**22.64**	**0.663**	**0.196**

Table 5: View optimization strategies on the sparser setting of the *stump* scene of the mip-NeRF 360 [3] dataset (50 views). All models are based on PyNeRF [52].

In Table 5, we show how our method offers the opportunity of optimizing for specific views. Toward this end, we train PyNeRF [52] augmenting with 4 randomly interpolated views synthesized by the baseline and compare it with another PyNeRF model trained augmenting with 4 test views synthesized by the same baseline. Both models are evaluated on the viewpoint of the 4 test views. While adding the randomly interpolated views benefits the output quality, specifically synthesizing the viewpoints of interest (i.e., the test views in this case) shows a higher margin on all metrics. In practice, this means that Re-Nerfing allows to improve on specific viewpoints of interest, which can be beneficial when the views available lead to a sub-optimal quality from specific areas. In such setting, Re-Nerfing can be used to densify the viewpoints around the area lacking in terms of quality, and ultimately improve the novel view synthesis.

A.2   Evaluation by Scene

In Tables from 6 to 12, we report the performance for each scene for the various pipelines on which we applied Re-Nerfing, namely Instant-NGP [34], PyNeRF [52], and 3D Gaussian Splatting [26]. The detailed results showcase the benefits of our method indoor and outdoor, for sparser and denser view settings, and for both implicit and explicit methods.

A.3   Additional Qualitative Results

We point the reader to the attached supplementary video https://youtu.be/8mo3s1XAwHQ for more qualitative results. The video interpolates across the test

views. Remarkably, when paired with the proposed Re-Nerfing 3DGS [26] reaches significantly higher rendering quality, even in the sparser setting showcased in the video for the *counter* scene (30 views). Thanks to our method, the scene geometry is substantially better captured by the Gaussians, eliminating major structural artifacts. As shown in Table 9, the *garden* scene is easier to render. In this case, our method helps with the scene geometry, especially visible in the rendered depths, as it eliminates the holes in the ground.

Method	*counter* - sparser			*counter* - denser		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Instant-NGP [34]	17.72	0.557	0.458	**25.52**	0.801	0.183
+ Re-Nerfing [ours]*	**18.08**	**0.574**	**0.422**	25.50	**0.805**	**0.180**
PyNeRF [52]	17.57	0.643	0.325	26.80	0.864	0.111
+ Re-Nerfing [ours]	**18.09**	**0.676**	**0.299**	**27.13**	**0.871**	**0.107**
3DGS [26]	19.31	0.609	0.296	27.69	**0.886**	**0.119**
+ Re-Nerfing [ours]*	**21.08**	**0.686**	**0.241**	**27.70**	0.879	0.127

Table 6: Results on the *counter* scene of the mip-NeRF 360 dataset [3], for sparser and denser settings. The proposed Re-Nerfing is applied on Instant-NGP [34], PyNeRF [52], and 3D Gaussian Splatting (3DGS) [26]. *: no uncertainty estimates were used.

Method	*bicycle* - sparser			*bicycle* - denser		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Instant-NGP [34]	20.98	0.477	0.370	23.30	0.625	0.301
+ Re-Nerfing [ours]*	**21.32**	**0.503**	**0.344**	**23.35**	**0.630**	**0.300**
PyNeRF [52]	22.13	0.608	0.234	24.92	0.779	0.154
+ Re-Nerfing [ours]	**22.96**	**0.643**	**0.200**	**25.19**	**0.784**	**0.150**
3DGS [26]	17.48	0.320	0.403	25.37	**0.775**	**0.136**
+ Re-Nerfing [ours]*	**18.36**	**0.324**	**0.377**	**25.39**	0.774	0.149

Table 7: Results on the *bicycle* scene of the mip-NeRF 360 dataset [3], for sparser and denser settings. The proposed Re-Nerfing is applied on Instant-NGP [34], PyNeRF [52], and 3D Gaussian Splatting (3DGS) [26]. *: no uncertainty estimates were used.

Method	stump - sparser			stump - denser		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
PyNeRF [52]	22.42	0.632	0.218	26.47	0.810	0.117
+ Re-Nerfing [ours]	**23.75**	**0.672**	**0.188**	**26.51**	**0.814**	**0.112**
3DGS [26]	19.72	0.409	0.311	23.81	0.666	0.177
+ Re-Nerfing [ours]*	**19.97**	**0.433**	**0.306**	**24.32**	**0.691**	**0.172**

Table 8: Results on the *stump* scene of the mip-NeRF 360 dataset [3], for sparser and denser settings. The proposed Re-Nerfing is applied on PyNeRF [52] and 3D Gaussian Splatting (3DGS) [26]. Instant-NGP [34] could not deal with this scene in the sparser setting (PSNR of 15.55), so it was not applied here. *: no uncertainty estimates were used.

Method	garden - sparser			garden - denser		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Instant-NGP [34]	23.03	0.733	0.163	25.54	0.829	0.111
+ Re-Nerfing [ours]*	**23.45**	**0.745**	**0.149**	**25.64**	**0.831**	**0.108**
PyNeRF [52]	25.80	0.841	0.073	**28.26**	**0.900**	**0.052**
+ Re-Nerfing [ours]	**26.18**	**0.845**	**0.071**	27.88	0.897	0.055
3DGS [26]	21.97	0.670	0.140	27.44	0.889	0.063
+ Re-Nerfing [ours]*	**23.18**	**0.730**	**0.106**	**27.93**	**0.900**	**0.048**

Table 9: Results on the *garden* scene of the mip-NeRF 360 dataset [3], for sparser and denser settings. The proposed Re-Nerfing is applied on Instant-NGP [34], PyNeRF [52], and 3D Gaussian Splatting (3DGS) [26]. *: no uncertainty estimates were used.

Method	bonsai - sparser			bonsai - denser		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Instant-NGP [34]	19.41	0.699	0.335	**26.96**	**0.894**	**0.098**
+ Re-Nerfing [ours]*	**19.63**	**0.718**	**0.299**	26.77	0.892	0.100
PyNeRF [52]	23.61	0.837	0.137	**29.40**	**0.932**	**0.063**
+ Re-Nerfing [ours]	**24.70**	**0.854**	**0.118**	29.37	0.931	**0.063**
3DGS [26]	**23.32**	**0.743**	**0.212**	**30.88**	**0.938**	**0.078**
+ Re-Nerfing [ours]*	23.09	0.705	0.217	30.61	0.931	0.087

Table 10: Results on the *bonsai* scene of the mip-NeRF 360 dataset [3], for sparser and denser settings. The proposed Re-Nerfing is applied on Instant-NGP [34], PyNeRF [52], and 3D Gaussian Splatting (3DGS) [26]. *: no uncertainty estimates were used.

Method	*kitchen* - sparser			*kitchen* - denser		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Instant-NGP [34]	20.75	0.709	0.232	**26.28**	0.837	**0.135**
+ Re-Nerfing [ours]*	**21.57**	**0.743**	**0.201**	25.97	**0.839**	0.136
PyNeRF [52]	23.16	0.840	0.119	**28.77**	**0.919**	**0.069**
+ Re-Nerfing [ours]	**24.08**	**0.872**	**0.099**	28.62	0.916	0.073
3DGS [26]	21.95	0.770	0.149	30.13	**0.928**	**0.062**
+ Re-Nerfing [ours]*	**22.56**	**0.797**	**0.130**	**30.21**	0.928	0.064

Table 11: Results on the *kitchen* scene of the mip-NeRF 360 dataset [3], for sparser and denser settings. The proposed Re-Nerfing is applied on Instant-NGP [34], PyNeRF [52], and 3D Gaussian Splatting (3DGS) [26]. *: no uncertainty estimates were used.

Method	*room* - sparser			*room* - denser		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Instant-NGP [34]	22.68	0.778	0.248	27.62	0.883	0.143
+ Re-Nerfing [ours]*	**24.33**	**0.806**	**0.203**	**28.07**	**0.886**	**0.139**
PyNeRF [52]	23.85	0.825	0.168	27.24	**0.913**	0.092
+ Re-Nerfing [ours]	**24.81**	**0.841**	**0.146**	**27.28**	0.911	**0.091**
3DGS [26]	19.67	0.677	0.301	29.50	0.903	0.129
+ Re-Nerfing [ours]*	**20.89**	**0.715**	**0.271**	**30.02**	**0.907**	**0.128**

Table 12: Results on the *room* scene of the mip-NeRF 360 dataset [3], for sparser and denser settings. The proposed Re-Nerfing is applied on Instant-NGP [34], PyNeRF [52], and 3D Gaussian Splatting (3DGS) [26]. *: no uncertainty estimates were used.

References

1. Ahn, Y.C., Jang, S., Park, S., Kim, J.Y., Kang, N.: PANeRF: Pseudo-view augmentation for improved neural radiance fields based on few-shot inputs. arXiv preprint arXiv:2211.12758 (2022) 4, 5

2. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5855–5864 (2021) 2

3. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5470–5479 (2022) 2, 3, 9, 10, 11, 12, 13, 15, 16, 17, 18

4. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Zip-NeRF: Anti-aliased grid-based neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 19697–19705 (2023) 3, 6

5. Bian, W., Wang, Z., Li, K., Bian, J.W., Prisacariu, V.A.: Nope-NeRF: Optimising neural radiance field with no pose prior. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4160–4169 (2023) 1, 3

6. Bortolon, M., Del Bue, A., Poiesi, F.: VM-NeRF: Tackling sparsity in NeRF with view morphing. In: Proceedings of the International Conference on Image Analysis and Processing. pp. 63–74. Springer (2023) 4, 5

7. Cao, A., Johnson, J.: HexPlane: A fast representation for dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 130–141 (2023) 3

8. Chen, A., Xu, Z., Geiger, A., Yu, J., Su, H.: Tensorf: Tensorial radiance fields. In: Proceedings of the European Conference on Computer Vision. Springer (2022) 3

9. Chen, D., Liu, Y., Huang, L., Wang, B., Pan, P.: GeoAug: Data augmentation for few-shot nerf with geometry constraints. In: Proceedings of the European Conference on Computer Vision. pp. 322–337. Springer (2022) 4, 5

10. Chen, L.C., Lopes, R.G., Cheng, B., Collins, M.D., Cubuk, E.D., Zoph, B., Adam, H., Shlens, J.: Naive-student: Leveraging semi-supervised learning in video sequences for urban scene segmentation. In: Proceedings of the European Conference on Computer Vision. pp. 695–714. Springer (2020) 5, 10

11. Chen, T., Wang, P., Fan, Z., Wang, Z.: Aug-NeRF: Training stronger neural radiance fields with triple-level physically-grounded augmentations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15191–15202 (2022) 5

12. Chen, Z., Funkhouser, T., Hedman, P., Tagliasacchi, A.: MobileNeRF: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16569–16578 (2023) 1

13. Choi, I., Gallo, O., Troccoli, A., Kim, M.H., Kautz, J.: Extreme view synthesis. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7781–7790 (2019) 3

14. Cover, T., Hart, P.: Nearest neighbor pattern classification. Transactions on Information Theory **13**(1), 21–27 (1967) 8

15. Deng, K., Liu, A., Zhu, J.Y., Ramanan, D.: Depth-supervised NeRF: Fewer views and faster training for free. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12882–12891 (2022) 2, 4, 6, 14

16. Feldmann, C., Siegenheim, N., Hars, N., Rabuzin, L., Ertugrul, M., Wolfart, L., Pollefeys, M., Bauer, Z., Oswald, M.R.: NeRFmentation: NeRF-based augmentation for monocular depth estimation. arXiv preprint arXiv:2401.03771 (2024) 5
17. Flynn, J., Neulander, I., Philbin, J., Snavely, N.: DeepStereo: Learning to predict new views from the world's imagery. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5515–5524 (2016) 3
18. Fridovich-Keil, S., Meanti, G., Warburg, F.R., Recht, B., Kanazawa, A.: K-planes: Explicit radiance fields in space, time, and appearance. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12479–12488 (2023) 3, 5
19. Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance fields without neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5501–5510 (2022) 3
20. Garbin, S.J., Kowalski, M., Johnson, M., Shotton, J., Valentin, J.: FastNeRF: High-fidelity neural rendering at 200FPS. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 14346–14355 (2021) 3
21. Gasperini, S., Morbitzer, N., Jung, H., Navab, N., Tombari, F.: Robust monocular depth estimation under challenging conditions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8177–8186 (2023) 5, 10
22. Ghiasi, G., Cui, Y., Srinivas, A., Qian, R., Lin, T.Y., Cubuk, E.D., Le, Q.V., Zoph, B.: Simple copy-paste is a strong data augmentation method for instance segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2918–2928 (2021) 4, 5
23. Goli, L., Reading, C., Selllán, S., Jacobson, A., Tagliasacchi, A.: Bayes' Rays: Uncertainty quantification for neural radiance fields. arXiv preprint arXiv:2309.03185 (2023) 8
24. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015) 5
25. Kania, K., Yi, K.M., Kowalski, M., Trzciński, T., Tagliasacchi, A.: CoNeRF: Controllable neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18623–18632 (2022) 3
26. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3D Gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4), 1–14 (2023) 2, 3, 9, 10, 11, 13, 15, 16, 17, 18
27. Kundu, A., Genova, K., Yin, X., Fathi, A., Pantofaru, C., Guibas, L.J., Tagliasacchi, A., Dellaert, F., Funkhouser, T.: Panoptic neural fields: A semantic object-aware neural scene representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12871–12881 (2022) 3
28. Lehner, A., Gasperini, S., Marcos-Ramiro, A., Schmidt, M., Navab, N., Busam, B., Tombari, F.: 3D adversarial augmentations for robust out-of-domain predictions. International Journal of Computer Vision pp. 1–33 (2023) 4, 5
29. Levoy, M., Hanrahan, P.: Light field rendering. In: Proceedings of the Conference on Computer Graphics and Interactive Techniques. p. 31–42. SIGGRAPH '96, Association for Computing Machinery (1996) 3
30. Lin, C.H., Ma, W.C., Torralba, A., Lucey, S.: BARF: Bundle-adjusting neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5741–5751 (2021) 1, 3
31. Martin-Brualla, R., Radwan, N., Sajjadi, M.S., Barron, J.T., Dosovitskiy, A., Duckworth, D.: NeRF in the wild: Neural radiance fields for unconstrained photo collections. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7210–7219 (2021) 2

32. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019) 3

33. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: Representing scenes as neural radiance fields for view synthesis. In: Proceedings of the European Conference on Computer Vision. pp. 405–421. Springer (2020) 1, 3, 5, 6

34. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Transactions on Graphics (ToG) 41(4), 1–15 (2022) 1, 2, 3, 5, 9, 10, 13, 14, 15, 16, 17, 18

35. Nekrasov, A., Schult, J., Litany, O., Leibe, B., Engelmann, F.: Mix3D: Out-of-context data augmentation for 3D scenes. In: Proceedings of the International Conference on 3D Vision (3DV). pp. 116–125. IEEE (2021) 4, 5

36. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019) 3

37. Park, K., Henzler, P., Mildenhall, B., Barron, J.T., Martin-Brualla, R.: CamP: Camera preconditioning for neural radiance fields. ACM Transactions on Graphics (TOG) 42(6), 1–11 (2023) 3

38. Park, K., Sinha, U., Barron, J.T., Bouaziz, S., Goldman, D.B., Seitz, S.M., Martin-Brualla, R.: Nerfies: Deformable neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5865–5874 (2021) 3

39. Park, K., Sinha, U., Hedman, P., Barron, J.T., Bouaziz, S., Goldman, D.B., Martin-Brualla, R., Seitz, S.M.: HyperNeRF: A higher-dimensional representation for topologically varying neural radiance fields. ACM Transactions on Graphics (ToG) 40(6) (2021) 3

40. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 652–660 (2017) 3

41. Rebain, D., Matthews, M., Yi, K.M., Lagun, D., Tagliasacchi, A.: LOLNeRF: Learn from one look. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1558–1567 (2022) 4

42. Rematas, K., Liu, A., Srinivasan, P.P., Barron, J.T., Tagliasacchi, A., Funkhouser, T., Ferrari, V.: Urban radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12932–12942 (2022) 4, 5

43. Roessle, B., Barron, J.T., Mildenhall, B., Srinivasan, P.P., Nießner, M.: Dense depth priors for neural radiance fields from sparse input views. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12892–12901 (2022) 2, 3, 4, 5

44. Schönberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4104–4113 (2016) 4

45. Shoemake, K.: Animating rotation with quaternion curves. In: Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques. pp. 245–254 (1985) 8

46. Sitzmann, V., Zollhöfer, M., Wetzstein, G.: Scene Representation Networks: Continuous 3d-structure-aware neural scene representations. Advances in Neural Information Processing Systems 32 (2019) 3

47. Sun, C., Sun, M., Chen, H.T.: Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5459–5469 (2022) 3

48. Tancik, M., Casser, V., Yan, X., Pradhan, S., Mildenhall, B., Srinivasan, P.P., Barron, J.T., Kretzschmar, H.: Block-nerf: Scalable large scene neural view synthesis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8248–8258 (2022) 3

49. Tancik, M., Weber, E., Ng, E., Li, R., Yi, B., Wang, T., Kristoffersen, A., Austin, J., Salahi, K., Ahuja, A., et al.: Nerfstudio: A modular framework for neural radiance field development. In: ACM SIGGRAPH 2023 Conference Proceedings. pp. 1–12 (2023) 9

50. Tosi, F., Tonioni, A., De Gregorio, D., Poggi, M.: Nerf-supervised deep stereo. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 855–866 (2023) 5

51. Truong, P., Rakotosaona, M.J., Manhardt, F., Tombari, F.: SPARF: Neural radiance fields from sparse and noisy poses. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4190–4200 (2023) 2, 3, 4, 5, 6, 14

52. Turki, H., Zollhöfer, M., Richardt, C., Ramanan, D.: PyNeRF: Pyramidal neural radiance fields. Advances in Neural Information Processing Systems 36 (2024) 2, 3, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18

53. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: From error visibility to structural similarity. IEEE Transactions on Image Processing 13(4), 600–612 (2004) 9

54. Xie, Y., Takikawa, T., Saito, S., Litany, O., Yan, S., Khan, N., Tombari, F., Tompkin, J., Sitzmann, V., Sridhar, S.: Neural fields in visual computing and beyond. In: Computer Graphics Forum. vol. 41, pp. 641–676. Wiley (2022) 3

55. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: pixelNeRF: Neural radiance fields from one or few images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4578–4587 (2021) 2, 4, 5

56. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 586–595 (2018) 9

57. Zhou, T., Tulsiani, S., Sun, W., Malik, J., Efros, A.A.: View synthesis by appearance flow. In: Proceedings of the European Conference on Computer Vision. pp. 286–301. Springer (2016) 3

58. Zhou, Y., Tuzel, O.: VoxelNet: End-to-end learning for point cloud based 3D object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4490–4499 (2018) 3