MixRF: Universal Mixed Radiance Fields With Points and Rays Aggregation

Haiyang Bai, Tao Lu, Jiaqi Zhu, Wei Huang, Chang Gou, *Graduate Student Member, IEEE*, Jie Guo, Lijun Chen, and Yanwen Guo

***Abstract*—Recent advancements in neural rendering methods, such as Neural Radiance Fields (NeRF) and 3D Gaussian Splatting (3D-GS), have significantly revolutionized photo-realistic novel view synthesis of scenes with multiple photos or videos as input. However, existing approaches within the NeRF and 3D-GS frameworks often assume the independence of point sampling and ray casting, which are intrinsic to volume rendering and alpha-blending techniques. These underlying assumptions limit the ability to aggregate context within subspaces, such as densities and colors in the radiance fields and pixels on the image plane, leading to synthesized images that lack fine details and smoothness. To overcome this, we propose a universal framework, MixRF, comprising a Radiance Field Mixer (RF-mixer) and a Color Domain Mixer (CD-mixer), to sufficiently aggregate and fully explore information in neighboring sampled points and casting rays, separately. The RF-mixer treats sampled points as an explicit point cloud, enabling the aggregation of density and color attributes from neighboring points to better capture local geometry and appearance. Meanwhile, the CD-mixer rearranges rendered pixels on the sub-image plane, improving smoothness and recovering fine details and textures. Both mixers employ a kernel-based mixing strategy to facilitate effective and controllable attribute aggregation, ensuring a more comprehensive exploration of radiance values and pixel information. Extensive experiments demonstrate that our MixRF framework is compatible with radiance field-based methods, including NeRF and 3D-GS designs. The proposed framework dramatically enhances performance in both qualitative and quantitative evaluations, with less than a 25% increase in computational overhead during inference.**

***Index Terms*—Novel view synthesis, neural radiance fields, 3D Gaussian splatting, radiance field mixer, color domain mixer.**

I. Introduction

Synthesizing photo-realistic images under novel viewpoints is a fundamental and pivotal task in computer vision and graphics, with applications ranging from virtual reality (VR) and gaming to augmented reality (AR) and autonomous driving [1], [2]. Classical methods typically generate a novel view by querying explicit 3D representations like meshes or point clouds [3], [4], [5], [6] to achieve high quality. However, acquiring such 3D auxiliaries is often challenging, and the visual quality can suffers significantly due to defects in 3D data. Neural Radiance Fields (NeRF) [7] and 3D Gaussian Splatting (3D-GS) [8] provide alternatives that directly optimize the spatial distribution of radiance fields using only 2D images through gradient descent, eliminating the requirement on precise 3D geometry. This enables the creation of high-quality renderings of scenes with only multiple photos or videos as input.

NeRF represents a scene as a continuous function mapping spatial points to their associated color and density attributes. These attributes are then accumulated from independent points along each view ray through a volumetric ray-marching process to generate pixels. Despite the remarkable success of NeRF and its variants [6], [7], [9], [10], [11], [12], [13], [14], their approach of mapping on a per-sample basis and rendering on a per-pixel basis restricts the effective representation of spatial relationship within both the 3D scene space and the image plane. Therefore, effectively incorporating local structural comprehension becomes challenging, hindering the full exploitation of the correlation of spatial points and imaged pixels. MipNeRF/MipNeRF360 [11], [12] have made significant progress by extending a single ray to a conical frustum, introducing heuristic spatial awareness. Their effectiveness in anti-aliasing verifies the necessity of spatial modeling. However, the information contained in a cone is not adequate for capturing inter-ray relationships, resulting in a lack of smooth rendering. Other works [9], [10], [13], [14] introduce locality in features by encoding scenes into voxel spaces and interpolating vertex features from neighboring voxels. Point-NeRF [6] further adopts a more flexible point cloud as the geometry prior containing high-frequency information, and applies a KNN-based local aggregator to extract local features, improving spatial smoothness and preserving details. While these approaches consider aggregation of neighborhood features to some extent, they do not fully exploit the intrinsic relationships among rays, which leaves a great space for further improving the visual quality. Works such as [15], [16], [17], [18] utilize convolution operations to share local information among neighboring points or rays, aiming to enhance image smoothness and improve detail recovery. However, querying the spatial relationships among adjacent samples often requires additional geometrical auxiliary, such as the mesh from

Received 17 April 2024; revised 22 April 2025; accepted 8 May 2025. Date of publication 20 May 2025; date of current version 5 September 2025. This work was by the Fundamental Research Funds for the Central Universities under Grant 14380079 and in part by the Collaborative Innovation Center of Novel Software Technology and Industrialization. Recommended for acceptance by X. Han. *(Corresponding authors: Lijun Chen; Jie Guo; Yanwen Guo.)*

Haiyang Bai, Jiaqi Zhu, Wei Huang, Chang Gou, Jie Guo, Lijun Chen, and Yanwen Guo are with the National Key Lab for Novel Software Technology, Nanjing University, Nanjing 210000, China (e-mail: baihy@smail.nju.edu.cn; jiaqizhu@smail.nju.edu.cn; huangwei@smail.nju.edu.cn; cgou@smail.nju.edu.cn; guojie@nju.edu.cn; chenlj@nju.edu.cn; ywguo@nju.edu.cn).

Tao Lu is with Shanghai Artificial Intelligence Laboratory, Shanghai 200240, China (e-mail: taolu@smail.nju.edu.cn).

Digital Object Identifier 10.1109/TVCG.2025.3572015

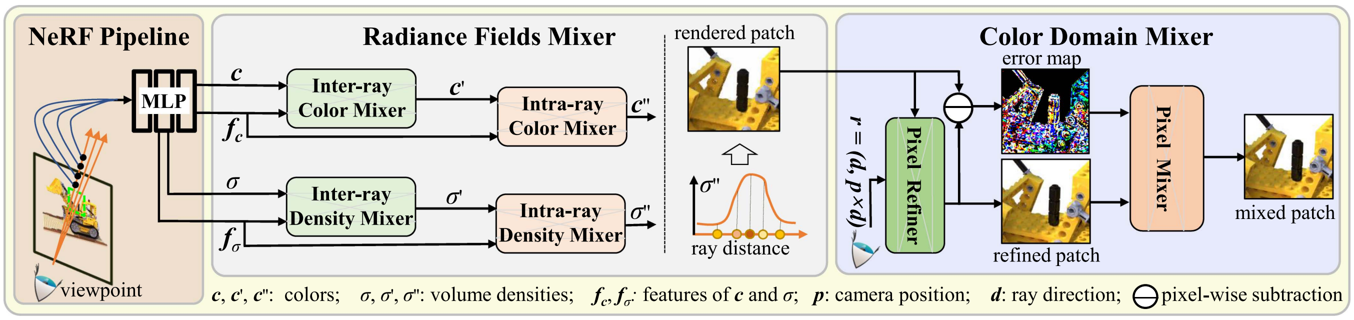

Fig. 1. *Overview of MixRF pipeline.* Our method incorporates two modules that seamlessly integrate into traditional neural rendering pipelines: the RF-mixer and CD-mixer. The RF-mixer performs a two-stage aggregation of radiance values (i.e., density and color) within a subspace. This involves aggregating sampled points at the same depth across neighboring rays (inter-ray) and adjacent points along the same ray (intra-ray), effectively fusing geometric and appearance information from surrounding points. The CD-mixer first refines rendered pixel patches with Plücker coordinates as a view consistency constraint, then mixes pixels within sub-image planes, guided by an error map derived from the comparison between the rendered and refined patches, thereby facilitating targeted texture recovery.

Shape-from-Silhouette (SFS) [16]. Furthermore, their reliance on redundant convolutional networks for decoding concatenated features leads to computational inefficiencies when processing high-dimensional tensors.

Recently, 3D-GS [8] has made significant advancements in rendering speed. It directly utilizes a sparse point cloud derived from structure-from-motion (SfM) [19] to construct Gaussian primitives associated with learnable attributes including position, opacity, color, and covariance. To generate a pixel, 3D-GS employs alpha-blending [20] to blend colors of N ordered Gaussians that overlap that pixel. While the covariance within each Gaussian primitive enhances correlations on local spatial attributes like opacity and color, the assumption of independence among pixels in tile-based rasterization limits the modeling of pixels in the local color domain, leading to visible non-smooth artifacts. To date, there has been no work dedicated to addressing this particular issue.

To simultaneously aggregate and explore information in sampled points and casting rays, we propose MixRF, a universal framework comprising two key components: the Radiance Field Mixer (RF-mixer) and Color Domain Mixer (CD-mixer), as illustrated in Fig. 1. The RF-mixer module interprets sampled points within the radiance field as an explicit point cloud associated with neural attributes including density and color. This module enables effective aggregation of both geometric and appearance information from neighboring points, facilitating the prediction of densities and colors with a localized understanding of the scene. However, most neural rendering frameworks fail to fully exploit the complementary relationships among neighboring rendered pixels due to the inherent independence of rays and pixels in volume rendering and alpha blending. As a result, pixel correlations on the sub-image plane are underutilized, leading to degraded quality in novel view synthesis. To address this limitation, the CD-mixer module processes rendered pixels by performing regional reconfiguration on the sub-image plane. This helps enhance the smoothness of novel views while simultaneously recovering finer details and textures.

To improve the efficiency of local context aggregating, we introduce a general and controllable kernel-based mixing strategy, integrated into both the RF-mixer and CD-mixer. Unlike conventional aggregation methods that first encode concatenated feature volumes into feature maps and then decode them into the target attributes—resulting in high computational cost—our strategy reformulates the process as a weighted summation of local attribute blocks, with dynamically predicted weights. This approach significantly reduces computational complexity while maintaining high performance in attribute aggregation.

We evaluate the effectiveness of our approach by conducting experiments on various radiance field-based methods against different benchmarks. Comprehensive experiments demonstrate that our MixRF is compatible with multiple mainstream NeRF- and 3D-GS-based methods, and can significantly improve the quality of novel view synthesis with reasonable computational overhead.

To summarize, our contributions are as follows.

- We introduce a universal mixed framework which is applicable to various radiance field-based methods such as NeRF and 3D-GS, significantly improving rendering quality.
- We design RF-mixer and CD-mixer to facilitate local contextual interaction in both the 3D space and image plane, yielding synthesized images that preserve high-frequency details and local smoothness.
- Our kernel-based mixing strategy allows for more effective and controllable attribute integration within the subspace, resulting in higher rendering quality with less than a 25% increase in computational overhead during inference.

II. Related Work

A. Novel View Synthesis

Novel view synthesis (NVS) refers to the generation of realistic views from various viewpoints based on a set of captured images. Traditional methods like Lumigraph [21], [22] and neural light field [23], [24], [25] directly synthesize views by interpolating input images, but require dense scene captures. Other approaches, such as [26], [27] exploring polar geometry correlations among source views to build a light field from sparse views, face challenges in optimization efficiency. Recent progress in Multi-Plane Images (MPIs) [28], [29], [30], [31], [32] has shown promising results by merging multiple image layers. Another branch involves generating views by accessing explicit 3D representations of the scene, such as meshes [33], [34], [35], [36], [37], [38], point clouds [5], [6], [39], [40], [41], or voxel grids [9], [10], [13], [14], [35]. However, acquiring these 3D representations is challenging. Recently, scene representation networks (SRNs) [28] and Neural Radiance Fields (NeRF) [7] have demonstrated impressive rendering performance by mapping 3D world coordinates to shading values and using a ray-marcher for the final RGB output. Additionally, 3D Gaussian Splatting (3D-GS) [8] produces photorealistic images by rasterizing from learnable Gaussians, achieving high-quality and real-time rendering.

B. Neural Radiance Fields

NeRF [7] has made significant strides in the task of novel view synthesis, sparking a wave of subsequent research endeavors. Departing from conventional explicit and discretized volumetric representations, NeRF utilizes a continuous 5D function to depict static scenes, optimized through a deep fully-connected neural network. Subsequent studies have focused on addressing NeRF's constraints, particularly by enhancing training and rendering speeds [10], [13], [14], [42], [43], [44], reducing aliasing artifacts to enhance rendering quality [10], [11], [12], modeling from sparse views [45], and expanding scene coverage [46], [47]. Furthermore, various works have extended NeRF's continuous neural volumetric representation for generative modeling [48], [49], [50], [51], dynamic scenes [52], [53], non-rigidly deforming objects [54], [55], [56], object detection [57], and editable scenes or geometry [56], [58], [59]. Additionally, impressive synthesis outcomes have been attained for unconstrained captures, such as varying illumination and dynamic occlusion [46], [60]. Remarkably, NeRF-based SLAM systems [1], [61] have been formulated for multiple large-scale bounded indoor or outdoor scenes. From our observations, all NeRF-related applications rely on their high-quality renderings. In this work, we mainly focus on further enhancing the performance of NeRF-based methods by a large margin.

C. 3D Gaussian Splatting

3D-GS [8] deviates from the conventional implicit scene representation approach used by MLPs. Instead, it adopts an explicit representation by utilizing Gaussian functions to model a static scene. This deliberate shift eliminates the necessity for point sampling during ray marching [7], [11] in rendering integration. It replaces this process with finite Gaussian splatting onto the image space, enabling real-time rendering capabilities. This change has garnered significant attention across various fields, including Simultaneous Localization and Mapping (SLAM) systems [2]. Moreover, 3D-GS has shown promise in scene editing [62] and segmentation [63]. Noteworthy advancements have been made in integrating 3D-GS with established diffusion models [64], [65] in the domain of Augmented Intelligence and Graphics Computing (AIGC) [66], [67]. In these applications, the diffusion model offers geometric priors, addressing inconsistencies in multi-view geometry. Another notable strategy involves tracking dynamic Gaussians to support continuous dynamic view synthesis [68], [69].

III. Method

Our method is a versatile framework that can significantly enhance the quality of synthesized images under new viewpoints. It can be seamlessly integrated with mainstream radiance field-based methods such as NeRF and 3D-GS, along with their variations. This generalization ability is achieved by exclusive operation for the intermediate or output variables, without modifying the original models. We begin with a brief review of NeRF and 3D-GS, followed by a detailed explanation of our proposed method.

A. Preliminaries

1) Neural Radiance Fields: NeRF represents the scene using sampled points with coordinates of 3D position $\boldsymbol{x} : (x, y, z) \in \mathbb{R}^3$ and 2D direction $\boldsymbol{d} : (\theta, \varphi) \in \mathbb{R}^2$ which are then mapped into volumetric density $\sigma \in \mathbb{R}^+$ and emitted RGB color $\boldsymbol{c} \in \mathbb{R}^3$ by MLP $\boldsymbol{F}_\theta$ as

$$(\boldsymbol{c}_i,\ \sigma_i) = \boldsymbol{F}_\theta\left(\boldsymbol{x}_i,\ \boldsymbol{d}_i\right). \tag{1}$$

The final pixel color $\hat{\boldsymbol{C}}$ is determined through volumetric rendering along a ray with intervals δ_i as

$$\hat{\boldsymbol{C}} = \sum_{i=1}^{N} \boldsymbol{T}_i\, \boldsymbol{\alpha}_i\, \boldsymbol{c}_i, \tag{2}$$

where

$$\boldsymbol{\alpha}_i = \left(1 - exp\left(-\sigma_i \delta_i\right)\right), \quad \boldsymbol{T}_i = \sum_{j=1}^{i-1} \left(1 - \boldsymbol{\alpha}_i\right).$$

2) 3D Gaussian Splatting: The scene representation in 3D-GS is achieved by optimizing a collection of 3D Gaussian functions. Each Gaussian function characterizes the local radiance field by forming an anisotropic distribution centered at $\boldsymbol{\mu}_i$ with covariance $\boldsymbol{\Sigma}_i$, opacity α_i, and view-dependent color $\boldsymbol{c}_i$. These properties are optimized using back-propagation. The ensemble of 3D Gaussians is denoted as

$$\boldsymbol{G}_i = \left\{ \left(\boldsymbol{\mu}_i,\ \boldsymbol{\Sigma}_i,\ \alpha_i,\ \boldsymbol{c}_i\right) \middle| \left(i = 0, 1, 2 \ldots\right) \right\}. \tag{3}$$

3D Gaussians $\boldsymbol{G}_i$ located within the view frustum are initially projected into 2D Gaussians $\boldsymbol{G}_i{}'$ with 2D covariance $\boldsymbol{\Sigma}'$ on the image plane. The final color is then computed in parallel by blending N ordered 2D Gaussians that overlap each pixel:

$$\hat{\boldsymbol{C}} = \sum_{i=1}^{N} \boldsymbol{c}_i \alpha_i{}' \prod_{j=1}^{i-1} \left(1 - \alpha_j{}'\right), \tag{4}$$

where

$$\alpha_i{}' = \alpha_i \cdot exp\left(-\frac{1}{2}\left(\boldsymbol{x}' - \boldsymbol{\mu}_i{}'\right)^{\mathrm{T}} \boldsymbol{\Sigma}_i{}'^{-1} \left(\boldsymbol{x}' - \boldsymbol{\mu}_i{}'\right)\right).$$

Here, $\boldsymbol{x}'$ and $\boldsymbol{\mu}_i'$ represent the coordinates in the projected space.

Analysis: Eqn. (1) demonstrates that the mapping process in NeRF occurs at the level of individual points, limiting the incorporation of spatial context. Similarly, (2) and (4) represent volume rendering and tile-based alpha-blending, perform on either a per-pixel or per-ray manner, which restricts the optimization of rendered pixels within the image plane's subspace.

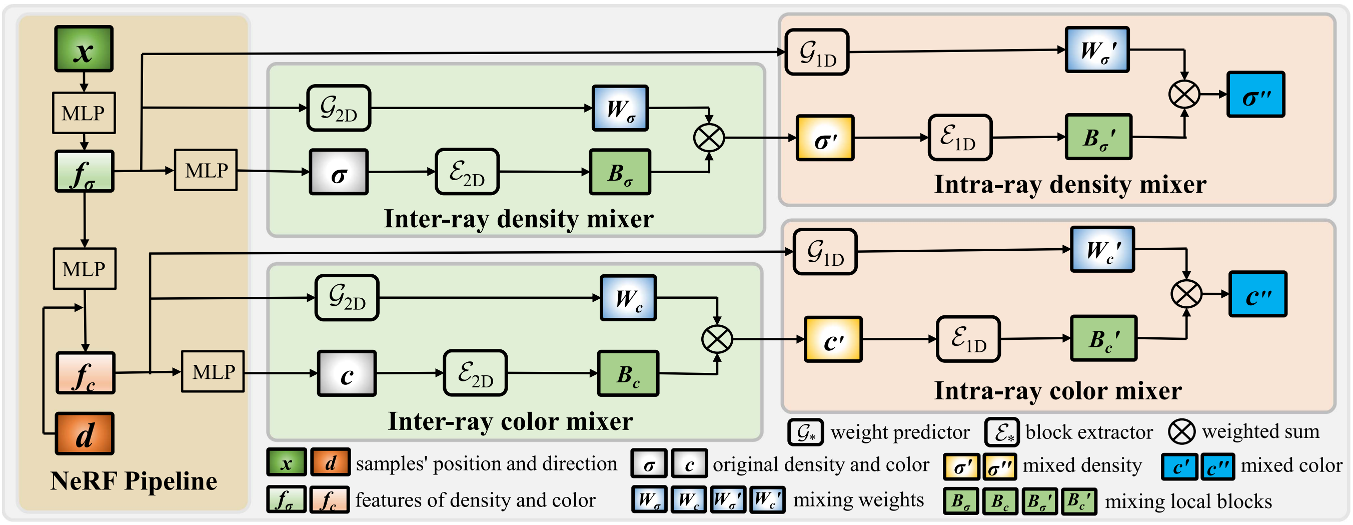

Fig. 2. *Overview of RF-mixer.* We perform a two-stage integration process that involves mixing sample attributes (*i.e.*, density and color) within neighboring rays (inter-ray) at the same sampling depth and among adjacent samples along the ray (intra-ray). Initially, we extract NeRF variables, including densities, colors, and their associated features $\{\sigma, \boldsymbol{c}, \boldsymbol{f}_\sigma, \boldsymbol{f}_c\}$. Subsequently, we feed $\{\sigma, \boldsymbol{f}_\sigma\}$ and $\{\boldsymbol{c}, \boldsymbol{f}_c\}$ into the weight predictor $\mathcal{G}$ and block extractor $\mathcal{E}$, separately deriving corresponding sets of mixing weights and local blocks $\{\boldsymbol{W}_\sigma, \boldsymbol{B}_\sigma\}$ and $\{\boldsymbol{W}_c, \boldsymbol{B}_c\}$. This is followed by a weighted summation of elements within each set, yielding the inter-ray mixed density and color $\{\sigma', \boldsymbol{c}'\}$. Analogously, the intra-ray mixed density and color $\{\sigma'', \boldsymbol{c}''\}$ are obtained from variables $\{\sigma', \boldsymbol{c}', \boldsymbol{f}_\sigma, \boldsymbol{f}_c\}$ in the second stage. Our RF-mixer is trained end-to-end, supervised by the sum of the L_2 distances between the ground truth and pixels by volume rendered (VR).

These limitations, as illustrated in Figs. 8(a) and 9(a), lead to incomplete geometric reconstruction and low-quality rendering. To address this, we introduce the Radiance Field Mixer (RF-mixer) and the Color Domain Mixer (CD-mixer), which utilize an efficient kernel-based mixing strategy to incorporate spatial priors, enabling the prediction of radiance values and the construction of high-quality, structurally coherent radiance fields.

B. Overview

Our method aims to improve the quality of novel view synthesis in neural rendering by introducing two key components: the Radiance Fields Mixer (RF-mixer) and the Color Domain Mixer (CD-mixer), as illustrated in Fig. 1. Using the NeRF framework as an example, we adopt a patch sampling technique [70], [71] that naturally facilitates convenient querying of both neighboring points along the same ray (intra-ray) and those on adjacent rays (inter-ray). Following such sampling, the RF-mixer treats the sampled points, mapped through (1), as explicit point clouds with neural attributes (*i.e.* color and density). It employs a two-stage mixing strategy to aggregate radiance values from points in subspace before volume rendering, resulting in a more precise radiance field reconstruction and enhanced rendering quality. The CD-mixer first processes pixel patches generated by volume rendering or alpha-blending, smoothing pixel values via a refinement module with Plücker coordinates as a view consistency constraint. The subsequent mixing operation further rearranges the refined pixel patches by blending pixel values within sub-image planes, recovering finer details, particularly along scene edges, to enhance texture fidelity. The details are elaborated as follows.

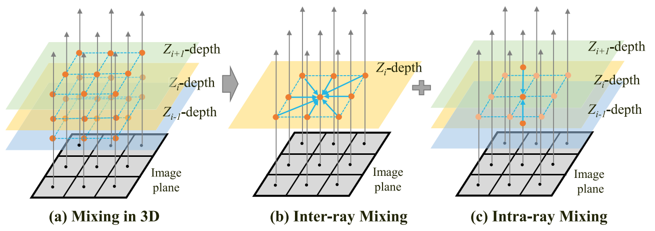

Fig. 3. *Two-stage mixing strategy.* We approximate the aggregation of spatial features in 3D by decomposing it into 2D (inter-ray) and 1D (intra-ray) operations, effectively reducing computational complexity. Inter-ray mixing aggregates features of points at the same sampling depth across adjacent rays, while intra-ray mixing focuses on features along each ray.

C. Radiance Fields Mixer (RF-Mixer)

The purpose of the RF-mixer is to aggregate densities and colors among neighboring points before volume rendering, as shown in Fig. 2. To achieve this, we consider combining adjacent $k \times k$ rays and neighboring k samples along each ray, forming a $k \times k \times k$ 3D local space. However, the computational complexity of direct 3D operations is prohibitively high. Therefore, we propose a two-stage mixing strategy: inter-ray mixing and intra-ray mixing, as illustrated in Fig. 3. This strategy decomposes the 3D operations into a combination of convolutional operations in 2D and 1D. Specifically, the 2D and 1D convolutional kernels can slide across the width/height dimensions and the depth dimension, which signifies inter-ray mixing and intra-ray mixing, respectively, of the density and color tensors.

Before delving into the details of the RF-mixer, we first outline the computational flow and the definition of the variables involved in it. Initially, a patch of pixels is randomly selected from a training image, represented as $\boldsymbol{C}_{gt} \in \mathbb{R}^{3\times P\times P}$. Tensors of positions $\boldsymbol{X} \in \mathbb{R}^{N\times D_x\times P\times P}$ and view directions $\boldsymbol{V} \in$

$\mathbb{R}^{N\times D_v\times P\times P}$ are obtained straightforwardly through sampling along the respective rays as [7], [11], [12]. Here, P denotes the patch size, and N indicates the number of samples along the ray. Subsequently, the sampled volume densities $\sigma \in \mathbb{R}^{N\times 1\times P\times P}$ and colors $\boldsymbol{c} \in \mathbb{R}^{N\times 3\times P\times P}$ are computed by feeding $\boldsymbol{X}$ and $\boldsymbol{V}$ into the radiance field pipeline [7], [11], [12]. Notably, the features used for decoding densities and colors are identified as $\boldsymbol{f}_\sigma \in \mathbb{R}^{N\times D_\sigma\times P\times P}$ and $\boldsymbol{f}_c \in \mathbb{R}^{N\times D_c\times P\times P}$, respectively. Unless specified otherwise, D_* represents the feature dimension of a particular tensor throughout the paper.

1) Inter-Ray Mixing of Densities and Colors: Inter-ray mixing aggregates the volume densities and colors of points sampled at the same depth across adjacent $k \times k$ rays. Specifically, with density features $\boldsymbol{f}_\sigma$ and color features $\boldsymbol{f}_c$, our 2D CNN-based weight predictors $\mathcal{G}_{2D}$ generate mixing weights $\boldsymbol{W}_\sigma \in \mathbb{R}^{N\times(k\times k)\times P\times P}$ for densities σ and $\boldsymbol{W}_c \in \mathbb{R}^{N\times(k\times k)\times P\times P}$ for colors $\boldsymbol{c}$, respectively. The receptive field of $\mathcal{G}_{2D}$ is strictly confined to $k \times k$, ensuring that during convolutions over $\boldsymbol{f}_\sigma$ and $\boldsymbol{f}_c$, each kernel aligns the densities and colors of neighboring $k \times k$ samples at the same depth within adjacent rays. To enable local mixing, densities σ and colors $\boldsymbol{c}$ are transformed into separate local blocks using 2D block extractors $\mathcal{E}_{2D}$: $\boldsymbol{B}_\sigma \in \mathbb{R}^{N\times(k\times k)\times 1\times P\times P}$ for densities and $\boldsymbol{B}_c \in \mathbb{R}^{N\times(k\times k)\times 3\times P\times P}$ for colors. The final inter-ray mixed densities $\sigma' \in \mathbb{R}^{N\times 1\times P\times P}$ and colors $\boldsymbol{c}' \in \mathbb{R}^{N\times 3\times P\times P}$ are then computed as

$$(\sigma',\ \boldsymbol{c}') = (sum\,(\boldsymbol{W}_\sigma \otimes \boldsymbol{B}_\sigma, 1)\,,\ sum\,(\boldsymbol{W}_c \otimes \boldsymbol{B}_c, 1))\,, \quad (5)$$

where $\otimes$ represents the element-wise multiplication, and $sum(\boldsymbol{T}, n)$ donates summation over the n^{th} dimension of tensor $\boldsymbol{T}$. The details of weight predictors and block extractors are explained in Section III-E.

2) Intra-Ray Mixing for Densities and Colors: This stage focuses on integrating mixed densities σ' and colors $\boldsymbol{c}'$ of k neighboring points along each ray. Differing slightly from Section III-C1, we redesign the weight predictors $\mathcal{G}_{1D}$ using 1D CNNs, where the kernels slide along the depth dimension. To adapt the data for 1D CNNs, we first reshape the features $\boldsymbol{f}_\sigma$: $\mathbb{R}^{N\times D_\sigma\times P\times P} \Rightarrow \mathbb{R}^{(P\times P)\times D_\sigma\times N}$ and $\boldsymbol{f}_c$: $\mathbb{R}^{N\times D_c\times P\times P} \Rightarrow \mathbb{R}^{(P\times P)\times D_c\times N}$. This reshaping ensures that the 1D convolutional kernels process features corresponding to neighboring points along the ray. Given $\boldsymbol{f}_\sigma$ and $\boldsymbol{f}_c$, the weight predictors $\mathcal{G}_{1D}$ produces the weights $\boldsymbol{W}'_\sigma \in \mathbb{R}^{(P\times P)\times k\times D_\sigma\times N}$ for σ' and $\boldsymbol{W}'_c \in \mathbb{R}^{(P\times P)\times k\times D_c\times N}$ for $\boldsymbol{c}'$. Similarly, local blocks are extracted using 1D block extractors $\mathcal{E}_{1D}$: $\boldsymbol{B}'_\sigma \in \mathbb{R}^{(P\times P)\times k\times 1\times N}$ for σ' and $\boldsymbol{B}'_c \in \mathbb{R}^{(P\times P)\times k\times 3\times N}$ for $\boldsymbol{c}'$. The mixing process is as

$$(\sigma'',\ \boldsymbol{c}'') = (sum\,(\boldsymbol{W}'_\sigma \otimes \boldsymbol{B}'_\sigma, 1)\,,\ sum\,(\boldsymbol{W}'_c \otimes \boldsymbol{B}'_c, 1))\,. \quad (6)$$

Here, σ'' and $\boldsymbol{c}''$ are volume densities and colors through two-stage mixing.

3) Supervision: After two-stage mixing, three sets of volume density and colors $\{\sigma, \boldsymbol{c}\}$, $\{\sigma', \boldsymbol{c}'\}$ and $\{\sigma'', \boldsymbol{c}''\}$ are obtained. Afterwards, we perform volume render on each set to predict pixel sets $\{\boldsymbol{C}, \boldsymbol{C}', \boldsymbol{C}''\}$. We train RF-mixer with the loss measuring the mean square error between the prediction and ground truth pixels over the sequence $\{\boldsymbol{C}, \boldsymbol{C}', \boldsymbol{C}''\}$. Given ground truth pixel $\boldsymbol{C}_{gt}$, the loss is defined as

$$\mathcal{L} = \frac{1}{P^2}\,(||\boldsymbol{C} - \boldsymbol{C}_{gt}||_2 + ||\boldsymbol{C}' - \boldsymbol{C}_{gt}||_2 + ||\boldsymbol{C}'' - \boldsymbol{C}_{gt}||_2)\,. \quad (7)$$

For frameworks that use hierarchical volume sampling, such as NeRF [7] and MipNeRF [11], we apply the RF-mixer exclusively at the fine stage, as the sampling density in the coarse stage is relatively sparse. This further demonstrates that our RF-mixer is not limited to uniform sampling and performs equally well with non-uniform sampling.

D. Color Domain Mixer (CD-Mixer)

The CD-mixer is designed to rearrange rendered pixels within the sub-image plane to recover fine textures and enhance smoothness in the synthesized image. The process involves two key steps: 1) refining the rendered pixels, and 2) mixing the refined pixels.

1) Refining Pixels With View Consistency: As shown in Fig. 4, two separate branches utilized to extract pixel features from the rendered pixels patch $\boldsymbol{C}_{vr}$ and view features from the camera rays, separately. The pixel feature extractor is a 4-layer 2D CNN with 3×3 kernels that predicts features of the same size as the input. The view feature branch ensures view consistency by establishing correspondences between rays and pixels. Specifically, we use Plücker coordinates $\boldsymbol{r} = (\boldsymbol{d},\ \boldsymbol{p} \times \boldsymbol{d})$ to represent a ray, where $\boldsymbol{p}$ is the camera position and $\boldsymbol{d}$ is the ray direction. A 4-layer 2D CNN with 1×1 kernels encodes view features from the rays. Subsequently, the extracted features are then concatenated and decoded into refined pixels $\boldsymbol{C}_{rf}$ using a 2D tiny CNN with 1×1 kernels. Throughout the process, all 1×1 kernels maintain view consistency by separating the rays.

2) Mixing Pixels Based on Kernel Weights: To predict mixing weights for locally rearranged pixels, we construct an error distribution map between the pixel patches of the rendered $\boldsymbol{C}_{vr} \in \mathbb{R}^{1\times 3\times P\times P}$ and the refined $\boldsymbol{C}_{rf} \in \mathbb{R}^{1\times 3\times P\times P}$. This map guides the 2D weight predictor $\mathcal{G}_{2D}$ to adaptively learn the weight contributions, particularly in high-frequency regions prone to reconstruction loss (see Section IV-E4 for details). The weight predictor decodes the mixing weights from the error map, which are then applied in a weighted sum with locally refined pixel blocks $\boldsymbol{B}_p \in \mathbb{R}^{3\times(k\times k)\times P\times P}$, extracted via a 2D block extractor $\mathcal{E}_{2D}$. The final mixed pixels $\boldsymbol{C}_m \in \mathbb{R}^{1\times 3\times P\times P}$ are

$$\boldsymbol{C}_m = sum\,(\boldsymbol{W}_p \otimes \boldsymbol{B}_p,\ \ 1)\,. \quad (8)$$

3) Supervision: We train CD-mixer using the following loss measuring the mean squared error (MSE) between the generated pixel patches $\{\boldsymbol{C}_{rf}, \boldsymbol{C}_m\}$ and the ground truth

$$\mathcal{L} = \frac{1}{P^2}\,(\lambda||\boldsymbol{C}_{rf} - \boldsymbol{C}_{gt}||_2 + (1-\lambda)||\boldsymbol{C}_m - \boldsymbol{C}_{gt}||_2)\,. \quad (9)$$

We set λ to 0.1in all our tests.

E. General Kernel-Based Mixing Strategy

As described in Section III-C and III-D, we reformulate local attribute aggregation for the RF-mixer and CD-mixer as

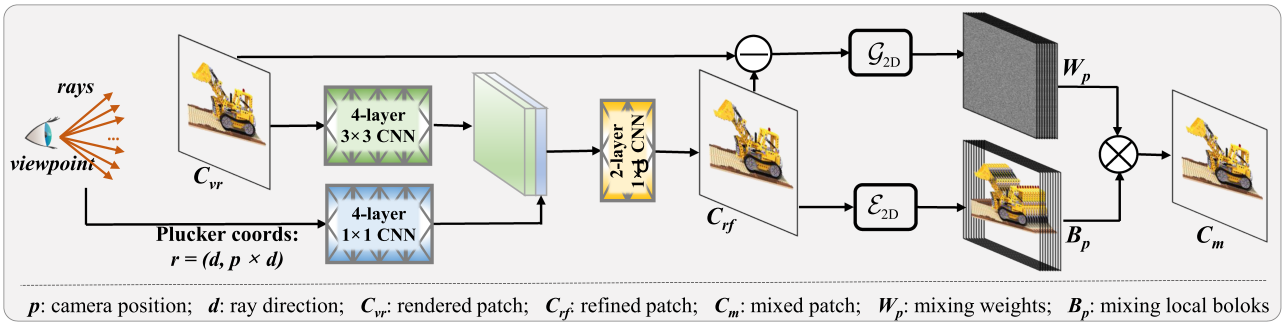
rays
viewpoint
C_{vr}
4-layer 3×3 CNN
4-layer 1×1 CNN
Plucker coords: r = (d, p × d)
2-layer 1×1 CNN
C_{rf}
$\mathcal{G}_{2D}$
$\mathcal{E}_{2D}$
W_p
B_p
C_m
p: camera position; d: ray direction; C_{vr}: rendered patch; C_{rf}: refined patch; C_m: mixed patch; W_p: mixing weights; B_p: mixing local boloks

Fig. 4. *Overview of CD-mixer.* Initially, we refine the rendered pixel patch $\boldsymbol{C}_{vr}$ by decoding concatenated features from the rendered pixel and the camera view. To ensure view consistency, we use Plucker coordinates to represent a ray $\boldsymbol{r} = (\boldsymbol{d}, \boldsymbol{p} \times \boldsymbol{d})$, establishing a correspondence with the pixel, where $\boldsymbol{p}$ and $\boldsymbol{d}$ are the camera position and ray direction, respectively. Subsequently, the weight predictor $\mathcal{G}_{2\mathrm{D}}$ utilizes the error map between the pixel patched of the rendered $\boldsymbol{C}_{vr}$ and refined $\boldsymbol{C}_{rf}$ to predict mixing weights $\boldsymbol{W}_p$. Meanwhile, the block extractor $\mathcal{E}_{2\mathrm{D}}$ converts the refined pixels into corresponding local blocks $\boldsymbol{B}_p$. The final mixed pixels are obtained through a weighted summation between $\boldsymbol{W}_p$ and $\boldsymbol{B}_p$.

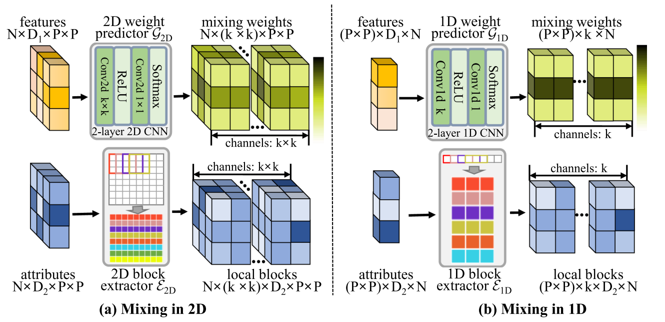

Fig. 5. *Computational flow of 2D/1D mixing.* (a) Mixing in 2D: Feature maps and attribute maps are processed in a spatially aware manner. A 2D block extractor segments attributes into local spatial blocks, and a 2D CNN predicts mixing weights across $k \times k$ neighborhoods for each spatial location. (b) Mixing in 1D: Features and attributes are flattened along the spatial dimension. A 1D block extractor segments the sequence, and a 1D CNN predicts mixing weights for each channel along the sequence. This design supports flexible integration of neighborhood information in either spatial or sequential domains.

a weighted sum of local attribute blocks and dynamically predicted weights. This approach involves two key parts: the weight predictor, which determines the contribution of neighboring samples to the source, and the block extractor, which gathers local samples as blocks for mixing.

1) Weight Predictor: As illustrated in Fig. 5, the weight predictors are compact two-layer CNNs with ReLU activation functions. In our setup, the 2D predictor $\mathcal{G}_{2\mathrm{D}}$ utilizes kernels of size $k \times k$, generating dynamically mixing weights also of dimension $k \times k$. This design allows $\mathcal{G}_{2\mathrm{D}}$ to effectively aggregate information from the $k \times k$ neighboring samples and compute each sample's contribution to the central sample during the mixing process, as detailed in Section III-C1 and III-D2. In a similar vein, the 1D predictors $\mathcal{G}_{1\mathrm{D}}$ use k-sized kernels, producing k weights that represent the contributions of neighboring samples for subsequent intra-mixing, as described in (used in Section III-C2). To ensure numerical stability and proper normalization, the $Softmax$ activation is applied to all predictor, ensuring that the weights of local samples sum to 1.

2) Block Extractor: To facilitate direct matrix operations between tensors representing attributes (such as density, color, and pixel) and their corresponding mixing weights, we utilize the PyTorch $unfold$ function to create block extractors. This method allows for efficient retrieval of local samples from the tensor without necessitating additional arithmetic calculations, as illustrated in Fig. 5. Specifically, the 2D extractors, denoted as, slide a $k \times k$ kernel across the entire attribute tensor. For each position of the kernel, the samples encompassed within its coverage are converted into a flattened column. All flattened columns are referred to as local blocks, encapsulating the local spatial context around each sample. Similarly, in the 1D case, the extractors $\mathcal{G}_{1\mathrm{D}}$ operate with kernels of size k, fetching local blocks consisting of k neighboring samples in each block.

Importantly, the value of k can be flexibly adjusted to control the size of the local region for integration.

IV. Experiments and Results

A. Experimental Settings

1) Dataset: We evaluate the performance of MixRF on NeRF-based methods through quantitative and qualitative evaluations using three widely recognized benchmarks for novel view synthesis: Realistic Synthetic 360°, Synthetic-NSVF, and DONeRF, a prevalent benchmark for multi-view reconstruction. The first two datasets contain path-traced images of objects with intricate geometry and realistic non-Lambertian materials. DONeRF has six complex scenes, featuring detailed textures with high-frequency components. Furthermore, we additionally utilize three more intricate datasets to evaluate MixRF on 3D-GS-based methods, namely MipNeRF360, Tanks & Temples (T&T) and Deep Blending (DB). MipNeRF360 includes nine scenes, each featuring complex central objects or areas and detailed backgrounds, representing the current state-of-the-art NeRF rendering quality. The T&T dataset comprises two handheld 360° captures of large-scale scenes, namely Truck and Train. The DB dataset comprises two scenes with distinct capture styles, including large bounded indoor scenes.

2) Baselines: To illustrate the versatility of our method as a broad framework applicable to nearly all radiance field-based approaches, we apply it to six representative methods: NeRF [7], MipNeRF [11], DVGO [9], instant-NGP (iNGP) [13], 3D-GS [8], and Scaffold-GS [72]. Notably, the initial four methods

TABLE I
QUANTITATIVE COMPARISON OF NERF-BASED METHODS

Methods	Realistic Synthetic 360° Dataset			DONeRF Dataset			Synthetic-NSVF Dataset		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF	31.01	0.943	0.079	29.94	0.861	0.271	31.04	0.951	0.067
Mix-NeRF	**(+1.14)** 32.15	**(+0.010)** 0.953	**(-0.018)** 0.061	**(+2.06)** 32.00	**(+0.040)** 0.901	**(-0.060)** 0.211	**(+2.16)** 33.20	**(+0.013)** 0.964	**(-0.019)** 0.048
MipNeRF	32.63	0.945	0.073	29.56	0.853	0.283	32.50	0.958	0.056
Mix-MipNeRF	**(+0.40)** 33.03	**(+0.015)** 0.960	**(-0.019)** 0.054	**(+1.84)** 31.40	**(+0.037)** 0.890	**(-0.055)** 0.228	**(+1.43)** 33.93	**(+0.011)** 0.969	**(-0.014)** 0.042
DVGO	32.80	0.955	0.054	30.30	0.883	0.233	35.14	0.976	0.032
Mix-DVGO	**(+0.76)** 33.56	**(+0.006)** 0.961	**(-0.008)** 0.046	**(+1.44)** 31.74	**(+0.018)** 0.901	**(-0.033)** 0.200	**(+0.82)** 35.96	**(+0.004)** 0.980	**(-0.005)** 0.027
iNGP	33.21	0.965	0.046	30.24	0.892	0.216	35.87	0.982	0.027
Mix-iNGP	**(+1.38)** 34.59	**(+0.005)** 0.970	**(-0.008)** 0.038	**(+1.85)** 32.09	**(+0.019)** 0.911	**(-0.037)** 0.179	**(+1.12)** 36.99	**(+0.003)** 0.985	**(-0.005)** 0.022

Our evaluation metrics are PSNR/SSIM (higher is better) and LPIPS [73] (lower is better). The mixed models exhibit substantial increments in overall metrics compared to their respective original frameworks. The increments are shown in **bold** inside the brackets.

are NeRF-based, whereas the latter two are 3D-GS-based. In particular, NeRF, as the pioneering neural radiance fields method, is enhanced by MipNeRF through the introduction of a multiscale representation for anti-aliasing radiance fields. DVGO and iNGP significantly boost both training and rendering speeds by leveraging voxel grids for scene reconstruction. Meanwhile, 3D-GS represents the most recent achievement with top-notch visual quality and training speed, while Scaffold-GS addresses the excessive memory consumption of 3D-GS without compromising performance. In our setup, we integrate both RF-mixer and CD-mixer modules into NeRF and MipNeRF, given that their sampled points and casting rays/cones are strictly independent. Conversely, only the CD-mixer is applied to DVGO, iNGP, 3D-GS, and Scaffold-GS due to their intrinsic capability to fuse spatial information. For a more straightforward comparison, we prefix the model names with Mix- to indicate methods that incorporate our mixing framework. In our comprehensive evaluation of our methods and various baselines, the rendering quality is evaluated by PSNR, SSIM [74], and LPIPS [73], which have been widely adopted by the novel view synthesis task.

3) Training Details: In all experiments involving RF-mixer, we set $P = 40$ and $k = 3$. The remaining training specifications such as learning rate, total training iterations, and so forth, strictly follow the original model setup. To train CD-mixer, we use the Adam optimizer [75] with a learning rate that begins at $5e - 4$ and decays to $5e - 6$ following a cosine schedule over the course of optimization. We set $k = 5$ and $P = 32$, with the batch size of 64 and the total iterations of 200 k. All training and evaluation procedures are performed on an NVIDIA GeForce RTX 3090 Graphics card.

B. Results on NeRF-Based Methods

1) Quantitative Comparisons: Table I illustrates that all baseline models show improved performance when augmented with MixRF. The improvement is noticeable across all metrics assessed, with a particularly notable enhancement seen in PSNR. The datasets reveal an average rise of $\mathbf{1.37}$, $\mathbf{1.80}$, and $\mathbf{1.38}$ in PSNR, highlighting the significant influence of MixRF and signaling a substantial advancement towards achieving more realistic synthesis. Upon examining the results, a remarkable performance on the DONeRF dataset becomes apparent, which is known for its intricate and complex textures. The results showcase an exceptional capability to extract high-frequency information and improve the smoothness of volume rendering by skillfully aggregating spatial context. We also noted that the Realistic Synthetic 360 and Synthetic-NSVF datasets, being object-centered, contain substantial areas of white background lacking rich texture details. These regions contribute minimally to the overall image quality assessment, leading to limited improvements in SSIM and LPIPS.

2) Qualitative Comparisons: In the realm of qualitative assessment, we conduct a rigorous comparison between the mixed and original models, as illustrated in Fig. 6. Notably, the mic exhibits apparent disparities among the original models, especially for NeRF, MipNeRF, and DVGO. These models exhibit a distinct blurriness in both structural and textural aspects. By contrast, mixed models excel in accurately rendering intricate details. The mic's overall structure is notably enhanced. Extending our analysis to a larger and more complex scene like the expansive $sanmiguel$ provides additional insights. NeRF and MipNeRF, in their original forms, fail to capture the depiction of water flowing from the fountain, thereby compromising their accuracy. DVGO and iNGP, on the other hand, struggle with artifacts that affect representation completeness.

C. Results on 3D-GS-Based Methods

1) Quantitative Comparisons: We evaluate our method using three datasets comparing against 3D-GS and Scaffold-GS. Table II demonstrates that integrating our MixRF into various 3D-GS-based methods enhances their performance on multiple benchmark datasets. For the 3D-GS method, there are average improvements in PSNR ($+\mathbf{0.52}$), SSIM ($+\mathbf{0.006}$), and LPIPS ($-\mathbf{0.005}$). The improvements over Scaffold-GS are $+\mathbf{0.28}$, $+\mathbf{0.011}$, and $-\mathbf{0.004}$, respectively. Particularly noteworthy is the substantial performance improvement observed on the MipNeRF360 and T&T datasets, mainly attributed to the complex

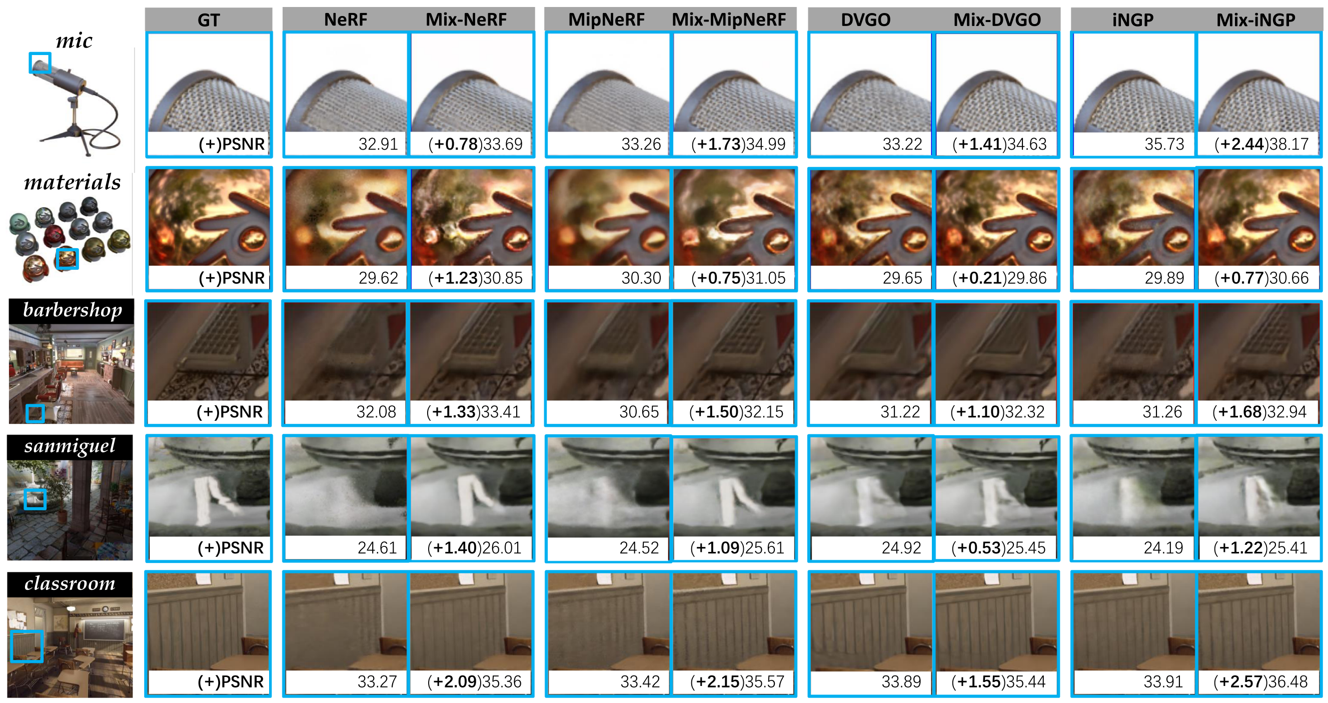

Fig. 6. *Qualitative comparison of the mixed and original NeRF-based methods.* We incorporate our method into the NeRF [7], MipNeRF [11], DVGO [9], and iNGP [13] frameworks, denoting them as Mix-NeRF, Mix-MipNeRF, Mix-DVGO, and Mix-iNGP, separately. Mixed models can more accurately capture fine details in both geometry and appearance, generating images that perceptually resemble the ground truth more closely than their respective original frameworks. The original models, especially NeRF and MipNeRF, struggle to capture clear and accurate textures and have trouble in representing delicate structures, resulting in ghosting artifacts from specular reflections on spherical materials and blurring of net structures. By contrast, our method reconstructs more details of the thin structure in the *mic* and presents a more realistic reflective effect for mirror materials. For more complex scenes *sanmiguel*, DVGO and iNGP miss details, including the incomplete fountain and floor patterns, while the mixed models successfully depict water columns on fountains. The increase in PSNR is emphasized in parentheses.

TABLE II
Quantitative Analysis of 3D-GS-Based Methods

Methods	MipNeRF360 Dataset			Tanks&Temples Dataset			Deep Blending Dataset		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
3D-GS	28.69	0.870	0.182	23.14	0.841	0.183	29.41	0.903	0.243
Mix-3D-GS	**(+0.62)** 29.31	**(+0.008)** 0.878	**(-0.010)** 0.172	**(+0.67)** 23.81	**(+0.007)** 0.848	**(-0.004)** 0.179	**(+0.25)** 29.66	**(+0.003)** 0.906	**(-0.000)** 0.243
Scaffold-GS	29.03	0.863	0.200	23.96	0.828	0.177	30.12	0.906	0.254
Mix-Scaffold-GS	**(+0.50)** 29.53	**(+0.010)** 0.873	**(-0.013)** 0.187	**(+0.21)** 24.17	**(+0.021)** 0.849	**(-0.001)** 0.176	**(+0.13)** 30.25	**(+0.001)** 0.907	**(-0.000)** 0.254

We use PSNR/SSIM (higher values are preferable) and LPIPS [73] (lower values are preferable) as the assessment criteria. The mixed models show significant improvements across all metrics compared to their original frameworks. Improvements in metrics compared to the original models are indicated in **bold** inside the brackets..

textures in the scene that align well with our mixing strategy. The degree of improvement on the DB dataset is relatively modest compared to the first two datasets. This is primarily due to the reason that most scenes in the DB dataset contain large texture-less regions of which our method cannot fully take advantage. However, we also observed that our method results in weaker performance improvements for 3D-GS-based frameworks compared to NeRF-based frameworks. We attribute this to the lack of error consistency in the synthesized images produced by 3D-GS methods, where errors are often concentrated in specific local regions (stemming from specific large Gaussian primitive). In contrast, NeRF errors are generally distributed more uniformly across the entire synthesized image. This makes CD-mixer challenging to learn a consistent aggregation pattern.

2) Qualitative Comparisons: We select representative scenes for qualitative comparison, as shown in Fig. 7. It can be seen that our MixRF significantly enhances the quality of view synthesis compared to 3D-GS-based approaches. Specifically, the mixed models accurately reproduce reflective regions in the scene, such as highlights on cutlery and metallic surfaces, in a more realistic manner. Furthermore, our method reconstructs more geometric structures, which can accurately preserve details such as stair railings and chair cushions, as highlighted by the green arrows. In comparison, both the 3D-GS and Scaffold-GS

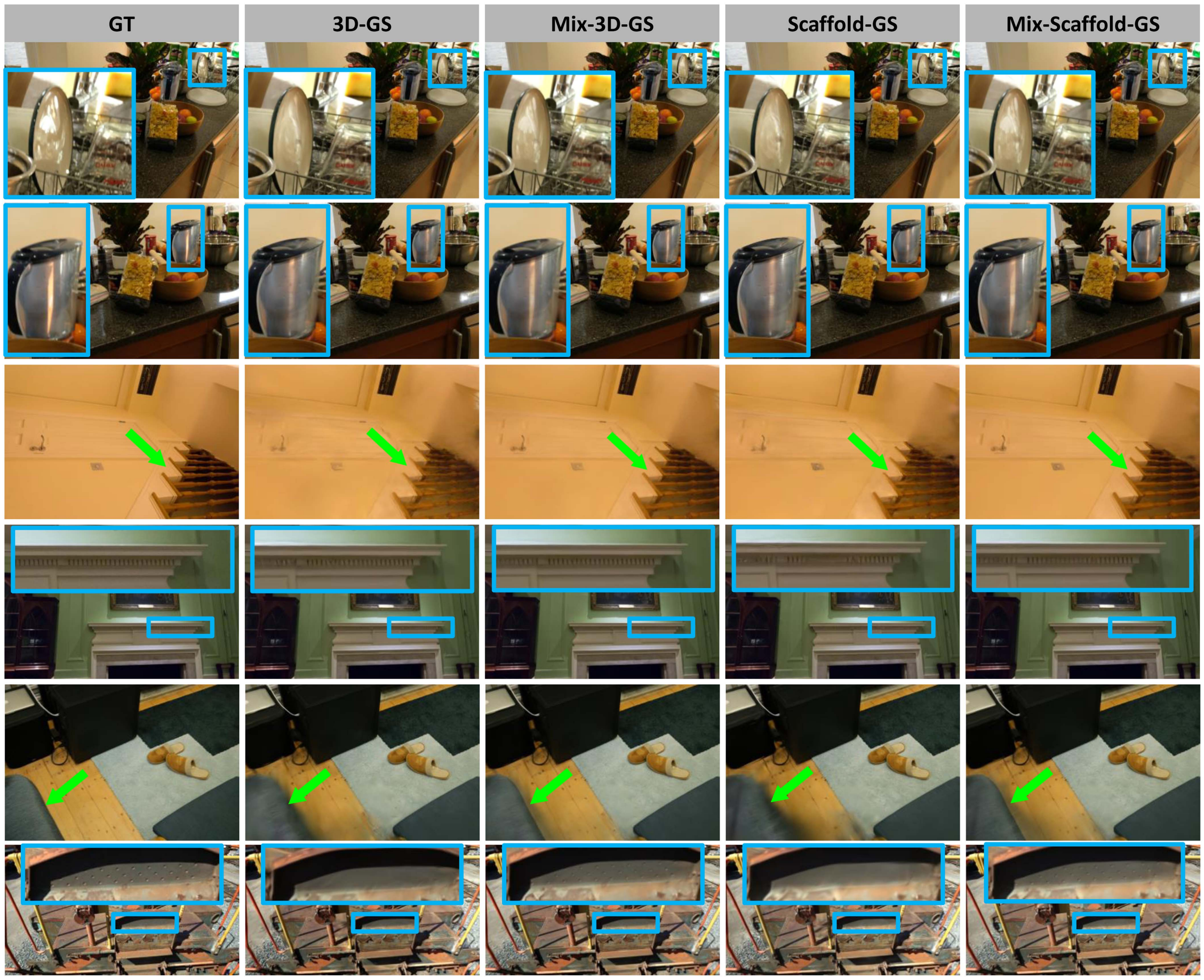

Fig. 7. *Qualitative comparison of mixed and original 3D-GS-based methods.* We integrate our MixRF into the 3D-GS [8] and Scaffold-GS [72] frameworks, dubbed as Mix-3D-GS and Mix-Scaffold-GS, respectively. Mixed models can capture more realistic scene appearance, such as the highlights on the surface of objects (first two rows). Additionally, mixed models can help compensate for the distortion in geometric reconstruction by the original models (indicated by the green arrows in the third and fifth rows). For more detailed context, the rendering results of the original model all exhibit a certain degree of blurriness, while mixed models faithfully reproduce synthetic results that are close to the ground truth (the fourth and last rows).

methods fail to recover these details, incurring severe blurring and distortion. Additionally, our method excels in reproducing regions with rich and fine details, such as the arrayed ventilation grilles and perforated metal sheets, with relatively accurate and realistic results. Conversely, 3D-GS-based methods struggle to effectively model such cases and tend to treat these regions as flat surfaces.

D. Additional Analysis

1) Contribution of RF-Mixer: The RF-mixer aims to aggregate local features, which can help radiance fields more accurately search for the scene's surfaces. To assess this, we compare meshes extracted from the original NeRF and Mix-NeRF models. As shown in Fig. 9, the meshes generated by Mix-NeRF demonstrate superior visual quality compared to NeRF. Mix-NeRF stands out in capturing complicated details and preserving more surface structures. In particular, the splash guard of the *Bike* scene shows improved geometric details, and most fragments in the *Lego* scene disappear. This suggests that Mix-NeRF has superior surface reconstruction capabilities, indicating that the mixed volume density distribution in the radiance field better aligns with the scene's surface. Our method emphasizes the beneficial incorporation of additional neighborhood information into NeRF, enhancing the scene's detailed representation and ultimately improving the visual quality of novel views.

2) Mixing Weights and Local Blocks: The core of our CD-mixer lies in obtaining the local blocks and their corresponding mixing weights. To verify the complementary effects among adjacent samples in the mixing process, we visualized these two components of the pre-trained CD-mixer on *Lego* scene with

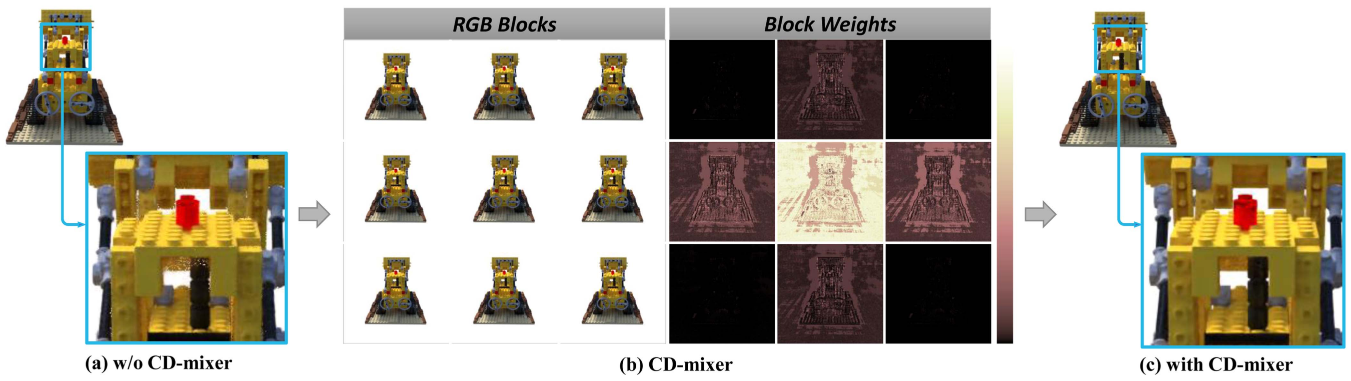

Fig. 8. *Visualization of local blocks and weights of the rendered image in CD-mixer with the kernel size of* 3×3. It is obvious that (a) the image produced through volume rendering appears grievously noisy and lacks smoothness. On the contrary, (b) the CD-mixer allows rendered pixels to rearrange from each other in sub-image space by weight learning, improving the overall quality and resulting in a smoother image (c). Additionally, the primary contribution comes from the source pixel itself, with its surrounding pixels also playing a complementary role, particularly in high-frequency regions.

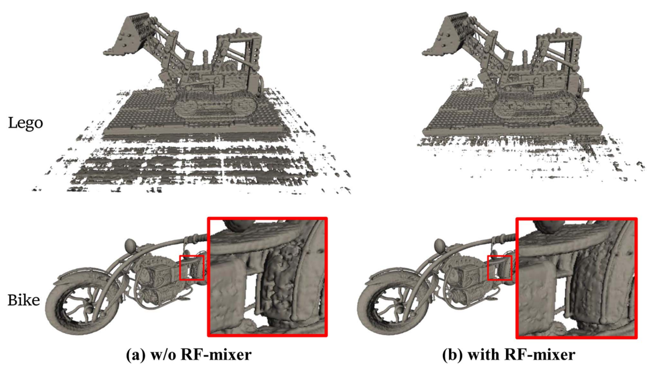

Fig. 9. *Contribution of RF-mixer to geometric optimization.* Visual comparison between the meshes extracted by the pre-trained NeRF and Mix-NeRF shows that RF-mixer helps radiance field-based models reconstruct a more detailed and precise surface.

the kernel size of $k = 3$. As shown in Fig. 8(b), the primary contribution of the mixed pixel comes from the source pixel itself, with some additional support from its surrounding pixels. This observation highlights the effectiveness of our mixing strategy in leveraging local information to improve reconstruction quality. Furthermore, we notice that the influence of neighboring pixels on the source is mainly concentrated in the contour and edge regions of the scene. These areas typically contain complicated details and are susceptible to noise, leading to synthesized images that lack smoothness. From Fig. 8(a), we observe a degree of randomness in pixel transitions near the edges. Nevertheless, after being processed by the CD-mixer module, as shown in Fig. 8(c) noise at the edges disappears, resulting in the synthesized image of more realistic visual quality.

3) Decomposition of 3D Operations Into 2D/1D: As depicted in Fig. 3, within a 3D subspace, our two-stage mixing strategy selectively aggregates spatial information for each sampling point by only considering neighboring points that are either at the same depth or along the same ray. Spatial information from regions outside these immediate neighborhoods is deliberately discarded. This focused approach not only simplifies the aggregation process but also ensures that the most relevant local context is preserved for each sampling point. As illustrated in Fig. 8(b), the contribution of neighboring samples to the source sample decreases as their relative offset increases during the mixing process. Therefore, we consider this approximation reasonable, as the discarded points are located farther from the source sample within the 3D subspace. By concentrating on closer, more relevant points, the method effectively captures essential local information while minimizing the influence of distant points, which are less likely to significantly impact the final result.

E. Ablation Study

1) Impact of Kernel Size on RF-Mixer: As described in Section III-E, our mixing strategy is controllable, allowing simultaneous adjustment of the kernel size k for the weights predictor and the block extractor to determine the number of local samples. We conduct experiments on the Realistic Synthetic 360° dataset to assess the impact of k on performance. Visualizations reveal that the synthesized image of the original NeRF presents minor noise (see Fig. 10 with $k = 1$). We attribute the noise to positional encoding (PE). PE aims to increase the model's sensitivity in high-frequency details by enabling significant prediction variations in the near region. However, the lack of local spatial aggregation also leads to noise in high-frequency areas. This is the issue that our MixRF aims to tackle. It shows that adding our RF-mixer with the kernel size of $k = 3$ leads to the best rendering quality. It is because the local spatial integration makes the prediction of colors and densities with local comprehension. Besides, setting k to 5 results in an unreasonable definition of the local space, posing challenges in predicting weights in the mixing process. This introduces ambiguity and ultimately leads to a decline in performance.

2) Impact of Kernel Size on CD-Mixer: In the process of pixel mixing on the image plane, the kernel size k determines the number of neighboring pixels that are used for mixing. We evaluate the performance of different values of k based on the 3D-GS model on MipNeRF360 and DB datasets. As shown in Fig. 11, it is evident that different values of k can improve

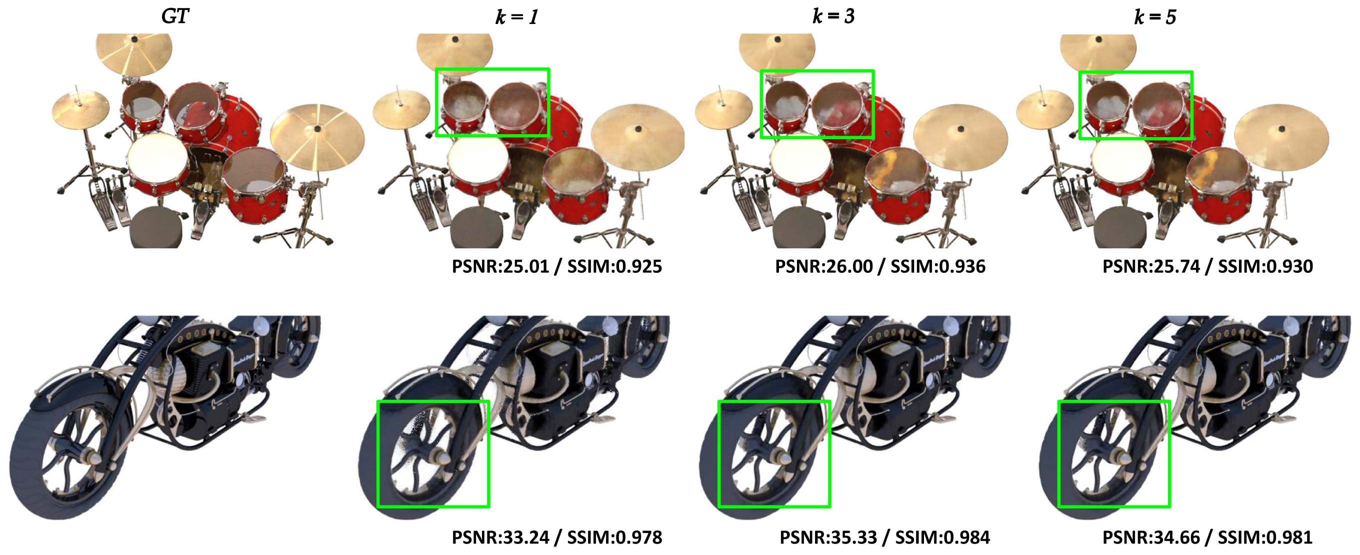

Fig. 10. *Effect of kernel size on the performance of RF-mixer.* The synthesized image of the original NeRF ($k = 1$) suffers from significant noise and blurriness. Setting k to 3 significantly improves the performance of the mixed model in terms of both visual aspects and metrics, allowing it to better learn the optimal scene representation and rectify geometric inaccuracies. When k is too large, the support region becomes too wide, resulting in uncertainty in weight learning and consequently, sub-optimal performance.

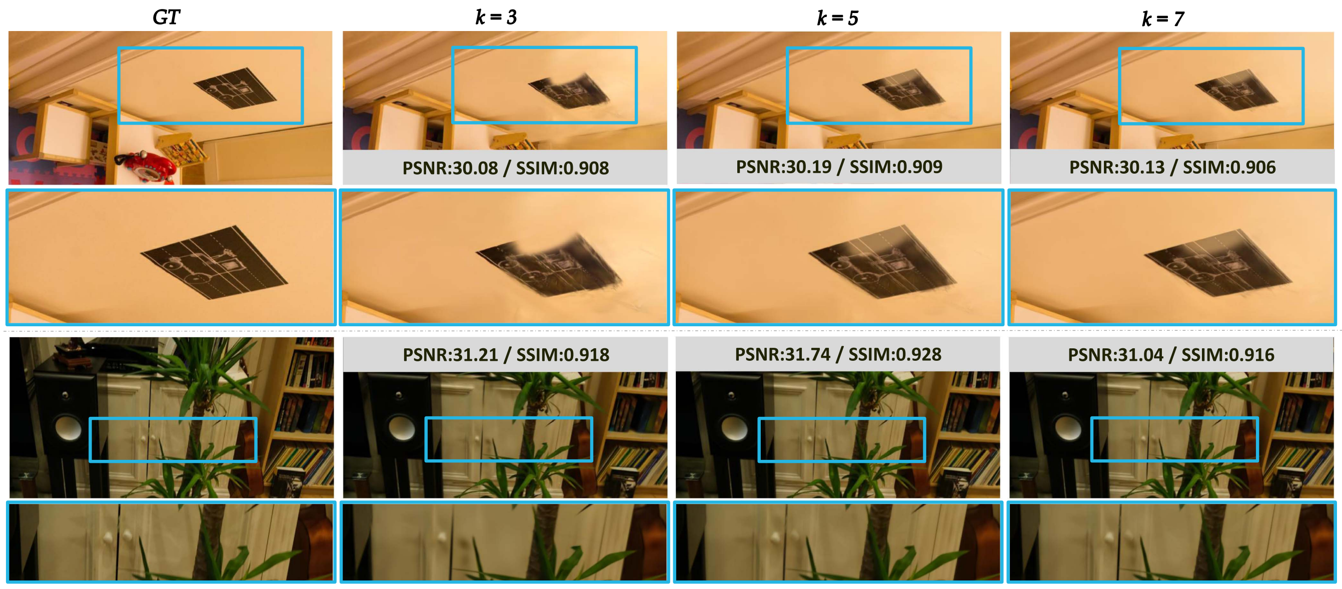

Fig. 11. *Effect of kernel size on the performance of CD-mixer.* Integrating CD-mixer with varying k values into the 3D-GS model enhances the quality of synthesized views to a certain degree. Through our experiments, setting k to 5 yields optimal performance. Excessive or insufficient kernel sizes can degrade the performance due to improperly sized local regions.

the performance of the original model to a certain extent in terms of visual quality and metrics. Particularly, $k = 5$ yields the optimal performance. The synthesized views exhibit clearer details and high-frequency textures as pixels borrow from and semantically complement each other on the local image plane, leading to higher-quality predictions. Nevertheless, the result of $k = 3$ is suboptimal, which we attribute to the inadequate aggregation of neighborhood information due to the improper size of local region. Additionally, we notice that setting a large local region can lead to a decrease in the synthesis quality, this is because pixels from distant areas not only have minimal impact on the source pixel, but also makes it harder to determine the appropriate weights during training.

3) Inter-Ray and Intra-Ray Mixing in RF-Mixer: The RF-mixer adopts a two-stage mixing strategy to aggregate both local geometric and appearance information from neighboring points along the same ray (intra-ray) as well as from sampling points at the same depth across different rays (inter-ray). As illustrated in Fig. 12, the original MipNeRF (Fig. 12(a)) struggles to accurately capture regions with complex spatial geometry, resulting

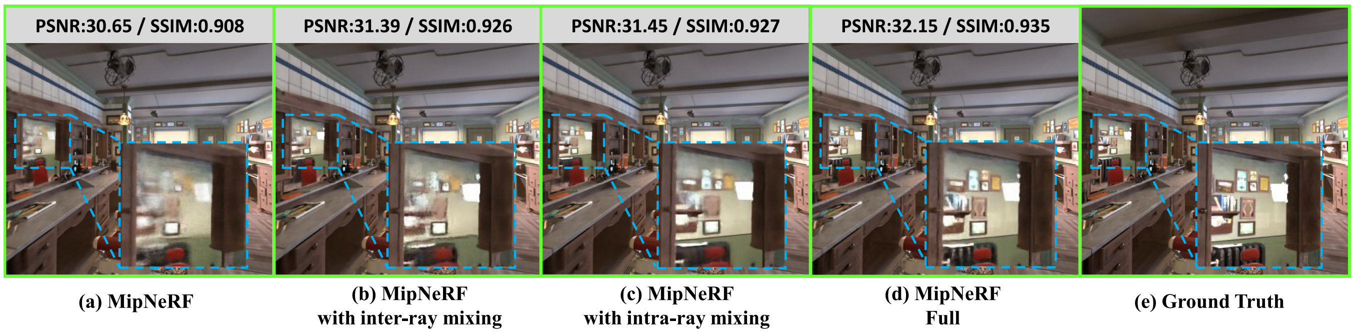

Fig. 12. *Ablation of inter-ray and intra-ray mixing.* (*a*) The rendering results produced by MipNeRF show significant blurring in the geometrically complex window region (highlighted by the blue dashed box). After aggregating information from nearby sampling points, both (*b*) and (*c*) partially recover the correct appearance and geometric details. In (*d*), the MipNeRF model combines the strengths of both (*b*) and (*c*), leading to renderings that more closely match the ground truth in (*e*).

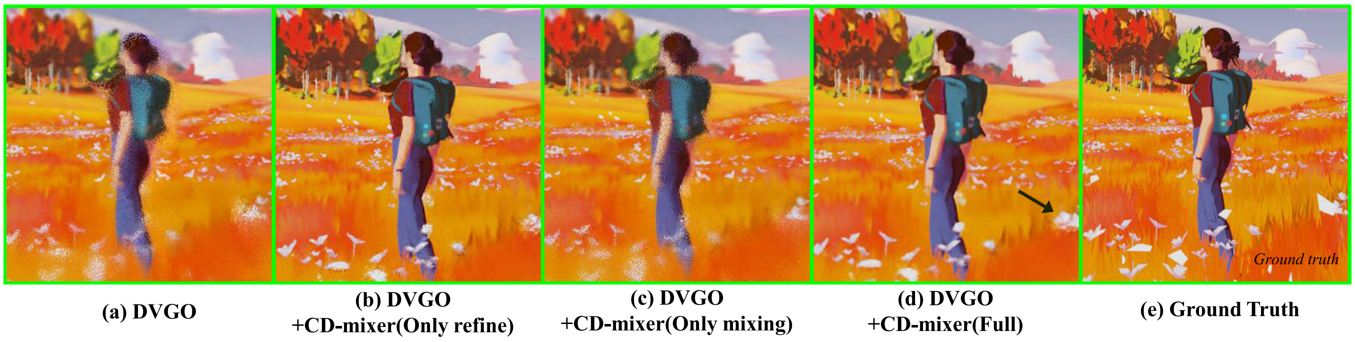
Ground truth
(a) DVGO
(b) DVGO +CD-mixer(Only refine)
(c) DVGO +CD-mixer(Only mixing)
(d) DVGO +CD-mixer(Full)
(e) Ground Truth

Fig. 13. *Ablation of the CD-mixer.* (*a*) The original DVGO-generated image exhibits significant noise and lacks smoothness due to insufficient pixel interaction. (*b*) The refining process effectively eliminates noise, resulting in a smoother output. (*c*) Directly mixing the rendered image enhances details, but cannot suppress noise well. (*d*) The DVGO model combines both (*b*) and (*c*), achieving noise reduction and smoothness enhancement in the synthetic image. It also focuses on enhancing high-frequency details, as indicated in the area marked by the black arrow, to make the rendering more closely resemble the ground truth (*e*).

in notable ambiguities. This issue is particularly evident in areas with rapid depth variations, such as the semi-transparent window (highlighted by the blue dashed box), where the synthesized images exhibit substantial blurring. The inter-ray (Fig. 12(b)) and intra-ray (Fig. 12(c)) mixing strategies alleviate these ambiguities by aggregating radiance values from neighboring sampling points, effectively enriching the source sample with additional geometric and appearance information. This approach enables a more accurate reconstruction of both the geometric structure and texture details in the rendered image. By integrating both mixing stages (Fig. 12(d)), the RF-mixer produces images with significantly improved fidelity to the ground truth (Fig. 12(e)). Consequently, the two-stage mixing not only reduces geometric inconsistencies in complex regions but also enhances overall visual quality, highlighting the method's effectiveness in challenging scenarios.

4) Refining and Mixing Pixels in CD-Mixer: The CD-mixer module starts by refining pixel patches and then mixes these refined patches among neighboring pixels to recover more detailed texture and smoothness in the new views. We deconstruct these procedures and assess their visual impact on the created views using the DVGO model. As shown in Fig. 13(a), pixels produced by volume rendering display independence, leading to sudden transitions among neighboring pixels with noticeable spots in the synthesized image. Furthermore, there is a distinct absence of fine texture details, resulting in a blurred appearance in the synthesized view and considerable image distortion. On the contrary, high-frequency details become more prominent in the image, and object edges appear smoother after pixel refining (Fig. 13(b)). When focusing solely on the mixing process, the details in the synthetic view show only marginal enhancement and noise is not well suppressed, as shown in Fig. 13(c). This failure stems from the use of rendered pixel patches instead of the error map (as elaborated in III-D2) to predict mixing weights. This simplification lacks guidance in the high-frequency region for the weights predictor, making it difficult to learn the mixing weights. Nonetheless, when combined with a comprehensive CD-mixer, the Mix-DVGO model reduces the noise and improves smoothness in the synthetic image. Simultaneously, it enhances image fidelity, particularly emphasizing high-frequency texture in regions indicated by arrows in Fig. 13(d).

5) Ablation of RF-Mixer and CD-Mixer: Our MixRF comprises two modules, RF-mixer and CD-mixer, designed to improve performance over the baseline. We isolate the two modules and conduct a series of ablation experiments to evaluate their individual contributions. By integrating these modules separately into the NeRF model under identical conditions, we observe notable differences. In Fig. 14(a), the original NeRF model exhibits considerable noise and loses smoothness, especially in

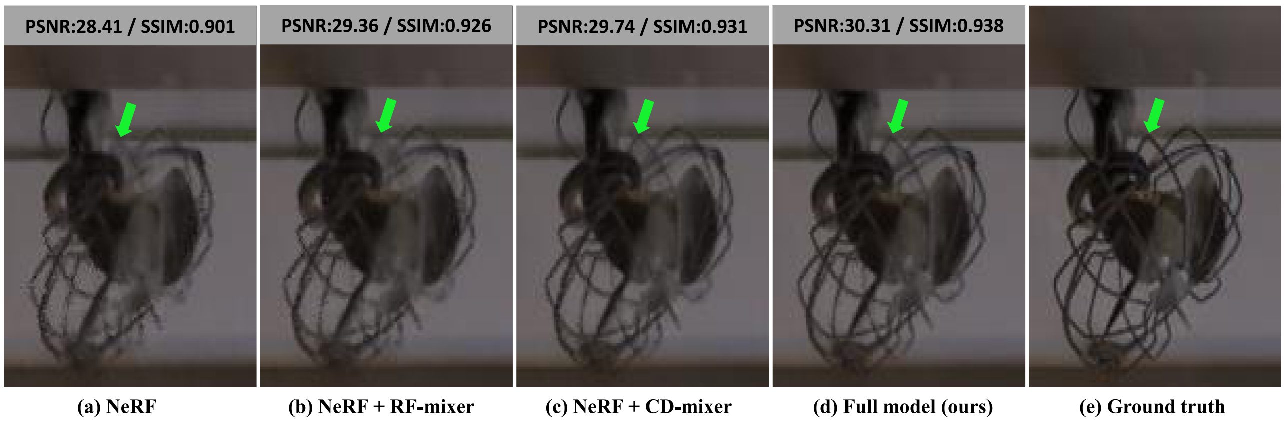

Fig. 14. *Ablation analysis of two mixers.* (**a**) The original NeRF introduces blurriness when reconstructing thin structures in the scene due to its independence of sampled points and casting rays, leading to noticeable noise. (**b**) The RF-mixer improves NeRF's capability to capture intricate details. However, noise-induced non-smoothness remains. (**c**) The CD-mixer enhances interactions of rendered pixels in the subspace, resulting in smoother synthetic outputs. (**d**) MixRF tackles the assumption of independence, leading to improved synthetic outcomes. This enhanced model is proficient in accurately reinstating intricate high-frequency elements within the scene.

TABLE III
COMPARISON OF PERFORMANCE IMPROVEMENT AND COMPUTATIONAL COST OF NERF-BASED MODELS WITH MIXRF INTEGRATION

Methods	PSNR ↑	SSIM ↑	LPIPS ↓	Time ↓		Additional cost ↓	
				Train (h)	Inference (s)	Train	Inference
NeRF	33.27	0.920	0.222	23.45	28.235	24.09%	24.27%
Mix-NeRF	**(+2.09)** 35.36	**(+0.022)** 0.942	**(-0.051)** 0.171	(+5.65) 29.10	(+6.853) 35.088		
MipNeRF	33.42	0.913	0.222	20.20	27.123	24.82%	16.71%
Mix-MipNeRF	**(+2.15)** 35.57	**(+0.032)** 0.945	**(-0.054)** 0.168	(+5.15) 25.35	(+4.533) 31.656		
DVGO	33.89	0.935	0.167	0.75	0.643	60.00%	1.40%
Mix-DVGO	**(+1.55)** 35.44	**(+0.009)** 0.944	**(-0.021)** 0.146	(+0.45) 1.20	(+0.009) 0.652		
iNGP	33.91	0.949	0.136	0.42	0.136	90.47%	9.76%
Mix-iNGP	**(+2.57)** 36.48	**(+0.012)** 0.961	**(-0.025)** 0.111	(+0.38) 0.80	(+0.008) 0.144		

Our MixRF can significantly improve all metrics of NeRF-based methods with a computational overhead of no more than 25% increasement. All metrics are tested on the *classroom* scene in DONeRF dataset. The magnitude of metric change is shown in parentheses, with improvements highlighted in **bold**.

TABLE IV
COMPARISON OF PERFORMANCE IMPROVEMENT AND COMPUTATIONAL COST OF 3D-GS-BASED MODELS WITH MIXRF INTEGRATION

Methods	PSNR ↑	SSIM ↑	LPIPS ↓	Train Time ↓	FPS ↑	Additional cost (Inference)
3D-GS	30.63	0.914	0.220	**$36min$**	109	22.02%
Mix-3D-GS	**(+1.09)** 31.72	**(+0.013)** 0.927	**(-0.023)** 0.197	(+45min) 81min	(-24) 95	
Scaffold-GS	31.39	0.925	0.202	**$45min$**	128	20.31%
Mix-Scaffold-GS	**(+0.86)** 32.25	**(+0.004)** 0.929	**(-0.015)** 0.187	(+43min) 88min	(-26) 102	

Our MixRF can significantly improve all metrics of 3D-GS-based methods with a computational overhead of no more than 25% increasement. All metrics are tested on the *room* scene in MipNeRF360 dataset. The magnitude of metric change is shown in parentheses, with improvements highlighted in **bold**.

detailed regions. Conversely, incorporating the RF-mixer into NeRF, as shown in Fig. 14(b), greatly enhances the model's ability to capture complex details like the fan, thereby improving its geometric perception and scene representation. For NeRF with CD-mixer, depicted in Fig. 14(c), we observe a marked reduction in noise in the synthesized image, resulting in an overall smoother output. This enhancement indicates that the CD-mixer optimizes image rendering at the sub-pixel level. Finally, combining both modules into the NeRF model, as illustrated in Fig. 14(d), yields a synthesized output closely resembling the ground truth in terms of structural accuracy and pixel smoothness.

6) Additional Computational Overhead: We evaluate the additional computational overhead introduced by integrating MixRF into baseline models. As shown in Tables III and IV, integrating both the RF-mixer and CD-mixer results in less than a 25% increase in inference overhead. By comparison, DVGO, iNGP, 3D-GS, and Scaffold-GS, which only integrate the CD-mixer, also maintain an inference overhead below 25% . The higher training overhead is mainly due to the CD-mixer being optimized over multiple batches, whereas inference requires only a single forward pass per image, resulting in a notable difference in efficiency. Given the significant performance improvements compared to the base models, this additional overhead is

TABLE V
WE REPORT THE EFFECT OF KERNEL SIZE K ON THE PERFORMANCE OF THE RF-MIXER AND CD-MIXER

RF-mixer (Test on the *Bike* scene in Synthetic-NSVF dataset)					
Kernel size	PSNR ↑	SSIM ↑	LPIPS ↓	Train time ↓	Infer. time ↓
k=1	33.24	0.978	0.040	**22.55**h	**26.432**s
k=3	**35.33**	**0.984**	**0.023**	27.23h	33.412s
k=5	34.66	0.981	0.031	29.50h	36.600s
CD-mixer (Test on the *room* scene in the MipNeRF360 dataset)					
Kernel size	PSNR ↑	SSIM ↑	LPIPS ↓	Train time ↓	FPS ↑
k=1	30.63	0.914	0.220	**51min**	**104**
k=3	31.21	0.918	0.192	86min	93
k=5	**31.74**	**0.928**	**0.187**	95min	90
k=7	31.04	0.916	0.190	98min	88

The best metrics are highlighted in **bold**.

acceptable. Additionally, Table V illustrates the effect of kernel size on computational efficiency. As expected, larger kernel sizes k negatively affect both training and inference efficiency. After balancing performance against efficiency, we determine that the optimal kernel sizes are $k = 3$ for the RF-mixer and $k = 5$ for the CD-mixer.

V. DISCUSSIONS AND LIMITATIONS

Experimental results demonstrate that our method significantly improves the performance of general neural rendering approaches by effectively aggregating local context, benefiting both NeRF-based and 3D Gaussian Splatting (3D-GS) methods. However, there are still some limitations to address. First, while the kernel-based mixing strategy simplifies the complexity of 3D operations, it introduces computational overhead due to the convolutional operations required for weight prediction. Second, although our RF-mixer can handle neural rendering models that utilize non-uniform sampling, it currently struggles with sampling techniques that produce an inconsistent number of samples per ray, such as adaptive sampling. Additionally, our method assumes that local spatial interactions can be effectively captured, which may not generalize well to highly dynamic or rapidly changing scenes. To overcome these limitations and extend the applicability of the method, further optimization of the mixing strategy and exploration of more advanced feature aggregation techniques are necessary.

VI. CONCLUSION

MixRF addresses limitations observed in NeRF- and 3D-GS-based methods, particularly the assumption of independence between point sampling and ray casting. This assumption can lead to synthesized images lacking fine details and smoothness. By tackling these challenges, MixRF significantly enhances the rendering quality of radiance field-based methods. Additionally, we introduce an efficient and controllable kernel-based mixing strategy that substantially improves performance both quantitatively and qualitatively, with less than a 25% increase of computational overhead during inference. Moreover, we suggest that our approach holds promise for broad applicability across various NeRF- and 3D-GS-based models, extending beyond those discussed in our paper.

REFERENCES

[1] Z. Zhu et al., “NICE-SLAM: Neural implicit scalable encoding for SLAM,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 12 786–12 796.

[2] H. Matsuki, R. Murai, P. H. Kelly, and A. J. Davison, “Gaussian splatting SLAM,” 2023, *arXiv:2312.06741*.

[3] M. Botsch, A. Hornung, M. Zwicker, and L. Kobbelt, “High-quality surface splatting on today’s GPUs,” in *Proc. Eurographics/IEEE VGTC Symp. Point-Based Graph.*, 2005, pp. 17–141.

[4] Y. Wang, I. Skorokhodov, and P. Wonka, “HF-NeuS: Improved surface reconstruction using high-frequency details,” in *PRoc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 1966–1978.

[5] T. Hu, X. Xu, S. Liu, and J. Jia, “Point2pix: Photo-realistic point cloud rendering via neural radiance fields,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 8349–8358.

[6] Q. Xu et al., “Point-NeRF: Point-based neural radiance fields,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 5438–5448.

[7] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, “NeRF: Representing scenes as neural radiance fields for view synthesis,” in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 405–421.

[8] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, “3D Gaussian splatting for real-time radiance field rendering,” *ACM Trans. Graph.*, vol. 42, no. 4, pp. 1–14, 2023.

[9] C. Sun, M. Sun, and H. Chen, “Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 5459–5469.

[10] C. Reiser et al., “MERF: Memory-efficient radiance fields for real-time view synthesis in unbounded scenes,” in *Proc. Conf. Special Int. Group Comput. Graph. Interactive Techn.*, 2023, pp. 1–12.

[11] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan, “Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields,” in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 5835–5844.

[12] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, “Mip-NeRF 360: Unbounded anti-aliased neural radiance fields,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 5470–5479.

[13] T. Müller, A. Evans, C. Schied, and A. Keller, “Instant neural graphics primitives with a multiresolution hash encoding,” *ACM Trans. Graph.*, vol. 41, no. 4, pp. 102:1–102:15, Jul. 2022, doi: 10.1145/3528223.3530127.

[14] L. Liu, J. Gu, K. Z. Lin, T.-S. Chua, and C. Theobalt, “Neural sparse voxel fields,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 15651–15663.

[15] X. Yang, D. Lu, Y. Li, C. Li, and C. Wang, “CeRF: Convolutional neural radiance fields for new view synthesis with derivatives of ray modeling,” 2023, *arXiv:2307.07125*.

[16] H. Luo et al., “Convolutional neural opacity radiance fields,” in *Proc. 2021 IEEE Int. Conf. Comput. Photography*, 2021, pp. 1–12.

[17] C. Wang, X. Wu, Y.-C. Guo, S.-H. Zhang, Y.-W. Tai, and S.-M. Hu, “NeRF-SR: High quality neural radiance fields using supersampling,” in *Proc. 30th ACM Int. Conf. Multimedia*, 2022, pp. 6445–6454.

[18] E. R. Chan et al., “Efficient geometry-aware 3D generative adversarial networks,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 16 123–16 133.

[19] J. L. Schonberger and J.-M. Frahm, “Structure-from-motion revisited,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 4104–4113.

[20] G. Kopanas, T. Leimkühler, G. Rainer, C. Jambon, and G. Drettakis, “Neural point catacaustics for novel-view synthesis of reflections,” *ACM Trans. Graph.*, vol. 41, no. 6, pp. 1–15, 2022.

[21] C. Buehler, M. Bosse, L. M. S. J. Gortler, and M. F. Cohen, “Unstructured lumigraph rendering,” in *Proc. Annu. Conf. Comput. Graph. Interactive Techn.*, 2002, pp. 425–432.

[22] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, “The lumigraph,” in *Proc. Annu. Conf. Comput. Graph. Interactive Techn.*, 1996, pp. 43–54.

[23] A. Davis, M. Levoy, and F. Durand, "Unstructured light fields," in *Proc. Conf. Comput. Graph. Forum*, Wiley Online Library, 2012, pp. 305–314.

[24] L. Shi, H. Hassanieh, A. Davis, D. Katabi, and F. Durand, "Light field reconstruction using sparsity in the continuous Fourier domain," *ACM Trans. Graph.*, vol. 34, no. 1, pp. 1–13, 2014.

[25] A. Levin and F. Durand, "Linear view synthesis using a dimensionality gap light field prior," in *Proc. 2010 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 1831–1838.

[26] M. Levoy and P. Hanrahan, "Light field rendering," in *Seminal Graphics Papers: Pushing the Boundaries*, vol. 2. New York, NY, USA: ACM, 2023, pp. 441–452.

[27] Q. Wang et al., "IBRNet: Learning multi-view image-based rendering," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 4690–4699.

[28] V. Sitzmann, M. Zollhöfer, and G. Wetzstein, "Scene representation networks: Continuous 3D-structure-aware neural scene representations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 1121–1132.

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

[30] J. Flynn et al., "DeepView: View synthesis with learned gradient descent," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2367–2376.

[31] J. Flynn, I. Neulander, J. Philbin, and N. Snavely, "DeepStereo: Learning to predict new views from the world's imagery," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 5515–5524.

[32] Z. Li, W. Xian, A. Davis, and N. Snavely, "Crowdsampling the plenoptic function," in *Proc. 16th Eur. Conf. Comput. Vis.*, Glasgow, UK, Springer, Aug. 23-28, 2020, pp. 178–196.

[33] J. Choi et al., "TMO: Textured mesh acquisition of objects with a mobile device by using differentiable rendering," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 16 674–16 684.

[34] H. Jiang et al., "Depth-NeuS: Neural implicit surfaces learning for multi-view reconstruction based on depth information optimization," 2023, *arXiv:2303.17088*.

[35] H. Li, X. Yang, H. Zhai, Y. Liu, H. Bao, and G. Zhang, "Vox-Surf: Voxel-based implicit surface representation," *IEEE Trans. Vis. Comput. Graph.*, vol. 30, no. 3, pp. 1743–1755, Mar. 2024.

[36] T. Wu et al., "Voxurf: Voxel-based efficient and accurate neural surface reconstruction," 2022, *arXiv:2208.12697*.

[37] H. Wu, A. Graikos, and D. Samaras, "S-VolSDF: Sparse multi-view stereo regularization of neural implicit surfaces," 2023, *arXiv:2303.17712*.

[38] W. Yifan, L. Rahmann, and O. Sorkine-Hornung, "Geometry-consistent neural shape representation with implicit displacement fields," 2021, *arXiv:2106.05187*.

[39] X. Huang, Y. Zhang, B. Ni, T. Li, K. Chen, and W. Zhang, "Boosting point clouds rendering via radiance mapping," in *Proc. AAAI Conf. Artif. Intell.*, 2023, pp. 953–961.

[40] K. Rematas et al., "Urban radiance fields," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 12 932–12 942.

[41] J. Ost, I. Laradji, A. Newell, Y. Bahat, and F. Heide, "Neural point light fields," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 18 419–18 429.

[42] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su, "TensoRF: Tensorial radiance fields," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2022, pp. 333–350.

[43] W. Hu et al., "Tri-MipRF: Tri-Mip representation for efficient anti-aliasing neural radiance fields," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 19774–19783.

[44] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, "Plenoxels: Radiance fields without neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 5501–5510.

[45] K. Deng, A. Liu, J.-Y. Zhu, and D. Ramanan, "Depth-supervised NeRF: Fewer views and faster training for free," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 12 882–12 891.

[46] R. Martin-Brualla, N. Radwan, M. S. M. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth, "NeRF in the wild: Neural radiance fields for unconstrained photo collections," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 7210–7219.

[47] B. Mildenhall, P. Hedman, R. Martin-Brualla, P. P. Srinivasan, and J. T. Barron, "NeRF in the dark: High dynamic range view synthesis from noisy raw images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 16190–16199.

[48] E. R. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein, "pi-GAN: Periodic implicit generative adversarial networks for 3D-aware image synthesis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 5799–5809.

[49] C.-H. Lin et al., "Magic3D: High-resolution text-to-3D content creation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 300–309.

[50] M. A. Groeber and M. A. Jackson, "Dream. 3D: A digital representation environment for the analysis of microstructure in 3D," *Integrating Mater. Manuf. Innov.*, vol. 3, pp. 56–72, 2014.

[51] Z. Liu, Y. Feng, M. J. Black, D. Nowrouzezahrai, L. Paull, and W. Liu, "MeshDiffusion: Score-based generative 3D mesh modeling," 2023, *arXiv:2303.08133*.

[52] A. Cao and J. Johnson, "HexPlane: A fast representation for dynamic scenes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 130–141.

[53] Y. Wang, Q. Han, M. Habermann, K. Daniilidis, C. Theobalt, and L. Liu, "NeuS2: Fast learning of neural implicit surfaces for multi-view reconstruction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 3295–3306.

[54] K. Park et al., "Nerfies: Deformable neural radiance fields," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 5865–5874.

[55] T. Xu and T. Harada, "Deforming radiance fields with cages," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2022, pp. 159–175.

[56] B. Attal et al., "Hyperreel: High-fidelity 6-DoF video with ray-conditioned sampling supplementary materials," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 16610–16620.

[57] B. Hu, J. Huang, Y. Liu, Y.-W. Tai, and C.-K. Tang, "NeRF-RPN: A general framework for object detection in NeRFs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 23 528–23 538.

[58] A. Mirzaei et al., "Spin-nerf: Multiview segmentation and perceptual inpainting with neural radiance fields," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 20 669–20 679.

[59] S. Li and Y. Pan, "Interactive geometry editing of neural radiance fields," 2023, *arXiv:2303.11537*.

[60] J. Sun et al., "Neural 3D reconstruction in the wild," in *Proc. ACM SIGGRAPH 2022 Conf.*, 2022, pp. 1–9.

[61] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, "iMAP: Implicit mapping and positioning in real-time," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 6229–6238.

[62] Y. Chen et al., "GaussianEditor: Swift and controllable 3D editing with Gaussian splatting," 2023, *arXiv:2311.14521*.

[63] J. Cen et al., "Segment any 3D gaussians," 2023, *arXiv:2312.00860*.

[64] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 6840–6851.

[65] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," 2020, *arXiv: 2011.13456*.

[66] X. Li, H. Wang, and K.-K. Tseng, "GaussianDiffusion: 3D Gaussian splatting for denoising diffusion probabilistic models with structured noise," 2023, *arXiv:2311.11221*.

[67] J. Tang, J. Ren, H. Zhou, Z. Liu, and G. Zeng, "DreamGaussian: Generative gaussian splatting for efficient 3D content creation," 2023, *arXiv:2309.16653*.

[68] G. Wu et al., "4D Gaussian splatting for real-time dynamic scene rendering," 2023, *arXiv:2310.08528*.

[69] Z. Yang, H. Yang, Z. Pan, X. Zhu, and L. Zhang, "Real-time photorealistic dynamic scene representation and rendering with 4D gaussian splatting," 2023, *arXiv:2310.10642*.

[70] M. Niemeyer, J. T. Barron, B. Mildenhall, M. S. Sajjadi, A. Geiger, and N. Radwan, "RegNeRF: Regularizing neural radiance fields for view synthesis from sparse inputs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 5480–5490.

[71] Q. Li, F. Li, J. Guo, and Y. Guo, "UHDNeRF: Ultra-high-definition neural radiance fields," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 23 097–23 108.

[72] T. Lu et al., "Scaffold-GS: Structured 3D Gaussians for view-adaptive rendering," 2023, *arXiv:2312.00109*.

[73] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 586–595.

[74] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[75] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

Haiyang Bai received the BEng degree from Northwestern Polytechnical University in 2018 and the master's degree from Xi'an Jiaotong University in 2021. He is currently working toward the doctor's degree with the Department of Computer Science and Technology, Nanjing University. His research interests include computer vision and computer graphics.

Tao Lu received the BEng degree from Southwest Jiaotong University in 2016 and doctor degree from Nanjing University in 2023. He currently works with Shanghai AI Lab. His research interests include 3D computer vision, neural scene representation and reconstruction, and generative models.

Jiaqi Zhu is currently working toward the graduate degree in computer science and technology with Nanjing University, China. He is currently engaged in research with the Meta Graphics & 3D Vision Lab. His primary interests lie in novel view synthesis and 3D reconstruction, focusing on developing innovative solutions in these advanced areas of computer vision.

Wei Huang is currently working toward the master's degree with the Department of Computer Science and Technology, Nanjing University. Her research interests include computer vision, 3D reconstruction, and deep learning.

Chang Gou (Graduate Student Member, IEEE) received the BEng degree from Penn State University in 2015 and the master's degree from UC Davis in 2017. He is currently working toward the doctoral degree with the Department of Computer Science and Technology, Nanjing University. His research interests include computer vision and computer graphics.

Jie Guo received the bachelor's and PhD degrees in computer science from Nanjing University, in 2008 and 2013, respectively. Currently, he is an assistant researcher with State Key Laboratory for Novel Software Technology, Nanjing University. He was a research intern with Microsoft Research Asia. His research interests include appearance modeling, real-time rendering, and virtual reality.

Lijun Chen received the BS degree in electrical engineering from the Xi'an University of Science and Technology, China, in 1982, and the MS and PhD degrees in automatic control from the China University of Mining and Technology, China, in 1993 and 1998, respectively. He was a postdoctoral fellow with Nanjing University, China, from 1998 to 2000, and Michigan State University, USA, from 2001 to 2002, and a visiting scholar with The Hong Kong Polytechnic University in 2007. His current research interests include distributed computing and ubiquitous networks.

Yanwen Guo received the PhD degree in applied mathematics from the State Key Lab of CAD&CG, Zhejiang University, China, in 2006. He is currently a professor with the National Key Lab for Novel Software Technology, Department of Computer Science and Technology, Nanjing University, Jiangsu, China. He worked as a visiting professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, in 2006 and 2009, respectively, and the Department of Computer Science, The University of Hong Kong, in 2008, 2012, and 2013, respectively. He was a visiting scholar with the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, from 2013 to 2015. His research interests include image and video processing, vision, and computer graphics.