Depth-Guided Robust and Fast Point Cloud Fusion NeRF for Sparse Input Views

Shuai Guo[1], Qiuwen Wang[1], Yijie Gao[1], Rong Xie[1], Li Song[1,2*]

[1]Institute of Image Communication and Network Engineering, Shanghai Jiao Tong University, Shanghai, China
[2]Cooperative Medianet Innovation Center, Shanghai Jiao Tong University, Shanghai, China
{shuaiguo, wangqiuwen, gaoyijie, xierong, song_li}@sjtu.edu.cn

Abstract

Novel-view synthesis with sparse input views is important for real-world applications like AR/VR and autonomous driving. Recent methods have integrated depth information into NeRFs for sparse input synthesis, leveraging depth prior for geometric and spatial understanding. However, most existing works tend to overlook inaccuracies within depth maps and have low time efficiency. To address these issues, we propose a depth-guided robust and fast point cloud fusion NeRF for sparse inputs. We perceive radiance fields as an explicit voxel grid of features. A point cloud is constructed for each input view, characterized within the voxel grid using matrices and vectors. We accumulate the point cloud of each input view to construct the fused point cloud of the entire scene. Each voxel determines its density and appearance by referring to the point cloud of the entire scene. Through point cloud fusion and voxel grid fine-tuning, inaccuracies in depth values are refined or substituted by those from other views. Moreover, our method can achieve faster reconstruction and greater compactness through effective vector-matrix decomposition. Experimental results underline the superior performance and time efficiency of our approach compared to state-of-the-art baselines.

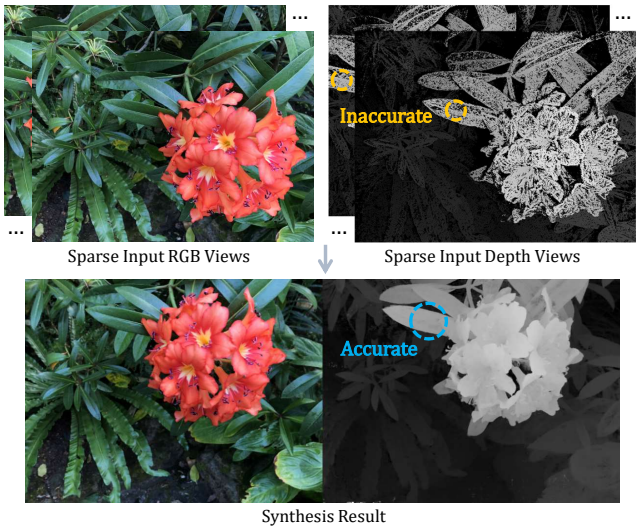

Figure 1: Depth-guided sparse input NeRF should overcome the effects of inaccurate depth values. This example illustrates a synthesis result of our method on the LLFF dataset.

Introduction

Novel-view synthesis (NVS) serves as a fundamental objective within the realm of computer vision. The recent surge in NVS popularity is largely attributable to the success of Neural Radiance Fields (NeRFs)(Mildenhall et al. 2021). However, NeRFs generally demand numerous images taken from a variety of views for efficient training. In real-world applications such as AR/VR and autonomous driving, where input views are typically sparse (Niemeyer et al. 2022), NeRF risks overfitting. This may lead to inconsistencies in reconstructions or failure in generating any useful solution.

Various strategies have significantly enhanced the performance of NeRF for sparse inputs by 1) optimizing training data utilization (Yu et al. 2021; Niemeyer et al. 2022; Truong et al. 2023), 2) incorporating prior information like depth and flow (Deng et al. 2022; Roessle et al. 2022), or 3) exploring new constraints and regularizations (Kim, Seo, and Han 2022; Yang, Pavone, and Wang 2023). Among these solutions, depth prior has garnered substantial interest due to its ease of access, its capacity to offer object positions, and its aid in handling occlusions and geometry understanding. Numerous methods have been proposed to integrate depth information into NeRFs for sparse input views.

Nevertheless, most existing depth-aware NeRFs for sparse input views disregard the holes, artifacts, and inaccurate values of depth maps. For example, DSNeRF (Deng et al. 2022) introduced depth supervision to leverage depth information, but it failed to account for inaccurate depth values. Certain practices might introduce additional unreliability. DDP-NeRF (Roessle et al. 2022), for instance, used a depth completion network to transform sparse depth into dense depth maps and uncertainty estimates, potentially leading to more inaccurate depth values. The issues inherent to the network itself could subsequently impact sparse view synthesis effects. SparseNeRF (Guangcong et al. 2023) employed the rough point cloud geometry provided by sparse RGB-D inputs to render more images and depict the approx-

*Corresponding author.

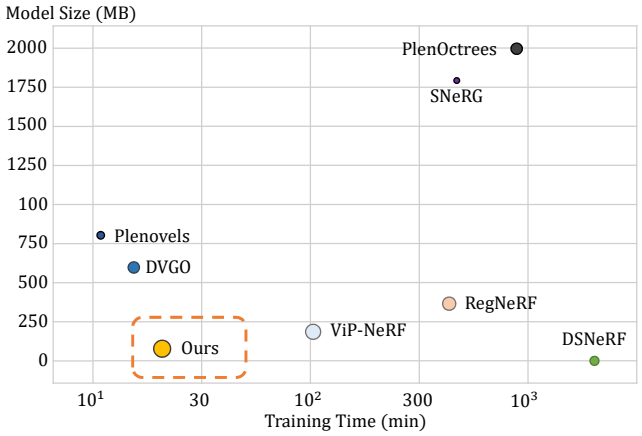

Figure 2: We compare our method with previous methods in terms of rendering quality (PSNR) and model size. Point sizes correspond to PNSRs. With effective vector-matrix decomposition and point cloud presentation, our work delivers superior rendering quality, faster reconstruction, and greater compactness.

imate scene appearance. However, the images rendered from sparse point clouds may be of low quality.

Moreover, few existing depth-aware NeRFs have used depth information to create faster NeRFs, resulting in overall low time efficiency. For instance, DSNeRF (Deng et al. 2022), despite its claims of improved speed, required several hours of training. On the other hand, Mip-NeRF (Dey, Ahmine, and Comport 2022), which employed depth supervision and depth-assisted local sampling, managed to train 3-5 times faster, but its training duration still approached an hour.

To address the challenges above, we introduce a depth-guided robust and fast point cloud fusion NeRF tailored for sparse input views. This is the first integration of point cloud fusion with NeRF volumetric rendering. In particular, inspired by TensoRF (Chen et al. 2022), we perceive radiance fields as an explicit voxel grid of features, delineated by a series of vectors and matrices that articulate scene appearance and geometry along their respective axes. The feature grid can be naturally seen as a 4D tensor, where three of its modes correspond to the XYZ axes of the grid, and the fourth mode represents the feature channel dimension. Utilizing sparse input RGB-D images and camera parameters, we map the 2D pixels of each input view to 3D space to generate a point cloud for each view. Subsequently, we convert depth values into densities, and encode both the depth and color information into the voxel grid utilizing two distinct sets of matrices and vectors. Volume density and view-dependent color can be decoded from the features, facilitating volumetric radiance field rendering. We aggregate the point cloud from each input view to assemble the fused point cloud of the entire scene. Each voxel determines its density and appearance within the scene by referencing this fused point cloud.

Since the planes and vectors are iteratively refined during the training process, and the point cloud of the entire scene is composed of all input views, inaccurate depth values are refined or replaced by depth values from other views. Figure 1 illustrates a synthesis example of our method. Additionally, as the vector-matrix decomposition technique effectively minimizes the number of components needed for the same expression capacity, our method can achieve faster reconstruction and greater compactness, as shown in Figure 2. This paper primarily contributes the following:

- We introduce the first depth-guided robust and fast point cloud fusion NeRF for sparse view input, minimizing the impact of inaccurate depth values.
- To our knowledge, this is the first NeRF framework that is integrated with point cloud fusion, offering a novel NeRF scene representation.
- Our method boosts time efficiency, and delivers superior results compared to state-of-the-art methods.

Related Work

In this section, we provide a comprehensive review of the relevant literature in the areas of novel-view synthesis and sparse input NeRF.

Novel-View Synthesis

The body of work about novel-view synthesis can generally be divided into two main categories: explicit representation based synthesis and implicit representation based synthesis.

Explicit Representations Explicit representation methods commonly employ point clouds (Ran, Liu, and Wang 2022; Huang et al. 2023), voxels (Sitzmann et al. 2019; Song, Jiang, and Yao 2022), meshes (Feng et al. 2019; Yang, Qiu, and Fu 2023), or MPI (Zhou et al. 2018; Kundu et al. 2020) to represent 3D geometry and appearance. Despite their computational efficiency, these techniques often pose optimization challenges due to their discontinuous nature.

Implicit Representations Implicit methods directly model the appearance of a 3D scene, thus eliminating the need for explicit geometric representation. A prime example of this approach is NeRF (Mildenhall et al. 2021). NeRF (Mildenhall et al. 2021) assigns a color and opacity to a given 3D location and 2D viewing direction, which correspond to the light emitted from that specific location in that particular direction. Owing to its inherent simplicity and superior rendering quality, NeRF has been widely adopted in recent studies for numerous extensions, including but not limited to, video synthesis (Li et al. 2022a,b), relighting (Yu et al. 2022; Rudnev et al. 2022), and scene editing (Yuan et al. 2022; Kobayashi, Matsumoto, and Sitzmann 2022).

Sparse Input NeRF

A number of works have been proposed to address the data-hungry problem of NeRF by exploiting training data, incorporating prior information like depth and flow, or introducing new constraints and regularizations.

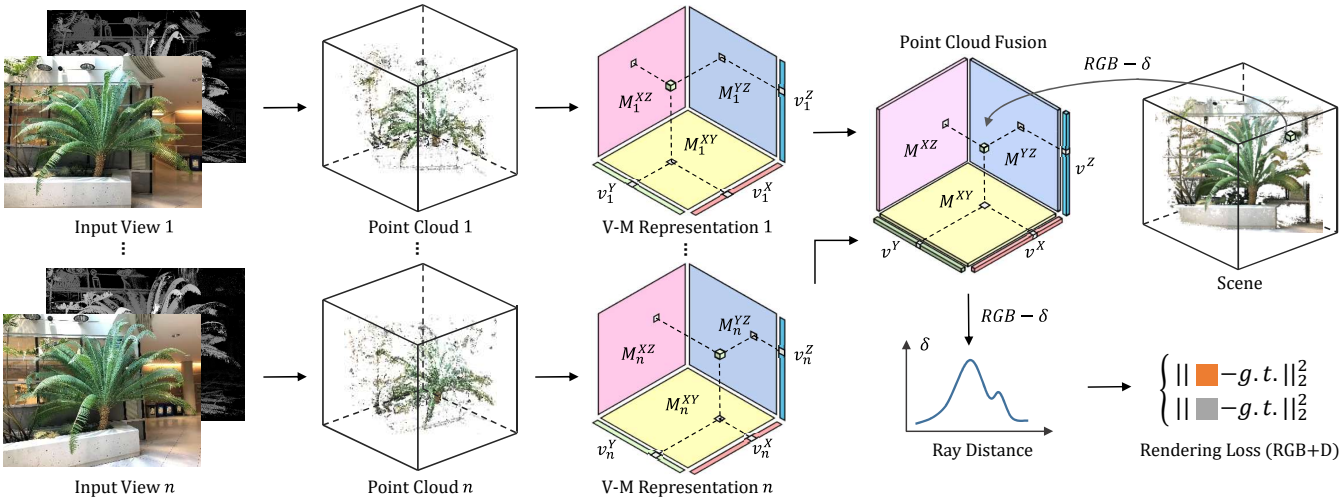

Figure 3: Overview of our method. We perceive radiance fields as an explicit voxel grid of features. With RGB-D images and camera parameters of n sparse input views, we first map pixel points into 3D space to construct a point cloud for each view, represented by vectors and matrices. Then we accumulate the point cloud of each input view to construct the fused point cloud of the entire scene. For each shading location $\mathrm{x}_w = (x, y, z)$, we use sampled values from the vectors and matrices to compute the corresponding values of the tensor component. The appearance values are sent to a decoding MLP S for color regression. The loss function is composed of RGB loss and depth loss.

Exploiting Training Data PixelNeRF (Yu et al. 2021) enhances scene comprehension by conditioning a NeRF on image inputs using a fully convolutional approach. RegNeRF (Niemeyer et al. 2022) applies regularization to the geometry and appearance of rendered patches from unseen views, helping to rectify inaccurately optimized scene geometry and divergent behavior at the optimization outset. SPARF (Truong et al. 2023) employs pixel matches between input views and depth consistency to generate realistic novel-view renderings with sparse inputs.

Leveraging Prior Information DSNeRF (Deng et al. 2022), for example, uses the sparse depth information created by COLMAP (Schonberger and Frahm 2016) as explicit supervision for sparse view synthesis. SparseNeRF (Guangcong et al. 2023), meanwhile, relies on depth ranking prior and spatial continuity distillation on NeRFs, enabling the synthesis of novel views with sparse view inputs. DDP-NeRF (Roessle et al. 2022) synthesizes novel views of entire rooms from significantly fewer images by employing dense depth priors to constrain the NeRF optimization, thereby enabling data-efficient novel-view synthesis on challenging indoor scenes.

Introducing New Regularizations InfoNeRF (Kim, Seo, and Han 2022) enhances the compactness of reconstructed scenes along individual rays while ensuring consistency across neighboring rays, an approach particularly beneficial for few-shot novel-view synthesis. Drawing on the importance of frequency, FreeNeRF (Yang, Pavone, and Wang 2023) regulates the frequency range of NeRF's inputs. It also imposes penalties on the density fields near the camera. Both methods demonstrate innovative approaches to manipulating input parameters to optimize output with sparse views.

Method

We first revisit the feature grids and radiance field, followed by an analysis of factorizing radiance fields and point cloud representation. Subsequently, we illustrate the process of continuous field representation, point cloud fusion, and rendering. The presentation concludes with a discussion on the optimization and loss function. Figure 3 is an overview of our method.

Feature Grids and Radiance Field Revisited

We construct a model of a radiance field which establishes a relationship between a 3D location x and a viewing direction d, with its volume density σ, and a view-dependent color c. Taking inspiration from TensoRF (Chen et al. 2022), we employ a standard 3D grid $\mathcal{G} \in \mathbb{R}^{I \times J \times K}$. Each voxel within this grid is equipped with multi-channel features, thus allowing us to simulate this function. Here, I, J, and K represent the resolutions of the feature grid along the X, Y, and Z axes, respectively. We segregate these feature channels into two distinct grids, one for geometry, represented by $\mathcal{G}_\sigma \in \mathbb{R}^{I \times J \times K}$, and another for appearance, represented by $\mathcal{G}_c \in \mathbb{R}^{I \times J \times K \times P}$. In this context, P indicates the number of appearance feature channels. These individual grids are designed to separately model the volume density σ, and the view-dependent color c.

Our model accommodates a range of appearance features within the appearance grid $\mathcal{G}_c$, which depend on a predefined function S. This function transforms an appearance feature vector in combination with a viewing direction d, into a color c. Here S is a small MLP where the appearance grid $\mathcal{G}_c$ comprises neural features and spherical harmonics (SH) coefficients, respectively. On the other hand, we introduce a single-channel grid $\mathcal{G}_\sigma$, where the values directly rep-

resent volume density, thus eliminating the need for an extra conversion function. This results in a continuous grid-based radiance field which can be expressed by the equation:

$$\sigma, c = \mathcal{G}_\sigma(x), S(\mathcal{G}_c(x), d). \tag{1}$$

In this equation, $\mathcal{G}_\sigma(x)$ and $\mathcal{G}_c(x)$ represent the features from the two grids at the location x, interpolated trilinearly.

Factorizing Radiance Fields

We model the geometry grid $\mathcal{G}_\sigma$ and the appearance grid $\mathcal{G}_c$ as factorized tensors. Utilizing the Vector-Matrix (VM) decomposition, we factorize the 3D geometry tensor $\mathcal{G}_\sigma$ as:

$$\begin{aligned}\mathcal{G}_\sigma &= \sum_{r=1}^{n} v^X_{\sigma,r} \circ M^{YZ}_{\sigma,r} + v^Y_{\sigma,r} \circ M^{XZ}_{\sigma,r} + v^Z_{\sigma,r} \circ M^{XY}_{\sigma,r} \\ &= \sum_{r=1}^{n} \sum_{m \in XYZ} \mathcal{A}^m_{\sigma,r},\end{aligned} \tag{2}$$

where $v^X_{\sigma,r} \in \mathbb{R}^I$, $v^Y_{\sigma,r} \in \mathbb{R}^J$, $v^Z_{\sigma,r} \in \mathbb{R}^K$, $M^{XY}_{\sigma,r} \in \mathbb{R}^{I \times J}$, $M^{YZ}_{\sigma,r} \in \mathbb{R}^{J \times K}$, and $M^{XZ}_{\sigma,r} \in \mathbb{R}^{I \times K}$.

Our approach differs from TensoRF in that the number of components of the 3D geometry tensor in our method is consistently fixed as the number of input views, denoted as n. Each component corresponds to an input view and is used to represent the volume density of the point cloud, constructed by this view. Similarly, the appearance tensor $\mathcal{G}_c$ is modeled using comparable vector-matrix spatial factors and additional feature basis vectors b_r, which express a multi-channel voxel feature grid:

$$\begin{aligned}\mathcal{G}_c = \sum_{r=1}^{n} \mathcal{A}^X_{c,r} \circ b_{3r-2} + \mathcal{A}^Y_{c,r} \circ b_{3r-1} + \mathcal{A}^Z_{c,r} \circ b_{3r}, \\ \mathcal{A}^X_{c,r} = v^X_{c,r} \circ M^{YZ}_{c,r}, \\ \mathcal{A}^Y_{c,r} = v^Y_{c,r} \circ M^{XZ}_{c,r}, \\ \mathcal{A}^Z_{c,r} = v^Z_{c,r} \circ M^{XY}_{c,r}.\end{aligned} \tag{3}$$

In this context, $v^X_{c,r} \in \mathbb{R}^I$, $v^Y_{c,r} \in \mathbb{R}^J$, $v^Z_{c,r} \in \mathbb{R}^K$, $M^{XY}_{c,r} \in \mathbb{R}^{X \times Y}$, $M^{YZ}_{c,r} \in \mathbb{R}^{J \times K}$, and $M^{XZ}_{c,r} \in \mathbb{R}^{I \times K}$.

Contrasting with TensoRF, our method also fixes the number of components of the appearance tensor as the number of input views n. Here, each component corresponds to an input view and represents the appearance of the point cloud constructed from this view. We maintain $3n$ vectors b_r to match the total number of components. By stacking all b_r, we create a $P \times 3n$ matrix B, which serves as a global appearance dictionary, abstracting the appearance commonalities across the entire scene. A density value $\mathcal{G}_{\sigma,ijk}$ of a single voxel at indices ijk can be calculated by the provided equation:

$$\mathcal{G}_{\sigma,ijk} = \sum_{r=1}^{n} \sum_{m \in XYZ} \mathcal{A}^m_{\substack{\sigma,r \\ ijk}}. \tag{4}$$

In parallel, the appearance grid $\mathcal{G}_{c,ijk}$, corresponding to $\mathcal{G}_c$ at fixed XYZ indices ijk, can also be calculated by the given method:

$$\begin{aligned}\mathcal{G}_{c,ijk} = \sum_{r=1}^{n} \mathcal{A}^X_{\substack{c,r \\ ijk}} \circ b_{3r-2} + \mathcal{A}^Y_{\substack{c,r \\ ijk}} \circ b_{3r-1} \\ + \mathcal{A}^Z_{\substack{c,r \\ ijk}} \circ b_{3r} = B \oplus ([\mathcal{A}^m_{\substack{\sigma,r \\ ijk}}]_{m,r}).\end{aligned} \tag{5}$$

Point Cloud Representation

We construct a point cloud for each input view, which is represented within the voxel grid using the corresponding feature vectors and matrices. Initially, we map depth values to the normalized device coordinate (NDC) space to ensure that all visible locations are normalized and represented within a predetermined cubic space. Subsequently, we map all pixels of the input view to the 3D space to generate the point cloud. For a 2D pixel location, x_p, in the $r-$th input image, we map it to a 3D world location $\mathrm{x}_w = (x, y, z)$, using the equation:

$$\mathrm{x}_w = R^{-1}(K^{-1}\mathrm{x}_p D - t), \tag{6}$$

where K, R, and t denote the intrinsic parameters, rotation matrix, and translation matrix of the input view corresponding to x_p.

Next, we represent the point cloud by constructing its geometry grid $\mathcal{G}_\sigma$ and appearance grid $\mathcal{G}_c$. As the 3D space is represented by a standard 3D grid $\mathcal{G} \in \mathbb{R}^{I \times J \times K}$, the world location $\mathrm{x}_w = (x, y, z)$ is projected onto the matrix $M^{XY}_{\sigma,r}, M^{YZ}_{\sigma,r}, M^{XZ}_{\sigma,r}$ and vectors $v^X_{\sigma,r}, v^Y_{\sigma,r}, v^Z_{\sigma,r}$. Elements of the matrices and vectors that are projections of the point cloud are assigned a value of 1, while all other elements are assigned a value of 0. This assignment method effectively indicates the presence of points in the geometry grid.

For the appearance grid, elements of the matrix $M^{XY}_{c,r}$, $M^{YZ}_{c,r}$, $M^{XZ}_{c,r}$ are assigned the average of R, G, B color values of the points projected onto this element. Meanwhile, vectors $v^X_{c,r}$, $v^Y_{c,r}$, and $v^Z_{c,r}$ are assigned random values. This provides a rough representation of the point clouds. After the training step, the representation of point clouds will be refined to overcome inaccurate depth values and will be fused together to characterize the entire scene.

Continuous Field Representation

We employ trilinear interpolation to represent a continuous field. For instance, consider a component tensor represented as $\mathcal{A}^X_r = v^X_r \circ M^{YZ}_r$. Each tensor element within this can be described as $\mathcal{A}^X_{r,ijk} = v^X_{r,i} M^{YZ}_{r,jk}$. The interpolated values can then be calculated using:

$$\mathcal{A}^X_r(\mathrm{x}) = v^X_r(x) M^{YZ}_r(y, z). \tag{7}$$

In the equation above, $\mathcal{A}^X_r(\mathrm{x})$ signifies the trilinearly interpolated value at the 3D location $\mathrm{x} = (x, y, z)$ of $\mathcal{A}_r$. The term $v^X_r(x)$ represents the linear interpolation at position x along the X-axis. Meanwhile, $M^{YZ}_r(y, z)$ denotes the bilinear interpolation at (y, z) of M^{YZ}_r in the YZ plane. Similarly, the following relations hold:

$$\begin{aligned}\mathcal{A}^Y_r(\mathrm{x}) = v^Y_r(y) M^{XZ}_r(x, z), \\ \mathcal{A}^Z_r(\mathrm{x}) = v^Z_r(z) M^{XY}_r(x, y).\end{aligned} \tag{8}$$

By trilinearly interpolating both grids and merging the point cloud, we obtain:

$$\mathcal{G}_\sigma(\mathrm{x}) = \sum_r \sum_m \mathcal{A}^m_{\sigma,r}(\mathrm{x}), \tag{9}$$

$$\mathcal{G}_c(\mathrm{x}) = B \oplus ([\mathcal{A}^m_{c,r}(\mathrm{x})]_{m,r}). \tag{10}$$

Point Cloud Fusion and Rendering

Integrating equations (1), (9) and (10), the factorized tensorial radiance field for the fused point cloud in our model is articulated as:

$$\sigma, c = \sum_r \sum_m \mathcal{A}^m_{\sigma,r}(x), S(B \oplus ([\mathcal{A}^m_{c,r}(\mathrm{x})]_{m,r}), d). \tag{11}$$

To render images, we march along a ray, and Q shading points are sampled along each ray. The color of the pixel is then determined using:

$$\begin{aligned} C &= \sum_{q=1}^{Q} \tau_q (1 - \exp(-\sigma_q \Delta_q)) c_q, \\ \tau_q &= \exp(-\sum_{p=1}^{q-1} \sigma_p \Delta_p). \end{aligned} \tag{12}$$

Here, σ_q and c_q denote the density and color respectively, determined by our model at their specific sampled locations x_q. Meanwhile, Δ_q is defined as the step size of the ray and τ_q stands for transmittance.

To further improve quality and avoid local minima, we apply coarse-to-fine reconstruction. Similar to TensoRF (Chen et al. 2022), our coarse-to-fine reconstruction is simply achieved by linearly and bilinearly upsampling our XYZ-mode vector and matrix factors.

Optimization

The network parameters are optimized using a collection of RGB-D frames, each containing color, depth, and camera pose data. Our loss function consists of two primary components. The first component is an RGB loss function. This component involves an L2 rendering loss combined with additional regularization terms to optimize our tensor factors for radiance field reconstruction. It can be represented as:

$$\mathcal{L}_{RGB} = \|C - \widetilde{C}\|_2^2 + \omega_{reg} \mathcal{L}_{reg}. \tag{13}$$

Here, $\widetilde{C}$ denotes the ground truth color, C represents the predicted color, $\mathcal{L}_{reg}$ is an L1 regularization term, and ω_{reg} is the weight assigned to this regularization. To promote sparsity in our tensor factors' parameters, we use the standard L1 regularization. This technique has proven effective in enhancing the extrapolation of views and eliminating anomalies like floaters or outliers in the final renderings.

The second component is a depth loss function, in which we employ an L2 rendering loss and can be expressed as:

$$\mathcal{L}_{depth} = \|D - \widetilde{D}\|_2^2. \tag{14}$$

In this equation, $\widetilde{D}$ is the ground truth depth, while D stands for the predicted depth. Contrary to the approach in DSNeRF (Deng et al. 2022), we normalize the ground truth depth in the NDC space. This normalization ensures that all visible locations are encapsulated within a predefined cubic space.

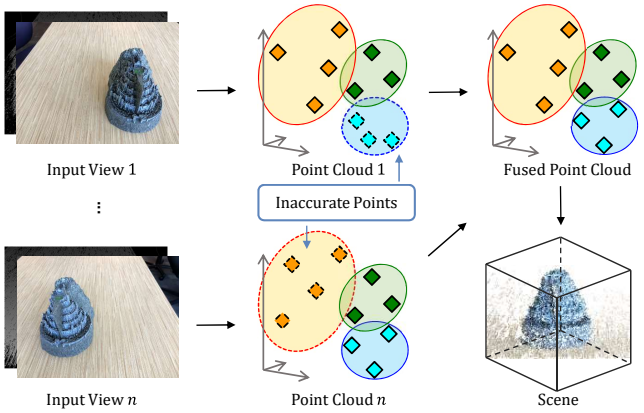

Figure 4: Inaccurate depth values can result in inaccurate 3D points. Through point cloud fusion and radiance field optimization, these inaccurate 3D points are substituted with accurate ones from other views. The squares represented by the dotted edges indicate inaccurate 3D points.

In summary, the composite loss function for our model, denoted by $\mathcal{L}$, is articulated as:

$$\mathcal{L} = \mathcal{L}_{RGB} + \lambda_{depth} \mathcal{L}_{depth}, \tag{15}$$

where λ_{depth} serves as a hyperparameter to strike a balance between color and depth supervision. Figure 4 illustrates how the inaccuracies are addressed through point cloud fusion and radiance field optimization.

Experiments

Datasets

Our results are derived from the real-world multi-view datasets LLFF (Mildenhall et al. 2019) and DTU (Jensen et al. 2014). The LLFF dataset includes 8 forward-facing scenes, each having a varied count of frames, all presented at a spatial resolution of 4032×3024. The DTU dataset features various objects captured from multiple perspectives in a controlled indoor setting.

For every scene, we selected subsets containing 2, 3, or 4 training views for our evaluation. Depth maps were generated using COLMAP (Schonberger and Frahm 2016), adhering to the dataset-provided camera parameters.

Baselines

We compare our method against several state-of-the-art models, including InfoNeRF (Kim, Seo, and Han 2022), DietNeRF (Jain, Tancik, and Abbeel 2021), RegNeRF (Niemeyer et al. 2022), DSNeRF (Deng et al. 2022), DDP-NeRF (Roessle et al. 2022), and ViP-NeRF (Somraj and Soundararajan 2023). It is noteworthy that all results of these methods are obtained through publicly accessible codes or papers.

Method	2 views			3 views			4 views		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
InfoNeRF	9.23	0.2095	0.7761	8.52	0.1859	0.7679	9.25	0.2188	0.7701
DietNeRF	11.89	0.3209	0.7265	11.77	0.3297	0.7254	11.84	0.3404	0.7396
RegNeRF	16.90	0.4872	0.4402	18.62	0.5600	0.3800	19.83	0.6056	0.3446
DSNeRF	17.06	0.5068	0.4548	19.02	0.5686	0.4077	20.11	0.6016	0.3825
DDP-NeRF	17.21	0.5377	0.4223	17.90	0.5610	0.4178	19.19	0.5999	0.3821
ViP-NeRF	16.76	0.5222	0.4017	18.92	0.5837	0.3750	19.57	**0.6085**	0.3593
Our Method	**17.83**	**0.5512**	**0.3832**	**19.30**	**0.6027**	**0.3682**	**20.86**	0.5967	**0.3247**

Table 1: Quantitative Comparisons on The LLFF Dataset.

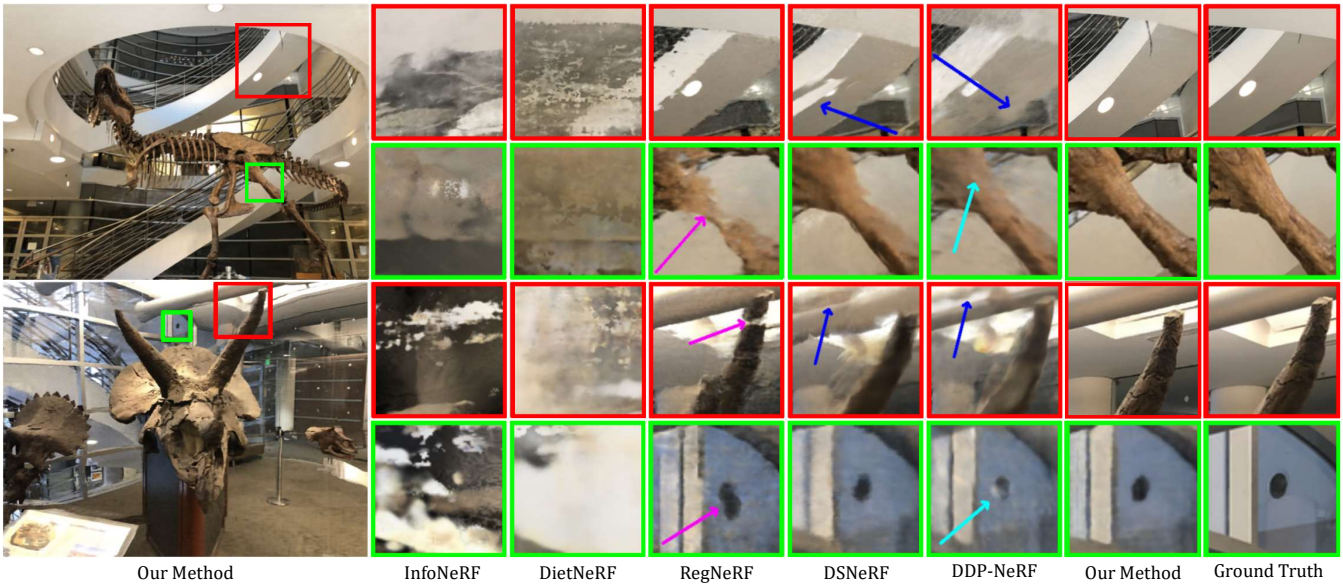

Figure 5: Qualitative comparisons on the LLFF dataset with two input views. Notably, the predictions from DSNeRF and DDP-NeRF exhibit noticeable floater artifacts. RegNeRF struggles to capture the finer details in bone structures. In contrast, our method significantly reduces these imperfections. In the second and fourth examples, we highlight the color changes predicted by DDP-NeRF. Our model's predictions are free from the aforementioned artifacts.

Implementation Details

Our implementation was carried out in PyTorch (Paszke et al. 2019), excluding any customized CUDA kernels. This model was optimized over T iterations, with a batch size of 4096 pixel rays, executed on a single NVIDIA RTX 4090 GPU (24GB). We introduced a feature decoding MLP and set $P = 27$. To facilitate a stepwise transition from coarse-to-fine reconstruction, we initiated with a grid of N_0^3, where $N_0 = 128$. This grid was upsampled at intervals of 2000, 3000, 4000, 5500, and 7000 steps, with voxel counts transitioning linearly in logarithmic space from N_0^3 to N^3.

Comparisons

Comparisons on The LLFF Dataset The quantitative and qualitative comparisons with competing models are respectively showcased in the referenced Table 1 and Figure 5. Here, we note that predictions from other models frequently exhibit blurriness, especially for views substantially distanced from the input. In contrast, our method performs well in geometry predictions, generating more realistic novel views. Our method consistently delivers sharp outputs accompanied by precise scene geometry across all tested scenarios. A standout feature of our approach is its ability to accurately represent and recover intricate details. Additionally, our model is good at eliminating visual noise, ensuring clearer visuals around objects in comparison to the baseline models.

Comparisons on The DTU Dataset The qualitative comparisons between the competing models can be found in Figure 6. Our approach consistently surpasses other models, especially in the perceptual metric domain. Additionally, these models often present inconsistent appearances for novel views, particularly when the camera perspectives deviate significantly from the input views. We can see that our model aligns more closely with the ground truth and avoids many of the artifacts evident in the predictions of other models.

Dataset	0% Noise			5% Noise			10% Noise		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
LLFF	17.83	0.5512	0.3832	15.98	0.4058	0.4597	13.26	0.2615	0.6433
DTU	19.34	0.7431	0.2576	18.17	0.5976	0.3814	16.63	0.4144	0.5561

Table 2: Influence of depth quality on our method. In this table, we add 5% and 10% white noise to depth maps respectively to observe the performance of our method. The original depth maps are obtained with COLMAP.

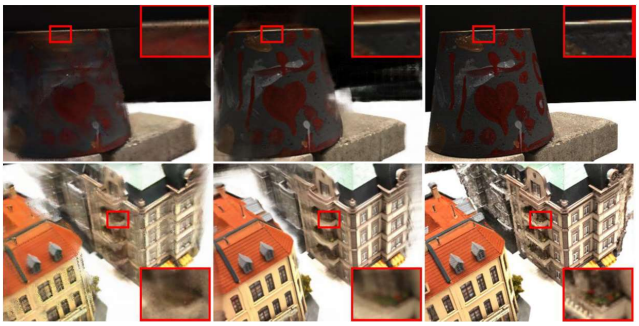

Figure 6: Qualitative comparisons on the DTU dataset with two input views. Predictions of DDP-NeRF and ViP-NeRF display pronounced floating cloud artifacts. Our method yields more lifelike and convincing novel views.

Figure 7: Influence of point cloud construction of our method. We show some qualitative examples on the LLFF dataset with two input views. Our method yields more lifelike and convincing novel views when point cloud construction and fusion are introduced.

Ablation Study

Ablation of Point Cloud Construction and Fusion In Figure 7, we highlight the significance of our point cloud construction and fusion, which effectively addresses the challenges posed by inaccurate depth values. When the point cloud construction is not applied, all matrices and vectors default to random values. The adoption of our point cloud construction and fusion approach yields superior quantitative and qualitative outcomes.

Depth Quality To delve deeper into the effects of varied depth quality on our method, we present synthesis outcomes as depth quality fluctuates in Table 2. For reference, the highest quality depth maps are sourced from COLMAP (Schonberger and Frahm 2016). We intentionally degrade the quality of these depth maps by converting 5% and 10% of the best depth values into white noise. Even with diminishing depth map quality, our method maintains commendable performance.

Discussion

With the depth supervision and our streamlined tensorial radiance field structure, our method boasts better performance and faster reconstruction.

To reduce the influence of inaccurate depth values, we use depth information to optimize the neural radiance field. We project points from each input view into 3D space, creating a unique point cloud for every view, represented by matrices and vectors. Although the initial representation of the point clouds is rudimentary and imprecise, the fusion and training processes refine inaccuracies in depth values or replace them with values from alternate views.

As shown in Figure 2, our method has less model size and less reconstruction time, as we effectively present the point cloud constructed for each input view with a few vectors and matrices. For example, for a $300 \times 300 \times 300$ feature grid with $P = 27$ channels (plus one density channel), the total number of parameters in a dense grid is 756 M, while the number of parameters used for our method is only about 0.36 M (four views input). We can achieve a compression rate of about 0.05%.

Conclusion

In this paper, we introduce the pioneering depth-guided robust and fast point cloud fusion NeRF tailored for sparse view input. We observed that existing depth-guided NeRFs for sparse input views tend to neglect inaccuracies in depth maps and often suffer from low time efficiency. To the best of our knowledge, this represents the first integration of point cloud fusion into the NeRF framework. Our method leverage depth information to construct a superior radiance field while reducing the influence of inaccurate depth values. It also enables faster reconstruction and greater compactness via efficient vector-matrix decomposition.

Limitations and FutureWork

We believe that depth-guided radiance fields based on matrix and vector representations hold significant promise in time efficiency enhancement. Moving forward, we will aim to further leverage depth information and tensorial structures to improve the performance and efficiency of NeRF.

Acknowledgments

This work was supported by the Fundamental Research Funds for the Central Universities, National Key R&D Project of China (2019YFB1802701), MoE-China Mobile Research Fund Project (MCM20180702), STCSM under Grant 22DZ2229005, 111 project BP0719010.

References

Chen, A.; Xu, Z.; Geiger, A.; Yu, J.; and Su, H. 2022. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision*, 333–350. Springer.

Deng, K.; Liu, A.; Zhu, J.-Y.; and Ramanan, D. 2022. Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12882–12891.

Dey, A.; Ahmine, Y.; and Comport, A. I. 2022. Mip-NeRF RGB-D: Depth Assisted Fast Neural Radiance Fields. *arXiv preprint arXiv:2205.09351*.

Feng, Y.; Feng, Y.; You, H.; Zhao, X.; and Gao, Y. 2019. Meshnet: Mesh neural network for 3d shape representation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, 8279–8286.

Guangcong; Chen, Z.; Loy, C. C.; and Liu, Z. 2023. SparseNeRF: Distilling Depth Ranking for Few-shot Novel View Synthesis. *Technical Report*.

Huang, X.; Zhang, Y.; Ni, B.; Li, T.; Chen, K.; and Zhang, W. 2023. Boosting point clouds rendering via radiance mapping. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 953–961.

Jain, A.; Tancik, M.; and Abbeel, P. 2021. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5885–5894.

Jensen, R.; Dahl, A.; Vogiatzis, G.; Tola, E.; and Aanæs, H. 2014. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 406–413.

Kim, M.; Seo, S.; and Han, B. 2022. Infonerf: Ray entropy minimization for few-shot neural volume rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12912–12921.

Kobayashi, S.; Matsumoto, E.; and Sitzmann, V. 2022. Decomposing nerf for editing via feature field distillation. *Advances in Neural Information Processing Systems*, 35: 23311–23330.

Kundu, J. N.; Seth, S.; Rahul, M.; Rakesh, M.; Radhakrishnan, V. B.; and Chakraborty, A. 2020. Kinematic-structure-preserved representation for unsupervised 3d human pose estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 11312–11319.

Li, L.; Shen, Z.; Wang, Z.; Shen, L.; and Tan, P. 2022a. Streaming radiance fields for 3d video synthesis. *Advances in Neural Information Processing Systems*, 35: 13485–13498.

Li, T.; Slavcheva, M.; Zollhoefer, M.; Green, S.; Lassner, C.; Kim, C.; Schmidt, T.; Lovegrove, S.; Goesele, M.; Newcombe, R.; et al. 2022b. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5521–5531.

Mildenhall, B.; Srinivasan, P. P.; Ortiz-Cayon, R.; Kalantari, N. K.; Ramamoorthi, R.; Ng, R.; and Kar, A. 2019. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 38(4): 1–14.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1): 99–106.

Niemeyer, M.; Barron, J. T.; Mildenhall, B.; Sajjadi, M. S.; Geiger, A.; and Radwan, N. 2022. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5480–5490.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Ran, H.; Liu, J.; and Wang, C. 2022. Surface representation for point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 18942–18952.

Roessle, B.; Barron, J. T.; Mildenhall, B.; Srinivasan, P. P.; and Nießner, M. 2022. Dense depth priors for neural radiance fields from sparse input views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12892–12901.

Rudnev, V.; Elgharib, M.; Smith, W.; Liu, L.; Golyanik, V.; and Theobalt, C. 2022. Nerf for outdoor scene relighting. In *European Conference on Computer Vision*, 615–631. Springer.

Schonberger, J. L.; and Frahm, J.-M. 2016. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4104–4113.

Sitzmann, V.; Thies, J.; Heide, F.; Nießner, M.; Wetzstein, G.; and Zollhofer, M. 2019. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2437–2446.

Somraj, N.; and Soundararajan, R. 2023. ViP-NeRF: Visibility Prior for Sparse Input Neural Radiance Fields. *arXiv preprint arXiv:2305.00041*.

Song, N.; Jiang, T.; and Yao, J. 2022. JPV-Net: Joint Point-Voxel Representations for Accurate 3D Object Detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 2271–2279.

Truong, P.; Rakotosaona, M.-J.; Manhardt, F.; and Tombari, F. 2023. Sparf: Neural radiance fields from sparse and noisy poses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4190–4200.

Yang, J.; Pavone, M.; and Wang, Y. 2023. FreeNeRF: Improving Few-shot Neural Rendering with Free Frequency Regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8254–8263.

Yang, Z.; Qiu, Z.; and Fu, D. 2023. Dmis: Dynamic mesh-based importance sampling for training physics-informed neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 5375–5383.

Yu, A.; Ye, V.; Tancik, M.; and Kanazawa, A. 2021. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4578–4587.

Yu, H.; Chen, A.; Chen, X.; Xu, L.; Shao, Z.; and Yu, J. 2022. Anisotropic fourier features for neural image-based rendering and relighting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 3152–3160.

Yuan, Y.-J.; Sun, Y.-T.; Lai, Y.-K.; Ma, Y.; Jia, R.; and Gao, L. 2022. Nerf-editing: geometry editing of neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 18353–18364.

Zhou, T.; Tucker, R.; Flynn, J.; Fyffe, G.; and Snavely, N. 2018. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*.