ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CVL Computer Vision Lab

Learning Neural Vector Fields for Implicit Surfaces and Multi-view Reconstruction

Master's Thesis

Albert Gassol Puigjaner
Department of Image Communication and Understanding

Advisors: Edoardo Mello Rella, Erik Sandström, Dr. Ajad Chhatkuli
Supervisor: Prof. Dr. Luc van Gool

October 30, 2023

Abstract

Neural implicit surface reconstruction methods have gained popularity due to their ability to recover accurate dense 3D surfaces via neural volume rendering. Despite the success in reconstructing richly textured objects, existing methods struggle when faced with areas of low texture, particularly planar surfaces. This difficulty with low-texture originates due to the lack of constraints, since these methods rely on photometric consistency only. To address this issue, we propose VF-NeRF, a novel neural surface reconstruction method that models the scene geometry as the Vector Field (VF), which consists of a unit vector directed to the nearest surface point. Specifically, we make use of a dual Multi-Layer Perceptron (MLP) to represent the VF and the color of the scene. Furthermore, we develop a novel density function as a transformation of the VF and use it to learn the VF representation through volume rendering. Our method is further refined by a tri-phased optimization routine comprising initial VF training, a self-learning stage, and a refinement phase. We strategically incorporate depth map priors into the optimization to aid the training process and enhance the scene representation. We demonstrate the capacity of our method to accurately reconstruct indoor scenes and show state-of-the-art results in public datasets such as Replica and ScanNet.

Contents

1 Introduction **1**

2 Related Work **3**

3 Method **5**

3.1 Neural Volume Rendering 6

3.2 Surface Representation with Vector Fields 6

3.2.1 Window cosine similarity and annealing 7

3.2.2 Learning the VF from the cosine similarity 8

3.3 Density as Transformed VF 9

3.4 Training 10

3.5 Indoor Scenes Training 10

4 Experiments and Results **13**

4.1 Experimental setup 13

4.1.1 Implementation details 13

4.1.2 Datasets, metrics and baselines 13

4.2 Object centric scenes 14

4.2.1 Non-explicit Vector Field learning 14

4.2.2 Multi-view 3D reconstruction 15

4.3 Indoor scenes 17

4.3.1 3D reconstruction 17

4.3.2 Novel view synthesis 20

4.3.3 Limitations 20

4.4 Ablation studies 21

5 Conclusion **23**

A Supplementary material **25**

A.1 Detailed networks' architecture 25

A.2 Metrics 25

A.3 Baselines 25

A.4 3D reconstruction quantitative results 26

A.5 3D reconstruction qualitative results 26

A.6 Novel view synthesis qualitative results 26

List of Figures

1.1 VF-NeRF 2

3.1 Overview of the method 5
3.2 Sliding window weights 8
3.3 Density function 9

4.1 Non-explicit VF learning qualitative results 14
4.2 Non-explicitly learned projected Vector Field 15
4.3 Object-centric qualitative results 16
4.4 Projected Vector Field of object-centric scenes 16
4.5 3D reconstruction qualitative results on Replica 18
4.6 3D reconstruction qualitative results on Scannet 19
4.7 3D reconstruction of plannar regions 19
4.8 Novel view synthesis qualitative results 20
4.9 Limitations 21
4.10 Losses ablations 22
4.11 Annealing and initialization ablations 22

A.1 Networks’ architecture 25
A.2 3D reconstruction qualitative results on Replica 28
A.3 3D reconstruction qualitative results on ScanNet 29
A.4 Novel view synthesis qualitative results 30

List of Tables

4.1 3D reconstruction quantitative results 17
4.2 Novel view synthesis quantitative results 20
4.3 Ablations quantitative results 21

A.1 Metric definitions 26
A.2 3D reconstruction quantitative results 27

Chapter 1

Introduction

Multi-view image-based 3D scene reconstruction is a cornerstone challenge in computer vision [15]. Such reconstructed geometries have a wide array of applications, spanning augmented reality to mapping for robotic applications. Traditional multi-view stereo (MVS) algorithms [11, 40, 41, 46, 57] 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.

Learning-based methods have led to significant improvements in multi-view reconstruction. Neural Radiance Fields (NeRF)[27, 31, 32, 38] model the scene surface density and color with a multi-layer perceptron (MLP) to map the points in space and their corresponding viewing directions to the density and RGB values. Even though these methods are capable of generating high-quality novel views via volume rendering, the reconstructed geometry is often noisy and inaccurate due to the low fidelity of the predicted surface density.

Neural implicit representations have gained significant attention in recent studies [34, 54], primarily due to their capability to address constraints inherent in traditional methods and improve the low-fidelity geometries reconstructed by NeRF. Employing an MLP to model the surface's implicit function and volume rendering [32] to render the scene into images, methods like [35] use occupancy grids as the implicit function, whilst, in contrast, works like [24, 48, 53] employ the Signed Distance Function (SDF). Recent works, such as [29], make use of the Unsigned Distance Function (UDF), which can accurately represent open and closed surfaces. However, the reconstructed geometries are often noisy. Moreover, the aforementioned methods' dependency on photometric consistency for implicit function learning can be problematic, particularly in low-textured planar regions.

To improve the geometry reconstruction of planar regions, the authors of ManhattanSDF [14] propose to use semantic and geometric constraints in the optimization while still leveraging the SDF as the neural implicit function. Even though this method has proven to be successful in reconstructing indoor scenes with a majority of planar surfaces, ManhattanSDF requires a considerable amount of assumptions and constraints.

Recently, Vector Field (VF) representation has emerged as a novel surface implicit representation, as illustrated in [39]. This method involves associating each position in the 3D space with a unit vector directed towards the nearest surface. Notably, the VF showcases its ability to faithfully represent both open and closed surfaces, with particular strength in piecewise planar surfaces. However, the study confines itself to a supervised learning paradigm, whereby the representation is directly learned from provided meshes, point clouds, or equivalent surface representations.

In this work, we suggest using VF as the implicit representation for 3D surfaces and learning it with multi-view images through neural volume rendering [53]. We adopt a dual-MLP strategy: one MLP estimates the VF for any spatial point, and the other predicts RGB color values. Drawing inspiration from [48, 53], we express the surface density in neural volume rendering based on the VF, improving training stability in the presence of sudden depth variations. Our approach proves particularly effective on piecewise planar surfaces, like indoor scenes, delivering state-of-the-art results. We rigorously evaluate our method against leading benchmarks for indoor scenes, including ManhattanSDF [14], NeuRIS [47], MonoSDF [56], and Neuralangelo [24], in indoor datasets such as Replica [44] and ScanNet [9]. Two examples of reconstructed indoor scenes using our method are presented in Figure 1.1. 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.

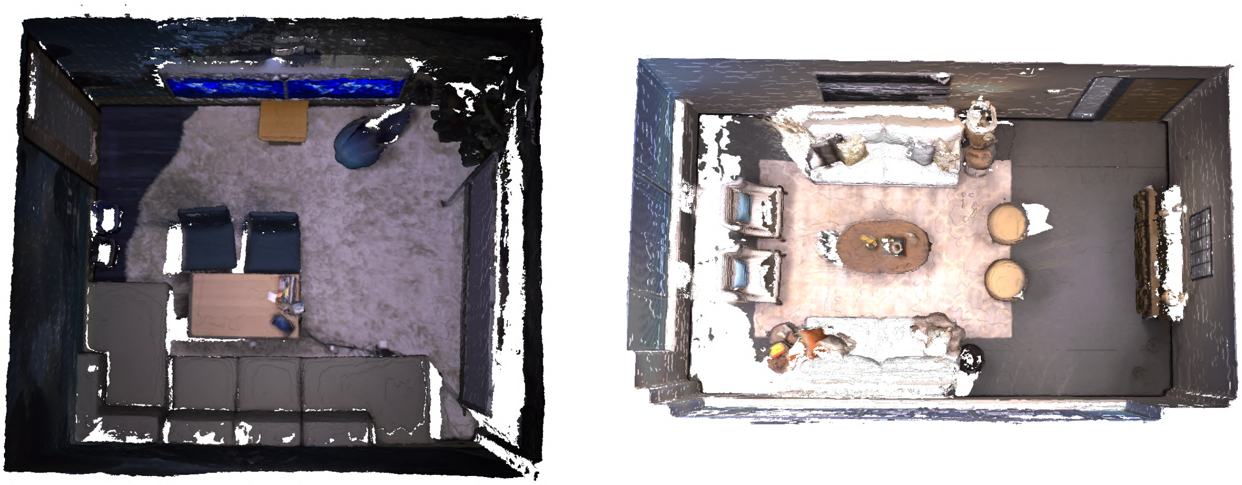


Figure 1.1: **VF-NeRF.** Indoor scenes reconstructed with VF-NeRF.

Thesis Organization

This project is divided into three main stages. In the initial stage, we investigate how to learn the VF without directly using the ground truth normal vectors as training targets. During this phase, our goal is to identify a function that, when applied to the VF, effectively represents a surface. Once we have this function, we delve into how it can be used in the training process to effectively learn the VF representation.

The second stage involves learning the VF representation through volume rendering using multi-view images. In this context, we delve into the design of a neural volume rendering density function that depends on the predicted VF. Furthermore, a cost function with competing objectives is engineered to achieve the desired results. To enhance the results, we also introduce some supplementary components.

In the final stage, we focus on applying the VF representation to indoor scenes. Within this scope, we introduce some indoor-specific objectives to the cost function. Additionally, we devise an optimization strategy that includes two phases of multi-view reconstruction using neural volume rendering, separated by a self-supervision phase. Notably, we find that this optimization approach enhances the VF representation.

The aforementioned phases, together with all the implementation details, are specified in Chapter 3. Following the methodology, the experiments, their results, and comparisons with benchmarks are elaborated in Chapter 4. Finally, concluding remarks and findings of this study are presented in Chapter 5.

Chapter 2

Related Work

Multi-view Surface Reconstruction. Traditional MVS approaches have often relied on feature matching for depth estimation [2, 3, 4, 12, 23, 40, 41, 42]. 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 the Poisson surface reconstruction [20, 21], are applied to generate 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 missing surfaces.

Recently, learning-based methods have gained attention, offering replacements for classic MVS methods. Methods like [5, 16, 50, 52] leverage 3D CNNs to extract features and predict depth maps, while others [6, 13] 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 [27, 31, 32, 38] the potential of MLPs to represent scenes, both in terms of density and appearance, using a singular network has been explored. While these techniques can produce photorealistic outcomes for novel view synthesis, determining an isosurface for the volume density to illustrate 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. Neural-based scene representation approaches employ deep learning to learn properties of 3D points and generate geometry representations. Traditional methods such as point clouds [10, 26] and voxel grids [7, 49] have been primary choices for detailing scene geometry. More recently, implicit functions, such as occupancy grids [34, 35] and SDF [18, 24, 28, 36, 48, 53, 54], have gained popularity due to their precision in capturing scene geometry. For instance, in [28, 34] a novel differentiable renderer to learn the scene geometry from images is proposed, while [54] focuses on modeling view-dependence appearance, which proves successful in non-Lambertian surfaces. However, these methods rely on masks to accurately reconstruct the geometry from multi-view images. Consequent works, such as VolSDF [53] and NeuS [48], 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 [24] takes inspiration from Instant Neural Graphics Primitives (Instant NGP) [33] 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 reconstruction.

Priors for Neural Scene Representations. Several studies have explored the integration of priors into the optimization to improve the reconstruction of large indoor scenes. For instance, Manhattan-SDF [14] suggests incorporating dense depth maps from COLMAP [41] to facilitate the 3D geometry learning 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 semantic 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 segmentation is less accurate. More recently, NeuRIS [47] proposes to use normal priors to guide the reconstruction of indoor scenes. Expanding on this work, MonoSDF [56] leverages both normal and depth priors, incorporating monocular cues into the optimization process. By using normal priors, these methods successfully remove the Manhattan world assumption, thereby enhancing the reconstruction of indoor scenes.

Chapter 3

Method

In this work, we propose VF-NeRF, a novel method for reconstructing high-quality scene geometry using multi-view posed images. Our method uses VF to represent the scene geometry and color to represent its appearance. To this end, we first review neural volume rendering as a means to derive geometry from images in Section 3.1. We then address how the scene geometry can be represented using VF in Section 3.2 and discuss how this representation is used to define the surface density at any point in space in Section 3.3. Finally, we outline the complete optimization process in Section 3.4 and present the optimization strategy for indoor scene reconstruction in Section 3.5. An overview of our method is presented in Figure 3.1.

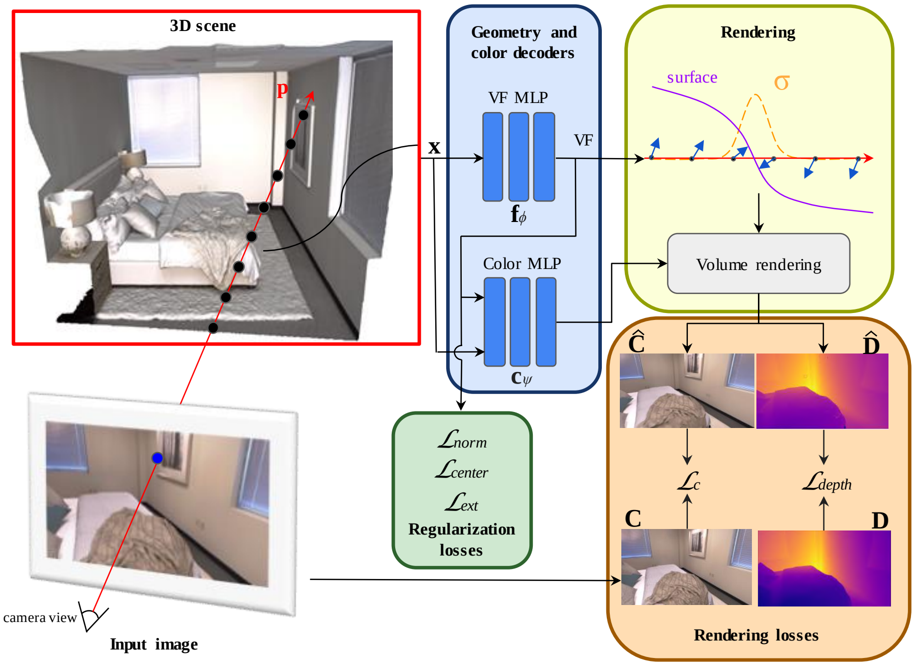


Figure 3.1: **Overview of the method.** We use VF to represent the geometry of a scene. Specifically, given a batch of rays, we predict the VF and color of the points of the ray to volume render the predicted RGB and depth. We design a specific function to transform VF to surface density.

3.1 Neural Volume Rendering

Neural volume rendering [19] is a method to generate images from spatial locations consisting of 3D points and their viewing direction. Given a posed image and its viewing direction, volume rendering integrates the color radiance of points along the ray. Denoting a ray emitted from a pixel belonging to the image as $\mathbf{p}(t) = \mathbf{o} + \mathbf{d}t$ with near and far bounds t_n and t_f, the color and depth of the pixel can be computed as:

$$C(\mathbf{p}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{p}(t))\mathbf{c}(\mathbf{p}(t), \mathbf{d})dt \tag{3.1}$$

$$D(\mathbf{p}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{p}(t))tdt \tag{3.2}$$

where $\mathbf{o}$ is the camera origin, $\mathbf{d}$ is the normalized camera viewing direction, $\sigma(\cdot)$ represents the surface density, $\mathbf{c}(\cdot, \cdot)$ denotes the color at a given point and direction, and $T(t)$ denotes the accumulated transmittance along the ray from t_n to t. In particular, the transparency function is defined as:

$$T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{p}(s))ds\right) \tag{3.3}$$

In this work, we adopt NeRF's strategy [32] to discretize Equation (3.1) and Equation (3.3) using numerical quadrature, namely the Riemann sum. Hence, we represent the color and depth of a pixel as follows:

$$C(\mathbf{p}) \approx \sum_{i=1}^{N} T_i(1 - \exp(-\sigma_i\delta_i))\mathbf{c}_i, \quad \text{where } T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j\delta_j\right) \tag{3.4}$$

$$D(\mathbf{p}) \approx \sum_{i=1}^{N} T_i(1 - \exp(-\sigma_i\delta_i))t_i \tag{3.5}$$

where $\delta_i = t_{i+1} - t_i$ is the distance between samples along a ray. Note that Equation (3.4) can be seen as traditional alpha composing with alpha values $\alpha_i = 1 - \exp(\sigma_i\delta_i)$.

Methods such as NeRF and its variants [27, 31, 32, 38] propose to model both surface density and color functions within a singular MLP with two heads. Notably, the viewing direction is only considered during color prediction. Nevertheless, surfaces generated through these methods tend to be noisy and unrealistic. More recent works, such as [24, 48, 53] define the surface density in terms of a learned SDF, which has proven to enhance the quality of the reconstructed surfaces. In this work, we adopt a similar approach by defining the surface density as a transformation of the learned VF. Section 3.2 and Section 3.3 provide the details of how the VF can be used to represent surfaces and the transformation needed to predict the surface density for volume rendering.

3.2 Surface Representation with Vector Fields

In VF-NeRF, the scene geometry is described using unit vectors that point towards the nearest surface. Let $\Omega \subset \mathbb{R}^3$ be an object's surface in $\mathbb{R}^3$ and $\Gamma \subset \mathbb{R}^3$ be the collection of normal unit 3D vectors. We define VF as a function $\mathbf{f} : \mathbb{R}^3 \to \Gamma$ that maps a point in space to its unit normal vector that points to the closest surface point of Ω. It's important to mention that for points on the surface Ω, the VF experiences sudden changes, shifting in direction. At this surface, the VF can correspond to either of the two opposing normal vectors. For a formal definition of VF, we reference [39] and present it as follows:

$$\mathbf{f}(\mathbf{x}) = \begin{cases} \dfrac{\mathbf{x}_S - \mathbf{x}}{||\mathbf{x}_S - \mathbf{x}||_2} & \text{if } \mathbf{x} \notin \Omega \\ \dfrac{\mathbf{x}_S - \widehat{\mathbf{x}}}{||\mathbf{x}_S - \widehat{\mathbf{x}}||_2} & \text{if } \mathbf{x} \in \Omega \end{cases} \tag{3.6}$$

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 $\widehat{\mathbf{x}} = \lim_{||\epsilon||_2 \to 0} \mathbf{x} + \epsilon$ is a surface point, with $\epsilon \in \mathbb{R}^3$ being an infinitessimal 3D vector.

Given the definition of the VF representation, we aim to identify a function that can determine, using the VF, whether a surface exists at a particular point in space. While the authors of [39] suggest using the flux density of the VF to represent the surface of scenes, this operation requires evaluating the VF within an infinitesimal spherical surface. This means that $\mathbf{f}$ needs to be evaluated at several points. For this reason, in this work, we propose to use the cosine similarity between the VF at a given point $\mathbf{x}$ and an infinitesimally close neighbor. This approach simplifies the process as it only demands the evaluation of $\mathbf{f}$ at two distinct points in space. Formally, we express the cosine similarity between two VF vectors $\mathbf{v}_1, \mathbf{v}_2 \in \Gamma$ as follows:

$$\cos(\mathbf{v}_1, \mathbf{v}_2) = \frac{\mathbf{v}_1 \cdot \mathbf{v}_2}{||\mathbf{v}_1||_2 ||\mathbf{v}_2||_2} \tag{3.7}$$

From a theoretical point of view, when considering a point in space and its neighboring point that is extremely close, their VFs will yield a cosine similarity of -1 if they are located on opposing sides of the surface. Yet, in practice, the cosine similarity can exceed -1 for such points due to the varying distances between neighboring points. To address this, we introduce a cosine similarity threshold to pinpoint surface points. More explicitly, we represent the surface Ω of a scene as follows:

$$\Omega = \{\mathbf{x}_1 \in \mathbb{R}^3, \mathbf{x}_2 = \mathbf{x}_1 + \epsilon \mid \cos(\mathbf{f}(\mathbf{x}_1), \mathbf{f}(\mathbf{x}_2)) < \gamma\} \tag{3.8}$$

where $\epsilon \in \mathbb{R}^3$ is an infinitesimal displacement and $|\gamma| \leq 1$ is a cosine similarity threshold.

3.2.1 Window cosine similarity and annealing

As mentioned at the beginning of this section, the VF experiences a sudden sign change when crossing a surface. Although this is the desired VF behavior, this can constitute training problems due to the non-smoothness behavior. To this end, we adopt a sliding window approach to compute the cosine similarity and smooth it at points near and at the surface. Additionally, the weights of the sliding window are annealed throughout the training process.

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 predicted sliding window and predicted VFs are used to compute the averaged cosine similarity associated with each point. The smoothed cosine similarity of a point is computed as the weighted average of the cosine similarities using 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 smoothed cosine similarities as:

$$c^i_{sim}(\mathbf{r}) = \begin{cases} \cos(\mathbf{f}(\mathbf{x}_i), \mathbf{f}(\mathbf{x}_{i+1})) & \text{if } i < M/2 + 1 \text{ or } i > N - M/2 - 1 \\ \sum_{j=0}^{M/2-1} \left[w_j \cos(\mathbf{f}(\mathbf{x}_i), \mathbf{f}(\mathbf{x}_{i-j-1})) + w_{j+M/2} \cos(\mathbf{f}(\mathbf{x}_i), \mathbf{f}(\mathbf{x}_{i+j+1})) \right] \end{cases} \tag{3.9}$$

Note that the cosine similarity of the first and last points of the ray is not smoothed 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 neighbor.

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

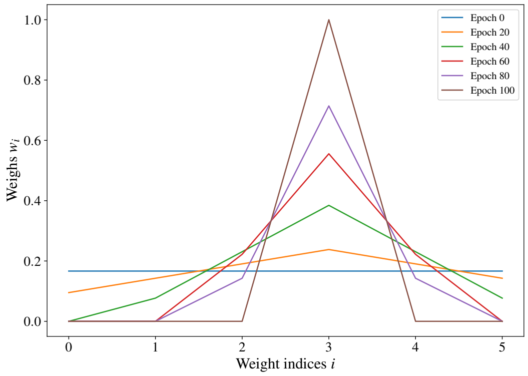


Figure 3.2: **Sliding window weights**. Weights of the sliding window at different stages of the annealing.

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. The annealing process is depicted in Figure 3.2. 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:

$$\widehat{w}_i = \frac{M}{2}\text{ReLU}\left(1 - \frac{n|i - M/2|}{N_{epochs}}\right) \tag{3.10}$$

$$w_i = \frac{\widehat{w}_i}{||\widehat{\mathbf{w}}||} \tag{3.11}$$

3.2.2 Learning the VF from the cosine similarity

To validate our proposition that the surface representation can be deduced through the cosine similarity of the VF for adjacent points, we initially formulate an experiment before delving into volume rendering from multi-view images. Given a scene representation, such as a mesh or a point cloud, our goal is to learn the VF representation by leveraging the cosine similarity.

Given a set of 3D points in space, we aim to minimize the cosine similarity of the VF between pairs of points on opposing sides of the surface. Conversely, we seek to maximize the cosine similarity for pairs of points located on the same side of the surface. Consequently, we introduce the subsequent cosine loss:

$$\mathcal{L}_{cos} = -\frac{1}{|\mathcal{S}|}\sum_{\{\mathbf{x}_1,\mathbf{x}_2\}\in\mathcal{S}} \cos\left(\mathbf{f}(\mathbf{x}_1), \mathbf{f}(\mathbf{x}_2)\right) + \frac{1}{|\mathcal{F}|}\sum_{\{\mathbf{x}_1,\mathbf{x}_2\}\in\mathcal{F}} \cos\left(\mathbf{f}(\mathbf{x}_1), \mathbf{f}(\mathbf{x}_2)\right) \tag{3.12}$$

where $\mathcal{S}$ is a set of pairs of points on opposite sides of the surface and $\mathcal{F}$ is a set of pairs of points on the same side of the surface. Note that it is important that the pairs are close enough so that when they are on opposite sides of the surface their ground truth VF cosine similarity is close to -1.

Even though the cosine similarity loss enforces the direction of the VF, the unit norm is not enforced with the aforementioned loss. Hence, we introduce a loss on the norm of the VF as follows:

$$\mathcal{L}_{norm} = \frac{1}{2(|\mathcal{S}| + |\mathcal{F}|)}\sum_{\{\mathbf{x}_1,\mathbf{x}_1\}\in\mathcal{S}\bigcup\mathcal{F}} (((||\mathbf{f}(\mathbf{x}_1)||_2 - 1)^2 + (||\mathbf{f}(\mathbf{x}_2)||_2 - 1)^2) \tag{3.13}$$

Additionally, we empirically find that it is important to introduce a loss to supervise the VF of far-away exterior points. To this end, given a set of exterior points $\mathcal{D} \subset \mathbb{R}^3$ and their respective ground truth VF

$\mathcal{V} \subseteq \Gamma$, we define the exterior VF loss as follows:

$$\mathcal{L}_{VF} = \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x} \in \mathcal{D}} ||\mathbf{f}(\mathbf{x}) - \mathbf{v}||_2^2 \tag{3.14}$$

where $\mathbf{v} \in \mathcal{V}$ is the ground truth VF.

The results of the experiment are depicted in Section 4.2.1

3.3 Density as Transformed VF

In neural volume rendering, the volume density is a function $\sigma : \mathbb{R}^3 \to \mathbb{R}_{\geq 0}$ that maps a point in space, $\mathbf{x}$, to the rate that which light is occluded at that $\mathbf{x}$. Previous works such as [32] model the density function with a simple MLP, while more recent methods [48, 53] propose to define the surface density as a transformation of the neural SDF. Drawing inspiration from the former methods, we propose to model the surface density as a function of the learnable VF. As mentioned in Section 3.2.2, we can examine if a point in space belongs to an object surface using the cosine similarity. Therefore, using sliding window cosine similarity introduced in Section 3.2.1, we redefine the surface density as a transformation that maps a point in the ray $\mathbf{r} \in \mathbb{R}^{(N+1)\times 3}$ to a scalar value, $\sigma : \mathbb{R}^{(N+1)\times 3} \times \mathbb{N}_{\geq 0} \to \mathbb{R}_{\geq 0}$. Leveraging the cosine similarity, we define the surface density as follows:

$$\sigma(\mathbf{r}, i) = \mathrm{ReLU}(\alpha \Psi_{\mu,\beta}(-c^i_{sim}(\mathbf{r})) - \alpha \Psi_{\mu,\beta}(\xi)) \tag{3.15}$$

where $\alpha, \mu, \beta > 0$ are learnable parameters and ξ 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. μ denotes the Laplacian mean, while β is Laplacian diversity and α is a scaling factor. Formally, the Laplacian CDF is defined as follows:

$$\Psi_{\mu,\beta}(x) = 0.5 + 0.5\mathrm{sgn}(x - \mu)\left(1 - \exp\left(-\frac{|x - \mu|}{\beta}\right)\right) \tag{3.16}$$

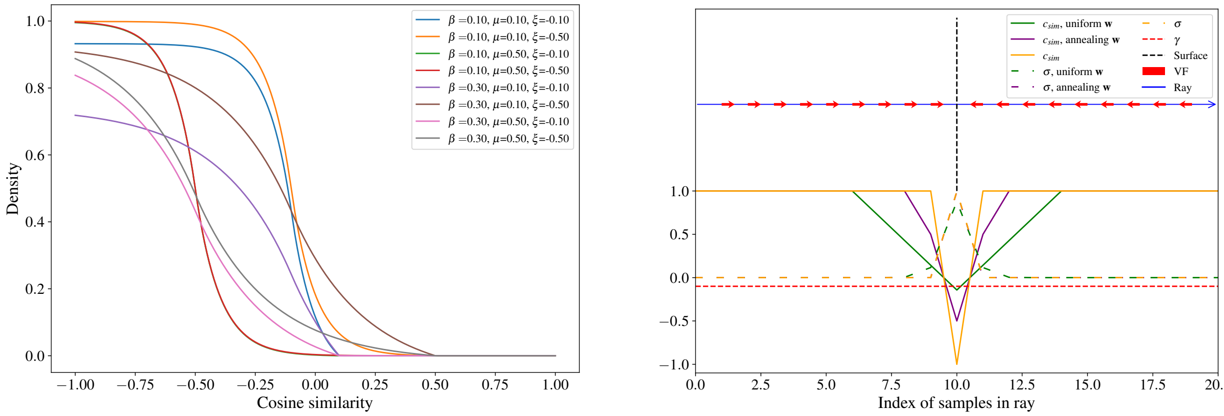


Figure 3.3: **Density function.** Left: Density function with different parameters. The maximum is always reached at -1. Right: Behavior of the window cosine similarity and the density during different stages of the annealing when crossing a surface. The figure is presented in 2D for simplicity. Additionally, we assume that the ray is perpendicular to the surface.

Note that σ is low when the cosine similarity is far from μ, which serves as a threshold to decide if there is a surface at the given point of space. On the other hand, when the cosine similarity is equal to or lower than μ, the density function increases. Additionally, the second Laplacian CDF is added to set the density to zero when the cosine similarity is larger than ξ. The behavior of the density function with respect to different parameter values, as well as an example of how the density changes when approaching and crossing a surface are depicted in Figure 3.3.

Intuitively, the goal is to train the VF network through volume rendering with multi-view images with the help of the density function. Ideally, after training, the density function yields high values at surface points, which are expected to have a cosine similarity smaller or equal to μ.

3.4 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 $\mathbf{z} \in \mathbb{R}^{256}$, where ϕ represents the network learnable parameters. Second, $\mathbf{c}_\psi : \mathbb{R}^{3+3+3+256} \rightarrow \mathbb{R}^3$ approximates the radiance field based on a given spatial point, viewing direction, VF, and global feature vector, where ψ 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 $\mathbf{c} = \mathbf{c}_\psi(\mathbf{x}, \mathbf{v}, \mathbf{d}, \mathbf{z})$. Additionally, our model incorporates three adjustable parameters for the density function as described in Equation (3.15), namely α, μ and β.

Positional encodings [32] 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 enhances the training process.

During the optimization process, 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 $C(\mathbf{p})$:

$$\mathcal{L}_c = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p} \in \mathcal{P}} ||\widehat{C}(\mathbf{p}) - C(\mathbf{p})||_1 \tag{3.17}$$

In addition to the color loss, we ensure that the VF has a unit vector property by applying the unit norm loss $\mathcal{L}_{norm}$ defined in Equation (3.13). Additionally, in object-centric scenes, the VF at distant points relative to the object usually resembles a vector directed toward the object. To simplify the training, 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 $\mathbf{c}_{scene}$.

$$\mathcal{L}_{ext} = \frac{1}{|\mathcal{P}_{ext}|} \sum_{\mathbf{x} \in \mathcal{P}_{ext}} \left|\left| \mathbf{f}(\mathbf{x}) - \frac{\mathbf{c}_{scene} - \mathbf{x}}{||\mathbf{c}_{scene} - \mathbf{x}||_2} \right|\right|_2 \tag{3.18}$$

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

$$\mathcal{L} = w_c \mathcal{L}_c + w_{norm} \mathcal{L}_{norm} + w_{ext} \mathcal{L}_{ext} \tag{3.19}$$

where w_c, w_{norm} and w_{ext} are hyperparameters for weighting the individual losses.

3.5 Indoor Scenes Training

Learning the geometry of indoor scenes solely from images presents a challenge in reconstructing accurate geometries, even in textured regions. To address this, we propose enhancing the learning of scene representation by introducing a depth consistency loss that compares the rendered depth, $\widehat{D}(\mathbf{p})$, with depth maps derived from multi-view stereo methods [40, 41, 57], symbolized as $D(\mathbf{p})$.

$$\mathcal{L}_{depth} = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p} \in \mathcal{P}} ||\widehat{D}(\mathbf{p}) - D(\mathbf{p})||_1 \tag{3.20}$$

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

$$\mathcal{L}_{center} = \frac{1}{|\mathcal{P}_{center}|} \sum_{\mathbf{x} \in \mathcal{P}_{center}} \left|\left| \mathbf{f}(\mathbf{x}) - \frac{\mathbf{x} - \mathbf{c}_{scene}}{||\mathbf{x} - \mathbf{c}_{scene}||_2} \right|\right|_2 \tag{3.21}$$

Hence, in indoor scenes, the overall loss function is defined as the following weighted sum:

$$\mathcal{L}_{indoor} = \mathcal{L} + w_{depth}\mathcal{L}_{depth} + w_{center}\mathcal{L}_{center} \tag{3.22}$$

Additionally, to improve the VF representation following the optimization of the loss detailed in Equation (3.22), we incorporate a self-supervision and refinement strategy post-initial optimization. In the self-supervision phase, we utilize the learning approach outlined in Section 3.2.2, where the desired scene representation is the VF from the initial training. In addition to the losses outlined in Section 3.2.2, we also integrate the center loss as presented in Equation (3.21). Subsequently, we further refine the scene representation, employing the indoor scene loss at a reduced learning rate.

Chapter 4

Experiments and Results

In this section, we first introduce the implementation details, datasets and metrics used in this work in Section 4.1. We then present a detailed analysis of qualitative and quantitative comparisons of our method against competitive baselines in Section 4.2 and Section 4.3. Finally, we showcase a set of ablation studies in Section 4.4.

4.1 Experimental setup

4.1.1 Implementation details

Our method is developed using PyTorch [37]. The VF and color functions are designed as MLPs consisting of 8 and 4 hidden layers, respectively. Alongside the use of positional encoding and the initialization method specified in Section 3.4, the Adam optimizer [22] is employed. The learning rate is initialized at 5×10^{-4} and is decreased using an exponential decay approach [25]. The training process spans 3300 epochs: 1900 for initial training, 500 for self-supervision, and the final 900 for refining the model. 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 a specific version of the Marching Cubes algorithm, presented in [39], to extract the surface mesh from the predicted VF with a resolution of 512.

We set the following weights of the multiobjective loss function: $w_c = 1.0$, $w_{norm} = 0.1$, $w_{ext} = w_{center} = 0.5$, $w_{depth} = 0.25$. 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 $\alpha = 100$.

4.1.2 Datasets, metrics and baselines

Datasets. For object-centric scenes, we perform experiments on the DTU and BlendedMVS datasets [17, 51], which contain real objects captured from multiple views. For indoor scenes, we test the performance of our algorithm on Replica [44] 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. The ScanNet dataset contains 16113 indoor scenes with 2.5 million views, with each view containing RGB-D images. Additionally, a Truncated Signed Distance Function (TSDF) integrated mesh is provided for each scene. For replica, we sample 1 of every 20 posed images for training, while in Scannet we sample 1 every 40.
Metrics. For 3D surface reconstruction, we focus on evaluating our method with Chamfer distance and F1-score [45]. Additionally, we also provide the peak signal-to-noise ratio (PSNR) to evaluate image synthesis. The detailed definitions of these metrics are included in the supplementary material.

Baselines. We compare our algorithm against state-of-the-art volume rendering based methods for indoor scenes: Manhattan-SDF [14], MonoSDF [56], NeuRIS [47] and Neuralangelo [24]. We use Marching Cubes [30] to extract the meshes rendered by the baselines. Additionally, similarly to [14], we render depth maps from the predicted mesh and re-fuse them using TSDF fusion to remove artifacts and possible arbitrary surfaces generated at unseen points in space.

4.2 Object centric scenes

In this section, we first present preliminary experiments that demonstrate the geometric representational power of the cosine similarity between VF adjacent points in Section 4.2.1. Consecutively, we showcase our method's capacity to reconstruct object-centric scenes from multi-view images in Section 4.2.2.

4.2.1 Non-explicit Vector Field learning

We conduct experiments on the DTU dataset to learn the VF from meshes generated with VolSDF. Given a mesh, we sample points on the surface and randomly from the surrounding area of the object. We then apply the losses outlined in Section 3.2.2. The qualitative outcomes of the mesh reconstruction, achieved through the adapted marching cubes technique alongside the VolSDF-generated meshes, are displayed in Figure 4.1. Our method introduces regularization and smoothing since the high-resolution details cannot be captured by the reconstructed mesh. The results suggest that the geometry representation we proposed, based on the cosine similarity of the VF, is apt for representing 3D scenes, even with the inherent smoothing.

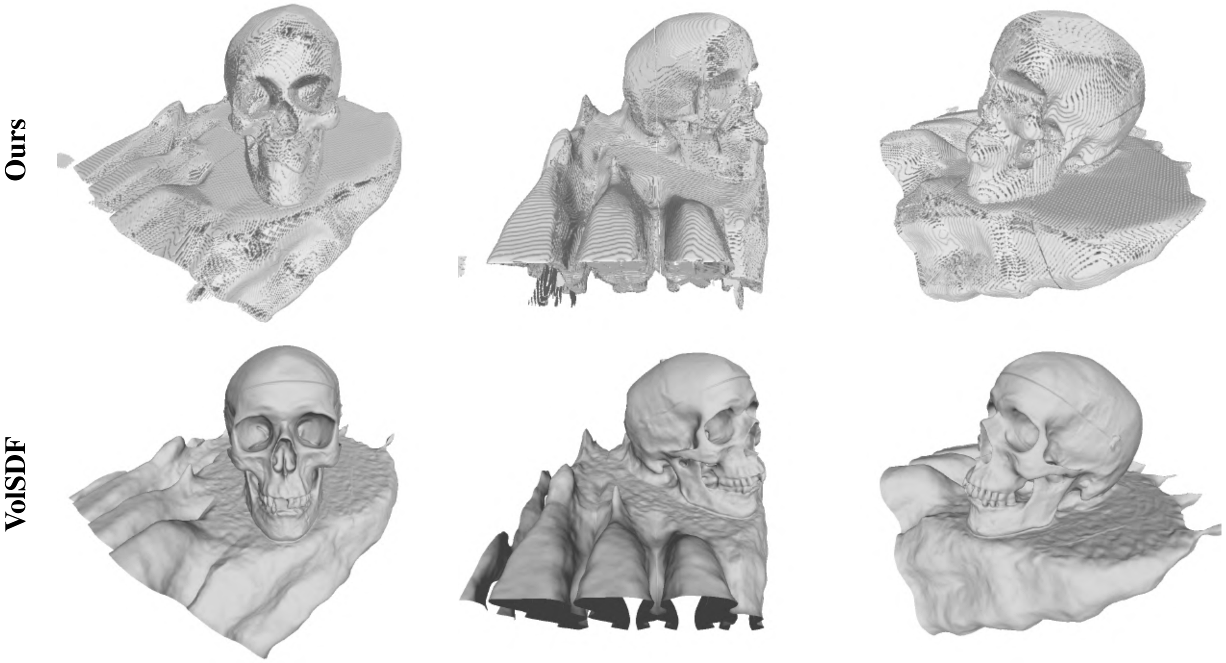


Figure 4.1: **Non-explicit VF learning qualitative results.** The top meshes represent the reconstruction using the learned VF. The bottom meshes are obtained using VolSDF and are used as the target geometry for the VF representation. The VF can generally represent 3D scenes, although it introduces smoothing and struggles to reconstruct high-frequency details

We analyze the representational capacity of an MLP to model the VF of a scene, using cosine similarity as the learning mechanism. Figure 4.2 presents the learned VF of the scene projected onto three different

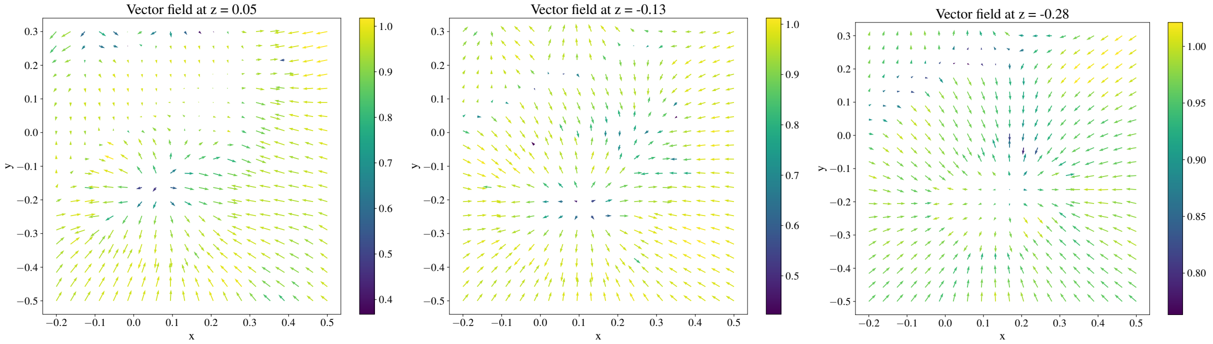


Figure 4.2: **Non-explicitly learned projected Vector Field.** Predicted VF projected into several planes of the z-axis. The color of the vectors represents the magnitude of its norm. The VF changes direction at surface points as expected.

z-axis planes. As depicted in the figures, the VF directs to the nearest surface, inverting its direction upon reaching surface points.

4.2.2 Multi-view 3D reconstruction

Prior to exploring the primary utility of our approach—its effectiveness in representing indoor scenes—we initially assess its performance in object-centric scenarios. Adhering to the optimization procedure outlined in Section 3.4, we learn the object geometry exclusively from a set of posed images. Our technique is evaluated on two distinct scenes, one from the DTU dataset and the other from the BlendedMVS dataset.

Overall, our method is capable of representing the geometry of the scene as depicted in Figure 4.3. Nevertheless, it lacks the high-level details in the scenes and tends to generate numerous arbitrary surfaces unrelated to the actual object geometry. A potential reason for these discrepancies may be the uneven distribution of posed images across the object's viewpoints. Particularly for the DTU scene, the majority of camera positions are concentrated in a specific area. Consequently, our technique struggles to accurately capture the true VF in areas not covered by the camera, resulting in the presence of artifacts and surfaces that blend with the object's true geometry. This phenomenon can be observed in the VF visualization of Figure 4.4, since we have vectors with opposite directions in regions of free space.

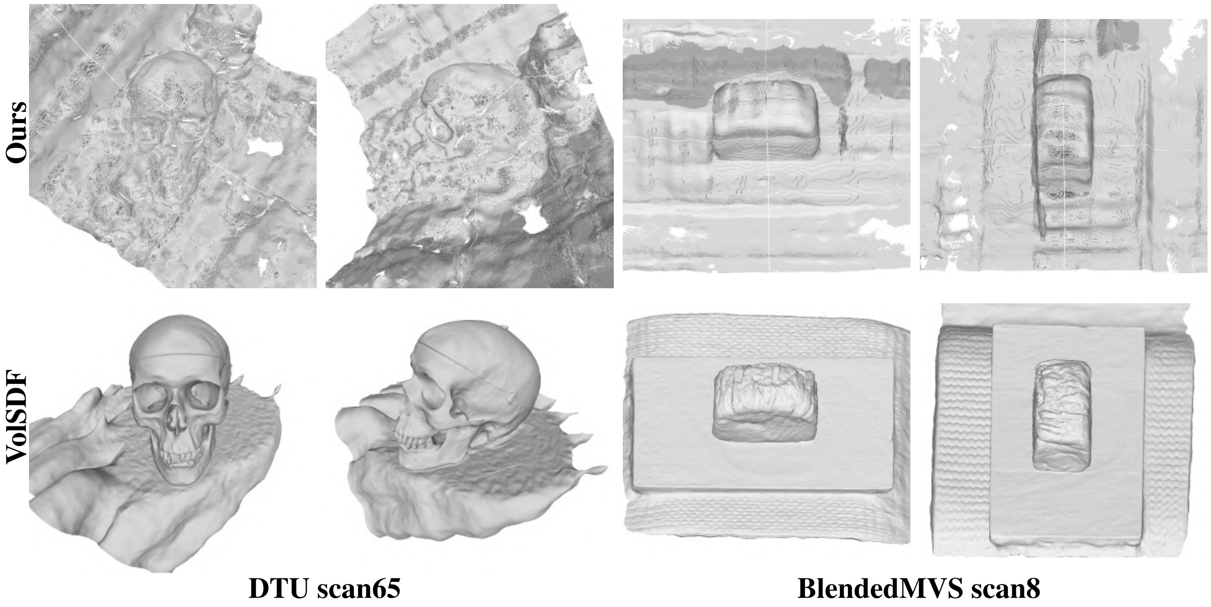


Figure 4.3: **Object-centric qualitative results.** Our method can generally recover the scene geometry, although it introduces smoothness and some artifacts.

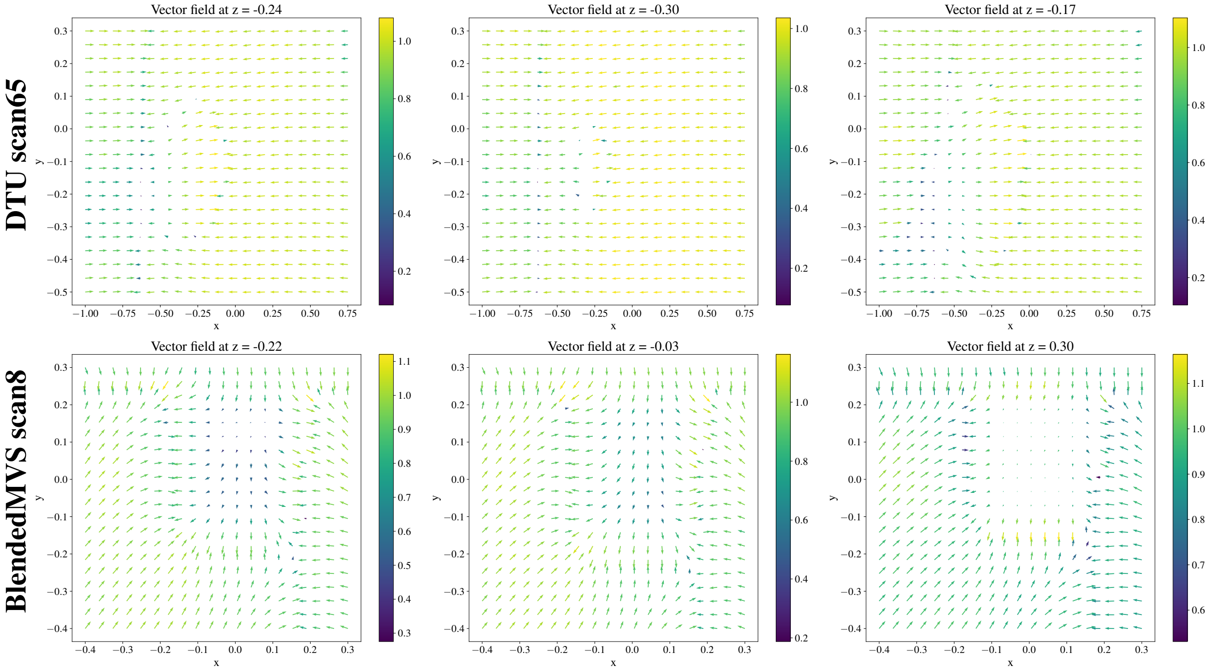


Figure 4.4: **Projected Vector Field of object-centric scenes.** Predicted VF projected into several planes of the z-axis. The color of the vectors represents the magnitude of their norm. The predicted VF generally changes direction at surface points, although it also presents this phenomenon in some regions of free space.

4.3 Indoor scenes

We evaluate our algorithm against state-of-the-art indoor scene reconstruction methods. We present qualitative and quantitative evaluations of 3D reconstruction in Section 4.3.1 and of novel view synthesis in Section 4.3.2.

4.3.1 3D reconstruction

We evaluate our method with the Replica and ScanNet datasets. The qualitative results on Replica and ScanNet are shown in Figure 4.5 and Figure 4.6, respectively. Quantitative results on both datasets are depicted in Table A.2. Additional detailed qualitative and quantitative results are included in the supplementary material. Our method generally outperforms volume rendering based benchmarks, except for MonoSDF, in terms of F-score and Chamfer distance in the Replica dataset. Additionally, we show competitiveness with the benchmarks in ScanNet in terms of F-score.

Note that MonoSDF makes use of multiresolution hash-encodings, as well as normal priors, thus achieving accurate results in 3D reconstruction. The performance of Neuralangelo in indoor scenes since it does not make use of depth information throughout the optimization. Manhattan-SDF 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, although it can struggle with high-frequency details. The capacity of our method to represent planar surfaces compared to the other methods is depicted in Figure 4.7.

	Replica								ScanNet				
	r-0	r-1	r-2	o-0	o-1	o-3	o-4	Mean	0050	0084	0580	0616	Mean
	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)	**0.919**	0.888	**0.808**	**0.921**	**0.838**	0.761	0.596	**0.819**	0.726	0.746	**0.800**	**0.687**	**0.740**
TSDF (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
	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)	**4.39**	**3.99**	**233**	**6.17**	**27.9**	471	**13.5**	**108.6**	60.8	72.5	18.2	90.6	60.5
TSDF (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

Table 4.1: **3D reconstruction quantitative results.** Our method outperforms Manhattan-SDF and Neuralangelo in most of Replica scenes. Additionally, it shows competitiveness in ScanNet in terms of F-score. Note that Neuralangelo fails to reconstruct the scene surface in scene 0084 of ScanNet. **Best result**. **Second best result**. Note that TSDF is only added as reference but not taken into account when ranking the methods.

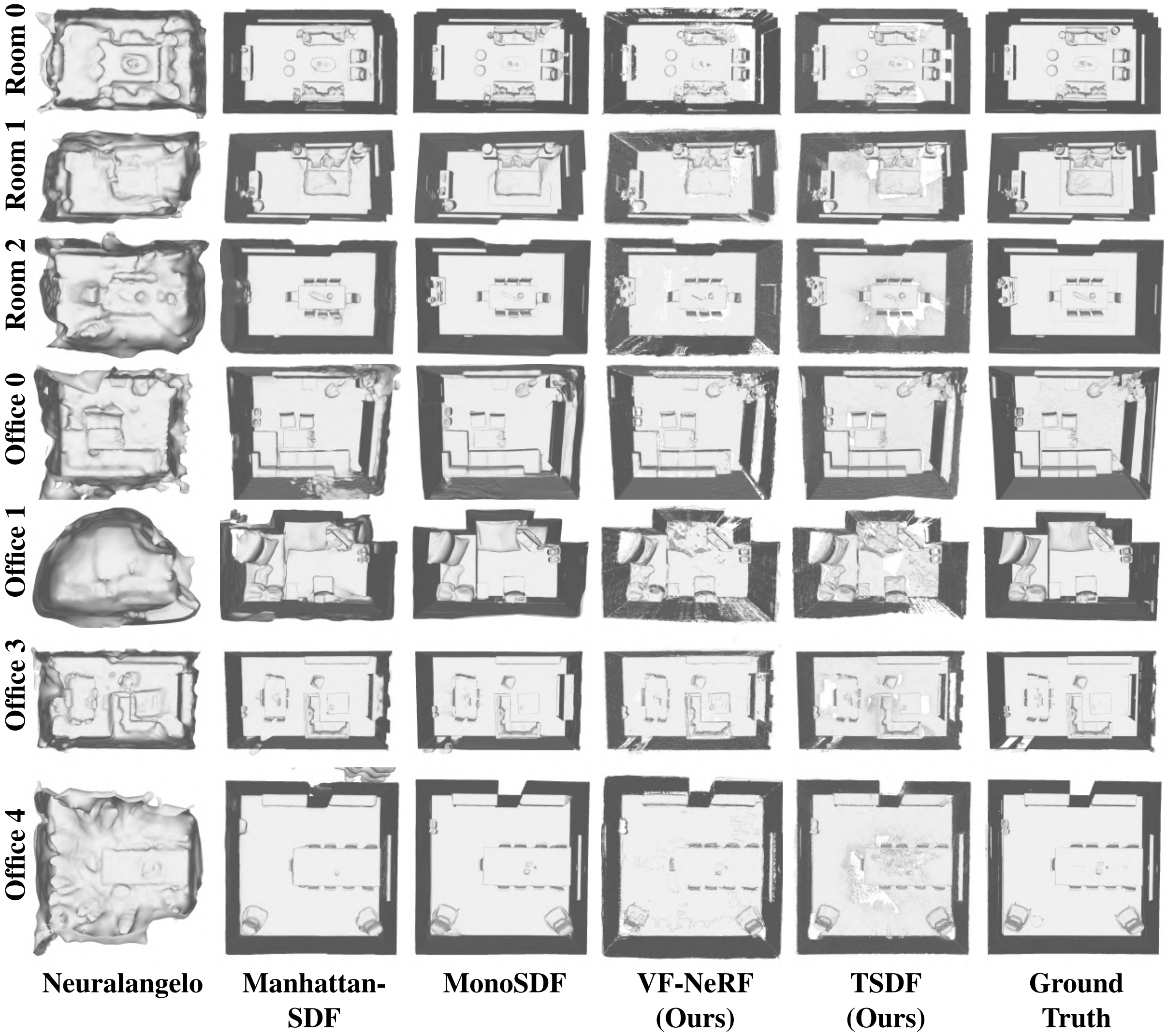


Figure 4.5: **3D reconstruction qualitative results on Replica.** Our method outperforms Neuralangelo and Manhattan-SDF in many scenes. Additionally, our method is more accurate on walls and floors.

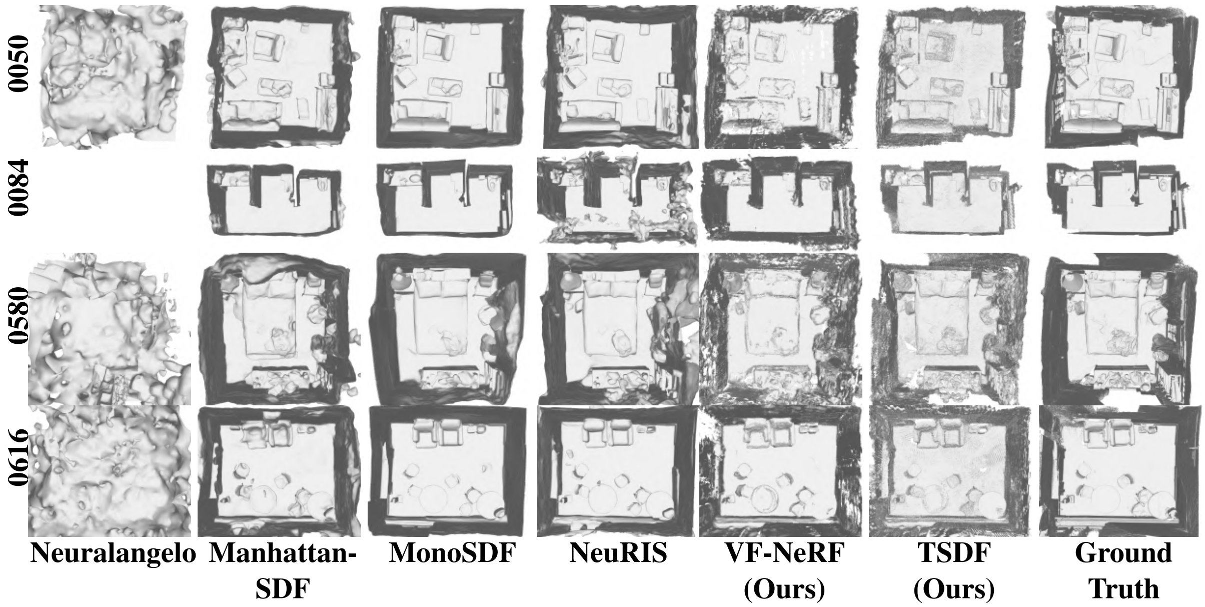


Figure 4.6: **3D reconstruction qualitative results on Scannet.** Our method outperforms Neuralangelo, Manhattan-SDF and NeuRIS in planar regions of the scenes such as walls and floors.

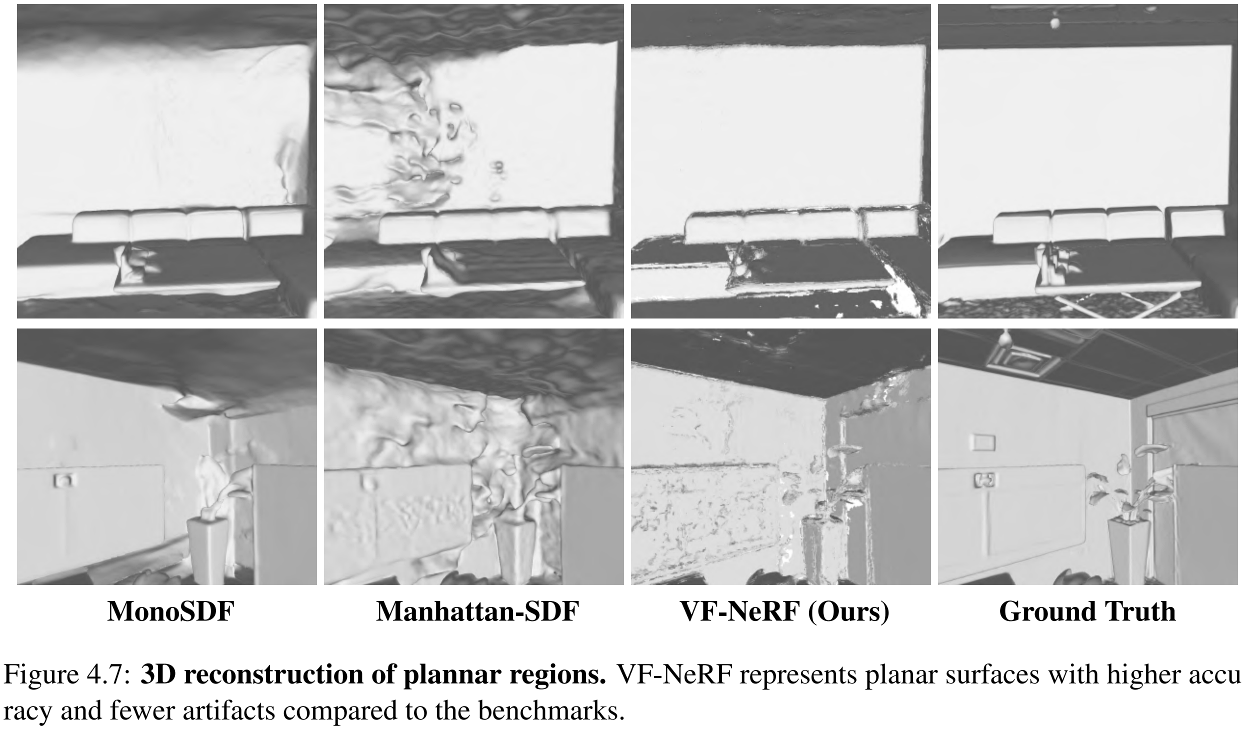


Figure 4.7: **3D reconstruction of plannar regions.** VF-NeRF represents planar surfaces with higher accuracy and fewer artifacts compared to the benchmarks.

4.3.2 Novel view synthesis

We present qualitative and quantitative novel view synthesis comparisons in Figure 4.8 and Table 4.2. Our method generally renders high-quality views and outperforms Manhattan-SDF in terms of PSNR. However, NeuRIS and multiresolution hash-encoding methods—such as Neuralangelo and MonoSDF—present outstanding results in Replica. However, our method demonstrates higher quality results compared to Manhattan-SDF, Neuralangelo and MonoSDF in Replica, a more challenging and realistic dataset. More qualitative results for each scene can be found in the supplementary material.

	PSNR↑												
	Replica								ScanNet				
	r-0	r-1	r-2	o-0	o-1	o-3	o-4	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	-	-	-	-	-	-	-	-	**28.06**	**32.41**	**28.33**	**27.91**	**29.18**
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 4.2: **Novel view synthesis quantitative results.** In general, VF-NeRF renders higher-quality images compared to Manhattan-SDF. **Best result**. **Second best result**.

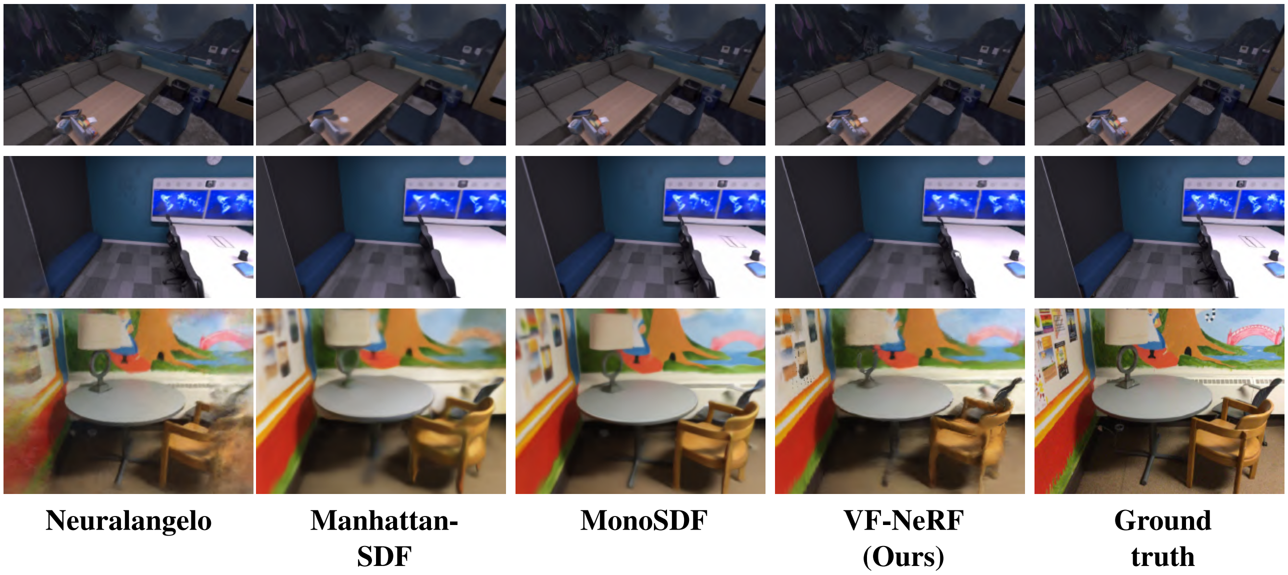


Figure 4.8: **Novel view synthesis qualitative results.** Our method can render accurate images with high-frequency details. Compared to Manhattan-SDF, our method is more accurate and introduces less smoothness in both datasets.

4.3.3 Limitations

Figure 4.9 presents the limitations of our method to reconstruct high-frequency details of scenes. As showcased, VF-NeRF introduces smoothness and therefore struggles in high-detailed areas. Additionally, as presented inSection 4.2.2, if the camera views do not homogeneously cover the scene, arbitrary artifacts appear in the VF representation.

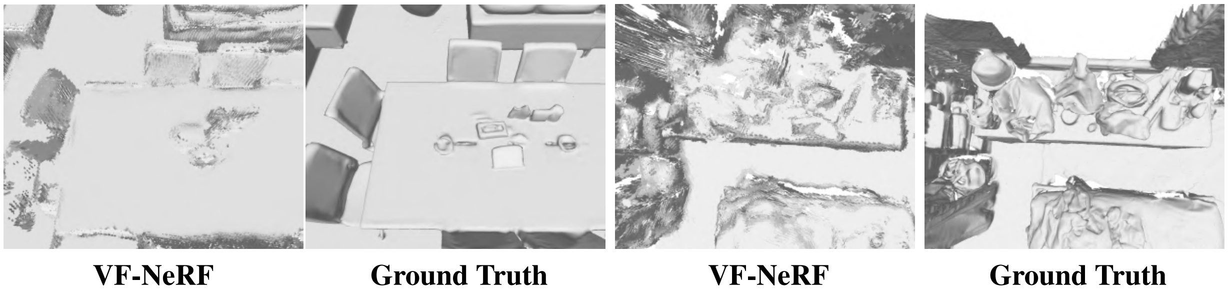


Figure 4.9: **Limitations.** VF-NeRF struggles to reconstruct high-frequency details due to its inherent smoothing.

4.4 Ablation studies

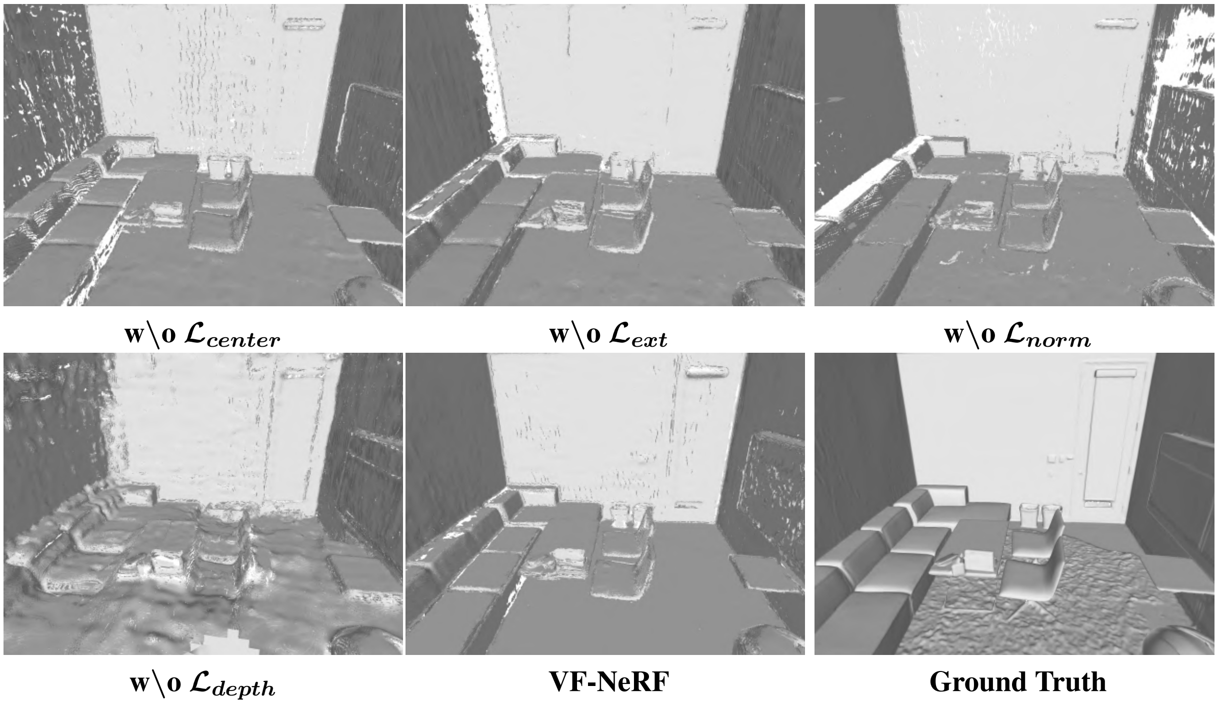
Loss terms. We analyze the impact of different loss terms on the surface representation and provide quantitative results. Specifically, we remove the following losses and study their effects: center supervision $\mathcal{L}_{center}$, exterior supervision $\mathcal{L}_{ext}$, unit norm $\mathcal{L}_{norm}$ and depth $\mathcal{L}_{depth}$. We demonstrate that removing these losses decreases the performance of our method, as presented in Figure 4.10 and Table 4.3. As shown, some surfaces disappear and holes are present in the scene. Additionally, removing the depth loss significantly decreases the performance since most of the surfaces of the scene do not have enough texture.

Sliding window annealing and initialization. Figure 4.11 and Table 4.3 show the results of ablating the sliding window cosine similarity and the custom VF network initialization. In the case of the annealing, we use normal cosine similarity with the next point of the ray instead of the weighted average to investigate the importance of the sliding window. The results show that both elements enhance the surface reconstruction of our method both qualitatively and quantitatively.

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 Table 4.3. As expected, we find that using more points per ray enhances the performance of our method, although it comes with an increase in training and inference time.

	Precision ↑	Recall ↑	**F-score**↑	**CD (mm)**↓
w\o $\mathcal{L}_{center}$	0.877	0.929	0.902	16.1
w\o $\mathcal{L}_{ext}$	0.533	0.915	0.674	168
w\o $\mathcal{L}_{norm}$	0.517	0.929	0.664	164
w\o $\mathcal{L}_{depth}$	0.292	0.453	0.355	344
w\o annealing	0.530	0.851	0.653	182
w\o initialization	0.735	0.898	0.808	83.9
100 points per ray	0.759	0.929	0.835	4.29
150 points per ray	0.847	0.929	0.886	6.79
VF-NeRF	**0.910**	**0.932**	**0.921**	**6.18**

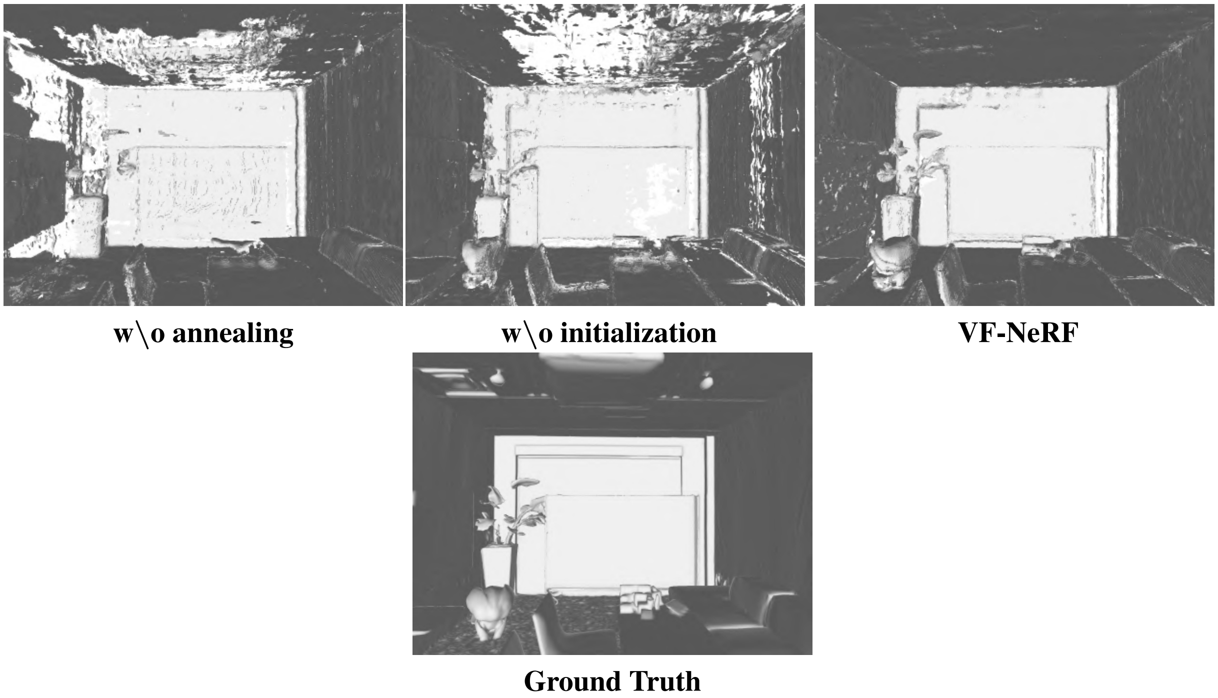
Table 4.3: **Ablations quantitative results.** Removing loss terms, sliding window weighs annealing or the VF network initialization decreases our method's performance.



w\o $\mathcal{L}_{center}$ **w\o $\mathcal{L}_{ext}$** **w\o $\mathcal{L}_{norm}$**

w\o $\mathcal{L}_{depth}$ **VF-NeRF** **Ground Truth**

Figure 4.10: **Losses ablations.** Removing $\mathcal{L}_{center}$, $\mathcal{L}_{ext}$ and $\mathcal{L}_{norm}$ generates holes in surfaces, mostly in planar regions. Our method without $\mathcal{L}_{depth}$ is less accurate, as most regions of the scene are low-textured.



w\o annealing **w\o initialization** **VF-NeRF**

Ground Truth

Figure 4.11: **Annealing and initialization ablations.** Our method without annealing or initialization generates more holes and artifacts, mostly in the ceiling and walls.

Chapter 5

Conclusion

In this work, we presented VF-NeRF, a novel neural approach for multiview surface reconstruction, utilizing Vector Fields (VFs) to encapsulate the scene's geometry. VFs consist of unit vectors directed towards the nearest learned surface point. 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 tri-phase optimization strategy, encompassing initial VF training through volume rendering, self-supervised learning, and a refinement stage, to enhance the accuracy of the learned VF. The experiments demonstrate the performance of our method to reconstruct indoor scenes, outperforming several state-of-the-art methods on indoor datasets.

One limitation of our method is its inherent smoothing, which makes it hard to represent high-frequency details. Moreover, a significant dependency of our technique is the necessity for consistent scene views during the reconstruction to prevent distortions, particularly in regions not observed during the learning phase. 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. Finally, recent state-of-the-art 3D reconstruction methods have leveraged multiresolution hash-encodings to increase the representational power of their models. Extending this concept to the VF representation in scenes may offer substantial improvements in the fidelity of the reconstructed surface.

Appendix A

Supplementary material

A.1 Detailed networks' architecture

We present an illustration of the VF and color networks in Figure A.1

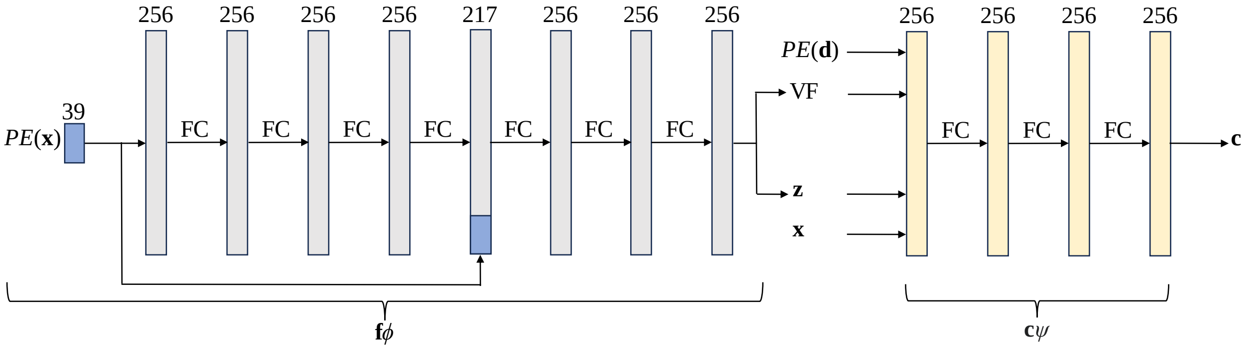


Figure A.1: **Networks' architecture.** The VF network takes a point in space $\mathbf{x}$ as input and applies positional encoding ($PE(\cdot)$) before feeding it to the MLP. The color network takes the spatial point, the predicted VF, the feature vector $\mathbf{z}$, and the viewing direction $\mathbf{d}$ with positional encoding as inputs to predict the color.

A.2 Metrics

The definitions of 3D reconstruction metrics are depicted in Table A.1.

A.3 Baselines

We use the official Manhattan-SDF [1], MonoSDF [2] and NeuRIS [3] implementations as baselines. We adapted the Replica dataset to use it in Manhattan-SDF, unfortunately, the NeuRIS does not support it. Additionally, we use SDFStudio's [55] implementation [4] of Neuralangelo.

[1] https://github.com/zju3dv/manhattan_sdf

[2] https://github.com/autonomousvision/monosdf

[3] https://github.com/jiepengwang/NeuRIS

[4] https://github.com/autonomousvision/sdfstudio

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	$\frac{2\cdot\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}}$
Chamfer Distance	$\sum_{p\in P}\min_{p^*\in P^*}\|\|p^*-p\|\|_2^2+\sum_{p^*\in P^*}\min_{p\in P}\|\|p^*-p\|\|_2^2$
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 A.1: **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.

A.4 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 A.2.

A.5 3D reconstruction qualitative results

We provide qualitative results for each scene of Replica and ScanNet in Figure A.2 and Figure A.3. We include visualizations of state-of-the-art 3D reconstruction methods as comparisons.

A.6 Novel view synthesis qualitative results

Figure A.4 presents qualitative comparisons of novel view synthesis for each scene on Replica and Scannet.

	Replica								ScanNet				
	r-0	r-1	r-2	o-0	o-1	o-3	o-4	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)	**0.938**	0.862	0.720	**0.910**	**0.777**	0.669	0.506	**0.769**	0.781	0.867	**0.886**	**0.893**	**0.857**
TSDF (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)	0.902	**0.915**	**0.919**	**0.932**	**0.910**	**0.882**	0.724	**0.883**	**0.686**	0.654	0.658	0.559	0.639
TSDF (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
	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)	**0.58**	0.71	1.06	**0.40**	**0.53**	1.29	3.39	**1.14**	1.81	2.71	**1.32**	2.73	2.14
TSDF (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 A.2: **3D reconstruction quantitative results of individual scenes on Replica and ScanNet.** **Best result**. **Second best result**. Note that TSDF is only added as reference but not taken into account when ranking the methods.

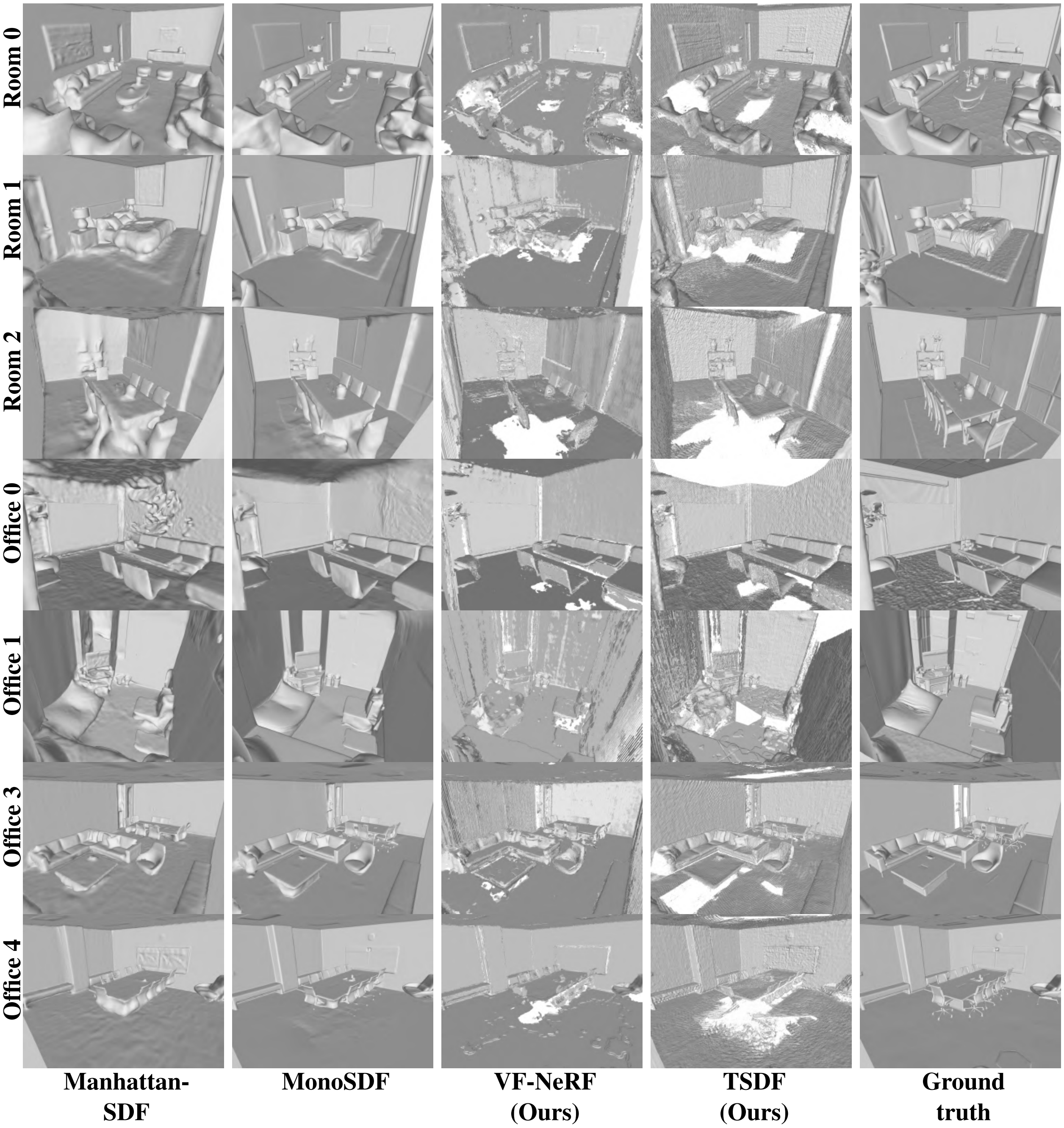


Figure A.2: **3D reconstruction qualitative results on Replica.**

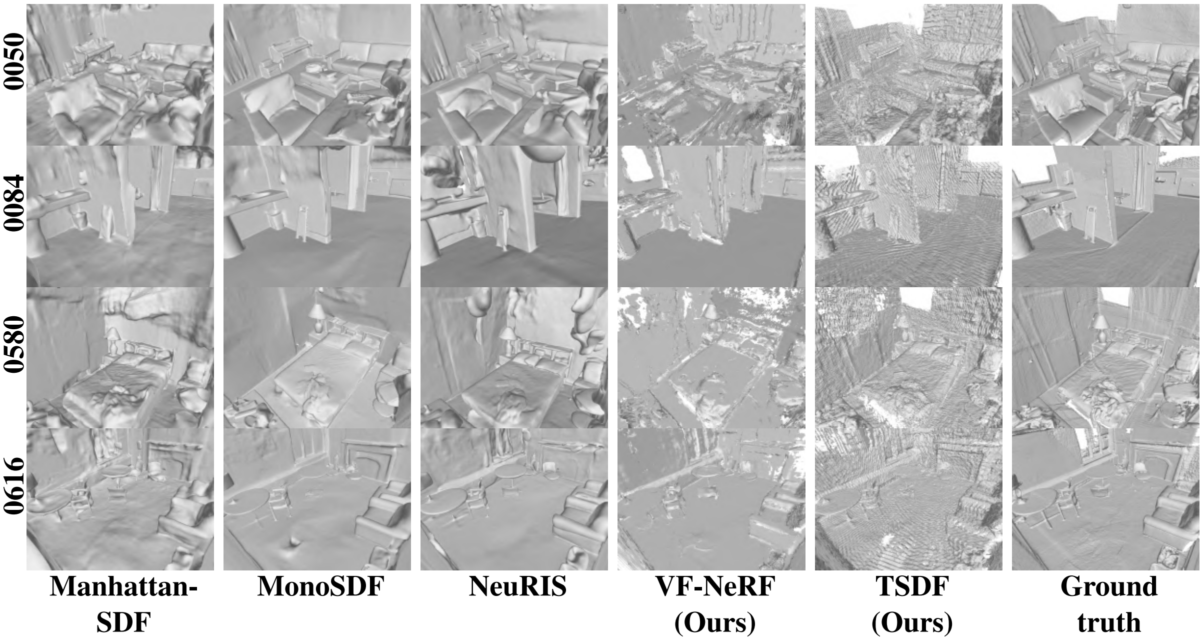


Figure A.3: **3D reconstruction qualitative results on ScanNet.**

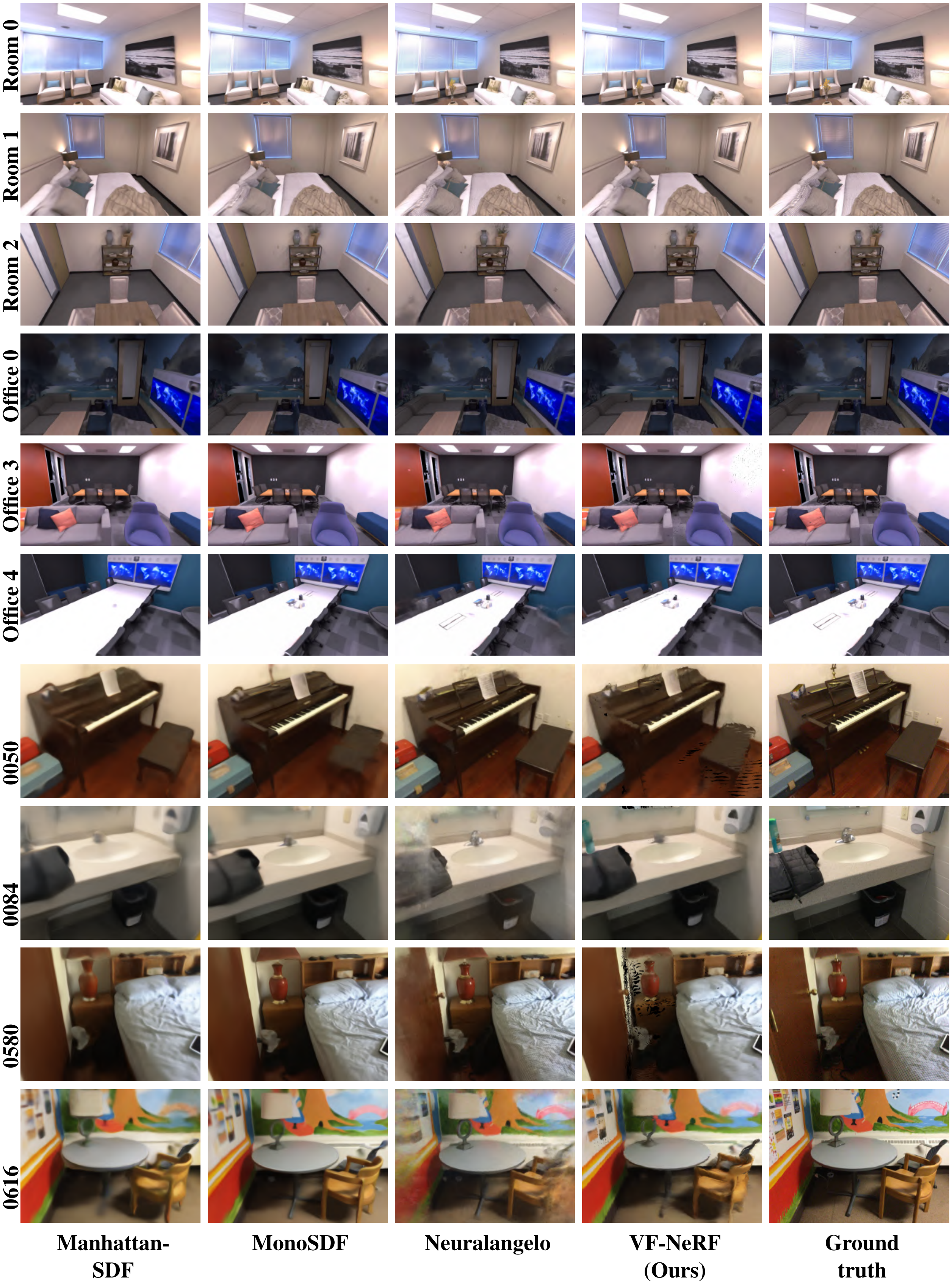


Figure A.4: **Novel view synthesis qualitative results.**

Bibliography

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

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

[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.

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

[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.

[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.

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

[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.

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

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

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

[13] 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.

[14] 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.

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

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

[17] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engil Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. *CVPR*, 2014.

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

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

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

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

[22] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

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

[24] 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.

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

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

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

[28] 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.

[29] Xiaoxiao Long, Cheng Lin, Lingjie Liu, Yuan Liu, Peng Wang, Christian Theobalt, Taku Komura, and Wenping Wang. Neuraludf: Learning unsigned distance fields for multi-view reconstruction of surfaces with arbitrary topologies. *CVPR*, 2023.

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

[31] 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.

[32] 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.

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

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

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

[36] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. *CVPR*, 2019.

[37] 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.

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

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

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

[41] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. *CVPR*, 2016.

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

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

[44] 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.

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

[46] 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.

[47] 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.

[48] 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.

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

[50] 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.

[51] Y. Yao, Z. Luo, S. Li, J. Zhang, Y. Ren, L. Zhou, T. Fang, and L. Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. *CVPR*, 2020.

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

[53] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *NeurIPS*, 2021.

[54] 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.

[55] 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.

[56] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. MonoSDF: Exploring monocular geometric cues for neural implicit surface reconstruction. 2022.

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