# Re-Nerfing: Enforcing Geometric Constraints on Neural Radiance Fields through Novel Views Synthesis

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

[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 approach that leverages NeRF's own view synthesis to address these limitations. With Re-Nerfing, we increase the scene's coverage and 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 next to the original ones to simulate a stereo or trifocal setup. Finally, we train a second NeRF with both original and pseudo views while enforcing structural, epipolar constraints via the newly synthesized images. Extensive experiments on the mip-NeRF 360 dataset show the effectiveness of Re-Nerfing across denser and sparser input scenarios, bringing improvements to the state-of-the-art Zip-NeRF, even when trained with all views.*

## 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 [26]. 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 [26]. These principles have been applied across various tasks, simplifying content creation, rendering, and reconstruction workflows.

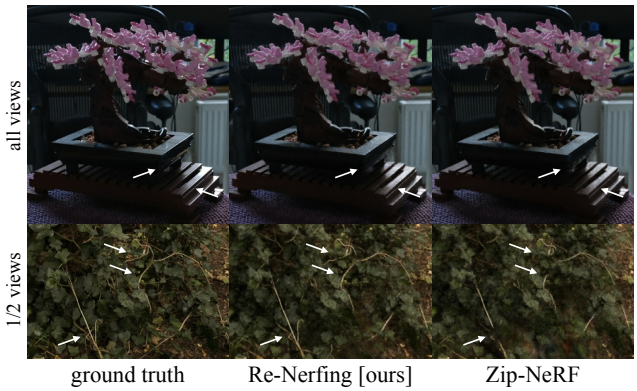



Figure 1. Examples of synthesized views by Zip-NeRF [3] with and without the proposed Re-Nerfing. Our approach improves the image quality thanks to the enforcement of geometric constraints on synthesized pseudo views. Applied to Zip-NeRF, the proposed method enables the reconstruction of finer details and improves upon the shape-radiance ambiguity. Image crops show test views from the mip-NeRF 360 [2] dataset. Training views have been reduced by half for the lower scene (*stump*).

Despite its remarkable success, NeRFs [26] have several limitations, such as slow training [27], failures when noisy or no camera poses are available [4, 23], and computationally intense rendering mostly incompatible with mobile applications [8]. While many works have been proposed to address these issues, 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 highly sparse scenes, with only a handful of views available [41, 45].

Moreover, complex geometries and large-scale environments often lead to artifacts due to the inability to consistently encode structures [1, 10, 35]. This is particularly severe in sparse settings, where the limited views lead to hallucinations manifesting as floaters in the free space due to

[*] The authors contributed equally.
Contact author: Felix Tristram (*felix.tristram@tum.de*).

the ambiguity of shape and radiance [10, 24, 41]. To mitigate these issues, depth and structural priors have been inserted into the optimization to leverage additional geometric information [10, 35, 41]. However, reliable dense depth estimates are also hard to obtain, and using wrongly estimated depth can further hurt performance [10]. This can also be seen with depth sensors, which can suffer from inherent sensor artifacts that then get propagated to the NeRF, limiting their performance in practice [18].

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 epipolar geometry on synthetic views generated by a previously trained NeRF on the same scene. Thanks to its ability to densify a scene and enforce geometric consistency, Re-Nerfing enables improvements in both sparse and dense settings, as shown in Figure 1. The main contributions of this paper can be summarized as follows:

- We show how NeRF's novel view synthesis is beneficial to enhance its own rendering quality.
- We propose Re-Nerfing: a simple and effective technique to enhance NeRF outputs via novel view synthesis and pseudo-stereo geometric constraints.
- We introduce a novel density loss for NeRF optimization derived from epipolar geometry that is generally applicable to any stereo-setup to be used with NeRF.
- We propose to evaluate the synthesized view regions according to their visibility within the training views.

With Re-Nerfing, we improve the state-of-the-art approach Zip-NeRF [3] in both denser and sparser settings.

## 2. Related Work

**View synthesis** With a dense set of views, traditional interpolation techniques based on light field sampling can synthesize photorealistic novel views [22]. 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 [11, 47]. More recently, methods building on previously reconstructed geometry have shown great promise in view synthesis tasks [20].

**Neural representations** Neural implicit representations have emerged as a shift from traditional explicit representations like point clouds, meshes, and voxels [32, 48], by encoding 3D shapes implicitly in the weights of a neural network. The pioneering works of Occupancy Networks [25] and DeepSDF [28] 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 [37].

**NeRF** Mildenhall et al. [26] 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 [44]. Among these, Instant-NGP [27], Fast-NeRF [14] and others [6, 12, 38] enable significantly faster training speeds, BARF [23], SPARF [41] and Nope-NeRF [4] cope with imperfect or absent camera poses, mip-NeRF 360 [2] deals with unbounded scenes and Nerfies [29] among others [5, 13, 30] deals with dynamic scenes. Additionally, the work of Rössle et al. [35] delivers more consistent geometry, CoNeRF [19] enables altering the output, Panoptic Neural Fields [21] incorporates scene understanding information, Block-NeRF [39] renders city-scale scenes, and CamP [31] optimizes the camera parameters. Zip-NeRF [3] 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 [9, 41]. Strong geometric information is required to reconstruct the scene and enable satisfactory novel views. Towards this end, researchers have explored the support of additional data and external models to provide geometric supervision [35, 41] or semantic knowledge [33, 45]. Such techniques enable novel view synthesis even from a single image. SPARF [41], for example, relies on a pixel correspondence network to enforce a geometrically accurate solution. PixelNeRF [45] 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 [35]. DS-NeRF [10] supervises NeRF's optimization with sparse depth obtained from COLMAP [36], thereby reducing overfitting and accelerating training. Rössle et al. [35] 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 [34] focuses on outdoor scenes where LiDAR data is available, which is used for losses that benefit surface estimation. VoxNeRF [42] leverages a volumetric representation as a sparse voxel octree within a probabilistic framework to account for the uncertainty of the surface-ray intersection point. SPARF [41] 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.

**Distillation and semi-supervised learning** Knowledge

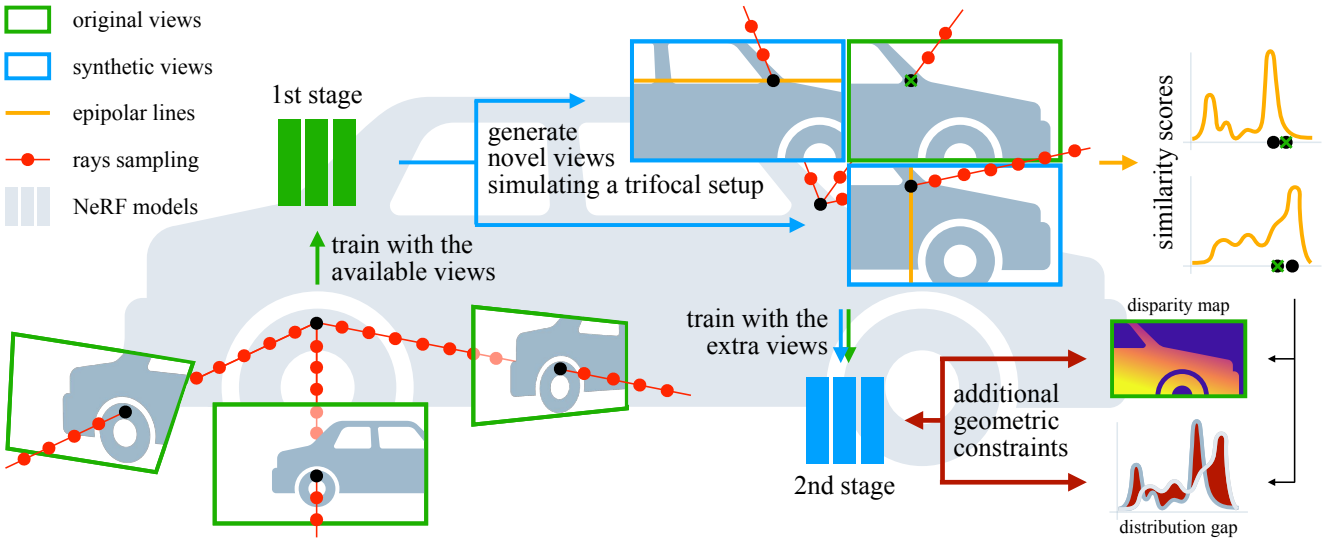



Figure 2. Re-Nerfing is a multi-stage framework. Compatible with any NeRF pipeline, it operates by first training a model with the available views (green, 1st). The trained model is then used to generate novel views that simulate a stereo or trifocal setup (blue). These new views are consecutively used to enforce geometric constraints on the depth and density (red, right) of a second model.

distillation involves transferring knowledge from a large, complex model (teacher) to a smaller, more efficient one (student) [17]. This paradigm has been widely adopted and extended to semi-supervised learning [7]. Naive-student [7] 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. [40] achieved remarkable depth estimates by training a stereo depth model using data synthesized by NeRF.

In this work, we enhance novel view synthesis in both dense and sparse settings by exploiting NeRF's inherent view synthesis capabilities and geometric constraints from epipolar geometry. Unlike existing approaches [34, 35, 41, 45], we do so by using only the available views without extra data or additional models. We achieve this by training a baseline model 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 generate such views to exploit the well-established constraints from epipolar geometry [16] with stereo and trifocal setups. In contrast to Tosi et al. [40], who trained a depth estimator with NeRF data, we train a NeRF with supervision from a pseudo-stereo setup. In our setting, the newly introduced stereo pairs are fully synthesized from the NeRF itself.

## 3. 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 3.1), then it is used to generate novel views mimicking a stereo or trifocal camera setup (Section 3.2), and lastly, a new NeRF is trained on the original and the paired synthetic views (Section 3.4), enforcing epipolar constraints between such paired views (Section 3.3). Re-Nerfing is a general framework that is compatible with any pipeline for novel view synthesis. Furthermore, Re-Nerfing follows an iterative process, so additional rounds (e.g., third) can be executed using as baseline the NeRF trained at the previous iteration.

### 3.1. Training a Standard NeRF

Neural Radiance Fields represent a continuous volumetric function in the weights of a neural representation, typically a Multi-Layer-Perceptron (MLP) [26], hash-grid [27] or (tri-)plane representation [13]. 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 [26].

### 3.2. Synthesizing Views in a Pseudo-Stereo Setup

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 [10, 41]. 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 [41], 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 [3].

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 while imposing additional geometric regularization. After training a baseline model, such as the one described in Section 3.1, we use it to generate novel views. While the synthesis procedure itself is standard [26], we do not just generate views for rendering or evaluation purposes as commonly done but leverage them as an intermediate training step. Specifically, we exploit the model's view synthesis purpose to enhance the output quality of another model trained at a later stage by adding more views of the same scene (Section 3.4).

Nevertheless, such pseudo-views are not as good as real images, otherwise, the novel view synthesis problem would be already solved. To mitigate this issue and further exploit such views, we not only synthesize them in the vicinity of the original views, where the displacement is small, producing better outputs, but we generate them to simulate a stereo camera setup in perfectly rectified scenarios. This allows us to enforce geometric constraints with epipolar geometry [16] (Section 3.3). So, for each training camera pose, we synthesize two more views. One to the right, one to the bottom of the real image to constitute two perfectly rectified setups and omit resampling artifacts arising from homography-induced rectification operations.

### 3.3. Loss Design and Epipolar Constraints

Given the additional rendered right and lower stereo-pairs of our approach in addition to the original ground truth image, we optimize our radiance field representation with the

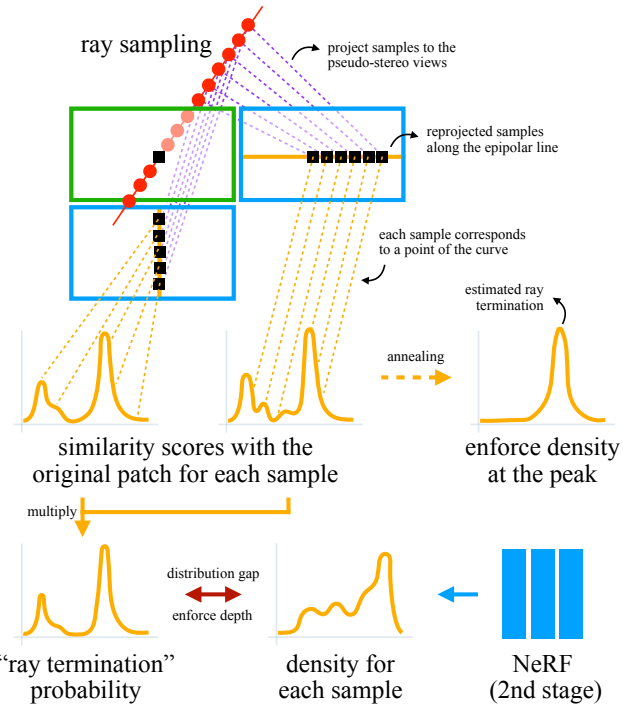



Figure 3. Illustration of the geometric constraints enforced by Re-Nerfing with the pseudo-views. A ray is cast through a pixel of the original view. The samples along the ray are projected to the pseudo views. The similarity scores between the patch around the original pixel and its epipolar line patches is computed and each ray sample is assigned a similarity. This is done independently for the side and lower pseudo views. Thanks to the epipolar principles, there is a direct correspondence of the samples along the epipolar lines of the pseudo views, so the respective scores are merged via multiplication, representing the estimated termination of the ray for each sample. The resulting values are then enforced against the density estimated by NeRF of the samples along the same ray. Additional constraints include the depth supervision driven by the estimated depth via stereo vision.

following losses:

$$\mathcal{L}_{total} = \mathcal{L}_{RGB} + \lambda_{depth} \cdot \mathcal{L}_{depth} + \lambda_{dist} \cdot \mathcal{L}_{dist}, \tag{4}$$

where $\mathcal{L}_{RGB}$, $\mathcal{L}_{depth}$, and $\mathcal{L}_{dist}$ define loss terms for color, depth, and density distribution, respectively. The parameters $\lambda_{depth}, \lambda_{dist} \in \mathbb{R}$ are balancing factors of the loss terms for depth and density distribution.

The photometric loss $\mathcal{L}_{RGB}$ is the standard pixel-wise photometric loss in the form of squared residuals with

$$\mathcal{L}_{RGB} = \sum_{\mathbf{r} \in \mathcal{R}} \left[ \left\| \hat{C}(\mathbf{r}) - C(\mathbf{r}) \right\|_2^2 \right] \tag{5}$$

between the synthesized colour $\hat{C}(\mathbf{r})$ and its ground truth image colour value $C(\mathbf{r})$ for a given ray $\mathbf{r}$ in the set $\mathcal{R}$ of visible rays from a certain perspective.

As depicted in Figure 3, we use the epipolar geometry of our pseudo-stereo setup to supervise the NeRF optimization in two ways, first by applying a loss $\mathcal{L}_{depth}$ on the expected depth and then further also applying a loss $\mathcal{L}_{dist}$ on the NeRF density weights directly.

**Depth Supervision:** We supervise the expected depth $\hat{D}$ directly via an L2 loss with the depth $D_{stereo}$ calculated from the best match of the relevant epipolar line such that

$$\mathcal{L}_{depth} = \sum_{s \in S} \left[ M_{occ} \odot \left\| \hat{D} - D^s_{stereo} \right\|_2 \right] \tag{6}$$

for all stereo pairs $s \in S$; in our setup, the pairs with the lower and the right image. $M_{occ}$ illustrates an occlusion mask and $\odot$ defines the element-wise Hadamard product. The stereo depth supervision signal $D^s_{stereo}$ is calculated using an L1-similarity between the patch-wise pixel values belonging to each ray and all other patches on its epipolar line in the stereo-pair. With the disparity $\Delta^s$ between the original left pixel and the maximum of the patch-wise L1-similarity, we can recover the depth $D^s_{Stereo}$ via:

$$D^s_{stereo} = \frac{f \cdot B^s}{\Delta^s}, \tag{7}$$

where $B^s$ depicts the baseline distance between the camera centers of the stereo pair $s \in S$ in the rectified setup.

Having not only side-by-side stereo pairs but also horizontal stereo pairs allows us to compare mutually calculated stereo depth values. We only consider the samples valid that fall within some threshold $\tau \in \mathbb{R}$ and create the mask

$$M_{occ} = \begin{cases} 1 & \text{if } \left\| D^{horizontal}_{stereo} - D^{vertical}_{stereo} \right\|_1 < \tau \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

This allows to mask out occlusions or wrong matches with minimal loss of information.

**Density Supervision:** In addition to the direct depth supervision we also constrain the density weights output from the NeRF. Since there is a direct geometric relationship between samples $t$ on each ray $\mathbf{r}$ and the epipolar line in the corresponding stereo-image, we project the ray samples onto the epipolar line and enforce a similarity between the density of each sample and the patch-wise L1-similarity

$$\text{Sim}^s\left(\mathbf{r}(u)\right) = \left\| \boldsymbol{P}^i_{S0} - \boldsymbol{P}^j_{S1} \right\|_1 \tag{9}$$

for corresponding patches $\boldsymbol{P}^i_{S0} \longleftrightarrow \boldsymbol{P}^j_{S1}$ between image stereo pairs $S0$ and $S1$ whose matches define a point $\mathbf{r}(u)$ along the ray $\mathbf{r}$. Since the ray samples $\mathbf{r}(t)$ are not necessarily integers but vary continuously, we interpolate their values on the similarity distribution, allowing for sub-pixel matching accuracy. With equation (7) we map ray-samples $\mathbf{r}(t)$ to the similarity distribution and interpolate their values from their closest neighbors $\mathbf{r}(u)$ by

$$\text{Sim}^s_{inter}\left(\mathbf{r}(t)\right) = \xi\left(\mathbf{r}(t), \text{Sim}^s\left(\mathbf{r}(u)\right)\right) \tag{10}$$

The interpolated similarity distribution, however, can be quite noisy. We apply a Gaussian filter centered around the maximum of the distribution with a variance that is annealed over the course of the optimization. This shifts our similarity distributions towards sharper geometric boundaries and gives a meaningful signal to enforce the density with

$$\text{Sim}^s_{filtered} = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[ -\frac{\left(\text{Sim}^s_{inter} - 1\right)^2}{2\sigma^2} \right] \tag{11}$$

using the shorthand notations $\text{Sim}^s_{inter}$ and $\text{Sim}^s_{filtered}$. The filtered distribution is then normalized to sum to one and applied to the density weights from equation (2) with an L1-Loss, while again occlusions and inconsistent matches are masked out by

$$\mathcal{L}_{dist} = \sum_{s \in S} \left[ M_{occ} \odot \left\| w(t) - \text{Sim}^s_{filtered} \right\|_1 \right]. \tag{12}$$

### 3.4. Re-Nerfing: Training NeRF (Again)

After training the first round and using it to generate novel views in a stereo setup, we train another NeRF from scratch while enforcing the geometric constraints defined in Section 3.3 that are derived from these novel stereo views.

To optimize this second round, we use the *pseudo-GT* images not only for the geometric constraints but also as part of the training images and query their rays during the course of optimization. The intuition behind using these *pseudo-GT* images is that they provide a more diverse set of views, which allows the NeRF representation to better triangulate the underlying scene geometry. This is especially important at the beginning of training, where the density of the scene still varies significantly during optimization and where the core geometrical components of the scene are being learned.

After a certain amount of iterations, the discrepancy in image quality between the *pseudo-GT* and GT images becomes detrimental to the overall reconstruction, so we remove them from the sampling pool.

## 4. 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 the *pseudo-GT* images. Interestingly, not only does using these *pseudo-GT* images improve convergence speed and quality on the test-views but it can also boost PSNR values on the GT training views by a small amount, further reinforcing our intuition.

### 4.1. Experimental Setup

**Dataset** We test our method on the 7 public scenes introduced in mip-NeRF 360 [2], comprising both bounded

| Method | all views | | | 1/2 views | | | 1/4 views | | | 1/8 views | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS |
| Zip-NeRF [3] | 30.17 | 0.903 | 0.058 | 27.06 | 0.818 | 0.101 | 22.54 | 0.648 | 0.222 | 21.12 | 0.741 | 0.211 |
| Zip-NeRF + DS-NeRF [10] | 30.16 | 0.903 | 0.058 | - | - | - | - | - | - | - | - | - |
| Zip-NeRF + syn. views | 30.25 | **0.906** | 0.057 | 27.28 | **0.829** | 0.094 | **22.62** | **0.655** | **0.213** | 21.37 | **0.761** | **0.195** |
| + constr. = Re-Nerfing [ours] | **30.29** | **0.906** | **0.056** | **27.29** | **0.829** | **0.093** | **22.62** | **0.656** | **0.213** | **21.49** | **0.761** | **0.195** |

Table 1. We report the standard RGB reconstruction metrics on the mip-NeRF 360 [2] dataset, with every 8th image used as test-view. To showcase the performance of our method on sparser views, we also report results for half, a quarter and an eights of the original views. The 1/8 views setting is only applied to the bonsai and room scene as for all others NeRF reconstruction failed. For DS-NeRF [10] we use the provided depth from colmap for the full scenes but find that it fails on some of the sparser-view scenes so do not report metrics there.

indoor and large unbounded outdoor settings. It features a large amount of views captured all around an object or scene. Given the large amount of views available, this dataset allows us to showcase the novel view quality in denser and sparser settings.

**Metrics** On top of evaluating the novel views on the standard metrics and errors, i.e., PSNR, SSIM [43], and LPIPS [46], we consider the number of times each test region has been seen during training. This visibility occurrence is computed considering the overlaps of each test region in 3D space with respect to the training camera-frustums. Since accurate scene geometry is not available, we simplify the computation as follows: all patches project at a constant distance, each region is considered seen from a training frustum if its center pixel projected in 3D space is inside such frustum, and patch rays with a rotational distance higher than 30° from a frustum are not treated as visible. This last measure is to avoid that views facing one another captured from opposite sides of the scene score complete overlap. Furthermore, each image is divided into 64 equal patches, on which the PSNR is computed. While additional details on this computation are described in the supplementary material, the computation aims to report the PSNR in greater detail according to the visibility of each region.

**Prior Works and Baselines** We showcase the effectiveness of our method by applying it on a state-of-the-art NeRF method, namely Zip-NeRF [3]. Furthermore, we compare with other methods enforcing geometric constraints, such as DS-NeRF [10].

**Implementation Details** We train our method on an image resolution down-sampled 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 for 25k iterations with a batch size of 8192 rays, a learning rate that is logarithmically decayed from $10^{-2}$ to $10^{-3}$ over the course of training and an initial warm-up phase of 5k iterations. The influence of the pseudo-views on the RGB loss is removed after 8k iterations. The loss balancing factors are equal as $\Lambda_{depth} = \Lambda_{dist} = 0.001$. We inherit the distortion, anti-interlevel, and hash-decay losses that were introduced in Zip-NeRF[3] and use the publicly available unofficial PyTorch implementation[1], since the official one was not public at the time of this writing. The $\sigma$ for our filtering is annealed logarithmically from 0.05 to 0.025 during the first 2k iterations, while our occlusion threshold is linearly decreased from 1.0 to 0.33 over the course of 20k iterations. For sparser settings, we downsampled the views while preserving coverage of the scene. We train all models on a single 24GB NVIDIA RTX 3090 or 4090 GPU.

### 4.2. Quantitative Results

In Table 1, we report the results on the mip-NeRF 360 dataset [2] across a decreasing number of available training views. Specifically, we used all training inputs available (all views), half of them (1/2 views), a quarter (1/4), or an eight (1/8). We compare with Zip-NeRF [3] as the baseline and DS-NeRF [10] applied on Zip-NeRF. The table also includes Zip-NeRF trained with 2 extra views synthesized for each available training view. Such views are generated using the baseline Zip-NeRF model (first row of the table) for each respective sparsity setting. The last row reports the results of the proposed Re-Nerfing which, on top of training with the extra synthesized views, enforces the geometric constraints described in Section 3.3.

**Pseudo-views** Compared to Zip-NeRF [3], the proposed technique of retraining the NeRF model with additional views synthesized by an earlier NeRF model brings an improvement over the standard Zip-NeRF across the board, albeit slight, even in the full views setup. In particular, the margin further increases by reducing the amount of training views. This finding is interesting as it shows that not only do real views help improve the output quality (all views compared to 1/2 views) but also generated ones. This simple technique is particularly effective even when the output quality of the pseudo-views is suboptimal (1/4 and 1/8 views), as the added inputs help against the shape-radiance ambiguity and offer novel viewpoints to capture the scene.

[1] https://github.com/SuLvXiangXin/zipnerf-pytorch

| Freq. | all views Zip-NeRF | our gain | 1/2 views Zip-NeRF | our gain |
|---|---|---|---|---|
| - | 30.17 | .12 | 27.06 | .23 |
| 1 | 30.62 | .14 | 28.08 | .15 |
| 2 | 31.52 | .15 | 28.04 | .18 |
| 3-4 | 31.44 | .08 | 28.70 | .28 |
| 5-9 | 31.60 | .09 | 29.81 | .18 |
| 10-14 | 32.00 | .04 | 31.46 | .16 |
| 15-19 | 33.39 | .09 | 32.71 | .17 |
| 20-29 | 36.07 | .01 | 33.24 | .37 |
| 30+ | 40.97 | .02 | 34.27 | .55 |

Table 2. Evaluation on the visibility of patches in the test-views and the resulting PSNR values for both the first-stage baseline and our Re-Nerfing approach. If all views are available our method is mainly useful in test-patches with low observability, but if fewer views in general are available the gain is more pronounced.

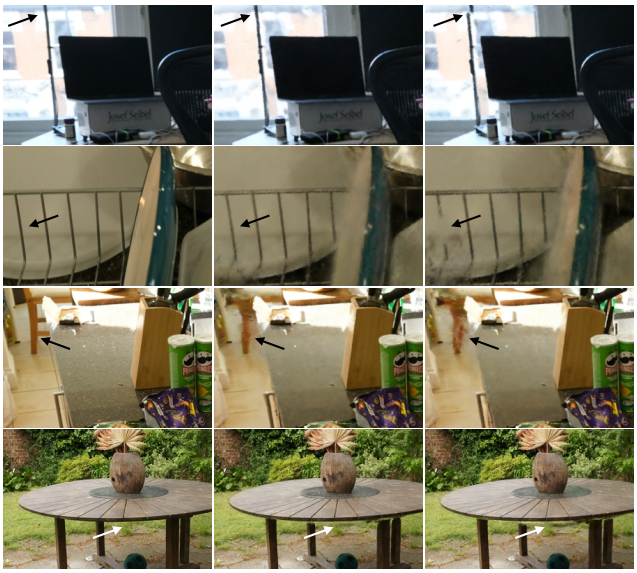



Figure 4. Qualitative results on cropped images from the test set of the mip-NeRF 360 dataset. The models were trained on all available images.

Furthermore, although NeRF is trained on a single scene and does not need to generalize to other scenes, overfitting is an issue considering the generalization to novel viewpoints. 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 [7, 15].

**Re-Nerfing** The proposed Re-Nerfing capitalizes on the positive effect of the synthetic views and further increases the gap over Zip-NeRF on which it is based, performing better across the board. While moderate, this gain is remarkable as it is obtained over the state-of-the-art Zip-NeRF, without any additional model to establish correspondences [41] or any other aid beyond the standard set of posed views already available to the baseline in the first stage (Section 3.1). The extra geometric constraints on the pseudo-views help enforce the structure of the scene, especially in the sparser setting (1/8). In particular, Re-Nerfing acts upon three fundamental factors of novel view synthesis: color, density, and depth. Re-Nerfing mainly addresses color by adding the synthesized pseudo-views, which contribute by disambiguating it from the radiance. Density is improved by the extra loss introduced, which further exploits the pseudo-views via epipolar constraints on the estimated density (Figure 3). Then, depth is enhanced by supervising it against the stereo depth computed through the pseudo-views. Together, these factors contribute to systematic improvements over the model on which the proposed Re-Nerfing is applied, which is the state-of-the-art Zip-NeRF in Table 1.

**DS-NeRF** DS-NeRF [10] supervises the NeRF's depth via the sparse reconstruction obtained by COLMAP [36]. COLMAP is typically run prior to NeRF to obtain the camera poses, so DS-NeRF's contribution comes for free, as ours. However, unlike ours, DS-NeRF does not bring any value compared to the standard Zip-NeRF, obtaining nearly identical results according to all three metrics. DS-NeRF was shown to improve upon the vanilla NeRF [26] in forward-facing, very sparse scenes where the COLMAP depth can be beneficial. However, in the unbounded scenes of the mip-NeRF 360 dataset [2], the COLMAP depth is too sparse, and Zip-NeRF's inherent depth estimation is already compelling, causing DS-NeRF to bring no added value over Zip-NeRF. It should be noted that the empty cells in Table 1 (-) are due to COLMAP failing to output a pose for all the given views, which is common in sparse settings [41]. In these cases, the comparison would not be fair because DS-NeRF relies on the input poses to be in the same space and scale as the COLMAP reconstruction. So, the only option would be excluding the failed views, which would mean training DS-NeRF with fewer views than the other approaches.

**Sparse settings** In sparser settings, the effect of the proposed Re-Nerfing and its pseudo-views are more pronounced, with a difference in PSNR of 0.37 dB (1/8). The pseudo-views are particularly beneficial as they triplicate the limited number of training images available. However, the quality of such views degrades significantly compared to the full-view setting, as the baseline drops over 9 PSNR points. A trade-off occurs between the output quality of the baseline and the impact of the pseudo-views. Nevertheless, this trade-off is mitigated by the larger room for improvement in sparser settings, which ultimately leads to


Figure 5. Qualitative results on cropped images from the test set of the mip-NeRF 360 dataset. The models were trained on a quarter of the available images (1/4).

bigger margins over Zip-NeRF. It should be noted that in the sparser 1/8 setting, only 2 scenes could be reconstructed, namely *room* and *bonsai*.

**Visibility-dependent evaluation** In Table 2, we report the PSNR results of Zip-NeRF [3] and the proposed Re-Nerfing applied on Zip-NeRF. It shows how the proposed Re-Nerfing performs better than Zip-NeRF regardless of how many times each region of the test images is seen during training. However, it is interesting how the gap varies. In the all-views setting, our method achieves the highest gain on those image regions that are approximately seen only once or twice during training. At higher frequencies, the gap reduces until it becomes negligible, with patches seen more than 20 times. When using only half of the training views (1/2), the margin is higher on the patches seen more often, although these are relatively few as only the *room* scene exhibits regions visible more than 15 times. Overall, the table indicates how Re-Nerfing mainly contributes to improving the novel view synthesis of the low-visibility regions. However, when fewer views are available (1/2), the gain in such rarely seen regions remains similar, while the overall scene benefits more from the multiple pseudo-views added by Re-Nerfing.

### 4.3. Qualitative Results

**All views available** Figure 4 shows examples of synthesized images with Zip-NeRF [3] given all training views available, with and without our method. As seen in the quantitative results (Section 4.2), overall, the output quality of the proposed Re-Nerfing improves upon the standard Zip-NeRF in structural regions of the scene (e.g., thin structures), while it is mostly similar in other areas. The improvements are particularly noticeable at the edges of the images or on regions of the scene mainly occluded by other parts (e.g., the chair). These are typically areas of lower visual overlap, which is aligned with the findings of Table 2. Figure 1 also reports qualitative results given all training views. There, it can be seen how while the outputs of the standard Zip-NeRF are compelling for the most part, Re-Nerfing has the upper hand in those areas that appear more challenging due to the occlusion or the high reflectivity (e.g., bonsai's support), as the extra pseudo-views help disambiguate shape and radiance.

**1/2 views available** Figure 1 shows a crop from the *stump* scene synthesized by models trained with half the views. The qualitative improvement of Re-Nerfing over Zip-NeRF is significant, with the latter failing to recover the details of the thin branches and foliage, resulting in a blurry and structureless output. Instead, the proposed method delivered a better novel view, better preserving the geometry of the scene and finer details.

**1/4 views available** In Figure 5, we show crops of the scenes synthesized by the models trained with 1/4 of the images. Here, as seen in Table 1, the quality decreases due to significant portions of the scene that are not adequately covered by the training views, inevitably causing artifacts. Although the proposed method relies on the same initial data as the state-of-the-art Zip-NeRF, its benefits in terms of geometric constraints and added pseudo-views are evident, with Re-Nerfing producing fewer artifacts overall.

## 5. Conclusion & Limitations

**Limitations** Since our method relies on the first stage to provide reasonable quality renderings from a good estimate of the underlying scene geometry, we cannot enhance extremely sparse scenes. Combining our approach with other methods [10, 41] that tackle these scenarios would, however, be an interesting direction. Furthermore, we also use simple stereo patch matching, which is sensitive to featureless or repetitive regions, introducing ambiguities. Incorporating some form of deep feature matching or regularization from a state-of-the-art stereo depth estimator could help alleviate these issues.

**Conclusion** This work enhances NeRFs by leveraging their own novel view synthesis capabilities. The introduced Re-Nerfing combines novel view synthesis with geometric constraints of newly synthesized views, yielding improvements in both rendering quality and convergence speed of sparser and denser scenes. Additionally, our novel density loss, derived from epipolar geometry, can be applied to any rectified stereo setup to be used in a NeRF optimization.

## References


[1] 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. 1

[2] 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. 1, 2, 5, 6, 7, 13

[3] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-NeRF: Anti-aliased grid-based neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19697–19705, 2023. 1, 2, 4, 6, 8, 12, 13, 14, 15

[4] Wenjing Bian, Zirui Wang, Kejie Li, Jia-Wang Bian, and Victor Adrian Prisacariu. Nope-NeRF: Optimising neural radiance field with no pose prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4160–4169, 2023. 1, 2

[5] Ang Cao and Justin Johnson. HexPlane: A fast representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 130–141, 2023. 2

[6] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *Proceedings of the European Conference on Computer Vision*. Springer, 2022. 2

[7] Liang-Chieh Chen, Raphael Gontijo Lopes, Bowen Cheng, Maxwell D Collins, Ekin D Cubuk, Barret Zoph, Hartwig Adam, and Jonathon Shlens. Naive-student: Leveraging semi-supervised learning in video sequences for urban scene segmentation. In *Proceedings of the European Conference on Computer Vision*, pages 695–714. Springer, 2020. 3, 7

[8] Zhiqin Chen, Thomas Funkhouser, Peter Hedman, and Andrea Tagliasacchi. 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*, pages 16569–16578, 2023. 1

[9] Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H Kim, and Jan Kautz. Extreme view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7781–7790, 2019. 2

[10] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised NeRF: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12882–12891, 2022. 1, 2, 4, 6, 7, 8, 14, 15

[11] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. DeepStereo: Learning to predict new views from the world's imagery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5515–5524, 2016. 2

[12] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, 2022. 2

[13] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12479–12488, 2023. 2, 3

[14] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. FastNeRF: High-fidelity neural rendering at 200FPS. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14346–14355, 2021. 2

[15] Stefano Gasperini, Nils Morbitzer, HyunJun Jung, Nassir Navab, and Federico Tombari. Robust monocular depth estimation under challenging conditions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8177–8186, 2023. 7

[16] Richard Hartley and Andrew Zisserman. *Multiple view geometry in Computer Vision*. Cambridge University Press, 2003. 3, 4

[17] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 3

[18] HyunJun Jung, Patrick Ruhkamp, Guangyao Zhai, Nikolas Brasch, Yitong Li, Yannick Verdie, Jifei Song, Yiren Zhou, Anil Armagan, Slobodan Ilic, et al. On the importance of accurate geometry data for dense 3d vision tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 780–791, 2023. 2

[19] Kacper Kania, Kwang Moo Yi, Marek Kowalski, Tomasz Trzciński, and Andrea Tagliasacchi. CoNeRF: Controllable neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18623–18632, 2022. 2

[20] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4):1–14, 2023. 2

[21] Abhijit Kundu, Kyle Genova, Xiaoqi Yin, Alireza Fathi, Caroline Pantofaru, Leonidas J Guibas, Andrea Tagliasacchi, Frank Dellaert, and Thomas Funkhouser. Panoptic neural fields: A semantic object-aware neural scene representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12871–12881, 2022. 2

[22] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the Conference on Computer Graphics and Interactive Techniques*, page 31–42. Association for Computing Machinery, 1996. 2

[23] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5741–5751, 2021. 1, 2

[24] 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. 2

[25] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[26] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European Conference on Computer Vision*, pages 405–421. Springer, 2020. 1, 2, 3, 4, 7

[27] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022. 1, 2, 3

[28] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[29] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 2

[30] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. HyperNeRF: A higher-dimensional representation for topologically varying neural radiance fields. *ACM Transactions on Graphics (ToG)*, 40 (6), 2021. 2

[31] Keunhong Park, Philipp Henzler, Ben Mildenhall, Jonathan T Barron, and Ricardo Martin-Brualla. CamP: Camera preconditioning for neural radiance fields. *arXiv preprint arXiv:2308.10902*, 2023. 2

[32] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. PointNet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017. 2

[33] Daniel Rebain, Mark Matthews, Kwang Moo Yi, Dmitry Lagun, and Andrea Tagliasacchi. LOLNeRF: Learn from one look. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1558–1567, 2022. 2

[34] Konstantinos Rematas, Andrew Liu, Pratul P Srinivasan, Jonathan T Barron, Andrea Tagliasacchi, Thomas Funkhouser, and Vittorio Ferrari. Urban radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12932–12942, 2022. 2, 3

[35] Barbara Roessle, Jonathan T Barron, Ben Mildenhall, Pratul P Srinivasan, and Matthias Nießner. Dense depth priors for neural radiance fields from sparse input views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12892–12901, 2022. 1, 2, 3

[36] Johannes L Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4104–4113, 2016. 2, 7

[37] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene Representation Networks: Continuous 3d-structure-aware neural scene representations. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[38] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022. 2

[39] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P Srinivasan, Jonathan T Barron, and Henrik Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8248–8258, 2022. 2

[40] Fabio Tosi, Alessio Tonioni, Daniele De Gregorio, and Matteo Poggi. Nerf-supervised deep stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 855–866, 2023. 3

[41] Prune Truong, Marie-Julie Rakotosaona, Fabian Manhardt, and Federico Tombari. SPARF: Neural radiance fields from sparse and noisy poses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4190–4200, 2023. 1, 2, 3, 4, 7, 8

[42] Sen Wang, Wei Zhang, Stefano Gasperini, Shun-Cheng Wu, and Nassir Navab. VoxNeRF: Bridging voxel representation and neural radiance fields for enhanced indoor view synthesis. *arXiv preprint arXiv:2311.05289*, 2023. 2

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

[44] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. In *Computer Graphics Forum*, pages 641–676. Wiley, 2022. 2

[45] 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. 1, 2, 3

[46] 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. 6

[47] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow.

In *Proceedings of the European Conference on Computer Vision*, pages 286–301. Springer, 2016. 2

[48] Yin Zhou and Oncel Tuzel. VoxelNet: End-to-end learning for point cloud based 3D object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018. 2

# Re-Nerfing: Enforcing Geometric Constraints on Neural Radiance Fields through Novel Views Synthesis

## Supplementary Material

This supplementary material provides additional details and experiments to complement the main paper. Specifically, in Section A, we provide more details about the visibility-based metric. Section B reports various experiments to motivate the design choices of Re-Nerfing. Lastly, Section C adds qualitative results of models trained with 1/8th of the original views.

## A. Additional Details on the Visibility Metric

As introduced in the main paper, for evaluation, we divided each synthesized view into 64 parts (8×8) and computed the PSNR according to the visibility of each part across the views used for training. By doing so, we obtain insights about where the benefits of the proposed method over the baseline Zip-NeRF [3] occur (Table 2).

To compute the visibility of each region, we proceeded as follows. For each training view, we computed its frustum in 3D space, given the camera pose and camera parameters. For this, we considered 0.1 as the close plane and 10.0 as the far plane. Then, for each of the 64 image regions being considered, we projected its center pixel in 3D space, fixing the scene depth for each view at 4.0. We count a region as visible if its projected center lies inside a training frustum. However, only frustums with an angular distance to the test view being evaluated closer than 30° are considered. This visibility is a simplification as it does not take into account occlusions within the scene. The angular threshold is meant to mitigate this problem, as no ground truth depth is available. Furthermore, we count how many training frustums each center is in and consider the respective region according to the visibility bin its center belongs to, e.g., 10-14 if the center is projected inside 12 frustums. Finally, we compute the PSNR for each region and average the resulting values within the bins.

Since this visibility computation is an approximation, the procedure described above runs relatively fast thanks to each image region center representing its entire region. Yet, more complex implementations may provide more accurate values. This visibility-based evaluation aims to provide a finer-grain assessment of performance differences compared to the metrics computed on the entire images.

## B. Additional Quantitative Results

**Density and Depth Supervision** In Table 3, we assess the impact of our geometric constraints introduced via the pseudo views. Therefore, we compare supervising the density alone, the depth alone, or both (full). Each of the two types of supervision is similarly beneficial over the standard Zip-NeRF [3]. However, when using both, the results are further improved. The annealing of the density supervision has a minor benefit, enforcing sharper estimates. Specifically, this density supervision depends on the points sampled along the ray, used to search their corresponding similarity on the epipolar lines. If these samples are not close to the maximum of the similarity distribution, supervising the density can introduce issues. The L2 loss on the expected depth mitigates this by pushing the Zip-NeRF proposal network to emit samples closer to the maximum of the similarity distribution of the sampled ray, further improving the results across the board.

| Method | PSNR | SSIM | LPIPS |
|---|---|---|---|
| baseline (Zip-NeRF) | 27.06 | 0.818 | 0.101 |
| [ours] w/ only density sup. | 27.27 | 0.831 | 0.103 |
| [ours] w/o annealing | 27.26 | 0.830 | 0.103 |
| [ours] w/ only depth sup. | 27.25 | 0.829 | 0.094 |
| [ours] w/ depth & density (full) | **27.36** | **0.834** | **0.096** |

Table 3. Ablation study on the geometric constraints evaluated on all scenes training only with half the camera views available (1/2). The proposed density supervision alone (w/ only density sup.), only the depth supervision (w/ only depth sup.), and removing the annealing of the Gaussian filter from the density supervision (w/o annealing) are compared with the full method supervising both depth and density (full).

| Stereo Baseline Distance | PSNR | SSIM | LPIPS |
|---|---|---|---|
| 0.5x | 21.73 | 0.540 | 0.236 |
| 1x, chosen distance | **21.89** | **0.560** | **0.211** |
| 2x | 21.66 | 0.537 | 0.243 |
| 4x | 21.63 | 0.533 | 0.243 |
| 8x | 21.60 | 0.531 | 0.244 |

Table 4. Experiments on the baseline distance for our stereo-pairs on the *stump* scene with half the original views for training (1/2). 1x indicates the selected distance of 0.05 and 0.025 for the right and lower stereo-pair respectively. The other multipliers (e.g., 0.5x and 8x) are multiples of these numbers.

**Baseline Distance vs. Quality Benefits** Intuitively, the further from the original views, the harder it is for the model


Figure 6. Visualisation of the pseudo views at different stereo baseline distances (0.5x, 1x, ..., 8x), for the *stump* scene with half the original views for training (1/2). The quality degrades with higher baseline distances as the synthesized views deviate more from the original inputs.

to synthesize high-quality images. In Table 4, we show the effect of this by varying the stereo baseline distance on the *stump* scene with half views available. Since the quality of the views decreases with the distance, using them as pseudo views worsens the metrics accordingly. However, when the baseline distance is too short (0.5x), the synthesized views are too similar to the original ones, so they do not bring significantly more information. Furthermore, the perspective advantage of the synthesized view is not substantial, so the supervision along the ray loses its importance. Therefore, by varying the baseline distance, a trade-off occurs between adding information and not introducing too many artifacts via the pseudo views. These results are confirmed in Figure 6, where we show the synthesized views depending on the baseline distance. Further viewpoints result in suboptimal synthesized images and a limited visual overlap with the original views, which reduce the impact of our geometric constraints.

**Overfitting on the Pseudo Views** The added pseudo views are helpful as they provide additional structure to the scene, which is particularly important in the early stages of the NeRF training. Later, details become more important. For these reasons, we only exploit the pseudo views in the earlier stages of training. Table 5 shows how sampling these synthesized views throughout the training reduces performance (w/o remove syn. views). This is because the quality of the pseudo views is worse than the original ones, and in the detailing stages of training, their inclusion is detrimental as the model overfits to the artifacts of the pseudo views and reaches worse results than the baseline. In particular, we observed this overfitting behavior through the validation curves, which show the benefits of the pseudo views diminishing after about 8k iterations. So, in our experiments, we remove their influence after 8k iterations and keep them only for the initial stages to help with the scene structure.

| Method | PSNR | SSIM | LPIPS |
|---|---|---|---|
| baseline (Zip-NeRF) | 30.17 | 0.903 | 0.058 |
| [ours] w/ synthetic views | 30.25 | 0.906 | 0.057 |
| [ours] w/o remove syn. views | 29.99 | 0.901 | 0.064 |
| [ours] w/ geom.constr. (full) | **30.29** | **0.906** | **0.056** |

Table 5. Ablations on the 7 public scenes with all views of the mip-NeRF 360 dataset [2]. The impact of not removing the synthetic views from the sampling pool is shown (w/o remove syn. views).

| Method | PSNR | SSIM | LPIPS |
|---|---|---|---|
| baseline (Zip-NeRF) | 25.71 | 0.858 | 0.107 |
| [ours] w/ syn. stereo | 25.89 | 0.862 | 0.104 |
| [ours] w/ syn. trifocal (full) | **26.00** | **0.863** | **0.103** |

Table 6. Comparison between a pseudo stereo and a pseudo trifocal setup on the *counter* scene with half the training views (1/2).

**Stereo vs. Trifocal** With Re-Nerfing, we generate synthetic views to simulate a trifocal setup. In Table 6, we compare the impact of simulating a stereo setup instead of a trifocal one. While both settings are beneficial over the standard Zip-NeRF [3], synthesizing an additional lower view (w/ syn. trifocal) instead of only a right one (w/ syn. stereo) further improves the results. The extra view of the trifocal setup is particularly helpful with a high amount of occlusions (e.g., the *counter* scene), as it helps disambiguate the structure of the scene.

**Results by Scene** For completeness, in Tables 7 to 13, we report the main results for each scene of the mip-NeRF 360 dataset [2]. In most of the sparser 1/8 view settings, the standard Zip-NeRF [3] failed to reconstruct the scene, so we did not report the results. The averages in Table 1 are computed only on the scenes that could be reconstructed, which are indicated in these scene-specific tables. For some scenes, such as *bicycle* and *stump*, the outputs of the standard Zip-NeRF were already suboptimal with 1/4 views, preventing the proposed Re-Nerfing from exploiting the pseudo views and leading to worse results.

| Method | *bicycle* - all views | | | *bicycle* - 1/2 views | | | *bicycle* - 1/4 views | | |
|---|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS |
| Zip-NeRF [3] | 26.78 | 0.831 | 0.090 | 21.88 | 0.673 | 0.154 | 16.51 | 0.227 | 0.503 |
| Zip-NeRF + DS-NeRF [10] | 26.82 | 0.830 | 0.089 | - | - | - | - | - | - |
| Zip-NeRF + syn. views | 26.89 | **0.836** | **0.085** | 22.00 | 0.685 | **0.144** | **16.52** | **0.236** | 0.484 |
| + constr. = Re-Nerfing [ours] | **26.91** | **0.836** | **0.085** | **22.57** | **0.719** | 0.163 | 16.45 | 0.235 | **0.483** |

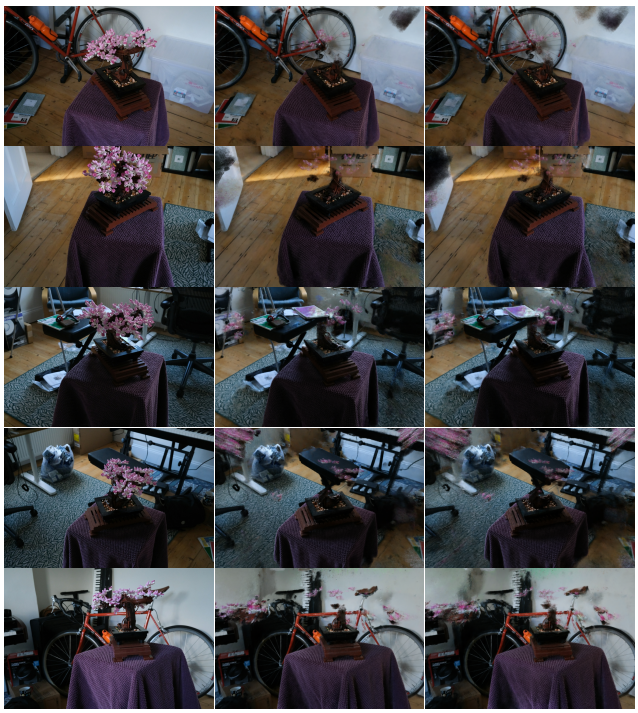
Table 7. Results on the *bicycle* scene, for all, half, and quarter views.

| Method | *counter* - all views | | | *counter* - 1/2 views | | | *counter* - 1/4 views | | |
|---|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS |
| Zip-NeRF [3] | 29.02 | 0.907 | 0.078 | 25.71 | 0.858 | 0.107 | 17.87 | 0.639 | 0.300 |
| Zip-NeRF + DS-NeRF [10] | 28.96 | 0.907 | 0.078 | - | - | - | - | - | - |
| Zip-NeRF + syn. views | 29.14 | 0.908 | 0.077 | 25.91 | 0.862 | 0.104 | 17.96 | 0.654 | 0.290 |
| + constr. = Re-Nerfing [ours] | **29.23** | **0.909** | **0.076** | **26.00** | **0.863** | **0.103** | **18.04** | **0.659** | **0.285** |

Table 8. Results on the *counter* scene, for all, half, and quarter views.

## C. Additional Qualitative Results

Figure 7 reports example outputs with 1/8 of the training views available, simulating sparse settings. In such a sparse setup, the output quality degrades significantly, as shown in Table 12. Nevertheless, as already seen in Figure 5 with 1/4 views, the proposed Re-Nerfing delivers sharper estimates than the standard Zip-NeRF on which it is based. This is particularly evident for floating artifacts, which our method mitigates substantially thanks to the added pseudo views and the geometric constraints, which help disambiguate the scene structure.



ground truth Re-Nerfing [ours] Zip-NeRF

Figure 7. Test-views of the *bonsai* scene with only 1/8th of the original images available. While for both the baseline Zip-NeRF and our method, the scene reconstruction is suboptimal in such sparse settings, the proposed Re-Nerfing exhibits significantly fewer floaters and artifacts.

| Method | *stump* - all views | | | *stump* - 1/2 views | | | *stump* - 1/4 views | | |
|---|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS |
| Zip-NeRF [3] | 27.08 | 0.819 | 0.085 | 21.44 | 0.513 | 0.242 | **16.19** | **0.152** | 0.470 |
| Zip-NeRF + DS-NeRF [10] | 27.14 | 0.820 | 0.086 | - | - | - | - | - | - |
| Zip-NeRF + syn. views | 27.32 | **0.830** | **0.081** | 21.86 | 0.559 | **0.211** | 15.97 | 0.148 | **0.456** |
| + constr. = Re-Nerfing [ours] | **27.35** | **0.830** | **0.081** | **21.89** | **0.560** | **0.211** | 15.96 | 0.148 | 0.458 |

Table 9. Results on the *stump* scene, for all, half, and quarter views.

| Method | *garden* - all views | | | *garden* - 1/2 views | | | *garden* - 1/4 views | | |
|---|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS |
| Zip-NeRF [3] | **29.67** | 0.914 | **0.038** | 27.75 | 0.884 | 0.053 | 25.51 | 0.830 | 0.074 |
| Zip-NeRF + DS-NeRF [10] | 29.61 | 0.915 | **0.038** | - | - | - | - | - | - |
| Zip-NeRF + syn. views | 29.61 | 0.913 | 0.039 | 27.86 | **0.890** | 0.051 | **25.69** | **0.842** | **0.069** |
| + constr. = Re-Nerfing [ours] | **29.67** | **0.915** | **0.038** | **27.88** | **0.890** | **0.050** | 25.65 | **0.842** | **0.069** |

Table 10. Results on the *garden* scene, for all, half, and quarter views.

| Method | *kitchen* - all views | | | *kitchen* - 1/2 views | | | *kitchen* - 1/4 views | | |
|---|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS |
| Zip-NeRF [3] | **32.89** | **0.952** | **0.036** | 30.54 | 0.927 | 0.053 | 26.34 | 0.885 | 0.075 |
| Zip-NeRF + DS-NeRF [10] | 32.81 | 0.950 | 0.037 | - | - | - | - | - | - |
| Zip-NeRF + syn. views | 32.85 | 0.951 | **0.036** | **30.66** | **0.928** | **0.052** | 26.45 | 0.886 | 0.073 |
| + constr. = Re-Nerfing [ours] | 32.87 | 0.951 | **0.036** | 30.63 | 0.927 | **0.052** | **26.59** | **0.888** | **0.073** |

Table 11. Results on the *kitchen* scene, for all, half, and quarter views.

| Method | *bonsai* - all views | | | *bonsai* - 1/2 views | | | *bonsai* - 1/4 views | | | *bonsai* - 1/8 views | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS |
| Zip-NeRF [3] | 33.58 | 0.960 | **0.027** | 31.90 | 0.949 | 0.035 | 26.43 | 0.887 | 0.075 | 18.98 | 0.716 | 0.215 |
| Zip-NeRF + DS-NeRF [10] | 33.59 | 0.960 | 0.028 | - | - | - | - | - | - | - | - | - |
| Zip-NeRF + syn. views | 33.71 | **0.961** | 0.028 | **32.31** | **0.952** | **0.033** | **26.69** | **0.896** | **0.070** | 19.40 | 0.747 | 0.191 |
| + constr. = Re-Nerfing [ours] | **33.75** | **0.961** | **0.027** | 32.25 | **0.952** | **0.033** | 26.54 | 0.895 | **0.070** | **19.50** | **0.748** | **0.190** |

Table 12. Results on the *bonsai* scene, for all, half, quarter and eight views.

| Method | *room* - all views | | | *room* - 1/2 views | | | *room* - 1/4 views | | | *room* - 1/8 views | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS |
| Zip-NeRF [3] | 32.20 | 0.941 | 0.051 | 30.19 | 0.926 | 0.062 | 28.94 | 0.916 | 0.056 | 23.26 | 0.766 | 0.207 |
| Zip-NeRF + DS-NeRF [10] | 32.21 | 0.941 | 0.051 | - | - | - | - | - | - | - | - | - |
| Zip-NeRF + syn. views | 32.24 | 0.941 | 0.051 | **30.38** | **0.928** | **0.061** | 29.08 | 0.921 | **0.053** | 23.34 | **0.775** | **0.198** |
| + constr. = Re-Nerfing [ours] | **32.27** | **0.942** | 0.050 | 30.33 | **0.928** | **0.061** | **29.11** | **0.922** | **0.053** | **23.48** | **0.775** | 0.200 |

Table 13. Results on the *room* scene, for all, half, quarter and eight views.