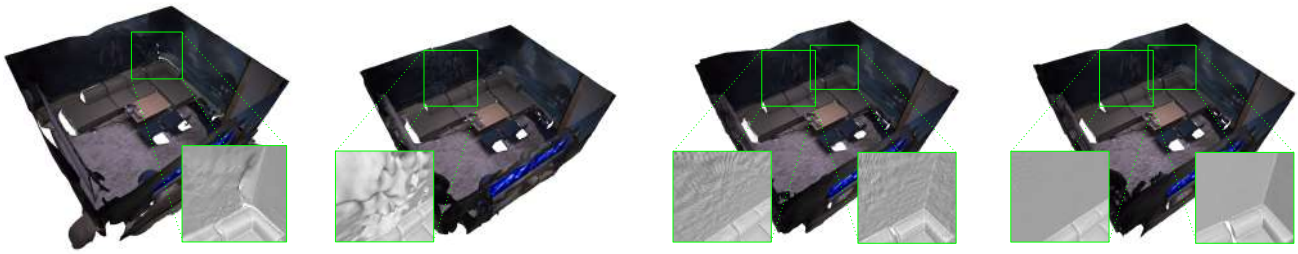
# VF-NeRF: Learning Neural Vector Fields for Indoor Scene Reconstruction

Albert Gassol Puigjaner*[1] Edoardo Mello Rella*[1] Erik Sandström[1]
Ajad Chhatkuli[1] Luc Van Gool[1,2,3]
[1] Computer Vision Lab, ETH Zurich [2] VISICS, KU Leuven [3] INSAIT, Sofia



MonoSDF [58] F-score: 0.790 | ManhattanSDF [16] F-score: 0.757 | VF-NeRF (Ours) F-score: **0.890** | Ground Truth

Figure 1. **VF-NeRF .** Using the recently proposed Vector Field (VF) [40] representation, our method reconstructs indoor scenes in the NeRF setting. Due to the planar inductive bias of VF, we can generally recover indoor scenes with high fidelity, providing State-of-the-Art (SOTA) performance.

## Abstract

*Implicit surfaces via neural radiance fields (NeRF) have shown surprising accuracy in surface reconstruction. Despite their success in reconstructing richly textured surfaces, existing methods struggle with planar regions with weak textures, which account for the majority of indoor surfaces. In this paper, we aim to solve indoor dense surface reconstruction by replacing traditional implicit representations such as the signed distance field (SDF) or surface density with the recently proposed vector field (VF). VF is defined by the unit vector directed to the nearest surface point. It therefore flips direction at the surface, and equals the explicit surface normals. Except for this flip or sign change around planar surfaces, VF remains constant and provides a strong inductive bias towards planar surfaces. We develop a novel density-VF relationship and a training scheme that allows us to learn VF via volume rendering. By doing this, VF-NeRF can model large planar surfaces without additional cues such as segmentations, depth or normals. Additionally, we show that, when depth cues are available, our method further improves and achieves state-of-the-art results in reconstructing indoor scenes. We extensively evaluate VF-NeRF on public datasets such as Replica and ScanNet and run comprehensive ablations of its components.*

*The authors contributed equally to the work

## 1. Introduction

Multi-view image-based 3D scene reconstruction is a cornerstone challenge in computer vision [17, 42, 45]. Traditional multi-view stereo (MVS) algorithms [11, 41, 42, 48, 59] leverage matching and triangulation to derive 3D point coordinates from given input images. Nonetheless, they often struggle in regions characterized by uniform low-texture or repetitive patterns. Equipped with volume rendering, Neural Radiance Fields (NeRF) [33, 50, 56] and its variants [27, 29, 32] have established themselves as powerful alternatives to previous methods for surface reconstruction. However, NeRF methods still struggle with low-texture indoor surfaces, surprisingly even when using Manhattan normal priors [8, 16].

NeRF for indoor scene reconstruction has currently two significant challenges. The first is that the classical NeRF surface density [33], which provides high-quality view rendering, stumbles significantly when it comes to scene geometry reconstruction. Even when an SDF [50, 56] representation is used for the geometry, any surface regularization for planar surfaces has to rely on the gradients of the SDF [16]. Note that these gradients are often noisy and unreliable for regularization. An additional downside of SDF is that its representation power is limited to water-tight surfaces. Therefore, it may not be able to faithfully reconstruct thin or open surfaces. The second challenge stems from poor texture in indoor surfaces, which provides weak multi-

view constraints for the indirect triangulation in NeRF or direct triangulation in MVS approaches.

In this paper, we address the first challenge, that of the implicit scene representation in NeRF. In the process, we also push towards mitigating the challenge of weak texture through an improved inductive bias towards planar surfaces. To that end, we make use of the recently proposed Vector Field (VF) representation [40] in order to encode the scene geometry. This involves associating each position in the 3D space with a unit vector directed towards the nearest surface. It has been shown that VF may exhibit superior performance to SDF even on closed surfaces and, furthermore, on planar surfaces. However, the study confines itself to a supervised learning paradigm, and the self-supervised learning with NeRF poses a significant challenge. Notably, without the ground-truth VF, a pair of points is required to compute the surface density given the VF predictions.

To ease the VF optimization, we propose to use a dual MLP network, one to predict the VF and the other to predict the RGB color values. We learn the VF and the color through a training scheme via volume rendering on multi-view posed images similarly to VolSDF [50, 56]. Specifically, we express the surface density via the cosine similarity of the VF predictions in the ray samples. Thus, neural volume rendering can be used to train the VF as in [50, 56]. As planar surfaces exhibit a constant VF around the surface (except for the direction flip at the surface), VF amounts to explicit normals for the most part, and thus provides a strong inductive bias. As a first study on VF for NeRF, we therefore consider its use for learning indoor scene geometry. We rigorously evaluate our method against leading benchmarks for indoor scenes, including ManhattanSDF [16], MonoSDF [58], and Neuralangelo [26], on indoor datasets such as Replica [46] and ScanNet [9], showing superior performance with depth priors.

In summary, our contributions are twofold:

- We propose to learn the VF representation of 3D scenes with multi-view images via volume rendering.
- We demonstrate the effectiveness of our method on different indoor scene datasets, showing state-of-the-art results.

## 2. Related work

**Multi-view Surface Reconstruction**. Traditional MVS approaches have often relied on feature matching for depth estimation [2–4, 12, 24, 41, 42, 44]. These classical methods extract image features, match them across views for depth estimation, and then fuse the obtained depth maps to form dense point clouds. Voxel-based representations [1, 11, 43] rely on color consistency between the projected images to generate an occupancy grid of voxels. Subsequently, meshing techniques, like Poisson surface reconstruction [21, 22] are applied to delineate the surface. However, these methods typically fail to reconstruct low-textured regions and non-Lambertian surfaces. Additionally, the reconstructed point clouds or meshes are often noisy and may fail to reconstruct some surfaces.

Recently, learning-based methods have gained attention, offering replacements for classic MVS methods. Methods like [5, 18, 53, 54] leverage 3D CNNs to extract features and predict depth maps, while others [6, 15] construct cost volumes hierarchically, yielding high-resolution outcomes. However, these methods often fail to accurately reconstruct the scene geometry due to the limited resolution of the cost volume.

**Neural Radiance Fields (NeRF)**. In recent studies [29, 32, 33, 39] the potential of MLPs to represent scenes both in terms of density and appearance has been explored. While these techniques can produce photorealistic results for novel view synthesis, determining an isosurface for the volume density to reconstruct scene geometry remains a challenge. Commonly, NeRF uses thresholding techniques to derive surfaces from the predicted density. However, these extracted surfaces can often exhibit noise and inaccuracies.

**Neural Scene Representations**. Approaches based on neural scene representations employ deep learning to learn properties of 3D points and to generate geometry. Traditional methods like point clouds [10, 28] and voxel grids [7, 51] have been primary choices for representing scene geometry. More recently, implicit functions, such as occupancy grids [35, 36], SDF [19, 26, 30, 37, 50, 55, 56], and VF [40, 52] have gained popularity due to their precision in capturing scene geometry. For instance, in [30, 35] a novel differentiable renderer to learn the scene geometry from images is proposed, while [55] focuses on modeling view-dependent appearance, which proves successful on non-Lambertian surfaces. However, these methods rely on masks to accurately reconstruct the geometry from multi-view images. Consequent works, such as VolSDF [56] and NeuS [50] introduce a second MLP in the NeRF context to represent the geometry as the SDF, further leveraging volume rendering to learn the geometry from images. Building upon these methods, Neuralangelo [26] takes inspiration from Instant Neural Graphics Primitives (Instant NGP) [34] to introduce hash encodings in neural SDF models, enhancing surface reconstruction resolution. However, a challenge persists as these methods tend to fail in large indoor planar scenes with low-texture regions, leading to inaccurate surface reconstructions.

**Priors for Neural Scene Representations**. Several works have explored the integration of priors during optimization to improve the reconstruction of indoor scenes. For instance, Manhattan-SDF [16] suggests incorporating dense depth maps from COLMAP [42] to facilitate the learning of 3D geometry and employs Manhattan world [8] priors to address the challenges posed by low-textured planar surfaces. A limitation of this approach is its reliance on seman-

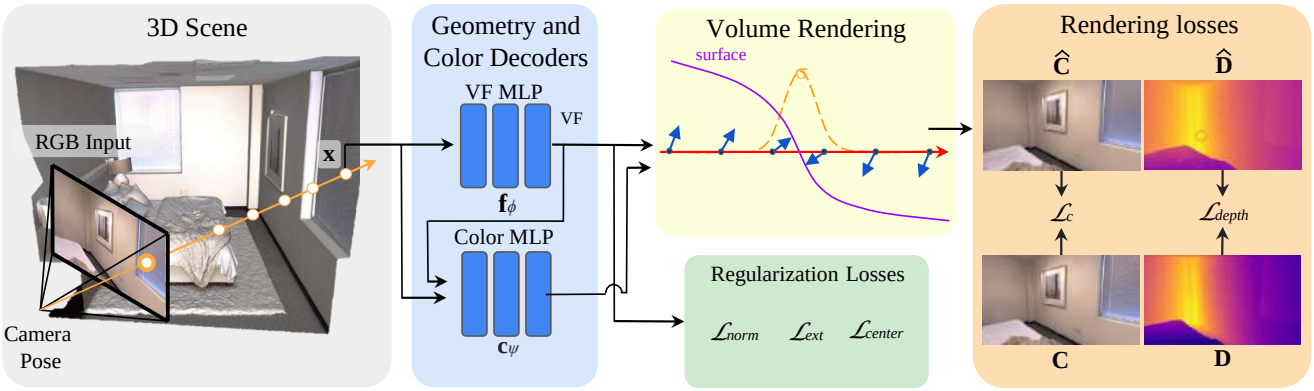



Figure 2. **VF-NeRF overview.** We use VF to represent the geometry of a scene. Specifically, given an input image taken from the camera view position, we shoot a batch of rays onto the 3D scene. We predict the VF and color of the points along the ray using geometry and color decoders, two sets of MLPs. By computing the cosine similarity between neighboring points on the ray, we can identify the surface as the locations where the value equals $-1$; this happens when the two predicted vectors have opposing directions. From the cosine similarity, we then differentiably compute the surface density required for volume rendering. We render the RGB and depth in order to compute the re-rendering losses. We then optimize them together with the regularization terms for the network parameters.

tic segmentation masks to pinpoint planar regions, adhering to the Manhattan world assumption. This dependency can lead to added complexity and potential inaccuracies in regions where segmentations are less accurate. More recently, NeuRIS [49] proposes to use normal priors to guide the reconstruction of indoor scenes. Expanding on this work, MonoSDF [58] introduces both normal and depth monocular cues into the optimization. By using normal priors, these methods successfully remove the Manhattan world assumption, thereby enhancing the reconstruction of indoor scenes.

## 3. Method

Given a set of posed images of an indoor scene, our goal is to reconstruct the dense scene geometry. We represent the surface geometry in NeRF with VF [40, 52], and describe its properties in Sec. 3.1. We then introduce the surface density as a parametrization of the VF in Sec. 3.2 . Finally, in Sec. 3.3, we formulate the optimization problem and introduce the loss terms of our method. We provide an overview in Fig. 2.

### 3.1. Vector Field Representation

In VF-NeRF , the scene geometry is defined using unit vectors that point towards the nearest surface. Let $\Omega \subset \mathbb{R}^3$ be the surface of an object in $\mathbb{R}^3$ and $\Gamma \subset \mathbb{R}^3$ be the set of unit norm 3-vectors. We make use of the VF definition [40]: VF is a function $\mathbf{f} : \mathbb{R}^3 \to \Gamma$ that maps a point in space to a unit vector directed to the closest surface point of $\Omega$:

$$\mathbf{f}(\mathbf{x}) = \begin{cases} \dfrac{\mathbf{x}_S - \mathbf{x}}{||\mathbf{x}_S - \mathbf{x}||_2} & \text{if } \mathbf{x} \notin \Omega \\ \dfrac{\hat{\mathbf{x}}_S - \hat{\mathbf{x}}}{||\hat{\mathbf{x}}_S - \hat{\mathbf{x}}||_2} & \text{if } \mathbf{x} \in \Omega \end{cases} \tag{1}$$

where $\mathbf{x}_S = \arg\min_{\mathbf{s}\in\Omega} ||\mathbf{x} - \mathbf{s}||_2$ is the closest surface point with respect to $\mathbf{x}$, and $\hat{\mathbf{x}} = \lim_{||\epsilon||_2 \to 0} \mathbf{x} + \epsilon$ is a point close to the surface, with $\epsilon \in \mathbb{R}^3$ being an infinitessimal 3D vector.

Given the definition of the VF representation, we identify a surface $\Omega$ between a point $\mathbf{x} \in \mathbb{R}^3$ and an infinitesimally close neighbor using the cosine similarity between the VF at the two points. When the two points are on opposite sides of the surface $\Omega$, their cosine similarity approaches $-1$. Conversely, it is close to 1 everywhere else, except at diverging discontinuities of the field.

$$\begin{aligned} \Omega = &\{\mathbf{x}_1, \mathbf{x}_2 = \mathbf{x}_1 + \epsilon | \cos(\mathbf{f}(\mathbf{x}_1), \mathbf{f}(\mathbf{x}_2)) < \tau\}, \\ &\cos(\mathbf{v}_1, \mathbf{v}_2) = \frac{\mathbf{v}_1 \cdot \mathbf{v}_2}{||\mathbf{v}_1||_2 ||\mathbf{v}_2||_2} \end{aligned} \tag{2}$$

where $\epsilon \in \mathbb{R}^3$ is an infinitesimal displacement and $|\tau| \leq 1$ is a cosine similarity threshold. Ideally, $\tau = -1$ for infinitesimally close neighbors.

From these definitions, we notice a similarity to the surface density $\sigma : \mathbb{R}^3 \to \mathbb{R}_{\geq 0}$, a function that indicates the rate at which a ray is occluded at location $\mathbf{x}$. Ideally, for non-translucid surfaces, $\sigma(\mathbf{x})$ behaves as a delta function, being zero everywhere except at the surface. To model this function typically used in volume rendering [20, 33], a simple transformation of the cosine similarity can be used. In fact, the cosine similarity between the VF of infinitesimally

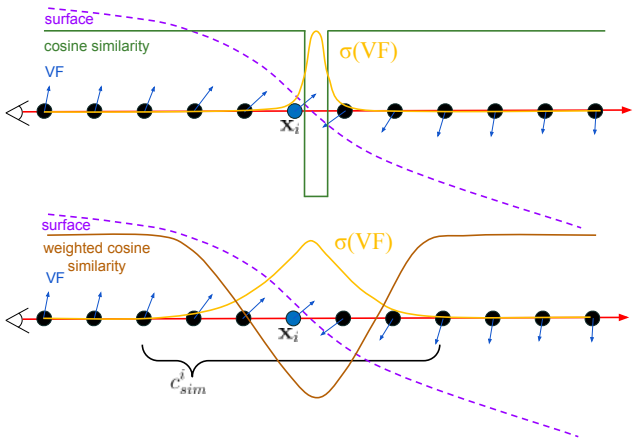



Figure 3. **Density using unaveraged and averaged cosine similarity**. The figures show the VF, cosine similarity and density of a ray crossing a surface. Top: density as a transformation of the cosine similarity. This yields a sharp function similar to the delta function centered at the surface. Bottom: Density as a transformation of the weighted average cosine similarity. This produces a smoother function with the maximum centered at the surface.

close neighbors is a delta function itself, yielding approximately 1 everywhere and $-1$ at the surface. However, as we show, a smooth function is necessary in order to ease the learning of VF through volume rendering.

### 3.2. Density as Transformed VF

We draw inspiration from existing methods [50, 56], which use neural volume rendering to learn the geometry of a scene as an implicit function. Contrary to these previous methods that use SDF, we propose to model the surface density as a function of the learnable VF. Given a viewing ray and the VF sampled at multiple points along the ray, we use a differentiable process to estimate the surface density. As previously highlighted, the cosine similarity of the VF at neighboring points along the ray can be used to indicate whether there is a surface between them. The resulting surface density function, showcased in Fig. 3 top row, closely resembles a delta function. This behavior is desirable in order to obtain sharp reconstructions; however, due to its discontinuity, the desired convergence is hard to achieve. In order to make the gradient-based optimization tractable, we first need a smoothing transformation. To this end, we adopt a sliding window approach and compute a weighted average cosine similarity. We thus smooth the function at points near the surface. The effect of the sliding window can be seen in the bottom row of Fig. 3.

Given a set of samples in a ray, we initially predict the VF at each point of the ray. We then define the weights of a sliding window of size $M$, where the size is an even number, as $\mathbf{w} = [w_0, w_1, ..., w_{M-1}]$. The sliding window and predicted VF are used to compute the weighted average cosine similarity associated with each point. The smoothened cosine similarity of a point is computed as the weighted average of the cosine similarities using multiple forward and backward neighbors of the ray. Therefore, given a ray of $N+1$ points $\mathbf{r} = [\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_N]$ , we can compute $N$ averaged cosine similarities as:

$$c^i_{sim}(\mathbf{r}) = \sum_{j=0}^{M/2-1} [w_j \cos(\mathbf{f}(\mathbf{x}_i), \mathbf{f}(\mathbf{x}_{i-j-1})) + w_{j+M/2} \cos(\mathbf{f}(\mathbf{x}_i), \mathbf{f}(\mathbf{x}_{i+j+1}))]. \quad (3)$$

For simplicity, we do not consider the boundary cases in Eq. (4). However, note that the cosine similarity of the first and last points of the ray is not smooth because the sliding window would go out of range. Additionally, given N+1 points, we can only compute N cosine similarities since the last point of the ray does not have a successor.

The effect of the weighted sliding window can be changed by modifying its weights. Initially, we start with a uniform distribution where all the weights are equal and sum up to 1. We introduce an annealing process to progressively add more weight to the closest neighbors with the final objective to end with a one-hot vector where all the weight is located at the next neighbor. Thanks to this approach, the network can be easily optimized, while preserving the desired sharpness during inference. The annealing process is depicted in Fig. 4. This process is linear and depends on the training epoch. Specifically, the weights of the sliding window are computed at the beginning of every epoch using the following equation:

$$\widehat{w}_i = \frac{M}{2}\text{ReLU}\left(1 - \frac{n|i - M/2|}{N_{epochs}}\right)$$
$$w_i = \frac{\widehat{w}_i}{||\widehat{\mathbf{w}}||}. \quad (4)$$

Using sliding window cosine similarity, we redefine the surface density as a transformation that maps a

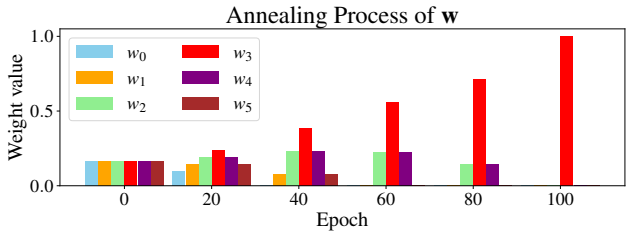



Figure 4. **Sliding window weights annealing example**. Example of weights at different stages of the annealing. In this case, the sliding window contains 6 weights. At the beginning of the training (epoch 0), the weights for each neighbor are equal. At the end of the training (epoch 100) the cosine similarity is computed only with respect to the next neighbor.

point in the ray $\mathbf{r} \in \mathbb{R}^{(N+1)\times 3}$ to a scalar value, $\sigma : \mathbb{R}^{(N+1)\times 3} \times \mathbb{N}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$. Leveraging the cosine similarity, we define the surface density as follows:

$$\sigma(\mathbf{r}, i) = \text{ReLU}(\alpha \Psi_{\mu,\beta}(-c^i_{sim}(\mathbf{r})) - \alpha \Psi_{\mu,\beta}(\xi)). \quad (5)$$

where $\alpha, \mu, \beta > 0$ are learnable parameters and $\xi$ is a cosine similarity threshold value left as a hyperparameter. ReLU is the rectified linear unit and $\Psi_{\mu,\beta}$ represents the Cumulative Distribution Function (CDF) of the Laplace distribution. $\mu$ denotes the Laplacian mean, while $\beta$ is Laplacian diversity and $\alpha$ is a scaling factor. Formally, the Laplacian CDF is defined as follows:

$$\Psi_{\mu,\beta}(x) = \begin{cases} 1 - \exp\left(-\frac{|x-\mu|}{\beta}\right) & \text{if } x > \mu \\ \exp\left(-\frac{|x-\mu|}{\beta}\right) & \text{if } x \leq \mu. \end{cases} \quad (6)$$

With this definition of the density function, we can accumulate the densities and colors using numerical quadrature [33] to render the color and depth of the pixel associated with the ray:

$$C(\mathbf{p}) = \sum_{i=1}^{N} T_i (1 - \exp(-\sigma_i \delta_i)) \mathbf{c}_i \quad (7)$$

$$D(\mathbf{p}) = \sum_{i=1}^{N} T_i (1 - \exp(-\sigma_i \delta_i)) t_i \quad (8)$$

where $T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right)$ and $\delta_i = t_{i+1} - t_i$ is the distance between samples along a ray. Note that Eqs. (7) and (8) can be seen as traditional alpha compositing with alpha values $\alpha_i = 1 - \exp(\sigma_i \delta_i)$.

### 3.3. Training

Our approach leverages a dual-MLP structure. First, $\mathbf{f}_\phi : \mathbb{R}^3 \rightarrow \mathbb{R}^{3+256}$ predicts the VF of the scene alongside a global geometry feature vector $\mathbf{z} \in \mathbb{R}^{256}$. Here, $\phi$ represents the network learnable parameters. Second, $\mathbf{c}_\psi : \mathbb{R}^{3+3+3+256} \rightarrow \mathbb{R}^3$ approximates the radiance field color values based on a given spatial point, viewing direction, VF, and global feature vector. Here, $\psi$ represents the radiance field network learnable parameters. Consequently, for a specific point on a ray $\mathbf{x}$ and its viewing direction $\mathbf{d}$ , we can predict the VF as $(\mathbf{v}, \mathbf{z}) = \mathbf{f}_\phi(\mathbf{x})$ and the radiance field as $\mathbf{c} = \mathbf{c}_\psi(\mathbf{x}, \mathbf{v}, \mathbf{d}, \mathbf{z})$. Our model also incorporates three adjustable parameters for the density function as described in Eq. (5), namely $\alpha$, $\mu$ and $\beta$.

During training, a batch of pixels $\mathcal{P}$ and their corresponding rays are sampled to minimize the difference between the rendered images $\widehat{C}(\mathbf{p})$ and the reference images $C(\mathbf{p})$:

$$\mathcal{L}_c = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p} \in \mathcal{P}} ||\widehat{C}(\mathbf{p}) - C(\mathbf{p})||_1 \quad (9)$$

Learning the geometry of indoor scenes solely from images presents a challenge in reconstructing accurate geometries, even in textured regions. To address this, we enhance the learning of scene representation by introducing a depth consistency loss similarly to [49, 58]. This loss compares the rendered depth, $\widehat{D}(\mathbf{p})$, with a reference depth, symbolized as $D(\mathbf{p})$. Depending on the availability of data, the depth $D(\mathbf{p})$ can be derived from multi-view stereo methods [41, 42, 59] or by monocular depth estimation [13, 14].

$$\mathcal{L}_{depth} = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p} \in \mathcal{P}} ||\widehat{D}(\mathbf{p}) - D(\mathbf{p})||_1 \quad (10)$$

In addition to the rendering losses, we add three regularization terms to impose the known properties of VF. First, we impose that the VF has a unit vector property by applying the unit norm loss $\mathcal{L}_{norm}$:

$$\mathcal{L}_{norm} = \frac{1}{N+1} \sum_{i=0}^{N} (||\mathbf{f}(\mathbf{x}_i)||_2 - 1)^2 \quad (11)$$

Additionally, in object-centric scenes, the VF at outer, distant points resembles a vector directed toward the center. In order to assess this, we incorporate a loss that guides the VF for points outside the scene, denoted as $\mathcal{P}_{ext}$, to point towards the object's center, represented by $\mathbf{c}_{scene}$.

$$\mathcal{L}_{ext} = \frac{1}{|\mathcal{P}_{ext}|} \sum_{\mathbf{x} \in \mathcal{P}_{ext}} \left|\left| \mathbf{f}(\mathbf{x}) - \frac{\mathbf{c}_{scene} - \mathbf{x}}{||\mathbf{c}_{scene} - \mathbf{x}||_2} \right|\right|_2 \quad (12)$$

Finally, considering that in indoor scenes, images are captured from within the scene's geometry, we introduce a loss function that guides points near the scene's center, represented as $\mathcal{P}_{cen}$, to point outwards:

$$\mathcal{L}_{cen} = \frac{1}{|\mathcal{P}_{cen}|} \sum_{\mathbf{x} \in \mathcal{P}_{cen}} \left|\left| \mathbf{f}(\mathbf{x}) - \frac{\mathbf{x} - \mathbf{c}_{scene}}{||\mathbf{x} - \mathbf{c}_{scene}||_2} \right|\right|_2 \quad (13)$$

The overall loss is defined as a weighted sum of the individual losses:

$$\begin{aligned} \mathcal{L} = & w_c \mathcal{L}_c + w_{norm} \mathcal{L}_{norm} + w_{ext} \mathcal{L}_{ext} + \\ & w_{depth} \mathcal{L}_{depth} + w_{cen} \mathcal{L}_{center} \end{aligned} \quad (14)$$

Our training scheme encompasses three distinct stages. Initially, we concentrate on optimizing the overall loss to learn an initial VF representation via neural volume rendering. The second stage involves encouraging the VFs to have opposite directions at proximal points across the surface boundary. This is accomplished by maximizing the cosine similarity between the VF vectors at these strategically identified points. To find points on opposite sides of the surface, we shoot rays and find the points where the current density function yields a maximum. The final stage is dedicated to refining our representation, achieved by optimizing the overall loss while employing a reduced learning rate.

## 4. Experiments

**Implementation details**. Our method is developed using PyTorch [38] and trained using the Adam optimizer [23]. The VF and color functions are designed as MLPs consisting of 8 and 4 hidden layers, respectively. Positional encodings [33] are used for the spatial positions $\mathbf{x}$ and viewing directions $\mathbf{d}$ to address the challenge of learning high-frequency details of the scene. Furthermore, we find that initializing the VF network to point toward the center of the scene eases the training process. The learning rate is initialized at $5 \times 10^{-4}$ and is decreased using an exponential decay approach [25]. The training process spans 3300 epochs. Notably, weight annealing for the sliding window technique is executed between the 300th and 1000th epochs. Each epoch's iteration count is equivalent to the dataset's training image count, and 1024 rays are sampled during each iteration. For each ray, we sample 200 stratified points perturbed with Gaussian noise. Additionally, we use Truncated Signed Distance Function (TSDF) fusion to extract the surface mesh from the predicted depth maps and images. We set the following weights of the multiobjective loss function: $w_c = 1.0$, $w_{norm} = 0.1$, $w_{ext} = w_{cen} = 0.5$, $w_{depth} = 0.195$. Regarding the density function parameters, we set the cosine similarity threshold to $\xi = -0.5$ and initialize the learnable parameters to $\mu = 0.7$, $\beta = 0.5$ and $\alpha = 100$.

**Datasets**. We test the performance of our algorithm on Replica [46] and ScanNet [9]. The Replica dataset consists of 18 synthetic indoor scenes, where each scene contains a dense ground truth mesh, and 2000 RGB and depth images. Similarly to MonoSDF [58], we focus on only seven scenes of this dataset for comparison purposes. The ScanNet dataset contains 16113 indoor scenes with 2.5 million views, with each view containing RGB-D images. Additionally, a fused mesh is provided for each scene. We select the four scenes of this dataset used by ManhattanSDF [16] and MonoSDF to evaluate our method. For replica, we sample 1 of every 20 posed images for training, while in Scannet we sample 1 of every 40.

**Metrics**. For 3D surface reconstruction, we focus on evaluating our method with Chamfer distance and F1-score [47]. Additionally, we also provide the peak signal-to-noise ratio (PSNR) to evaluate view synthesis. The detailed definitions of these metrics are included in the supplementary material.

**Baselines**. We compare our method against the State of the Art (SOTA), which use volume rendering for indoor scene reconstruction: ManhattanSDF [16], MonoSDF [58], NeuRIS [49] and Neuralangelo [26]. We use Marching Cubes [31] to extract the meshes rendered by the baselines.

### 4.1. Comparisons with baselines

**3D reconstruction**. We evaluate our method with the Replica and ScanNet datasets. The qualitative results on Replica and ScanNet are shown in Fig. 5. Quantitative results on both datasets are depicted in Tab. 4. Additional detailed qualitative and quantitative results are included in the supplementary material. Our method outperforms volume rendering based benchmarks in terms of F-score on both datasets. Most interestingly, the gap in performance is significantly higher in ScanNet, a more challenging dataset containing noisy depth maps. The ability to perform extremely well on this dataset might be explained by the strong inductive bias offered by VF that allows it to learn planar regions even in the presence of noisy data.

Note that MonoSDF makes use of monocular depth as well as surface normal cues to achieve accurate results in 3D reconstruction. The performance of Neuralangelo in indoor scenes drops since it does not make use of any of the two priors throughout the optimization. ManhattanSDF can generally recover high-quality scenes, although it struggles in some planar areas due to its dependency on semantic segmentation masks. In contrast, our method can recover planar surfaces with great fidelity using only depth. The capacity of our method to represent planar surfaces compared to the other methods is depicted in Fig. 7.

**Novel view synthesis**. We present qualitative and quantitative novel view synthesis comparisons in Fig. 8 and Table Tab. 4. VF-NeRF generally renders high-quality views and outperforms ManhattanSDF in terms of PSNR in both datasets. Additionally, VF-NeRF outperforms all baselines in ScanNet, a more realistic dataset. However, Neuralangelo and MonoSDF present outstanding results in Replica. More qualitative results for each scene can be found in the supplementary material.

### 4.2. Ablations

**Loss terms**. We analyze the impact of different loss terms on the surface representation and provide quantitative results. Specifically, we remove the following losses and study their effects: center supervision $\mathcal{L}_{center}$, exterior supervision $\mathcal{L}_{ext}$, unit norm $\mathcal{L}_{norm}$ and depth $\mathcal{L}_{depth}$. We demonstrate that removing these losses decreases the performance of our method, as presented in Tab. 2. Even though in some cases the precision improves, the overall performance in terms of F-score drops. Additionally, removing the depth loss significantly decreases the performance as demonstrated in Fig. 6 since most of the surfaces of the scene do not have enough texture.

**Sliding window annealing and initialization**. Tab. 2 presents the results of ablating the sliding window cosine similarity and the custom VF network initialization. In the case of the sliding window annealing, we use cosine similarity with the next point of the ray instead of the weighted average. This is the same as just using the sliding window as a one-hot vector where all the weight is located at the next neighbor. The results show that both elements enhance the

| | Replica | | | | | ScanNet | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | PSNR ↑ | CD ↓ | Precision ↑ | Recall ↑ | F-score ↑ | PSNR ↑ | CD ↓ | Precision ↑ | Recall ↑ | F-score ↑ |
| COLMAP [42] | - | - | 0.760 | 0.403 | 0.527 | - | - | 0.604 | 0.485 | 0.538 |
| NeRF [33] | - | - | 0.153 | 0.295 | 0.201 | - | - | 0.085 | 0.166 | 0.112 |
| UNISURF [36] | - | - | 0.195 | 0.338 | 0.247 | - | - | 0.298 | 0.335 | 0.315 |
| NeuS [50] | - | - | 0.524 | 0.465 | 0.493 | - | - | 0.406 | 0.437 | 0.421 |
| VolSDF [56] | - | - | 0.317 | 0.442 | 0.369 | - | - | 0.489 | 0.546 | 0.516 |
| ManhattanSDF [16] | 27.48 | 350 | 0.723 | 0.856 | 0.779 | 20.78 | 22.80 | 0.778 | 0.694 | 0.730 |
| Neuralangelo [26] | **31.44** | 1336 | 0.243 | 0.323 | 0.262 | 17.83 | 272 | 0.269 | 0.188 | 0.220 |
| MonoSDF [58] | **32.25** | **5.33** | **0.906** | **0.889** | **0.897** | 23.84 | **17.83** | **0.863** | **0.730** | **0.788** |
| NeuRIS [49] | - | - | - | - | - | **24.40** | 20.19 | 0.773 | 0.682 | 0.723 |
| VF-NeRF (Ours) | 29.25 | **25.28** | **0.960** | **0.857** | **0.905** | **25.01** | **19.2** | **0.913** | **0.868** | **0.888** |

Table 1. **Quantitative results.** Our method outperforms all baselines in terms of the averaged F-score. Additionally, we beat all baselines except MonoSDF in terms of Chamfer Distance. On novel view synthesis, VF-NeRF outperforms all baselines in ScanNet, and renders high-quality images in Replica. **Best result**. **Second best result**.

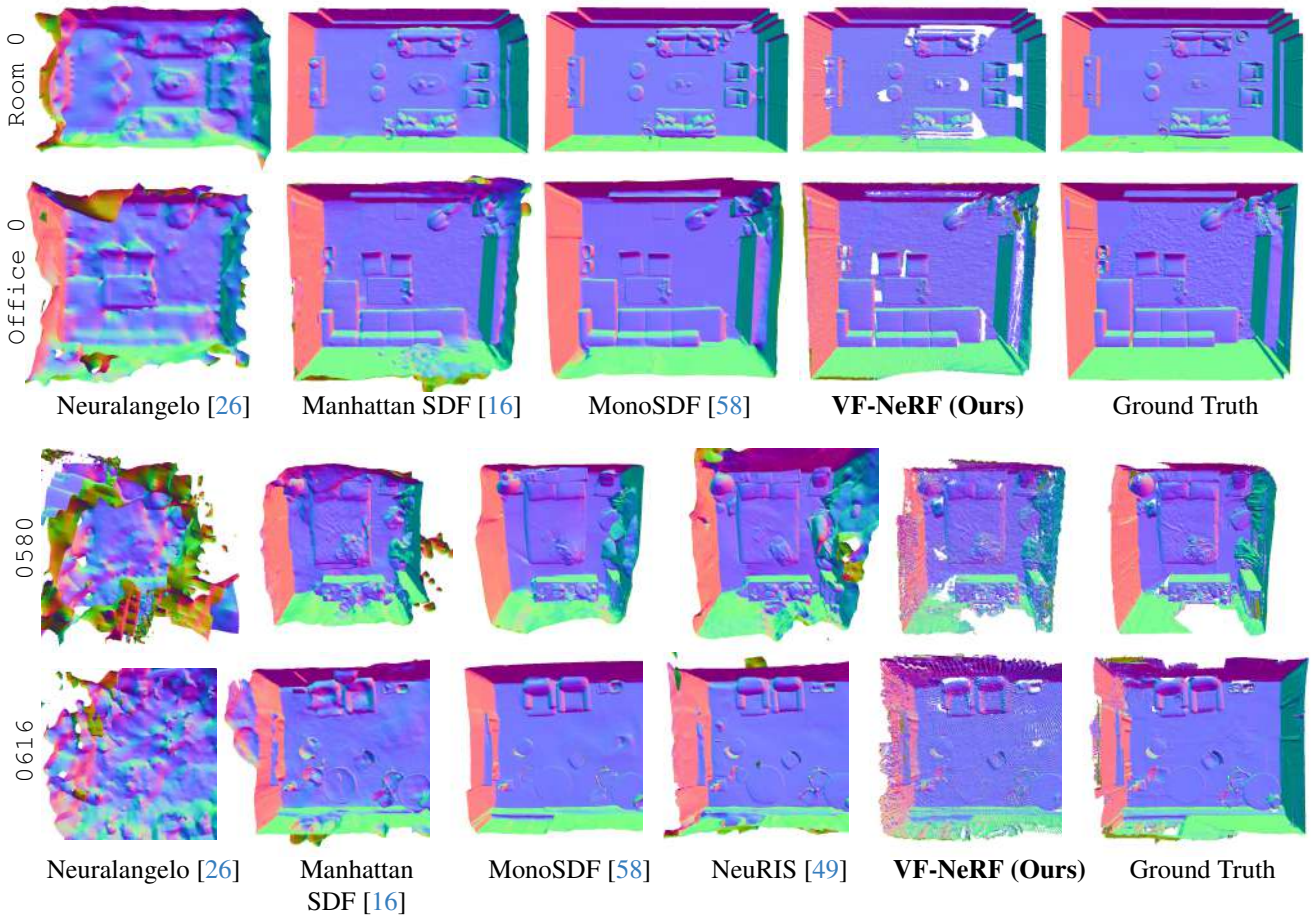
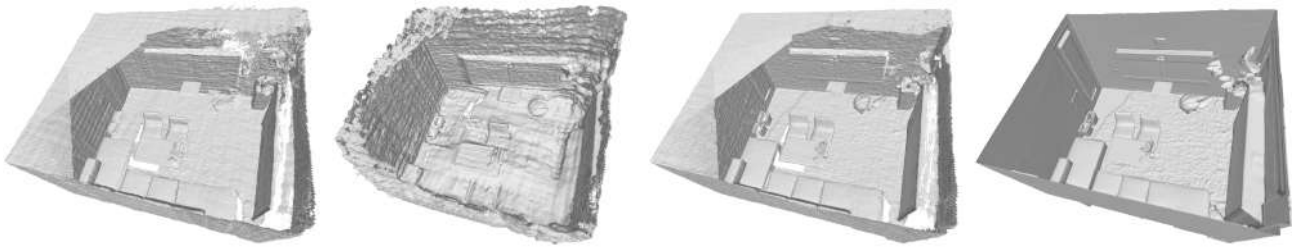



Figure 5. **3D reconstruction qualitative results.** We outperform the SOTA in planar regions of the scenes such as walls and floors.

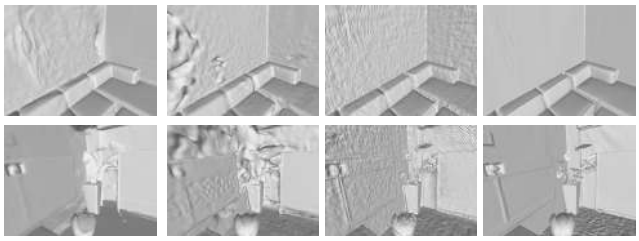
surface reconstruction. Additionally, we find that removing the annealing generates many artifacts as depicted in Fig. 6.

**Number of points per ray**. We investigate the importance of the number of sampled points per ray during training. We provide results using 100, 150 and 200 points per ray in Tab. 2. As expected, we find that using more points per



w\o annealing | w\o $\mathcal{L}_{depth}$ | **VF-NeRF (Ours)** | Ground Truth

Figure 6. **Ablations.** Removing the sliding window annealing generates holes in the reconstructed surfaces and artifacts. Our method without $\mathcal{L}_{depth}$ is less accurate, as most regions of the scene are low-textured. Nonetheless, it still captures the overall scene layout with decent accuracy.



MonoSDF [58] | Manhattan SDF [16] | **VF-NeRF (Ours)** | Ground Truth

Figure 7. **3D reconstruction of planar regions.** VF-NeRF represents planar surfaces with higher accuracy and fewer artifacts compared to the SOTA.

Manhattan SDF [16] | MonoSDF [58] | **VF-NeRF (Ours)** | Ground Truth

Figure 8. **Novel view synthesis qualitative results.** Our method can render accurate images with high-frequency details. Compared to ManhattanSDF, our method is more accurate and introduces less smoothness in both datasets.

ray enhances the performance of our method, although it comes with an increase in training and inference time.

### 4.3. Limitations

One limitation of our method is its inherent smoothing bias, which makes it hard to represent high-frequency details by self-supervised learning. Additionally, VF-NeRF assumes homogeneous density with three hyperparameters. Future works could explore using different hyperparameters depending on the geometry characteristics. Furthermore, our method currently uses a uniform ray sampling strategy; hence, future research could consider integrating a more sophisticated sampling strategy to refine the reconstruction accuracy.

| | Precision↑ | Recall↑ | **F-score↑** | **CD (mm)↓** |
|---|---|---|---|---|
| w\o annealing | 0.844 | 0.801 | 0.822 | 49.7 |
| w\o initialization | 0.994 | 0.821 | 0.899 | 79.46 |
| 100 points per ray | 0.966 | 0.823 | 0.889 | 80.1 |
| 150 points per ray | 0.915 | 0.893 | 0.904 | 20.68 |
| w\o $\mathcal{L}_{ext}$ | 0.990 | 0.822 | 0.898 | 78.92 |
| w\o $\mathcal{L}_{norm}$ | 0.945 | 0.828 | 0.883 | 76.13 |
| w\o $\mathcal{L}_{depth}$ | 0.531 | 0.370 | 0.436 | 161 |
| VF-NeRF | **0.883** | **0.941** | **0.911** | **7.54** |

Table 2. **Ablations quantitative results.** Removing loss terms, sliding window annealing or the VF network initialization decreases our method's performance.

## 5. Conclusion

In this work, we presented VF-NeRF, a novel NERF approach for multiview surface reconstruction, utilizing Vector Fields (VFs) to encapsulate the scene's geometry. By transforming the VF, we can represent the volume density of each point of the scene. The key idea is to learn the VF of the scene through volume rendering. Additionally, we proposed a training scheme, encompassing initial VF training through volume rendering, a VF enhancement by encouraging opposite VF directions at the surface, and a refinement stage, to improve the accuracy of the learned VF. The experiments demonstrate the performance of our method to reconstruct indoor scenes, outperforming state-of-the-art methods on indoor datasets. Additionally, we showcase the effectiveness of our method to reconstruct planar surfaces.

## References


[1] M. Agrawal and L.S. Davis. A probabilistic framework for surface reconstruction from multiple images. *CVPR*, 2001. 2

[2] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *SIGGRAPH*, 2009. 2

[3] Michael Bleyer, Christoph Rhemann, and Carsten Rother. Patchmatch stereo - stereo matching with slanted support windows. *BMVC*, 2011.

[4] A. Broadhurst, T.W. Drummond, and R. Cipolla. A probabilistic framework for space carving. *ICCV*, 2001. 2

[5] R. Chen, S. Han, J. Xu, and H. Su. Point-based multi-view stereo network. *ICCV*, 2019. 2

[6] Shuo Cheng, Zexiang Xu, Shilin Zhu, Zhuwen Li, Li Erran Li, Ravi Ramamoorthi, and Hao Su. Deep stereo using adaptive thin volume representation with uncertainty awareness. *CVPR*, 2020. 2

[7] Christopher B. Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. *ECCV*, 2016. 2

[8] J.M. Coughlan and A.L. Yuille. Manhattan world: compass direction from a single image by bayesian inference. *ICCV*, 1999. 1, 2

[9] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. *CVPR*, 2017. 2, 6

[10] Haoqiang Fan, Hao Su, and Leonidas Guibas. A point set generation network for 3d object reconstruction from a single image. *CVPR*, 2017. 2

[11] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE TPAMI*, 2010. 1, 2

[12] Silvano Galliani, Katrin Lasinger, and Konrad Schindler. Massively parallel multiview stereopsis by surface normal diffusion. *ICCV*, 2015. 2

[13] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. 2017. 5

[14] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. 2019. 5

[15] Xiaodong Gu, Zhiwen Fan, Siyu Zhu, Zuozhuo Dai, Feitong Tan, and Ping Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. *CVPR*, 2020. 2

[16] Haoyu Guo, Sida Peng, Haotong Lin, Qianqian Wang, Guofeng Zhang, Hujun Bao, and Xiaowei Zhou. Neural 3d scene reconstruction with the manhattan-world assumption. *CVPR*, 2022. 1, 2, 6, 7, 8, 10

[17] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004. 1

[18] Sunghoon Im, Hae-Gon Jeon, Steve Lin, and In So Kweon. Dpsnet: End-to-end deep plane sweep stereo. *ICLR*, 2019. 2

[19] Yue Jiang, Dantong Ji, Zhizhong Han, and Matthias Zwicker. Sdfdiff: Differentiable rendering of signed distance fields for 3d shape optimization. *CVPR*, 2019. 2

[20] James T. Kajiya and Brian Von Herzen. Ray tracing volume densities. *SIGGRAPH*, 1984. 3

[21] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *SIGGRAPH*, 2013. 2

[22] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson Surface Reconstruction. *Symposium on Geometry Processing*, 2006. 2

[23] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2015. 6

[24] K.N. Kutulakos and S.M. Seitz. A theory of shape by space carving. *ICCV*, 1999. 2

[25] Zhiyuan Li and Sanjeev Arora. An exponential learning rate schedule for deep learning. *ICLR*, 2020. 6

[26] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. *CVPR*, 2023. 2, 6, 7, 10

[27] Zhengqi Li, Qianqian Wang, Forrester Cole, Richard Tucker, and Noah Snavely. Dynibar: Neural dynamic image-based rendering. 2023. 1

[28] Chen-Hsuan Lin, Chen Kong, and Simon Lucey. Learning efficient point cloud generation for dense 3d object reconstruction. *AAAI Conference on Artificial Intelligence*, 2018. 2

[29] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *NeurIPS*, 2020. 1, 2

[30] Shaohui Liu, Yinda Zhang, Songyou Peng, Boxin Shi, Marc Pollefeys, and Zhaopeng Cui. Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. *CVPR*, 2019. 2

[31] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH*, 1987. 6

[32] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. *CVPR*, 2021. 1, 2

[33] 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. *ECCV*, 2020. 1, 2, 3, 5, 6, 7, 10

[34] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *SIGGRAPH*, 2022. 2

[35] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. *CVPR*, 2020. 2

[36] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. *ICCV*, 2021. 2, 7

[37] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning con-

tinuous signed distance functions for shape representation. *CVPR*, 2019. 2

[38] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. *NeurIPS*, 2019. 6

[39] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. *CVPR*, 2020. 2

[40] Edoardo Mello Rella, Ajad Chhatkuli, Ender Konukoglu, and Luc Van Gool. Neural vector fields for implicit surface representation and inference. 2022. 1, 2, 3

[41] Johannes L. Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. *ECCV*, 2016. 1, 2, 5

[42] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. *CVPR*, 2016. 1, 2, 5, 7

[43] S.M. Seitz and C.R. Dyer. Photorealistic scene reconstruction by voxel coloring. *CVPR*, 1997. 2

[44] S.M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. *CVPR*, 2006. 2

[45] Noah Snavely, Steven M Seitz, and Richard Szeliski. Modeling the world from internet photo collections. *IJCV*, 2008. 1

[46] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Yuheng Ren, Shobhit Verma, Anton Clarkson, Ming Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke Malte Strasdat, Renzo De Nardi, Michael Goesele, S. Lovegrove, and Richard A. Newcombe. The replica dataset: A digital replica of indoor spaces. *ArXiv*, 2019. 2, 6

[47] Jiaming Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. Neuralrecon: Real-time coherent 3d reconstruction from monocular video. *CVPR*, 2021. 6

[48] Engin Tola, Christoph Strecha, and Pascal V. Fua. Efficient large-scale multi-view stereo for ultra high-resolution image sets. *Machine Vision and Applications*, 2011. 1

[49] Jiepeng Wang, Peng Wang, Xiaoxiao Long, Christian Theobalt, Taku Komura, Lingjie Liu, and Wenping Wang. Neuris: Neural reconstruction of indoor scenes using normal priors. *ECCV*, 2022. 3, 5, 6, 7

[50] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *NeurIPS*, 2021. 1, 2, 4, 7

[51] Haozhe Xie, Hongxun Yao, Xiaoshuai Sun, Shangchen Zhou, and Shengping Zhang. Pix2vox: Context-aware 3d reconstruction from single and multi-view images. *ICCV*, 2019. 2

[52] Xianghui Yang, Guosheng Lin, Zhenghao Chen, and Luping Zhou. Neural vector fields: Implicit representation by explicit learning. 2023. 2, 3

[53] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. *ECCV*, 2018. 2

[54] Y. Yao, Z. Luo, S. Li, T. Shen, T. Fang, and L. Quan. Recurrent mvsnet for high-resolution multi-view stereo depth inference. *CVPR*, 2019. 2

[55] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *NeurIPS*, 2020. 2

[56] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *NeurIPS*, 2021. 1, 2, 4, 7, 10

[57] Zehao Yu, Anpei Chen, Bozidar Antic, Songyou Peng, Apratim Bhattacharyya, Michael Niemeyer, Siyu Tang, Torsten Sattler, and Andreas Geiger. Sdfstudio: A unified framework for surface reconstruction, 2022. 11

[58] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. MonoSDF: Exploring monocular geometric cues for neural implicit surface reconstruction. *NeurIPS*, 2022. 1, 2, 3, 5, 6, 7, 8, 10

[59] Enliang Zheng, Enrique Dunn, Vladimir Jojic, and Jan-Michael Frahm. Patchmatch based joint view selection and depthmap estimation. *CVPR*, 2014. 1, 5

## A. Detailed networks' architecture

We present an illustration of the VF and color networks in Figure 9. As the first MLP, the VF network predicts the VF at a location in space $\mathbf{x}$ and a feature vector $\mathbf{z}$ that is later used to predict the color at that point. The second smaller MLP is the color network. It takes as input the VF, the position in space $\mathbf{x}$, the direction $\mathbf{d}$[1] and the predicted feature vector $\mathbf{z}$. As explained in the main paper, the two networks are applied in a rendering pipeline similar to [33] and optimized jointly.

## B. Metrics

To evaluate the method, we use the standard metrics for the task [16, 26, 56, 58]. To evaluate the reconstruction accuracy, we use the precision, recall and F1-score together with the Chamfer Distance (CD). Given that the CD is sensitive to outliers, we report both its mean and median value. To evaluate the rendering capability of the network, we use the standard Peak Signal-to-Noise Ratio (PSNR). The definitions of metrics are reported in Table 3.

[1] Positional encoding $PE(\cdot)$ is applied to it

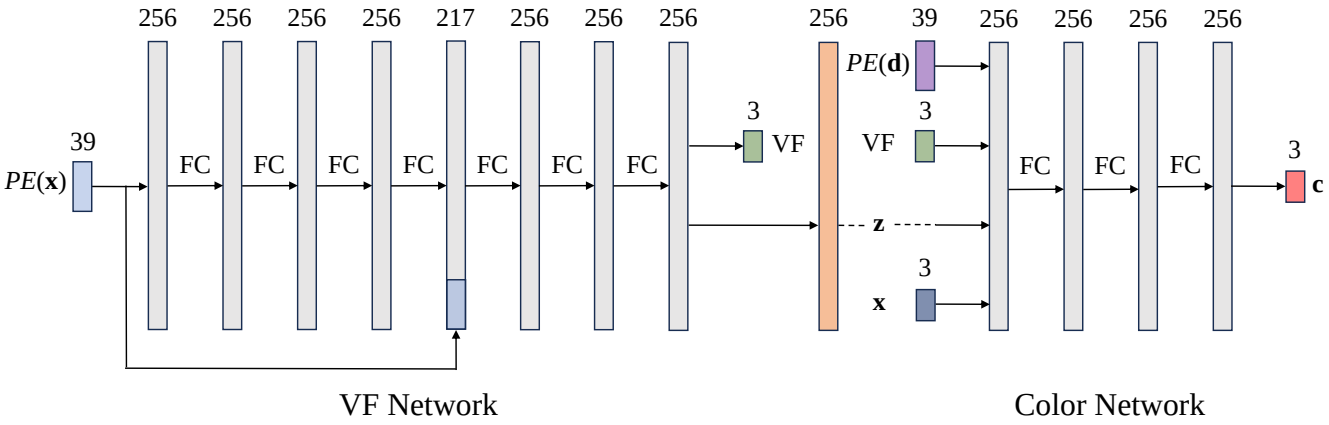



Figure 9. **Networks' architecture.** The VF network takes a point in space $\mathbf{x}$ as input and applies positional encoding ($PE(\cdot)$) before feeding it to the MLP. The color network takes the spatial point, the predicted VF, the feature vector $\mathbf{z}$, and the viewing direction $\mathbf{d}$ with positional encoding as inputs to predict the color.

| Metric | Definition |
|---|---|
| Precision | $\text{mean}_{p \in P}(\min_{p^* \in P^*} \|\|p - p^*\|\| < 0.05)$ |
| Recall | $\text{mean}_{p^* \in P^*}(\min_{p \in P} \|\|p - p^*\|\| < 0.05)$ |
| F1-score | $\frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$ |
| CD | $\sum_{p \in P} \min_{p^* \in P^*} \|\|p^* - p\|\|_2^2 + \sum_{p^* \in P^*} \min_{p \in P} \|\|p^* - p\|\|_2^2$ |
| MSE | $\frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \|\|C(i,j) - \widehat{C}(i,j)\|\|_2^2$ |
| PSNR | $-10 \cdot \log_{10}(\text{MSE})$ |

Table 3. **Metric definitions.** $P$ and $P^*$ are the point clouds sampled from the rendered and ground truth meshes. $M$ and $N$ are the height and weight of the images. $C$ and $\widehat{C}$ are the ground truth and rendered images.

## C. Baselines

We use the official Manhattan-SDF [2], MonoSDF [3] and NeuRIS [4] implementations as baselines. We adapted the Replica dataset to use it in Manhattan-SDF, while unfortunately NeuRIS does not support it. Additionally, we use SDFStudio's [57] implementation [5] of Neuralangelo.

## D. 3D reconstruction quantitative results

We quantitatively evaluate the capacity of our method to reconstruct Replica and ScanNet scenes and compare it against state-of-the-art neural volume rendering methods. We present these quantitative results for each scene in Table 4. VF-NeRF is among the 2 best methods on every Replica scene and the best method on every ScanNet scene in terms of F-score. Additionally, we observe that our method outperforms by a large margin all the others on the median CD and is among the best in mean CD. We note that median CD is more representative of the overall performance of the methods due to the high sensitivity to outliers of the mean CD. The large difference between mean CD and median CD (over 2 orders of magnitude in many cases) is caused by the overall poor performance of every method in representing the ceiling of the rooms. This is mainly due to the lack of observations that actually include the ceiling.

## E. 3D reconstruction qualitative results

We provide qualitative results for each scene of Replica and ScanNet in Figure 10 and Figure 11. We include visualizations of state-of-the-art 3D reconstruction methods as comparisons. VF-NeRF always achieves very accurate results and avoids generating artifacts on the walls, something that often happens in SDF-based methods.

## F. Novel view synthesis qualitative results

Figure 12 presents qualitative comparisons of novel view synthesis for each scene on Replica and Scannet. Tab. 5 showcases the quantitative results for each individual scene in both datasets.

[2] https://github.com/zju3dv/manhattan_sdf
[3] https://github.com/autonomousvision/monosdf
[4] https://github.com/jiepengwang/NeuRIS
[5] https://github.com/autonomousvision/sdfstudio

| | Replica | | | | | | | | ScanNet | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | room0 | room1 | room2 | office0 | office1 | office3 | office4 | Mean | 0050 | 0084 | 0580 | 0616 | Mean |
| | **Precision↑** | | | | | | | | | | | | |
| Manhattan-SDF | 0.674 | 0.867 | 0.746 | 0.703 | 0.382 | 0.905 | 0.784 | 0.723 | 0.819 | 0.892 | 0.685 | 0.714 | 0.778 |
| Neuralangelo | 0.265 | 0.458 | 0.176 | 0.261 | 0.269 | 0.122 | 0.153 | 0.243 | 0.359 | - | 0.310 | 0.138 | 0.269 |
| MonoSDF | **0.924** | **0.959** | **0.944** | **0.778** | **0.883** | **0.915** | **0.941** | **0.906** | **0.857** | **0.928** | **0.814** | **0.854** | **0.863** |
| NeuRIS | - | - | - | - | - | - | - | - | 0.822 | 0.776 | 0.740 | 0.755 | 0.773 |
| VF-NeRF (Ours) | **0.984** | **0.961** | **0.987** | **0.976** | **0.968** | **0.934** | **0.908** | **0.960** | **0.910** | **0.922** | **0.938** | **0.881** | **0.913** |
| | **Recall↑** | | | | | | | | | | | | |
| Manhattan-SDF | **0.924** | **0.926** | 0.854 | **0.819** | 0.691 | **0.882** | **0.899** | 0.856 | 0.662 | 0.854 | **0.735** | 0.523 | 0.694 |
| Neuralangelo | 0.338 | 0.370 | 0.279 | 0.417 | 0.207 | 0.482 | 0.166 | 0.323 | 0.233 | - | 0.262 | 0.070 | 0.188 |
| MonoSDF | **0.964** | **0.912** | **0.934** | 0.802 | **0.829** | **0.878** | **0.904** | **0.889** | 0.660 | **0.896** | 0.725 | **0.637** | **0.730** |
| NeuRIS | - | - | - | - | - | - | - | - | **0.699** | 0.741 | 0.719 | 0.568 | 0.682 |
| VF-NeRF (Ours) | 0.893 | 0.887 | **0.871** | **0.818** | **0.779** | 0.852 | 0.897 | **0.857** | **0.830** | **0.950** | **0.943** | **0.748** | **0.868** |
| | **F-score↑** | | | | | | | | | | | | |
| Manhattan-SDF | 0.778 | 0.896 | 0.796 | 0.757 | 0.492 | 0.893 | 0.838 | 0.779 | 0.732 | 0.873 | 0.709 | 0.604 | 0.730 |
| Neuralangelo | 0.297 | 0.410 | 0.216 | 0.321 | 0.234 | 0.195 | 0.159 | 0.262 | 0.283 | - | 0.284 | 0.093 | 0.220 |
| MonoSDF | **0.944** | **0.935** | **0.939** | **0.790** | **0.855** | **0.896** | **0.922** | **0.897** | 0.745 | **0.911** | **0.767** | **0.730** | **0.788** |
| NeuRIS | - | - | - | - | - | - | - | - | **0.755** | 0.758 | 0.729 | 0.648 | 0.723 |
| VF-NeRF (Ours) | **0.937** | **0.923** | **0.925** | **0.890** | **0.864** | **0.891** | **0.902** | **0.905** | **0.868** | **0.936** | **0.940** | **0.809** | **0.888** |
| | **Mean Chamfer Distance (mm)↓** | | | | | | | | | | | | |
| Manhattan-SDF | 494 | 65.2 | 392 | **77.5** | 1266 | 7.68 | 149 | 350 | **11.0** | **9.18** | 23.1 | 47.9 | 22.80 |
| Neuralangelo | 1113 | 67.9 | 1107 | 317 | 280 | 5464 | 1002 | 1336 | 95.6 | - | 196 | 523 | 272 |
| MonoSDF | **2.72** | **3.44** | **2.95** | **9.23** | **12.0** | **4.50** | **2.49** | **5.33** | 12.1 | **5.80** | **12.6** | **40.8** | **17.83** |
| NeuRIS | - | - | - | - | - | - | - | - | 11.3 | 10.6 | 24.6 | **34.3** | 20.2 |
| VF-NeRF (Ours) | **7.72** | **6.73** | **13.9** | 80.6 | **56.1** | **6.95** | **4.96** | **25.28** | **6.78** | 11.9 | **6.74** | 51.3 | **19.2** |
| | **Median Chamfer Distance (mm)↓** | | | | | | | | | | | | |
| Manhattan-SDF | 0.87 | 0.34 | 0.91 | **0.42** | 35.6 | 0.41 | 0.63 | 5.60 | 1.25 | 1.09 | 1.76 | 2.71 | 1.45 |
| Neuralangelo | 39.0 | 12.5 | 215 | 39.0 | 44.5 | 3754 | 174 | 611 | 28.8 | - | 29.2 | 251 | 103 |
| MonoSDF | **0.21** | **0.23** | **0.46** | 0.82 | **0.34** | **0.33** | **0.22** | **0.37** | 2.13 | **0.90** | **1.45** | **1.19** | **1.42** |
| NeuRIS | - | - | - | - | - | - | - | - | **0.57** | 2.35 | 1.48 | 2.42 | 1.71 |
| VF-NeRF (Ours) | **0.12** | **0.09** | **0.11** | **0.09** | **0.06** | **0.16** | **0.14** | **0.11** | **0.28** | **0.07** | **0.18** | **0.37** | **0.23** |

Table 4. **3D reconstruction quantitative results of individual scenes on Replica and ScanNet.** **Best result**. **Second best result**. Note Neuralangelo fails to reconstruct a valid geometry for scene 0580 of ScanNet.

| | PSNR↑ | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Replica | | | | | | | | ScanNet | | | | |
| | room0 | room1 | room2 | office0 | office1 | office3 | office4 | Mean | 0050 | 0084 | 0580 | 0616 | Mean |
| Manhattan-SDF | 25.06 | 26.38 | 29.36 | 28.87 | 26.39 | 28.35 | 27.92 | 27.48 | 22.44 | 18.92 | 22.87 | 18.90 | 20.78 |
| Neuralangelo | **28.22** | **30.45** | **29.59** | **36.02** | **36.15** | **29.54** | **30.14** | **31.44** | 17.48 | 18.66 | 18.40 | 16.78 | 17.83 |
| MonoSDF | **27.91** | **30.29** | **31.16** | **36.26** | **36.80** | **30.70** | **32.63** | **32.25** | 17.61 | **33.11** | **27.16** | 17.46 | 23.84 |
| NeuRIS | - | - | - | - | - | - | - | - | **23.29** | 27.63 | **23.56** | **23.14** | **24.41** |
| VF-NeRF (Ours) | 27.18 | 27.14 | 29.10 | 34.95 | 27.84 | 28.65 | 29.91 | 29.25 | **23.47** | **30.40** | 20.02 | **26.14** | **25.01** |

Table 5. **Novel view synthesis quantitative results.** **Best result**. **Second best result**.




Figure 10. **3D reconstruction qualitative results on Replica.**

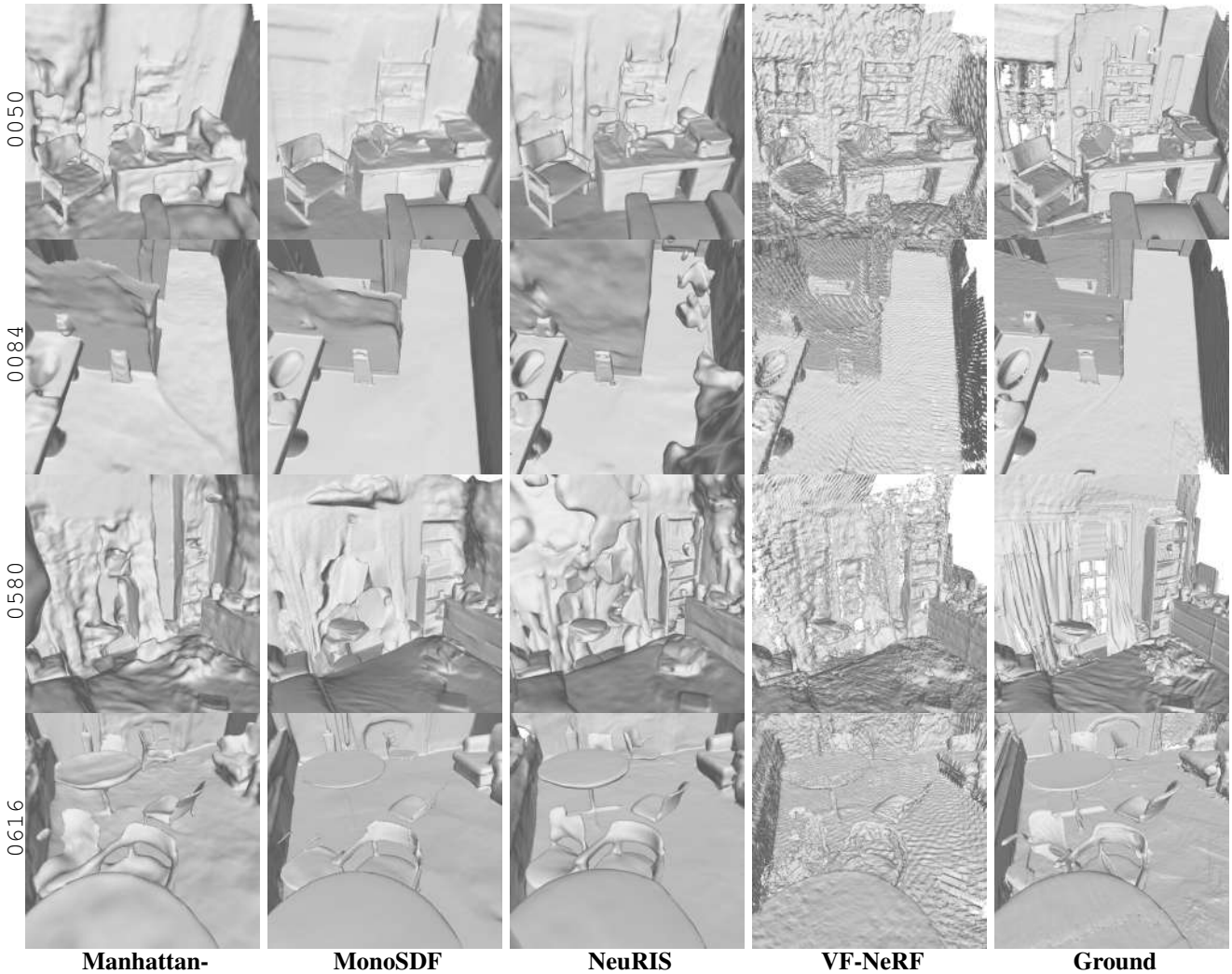



Figure 11. **3D reconstruction qualitative results on ScanNet.**

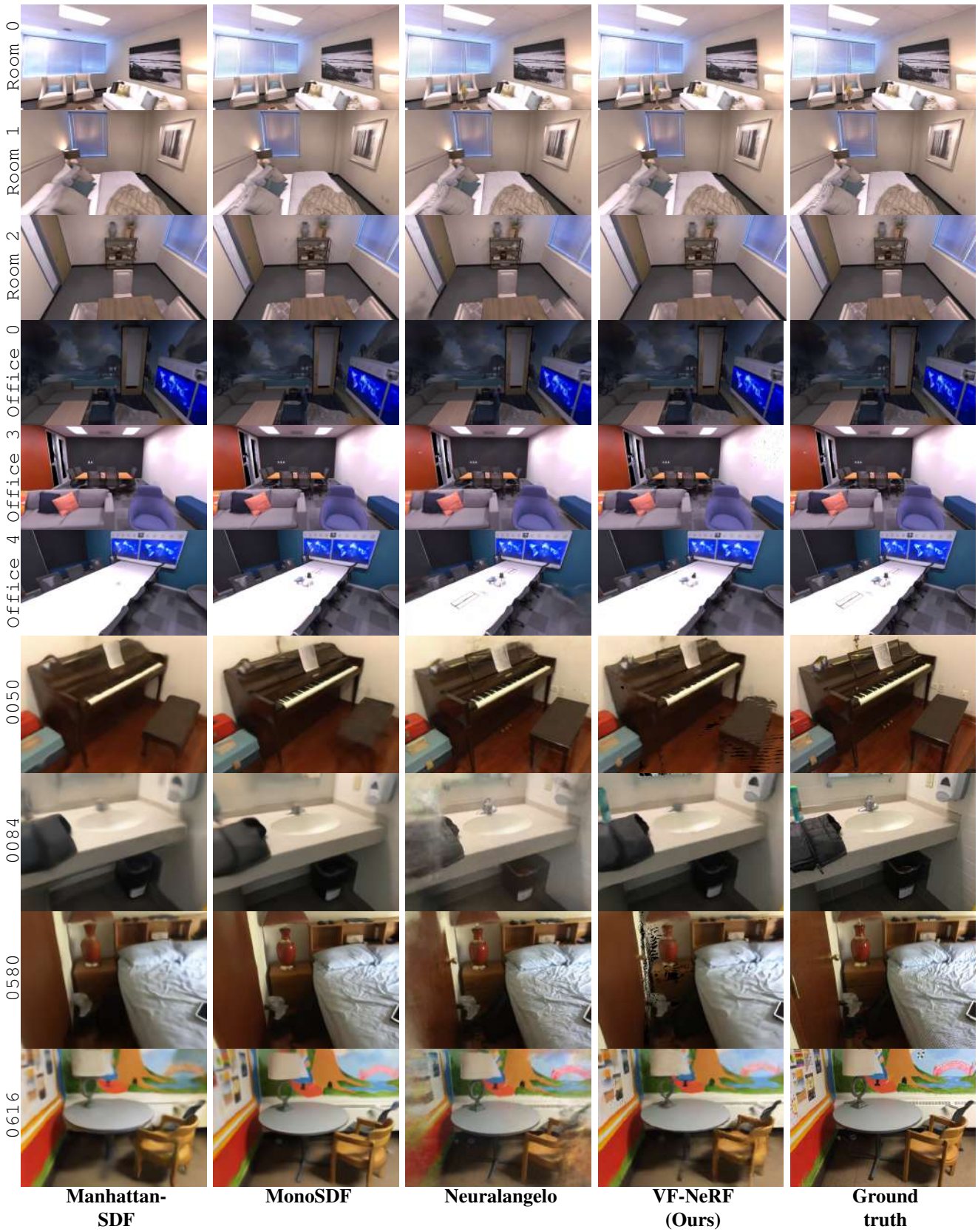



Figure 12. **Novel view synthesis qualitative results.**