Rip-NeRF: Anti-aliasing Radiance Fields with Ripmap-Encoded Platonic Solids

Junchen Liu*
Beihang University
Beijing, China
liujunchen0214@gmail.com

Wenbo Hu*
Tencent AI Lab
Shenzhen, China
wbhu@tencent.com

Zhuo Yang*
Beijing Institute of Technology
Beijing, China
zhuoyang@bit.edu.cn

Jianteng Chen
Beijing Institute of Technology
Beijing, China
chenjiantengx@gmail.com

Guoliang Wang
Tsinghua University
Beijing, China
wanggl199705@gmail.com

Xiaoxue Chen
Tsinghua University
Beijing, China
chenxx21@mails.tsinghua.edu.cn

Yantong Cai
Dermatology Hospital
Guangzhou, China
Southern Medical University
Guangzhou, China
yangtcai@gmail.com

Huan-ang Gao
Tsinghua University
Beijing, China
gha20@mails.tsinghua.edu.cn

Hao Zhao†
Tsinghua University
Beijing, China
zhaohao@air.tsinghua.edu.cn

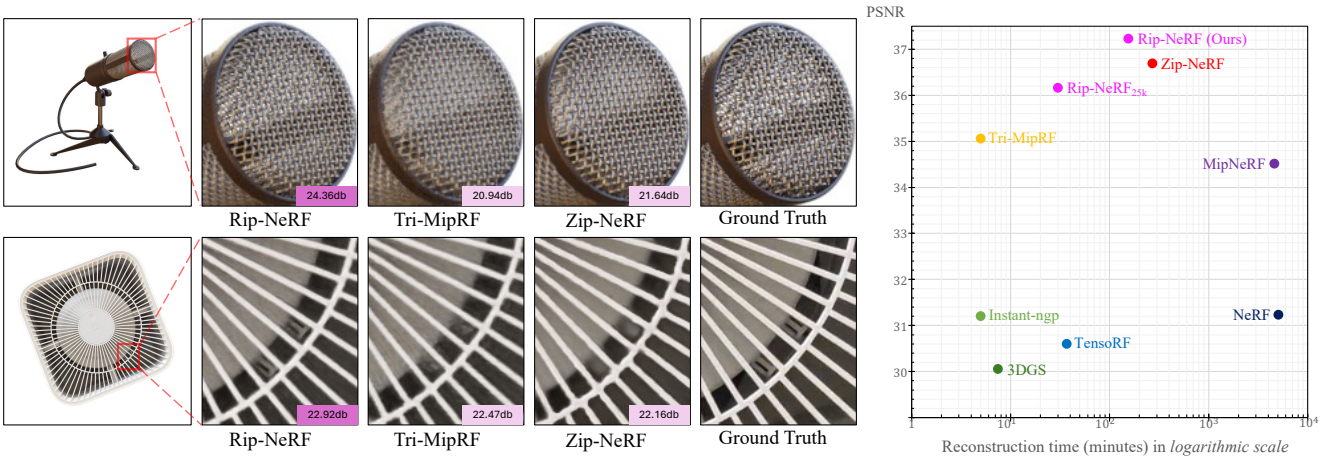


Figure 1: Qualitative and quantitative results of our Rip-NeRF and several representative baseline methods, *e.g.*, Zip-NeRF [Barron et al. 2023], Tri-MipRF [Hu et al. 2023], etc. Rip-NeRF$_{25k}$ is a variant of Rip-NeRF that reduces the training iterations from $120k$ to $25k$ for better efficiency. The first and second rows in the left panel are results from the multi-scale Blender dataset [Barron et al. 2021] and our newly captured real-world dataset, respectively. Our Rip-NeRF can render high-fidelity and anti-aliasing images from novel viewpoints while maintaining efficiency.

ABSTRACT

Despite significant advancements in Neural Radiance Fields (NeRFs), the renderings may still suffer from aliasing and blurring artifacts, since it remains a fundamental challenge to effectively and efficiently characterize *anisotropic areas* induced by the cone-casting procedure. This paper introduces a *Ripmap-Encoded Platonic Solid* representation to precisely and efficiently featurize 3D *anisotropic* areas, achieving high-fidelity anti-aliased renderings. Central to our approach are two key components: *Platonic Solid Projection* and *Ripmap encoding*. The Platonic Solid Projection factorizes the 3D space onto the unparalleled faces of a certain Platonic solid, such that the anisotropic 3D areas can be projected onto planes with distinguishable characterization. Meanwhile, each face of the Platonic solid is encoded by the Ripmap encoding, which is constructed by anisotropically pre-filtering a learnable feature grid, to enable featurzing the projected anisotropic areas both precisely and

*Equal Contribution
†Corresponding Author

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

SIGGRAPH Conference Papers '24, July 27-August 1, 2024, Denver, CO, USA

© 2024 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-0525-0/24/07.
https://doi.org/10.1145/3641519.3657402

efficiently by the anisotropic area-sampling. Extensive experiments on both well-established synthetic datasets and a newly captured real-world dataset demonstrate that our Rip-NeRF attains state-of-the-art rendering quality, particularly excelling in the fine details of repetitive structures and textures, while maintaining relatively swift training times, as shown in Fig.1. The source code and data for this paper are at https://github.com/JunchenLiu77/Rip-NeRF.

CCS CONCEPTS

• **Computing methodologies** → **Reconstruction**; **Hierarchical representations**.

KEYWORDS

novel view synthesis, radiance fields, anti-aliasing, anisotropic area-sampling

ACM Reference Format:
Junchen Liu, Wenbo Hu, Zhuo Yang, Jianteng Chen, Guoliang Wang, Xiaoxue Chen, Yantong Cai, Huan-ang Gao, and Hao Zhao. 2024. Rip-NeRF: Anti-aliasing Radiance Fields with Ripmap-Encoded Platonic Solids. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers '24 (SIGGRAPH Conference Papers '24), July 27-August 1, 2024, Denver, CO, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3641519.3657402

1 INTRODUCTION

The epoch-making Neural Radiance Fields (NeRFs) [Mildenhall et al. 2020] employ a neural network to represent the scene as a continuous 5D function, which is defined as the radiance and density along a ray at a certain 3D location and direction. They have promoted significant progress in numerous tasks, *e.g.*, novel view synthesis [Chen et al. 2022; Martin-Brualla et al. 2021; Yuan and Zhao 2023], geometry reconstruction [Wang et al. 2021; Chen et al. 2023a], content generation [Chan et al. 2022; Peng et al. 2023], simulation [Wu et al. 2023a; Wei et al. 2024] and automation [Zhu et al. 2022; Park et al. 2023a; Zhu et al. 2023; Zhou et al. 2024].

The aliasing artifacts remain a challenging problem in NeRFs, which are caused by the discrete sampling of the continuous physical world. As a pioneer, Mip-NeRF [Barron et al. 2021] proposed an analytical integrated positional encoding for the purely implicit representation to facilitate anti-aliasing rendering. However, both the training speed and rendering quality of it are limited by the purely implicit representation. To this end, Zip-NeRF [Barron et al. 2023] and Tri-MipRF [Hu et al. 2023] proposed anti-aliasing mechanisms based on the hybrid representation in the form of multi-sampling and area-sampling (*a.k.a.*pre-filtering), respectively. Nevertheless, multi-sampling inherently demands a large number of samples to featurize a single area, which puts it in a dilemma between the rendering quality and computational overhead. On the other hand, area-sampling of Tri-MipRF is more efficient as it can directly featurize a sub-volume. However, its isotropic mechanism significantly limits the ability to represent anisotropic areas that are ubiquitously induced by cone casting methods [Barron et al. 2022; Hu et al. 2023] in the volume rendering. As shown in Fig. 2 (a), the isotropic area-sampling cannot differentiate the anisotropic areas

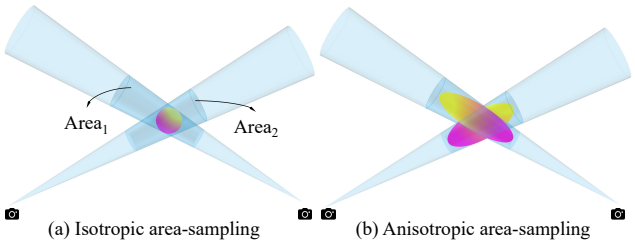


Figure 2: The two anisotropic areas Area$_1$ and Area$_2$ from different cones are ambiguously mapped to the same sampling area under the isotropic area-sampling (a), while are distinguishable under anisotropic area-sampling (b).

from different cones, which leads to ambiguities in the representation. Consequently, as shown in Fig. 1, blurriness on the surface of the microphone can be observed under this isotropic area-sampling.

In this paper, we propose a Ripmap-encoded Platonic solids representation, termed *Rip-NeRF*, for high-fidelity anti-aliased neural radiance fields. It enables featurizing 3D areas more precisely with various shapes using only one sample per area, such that anisotropic areas from different cones are distinguishable under our representation, as shown in Fig. 2 (b). The key to achieving this lies in two techniques, *i.e.*, the *Platonic Solid Projection* and the *Ripmap Encoding*. On one hand, the Platonic Solid Projection is a 3D space factorization method, which projects 3D areas onto the unparalleled faces of Platonic solids. By doing so, we can precisely represent 3D scenes with 2D feature grids, rather than 3D volumes, and the memory consumption is significantly reduced from $O(n^3)$ to $O(n^2)$. Note that, the orthogonal tri-plane representation adopted in [Chan et al. 2022; Hu et al. 2023] can be derived from a regular hexahedron (cube) in this perspective. On the other hand, the Ripmap Encoding can featurize the projected anisotropic 2D areas with a *Ripmap* [McReynolds et al. 1998] (*a.k.a.*anisotropic Mipmap), which is a feature grid pre-filtered with anisotropic kernels to represent the face of the Platonic solid. Compared with the tri-plane factorization and isotropic area-sampling (Mipmap) of Tri-MipRF [Hu et al. 2023], our Platonic Solid Projection together with the Ripmap Encoding enables more precisely featurizing anisotropic 3D areas in an efficient pre-filtering manner, such that sharper and more accurate details on the repetitive patterns of microphone can be reconstructed and rendered as shown in Fig. 1.

To evaluate the effectiveness of our Rip-NeRF, we conduct extensive experiments on both well-established public benchmarks and a newly captured real-world dataset, where the quantitative and qualitative results reveal that our Rip-NeRF achieves state-of-the-art rendering quality while maintaining efficient reconstruction. Besides, the ablation studies also demonstrate the effectiveness of our individual proposed components, *i.e.*the Platonic Solid Projection and the Ripmap Encoding. Furthermore, our Platonic Solid Projection introduces a flexible trade-off between rendering quality and efficiency, *e.g.*, training time and GPU memory consumption, by selecting different Platonic solids with a certain number of faces. Our contributions are summarized below.

- We propose a 3D space factorization method, *Platonic Solid Projection*, to represent a 3D scene with the 2D faces of a Platonic solid, such that the anisotropic 3D areas can be projected onto planes with distinguishable characterization.

- We propose to represent the faces of a Platonic solid by *Ripmap Encoding*, such that the projected anisotropic 2D areas can be precisely and efficiently featurized by the anisotropic area-sampling.
- Our *Rip-NeRF* achieves state-of-the-art rendering quality on both the public benchmarks and a newly captured real-world dataset while maintaining efficient reconstruction. And it enables a flexible trade-off between quality and efficiency.

2 RELATED WORK

2.1 Hybrid Representations in Neural Rendering

With the rise of deep learning, neural rendering, especially neural radiance fields (NeRF) [Mildenhall et al. 2020], has drawn increasing attention in novel view synthesis [Tancik et al. 2023; Takikawa et al. 2022; Bemana et al. 2022; Karnewar et al. 2022; Park et al. 2023b; Zhang et al. 2022a; Belhe et al. 2023; Li et al. 2023], avatar reconstruction [Xu et al. 2023; Wang et al. 2022; Duan et al. 2023; Kirschstein et al. 2023; Işık et al. 2023; Zheng et al. 2023; Dong et al. 2023; Jiang et al. 2023; Lin et al. 2023; Trevithick et al. 2023], reconstruction from sparse views [Somraj and Soundararajan 2023; Somraj et al. 2023; Zhang et al. 2022b; Lao et al. 2024], reconstruction of large-scale scenes [Wu et al. 2023d], scene editing [Gong et al. 2023; Huang et al. 2023; Zhuang et al. 2023; Zeng et al. 2023; Wu et al. 2023c,b; Jiang et al. 2022], and dynamic scene reconstruction [Lin et al. 2022; Park et al. 2021]. NeRF methods typically rely on neural networks to act as continuous representations. However, the pure implicit representations are computationally intensive and hard to represent high-frequency details.

On the other hand, recent works explored representing 3D scenes with explicit data structures, *e.g.*, octrees [Yu et al. 2021; Shu et al. 2023], sparse voxels [Fridovich-Keil et al. 2022], and VDB [Yan et al. 2023]. Nevertheless, explicit representations often suffer from large storage footprints and low rendering quality. To this end, hybrid representations, combining a tiny MLP and explicit data structures, like hash table [Müller et al. 2022], tri-plane [Chan et al. 2022], and Vector-Matrix [Chen et al. 2022], have emerged to improve rendering quality and efficiency [Reiser et al. 2023; Duckworth et al. 2023; Gupta et al. 2023; Chen et al. 2023b]. But these representations still suffer from aliasing artifacts, due to the ray casting procedure that discretely samples a continuous signal.

2.2 Anti-Aliasing in Neural Radiance Fields

Essentially, aliasing occurs as the overlapping frequency components due to insufficient sampling rates. Therefore, to alleviate this issue, we can directly increase the sampling rate by multi-sampling, or appropriately decrease the frequency of the scene by pre-filtering (*a.k.a.*area-sampling). In the neural radiance fields (NeRF) [Mildenhall et al. 2020] context, Mip-NeRF [Barron et al. 2021] pioneered the anti-aliasing for NeRF by the integrated positional encoding that enables area-sampling. Mip-NeRF 360 [Barron et al. 2022] further explored the anti-aliasing for unbounded scenes to improve the applicability. However, both the training and rendering of them are computationally intensive due to their pure implicit representation.

Recently, Zip-NeRF [Barron et al. 2023] presented a multi-sampling strategy to enable anti-aliasing for more efficient hybrid grid-based representation [Müller et al. 2022]. However, multi-sampling inherently demands many samples to featurize a single area, which puts it in a dilemma between the rendering quality and computational overhead. Conversely, Tri-MipRF [Hu et al. 2023] proposed an area-sampling strategy that models the scene as three orthogonal mipmaps, benefiting from the efficiency and compactness of the hybrid plane-based representation [Chan et al. 2022]. Nevertheless, its isotropic mechanism significantly limits the ability to represent anisotropic areas induced by cone casting. In contrast, our method not only enables precisely featurizing anisotropic 3D areas in an efficient area-sampling manner but also maintains the compactness of the hybrid plane-based representation.

3 METHOD

3.1 Overview

Given a set of calibrated multi-view images, our goal is to render *high-fidelity anti-aliasing* images in a NeRF [Mildenhall et al. 2020] fashion. Following [Müller et al. 2022; Barron et al. 2023; Hu et al. 2023; Chen et al. 2022], our Rip-NeRF adopts a hybrid representation, to benefit from both the efficiency and flexibility of explicit and implicit representations. As shown in Fig. 3, to render a pixel, we cast a cone for it and divide the cone into multiple conical frustums, similar to [Barron et al. 2021, 2022, 2023; Hu et al. 2023], which effectively avoids the discrete point-based sampling of the continuous signals in the image plane. To featurize a conical frustum, multi-sampling [Barron et al. 2023] and area-sampling [Hu et al. 2023] are two types of strategies in the hybrid representation. For the efficiency consideration, we adopt the latter one to first characterize the conical frustum by an anisotropic 3D Gaussian [Barron et al. 2021] with the mean and covariance as $(\boldsymbol{\mu}, \Sigma)$, and then featurize it with our proposed *Platonic Solid Projection* and *Ripmap Encoding*. Different from the Tri-Mip encoding [Hu et al. 2023] which roughly characterizes the frustum as isotropic balls, our Platonic Solid Projection together with Ripmap Encoding can accurately featurize the anisotropic 3D Gaussians, which are to be presented in the following sections. After featurizing the conical frustums, we employ a tiny MLP F_θ to estimate the color c and density σ of the frustums, and then render the pixel color by the volume rendering [Mildenhall et al. 2020]. The whole system is optimized end-to-end with a photometric loss between the rendered and observed images.

3.2 Ripmap Encoding

Before introducing how to featurize an anisotropic 3D Gaussian by projection, we first present the featurization of an anisotropic 2D Gaussian, which is the basis of the 3D case. Tri-MipRF [Hu et al. 2023] proposed to use a 2D Mipmap containing learnable features to support area sampling of an *isotropic* disc. However, as shown in Fig. 4 (a), the isotropic Mipmap structure of Tri-MipRF makes its sampling area a square, which can not precisely characterize the projected anisotropic 2D Gaussian, whose axis-aligned bounding-box is essentially a rectangle. In conventional graphics, anisotropic area sampling is proposed to address the aliasing issue when the view direction closely aligns with an axis of the UV

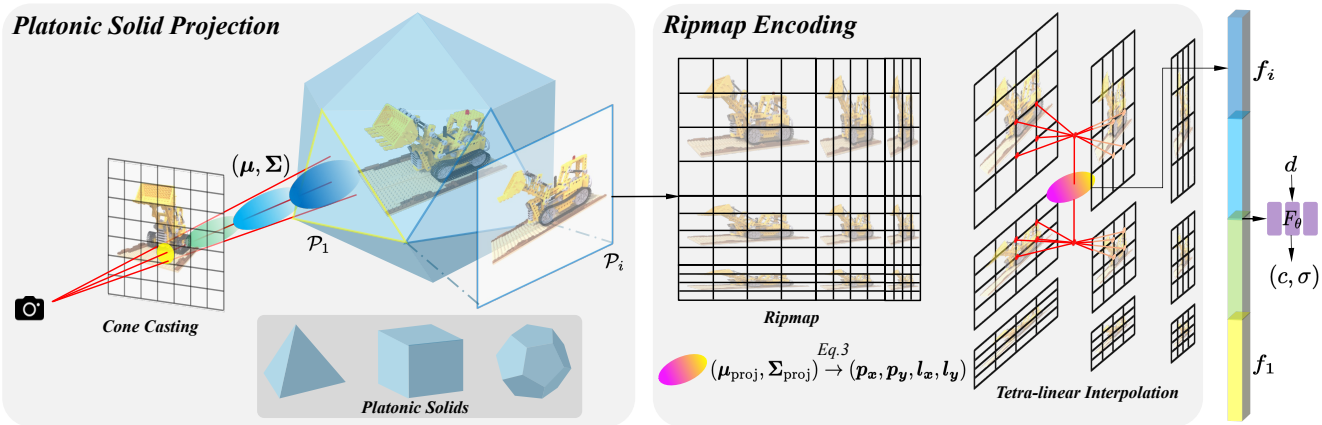
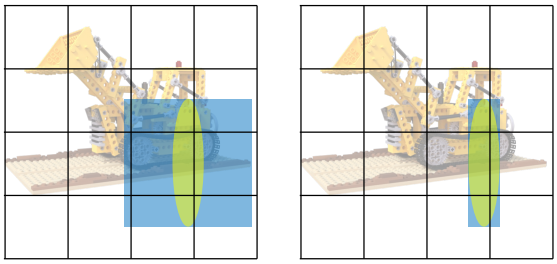


Figure 3: Overview of our Rip-NeRF. We first cast a cone for each pixel, and then divide the cone into multiple conical frustums, which are further characterized by anisotropic 3D Gaussians parameterized by their mean and covariance $(\mu,\ \Sigma)$. Next, to featurize a 3D Gaussian, we project it onto the unparalleled faces of the Platonic solid, denoted as $\{\mathcal{P}_i \mid i = 1, ..., n\}$ to form a 2D Gaussian $(\mu_{\text{proj}},\ \Sigma_{\text{proj}})$, while the Platonic solid's faces are represented by the Ripmap Encoding with learnable parameters. Subsequently, we perform tetra-linear interpolation on the Ripmap Encoding to query corresponding feature vectors f_i for the 2D Gaussian, where the position $(p_x,\ p_y)$ and level $(l_x,\ l_y)$ used in the interpolation are determined by the mean and covariance $(\mu_{\text{proj}},\ \Sigma_{\text{proj}})$ of the 2D Gaussian, respectively. Finally, feature vectors f_i from all Platonic solids' faces and the encoded view direction d are aggregated together to estimate the color c and density σ of the conical frustums by a tiny MLP F_θ.



(a) Isotropic area sampling (b) Anisotropic area sampling

Figure 4: Comparison between isotropic (a) and anisotropic (b) area sampling in Mipmap and Ripmap, for characterizing the projected Gaussian.

texture since the occurrence of aliasing is pronounced due to the significant variance in sampling density across the texture space. Among various solutions [Heckbert 1986], Ripmap [McReynolds et al. 1998] is a popular one due to its effectiveness and simplicity, which dynamically adjusts the the aspect ratio of the pre-filtering kernel based on the angle of incidence. Inspired by this, we propose *Ripmap Encoding* to employ a Ripmap with learnable parameters to enable anisotropic area-sampling in the neural rendering context, such that the anisotropic ellipsoidal footprint of a Gaussian can be characterized more precisely, as shown in Fig. 4 (b).

Ripmap Encoding construction. The Ripmap Encoding $\mathcal{R}$ contains $L \times L$ levels, while the base level $\mathcal{R}^{0,\,0}$ is a 2D feature grid $\mathcal{F}$ with the shape of $H \times W \times C$, where the H, W, C are the height, width, and number of channels, respectively. Other levels are constructed by performing anisotropic average pooling $\text{Avg}_{2\times1}$ and $\text{Avg}_{1\times2}$ on the lower level feature grid:

$$\mathcal{R}^{i,\,j} = \begin{cases} \text{Avg}_{2\times1}(\mathcal{R}^{i,\,j-1}) & \text{if } j \neq 0 \\ \text{Avg}_{1\times2}(\mathcal{R}^{i-1,\,j}) & \text{if } i \neq 0 \ \ \& \ \ j = 0 \\ \mathcal{F} & \text{otherwise,} \end{cases} \tag{1}$$

where i and j are the indices of the levels in the x and y directions, respectively. Note that, only the base level $\mathcal{R}^{0,\,0}$ is learnable, while other levels are derived from it, which makes the Ripmap Encoding compact and consistent among levels.

Ripmap Encoding querying. Once the Ripmap Encoding is constructed, we can featurize an anisotropic 2D Gaussian by querying the Ripmap Encoding using the tetra-linear interpolation:

$$\boldsymbol{f} = \mathcal{R}(p_x,\ p_y,\ l_x,\ l_y), \tag{2}$$

where $(p_x,\ p_y)$ and $(l_x,\ l_y)$ are the position and level used in the interpolation, respectively. The formal mathematical expression of tetra-linear interpolation is presented in the supplementary material. Since the querying position and level correspond to the location and size of the sampling area, respectively, we derive them from the mean and covariance $(\boldsymbol{\mu}_{\text{proj}},\ \Sigma_{\text{proj}})$ of the Gaussian as:

$$\begin{aligned} p_d &= \mu_d \\ l_d &= \log_2\left(\frac{w\sigma_d}{r}\right), \quad d \in \{x, y\}, \end{aligned} \tag{3}$$

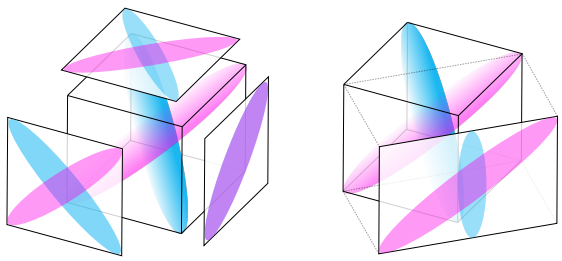
where $\sigma_x,\ \sigma_y = \sqrt{\text{diag}(\Sigma_{\text{proj}})}$ are the standard deviations along the $\mathbf{x}$ and $\mathbf{y}$ axes, respectively, w is a hyper-parameter to adjust how much probability mass of the Gaussian footprint is covered, and r is the radius of the bounding sphere for the reconstructed scene.

3.3 Platonic Solid Projection

Factorizing 3D space into a group of 2D planes has been proven to be effective and compact [Peng et al. 2020; Chan et al. 2022; Hu et al.

			PSNR ↑					SSIM ↑					LPIPS ↓				
	Train ↓	Size ↓	Full Res.	1/2 Res.	1/4 Res.	1/8 Res.	Avg.	Full Res.	1/2 Res.	1/4 Res.	1/8 Res.	Avg.	Full Res.	1/2 Res.	1/4 Res.	1/8 Res	Avg.
NeRF w/o $\mathcal{L}_{\text{area}}$	3 days	5.00 MB	31.20	30.65	26.25	22.53	27.66	0.950	0.956	0.930	0.871	0.927	0.055	0.034	0.043	0.075	0.052
NeRF [Mildenhall et al. 2020]	3 days	5.00 MB	29.90	32.13	33.40	29.47	31.23	0.938	0.959	0.973	0.962	0.958	0.074	0.040	0.024	0.039	0.044
TensoRF [Chen et al. 2022]	19 mins	71.8 MB	32.11	33.03	30.45	26.80	30.60	0.956	0.966	0.962	0.939	0.956	0.056	0.038	0.047	0.076	0.054
Instant-NGP [Müller et al. 2022]	5 mins	64.1 MB	30.00	32.15	33.31	29.35	31.20	0.939	0.961	0.974	0.963	0.959	0.079	0.043	0.026	0.040	0.047
Mip-NeRF [Barron et al. 2021]	3 days	2.50 MB	32.63	34.34	35.47	35.60	34.51	0.958	0.970	0.979	0.983	0.973	0.047	0.026	0.017	0.012	0.026
Tri-MipRF [Hu et al. 2023]	5.5 mins	48.0 MB	33.57	35.21	35.96	36.46	35.30	0.962	0.975	0.982	0.987	0.976	0.052	0.029	0.019	0.013	0.028
Zip-NeRF [Barron et al. 2023]	4.5 hrs	592 MB	34.21	36.55	37.88	38.13	36.69	0.974	0.985	0.990	0.992	0.985	0.036	0.019	0.014	0.015	0.021
3DGS [Kerbl et al. 2023]	7.5 mins	27.0 MB	29.00	30.94	32.06	28.21	30.05	0.946	0.965	0.976	0.964	0.963	0.064	0.037	0.024	0.030	0.039
Rip-NeRF$_{25k}$	32 mins	160 MB	34.30	35.94	36.92	37.47	36.16	0.966	0.978	0.984	0.989	0.979	0.045	0.025	0.016	0.011	0.024
Rip-NeRF (Ours)	2.6 hrs	160 MB	35.30	37.01	38.07	38.54	37.23	0.973	0.983	0.988	0.991	0.984	0.037	0.019	0.011	0.008	0.019

Table 1: Quantitative performance on the multi-scale Blender dataset [Barron et al. 2021]. We compared our Rip-NeRF, and its variant, Rip-NeRF$_{25k}$, which reduces the training iterations from $120k$ to $25k$ for fast reconstruction, against several representative methods. The best, second-best, and third-best results are marked in red, orange, and yellow, respectively.



(a) Projection on orthogonal tri-plane (b) Projection on an additional plane

Figure 5: Two 3D ellipsoids, whose major axes are aligned along two different body diagonals of a cube, share the same 2D AABBs on the orthogonal tri-plane (a), making them indistinguishable under the Ripmap encoding. However, their difference can be captured by an additional different-oriented plane (b).

2023; Fridovich-Keil et al. 2023; Cao and Johnson 2023; Chen et al. 2022]. Therefore, with the presented Ripmap Encoding for 2D area-sampling, the key to precisely featurizing anisotropic 3D Gaussians is how to project them onto the 2D planes. Tri-MipRF [Hu et al. 2023] projects isotropic 3D spheres onto three orthogonal 2D planes, however, this strategy falls short when dealing with anisotropic 3D Gaussians. The 3D Gaussians, produced by the cone casting, are almost randomly distributed in the Euclidean space with various shapes due to the arbitrariness of the intrinsic and extrinsic camera parameters. Thus, different 3D Gaussians may have the same 2D Axis-Aligned Bounding Box (AABB) when being projected onto the planes, *e.g.*, two 3D ellipsoids with major axes aligned along two different body diagonals of a cube in Fig. 5 (a), which makes them indistinguishable under the Ripmap Encoding. To address this issue, we propose to project the 3D Gaussians onto a larger number of planes, which are appropriately distributed in the 3D space, and then concatenate the features queried from those planes together, such that the 2D AABBs of different Gaussians can be more distinguishable and the derived features of Gaussians more discriminative, as shown in Fig. 5 (b).

Orientations and axes of the planes. Based on the above example, we intuitively think the more diverse the planes are oriented, the better their representation capability. To verify it, we tried three methods to evenly distribute the planes (the Platonic solids' faces, golden spiral [Keinert et al. 2015], and spherical blue noise [Wong and Wong 2018]). We also tried one control group method to adopt spherical white noise to determine the planes, which are slightly less diverse. We find the evenly distributed group performs much better than the control group and the three even groups have similar performance. Therefore, for simplicity and good performance, we opted for the *Platonic Solid Projection* that projects anisotropic 3D Gaussians into the unparalleled faces of a specific Platonic solid, *i.e.*, tetrahedron, cube, octahedron, dodecahedron, and icosahedron, whose faces are congruent (identical in shape and size) regular polygons that have equivalent dihedral angles. Note that, Mip-NeRF 360 [Barron et al. 2022]also adopted a similar strategy to derive off-axis integrated positional encoding features for 3D Gaussians. However, we propose the Platonic solid projection to explicitly project Gaussians onto planes for the area sampling using Ripmap encoding. Our Platonic Solid Projection also provides a flexible trade-off between rendering quality and efficiency by selecting different Platonic solids with a certain number of faces, as to be demonstrated in Tab. 4. Specifically, we denote the planes and their outward normals as $\{\mathcal{P}_i \mid i = 1, ..., n\}$ and $\{\boldsymbol{\phi}_i \in \mathbb{R}^3 \mid i = 1, ..., n\}$, respectively, where n is the number of faces of the selected Platonic solid. And the local 2D axes $\mathbf{x}_i \in \mathbb{R}^3$ and $\mathbf{y}_i \in \mathbb{R}^3$ of the plane $\mathcal{P}_i$, which is used to allocate grids for the Ripmap Encoding, should be perpendicular to the plane's normal $\boldsymbol{\phi}_i$. Given the unit vectors $\boldsymbol{X}, \boldsymbol{Y}$ and $\boldsymbol{Z}$ of the world coordinate system, we empirically define them as:

$$\begin{array}{lll} \mathbf{x}_i = \boldsymbol{X} & \mathbf{y}_i = \boldsymbol{Y} & \text{if } \boldsymbol{\phi}_i = \boldsymbol{Z}, \\ \mathbf{x}_i = \boldsymbol{Z} \times \boldsymbol{\phi}_i & \mathbf{y}_i = \mathbf{x}_i \times \boldsymbol{\phi}_i & \text{otherwise} \end{array} \tag{4}$$

Featurization of anisotropic 3D Gaussians. After defining the orientations and axes of the planes, we can project the anisotropic 3D Gaussians, characterized by mean $\boldsymbol{\mu}$ and covariance Σ, onto each plane $\mathcal{P}_i$:

$$\begin{aligned} \mathcal{M}_i &= [\mathbf{x}_i, \mathbf{y}_i], \\ \boldsymbol{\mu}^{\boldsymbol{i}}_{\text{proj}} &= \mathcal{M}_i^T \boldsymbol{\mu}, \\ \Sigma^{\boldsymbol{i}}_{\text{proj}} &= \mathcal{M}_i^T \Sigma \mathcal{M}_i, \end{aligned} \tag{5}$$

where $\mathcal{M}_i \in \mathbb{R}^{3\times 2}$ is the projection matrix for mapping the 3D world coordinate system into the 2D coordinate system of plane $\mathcal{P}_i$, $\boldsymbol{\mu}^{\boldsymbol{i}}_{\text{proj}}$ and $\Sigma^{\boldsymbol{i}}_{\text{proj}}$ are the mean and covariance of the projected 2D Gaussians on plane $\mathcal{P}_i$. Then, we can query a feature vector $\boldsymbol{f}_i$ from the Ripmap Encoding of each plane $\mathcal{P}_i$ using Eq. 2 and Eq. 3. Finally, we concatenate all the feature vectors $\{\boldsymbol{f}_i \mid i = 1, ..., n\}$

from the corresponding planes $\{\mathcal{P}_i \mid i = 1, ..., n\}$ to form the final feature vector $\boldsymbol{f}$ of the 3D Gaussian.

4 EXPERIMENTS

4.1 Implementation Details

Since our Rip-NeRF acquires the 3D structure only from the calibrated multi-view 2D images, we optimize the whole system end-to-end with a photometric loss, assessing the discrepancy between rendered pixels and captured images. To make the photometric loss area-aware, we scale the loss for each pixel by incorporating the area of its footprint on the image plane, denoted as "area loss" $\mathcal{L}_{\text{area}}$, following [Barron et al. 2021; Hu et al. 2023]. Except for the ablation study, we set the shape of the base level in the Ripmap Encoding $\mathcal{R}^{0,\,0}$ to $H = 512$, $W = 512$, and $C = 16$, aligned with Tri-MipRF [Hu et al. 2023]. For the Platonic Solid Projection, we by default employ the icosahedron, which contains ten unparalleled faces. We set the value of hyper-parameter w in Eq. (3) to 2.0, which is used to adjust how much probability mass of the Gaussian footprint is covered.

For efficiency, we not only employ the tiny-cuda-nn [Müller 2021] library for its highly optimized MLP implementation but also implement our Platonic Solid Projection and Ripmap Encoding in CUDA kernels. Besides, we also adopt the NerfAcc [Li et al. 2022] library to incorporate a binary occupancy grid to indicate empty *vs.*non-empty space similar to [Müller et al. 2022; Hu et al. 2023], which enables efficiently skipping samples in the empty area. All the modules are integrated into the PyTorch framework [Paszke et al. 2019] since PyTorch is widely used in the research community. Training of our Rip-NeRF is conducted using the AdamW optimizer [Loshchilov and Hutter 2019] with $120k$ iterations. We also present a variant of our method, Rip-NeRF$_{25\text{k}}$, which is trained with $25k$ iterations, for faster reconstruction that only slightly sacrifices the rendering quality. We apply a weight decay of 1×10^{-5} and an initial learning rate of 2×10^{-3}, which is modulated using PyTorch's MultiStepLR scheduler. Following Tri-MipRF [Hu et al. 2023], the learning rate for Ripmap encoding is scaled up by 10.0 times since it directly represents the scene.

4.2 Evaluation on the Muti-scale Blender Dataset

Following Mip-NeRF [Barron et al. 2021] and Tri-MipRF [Hu et al. 2023], to evaluate the capability of rendering anti-aliasing and fine image details, we benchmarked our Rip-NeRF on the multi-scale Blender dataset [Barron et al. 2021], which is a combination of the Blender dataset [Mildenhall et al. 2020] and its down-scaled versions with a factor of 2, 4, and 8. We compared our Rip-NeRF with the representative cutting-edge methods, *i.e.*, NeRF [Mildenhall et al. 2020], Mip-NeRF [Barron et al. 2021], TensoRF [Chen et al. 2022], Instant-NGP [Müller et al. 2022], Tri-MipRF [Hu et al. 2023], Zip-NeRF [Barron et al. 2023], and 3D Gaussian Splatting (denoted as 3DGS) [Kerbl et al. 2023]. Since some methods are not optimized for captures with variable distances or multiple resolutions, we incorporate the area loss $\mathcal{L}_{\text{area}}$ with all the methods for a fair comparison. All the methods are retrained on the combined training and validation splits and evaluated on the testing split, using their official implementations, in alignment with Tri-MipRF [Hu et al. 2023].

Quantitative results. The quantitative results are presented in Tab. 1, where we assess rendering quality using PSNR, SSIM [Wang et al. 2004], and VGG LPIPS [Zhang et al. 2018] metrics. Additionally, to evaluate the efficiency of computation and storage, we report the average training time on an NVIDIA A100-SXM4-80GB GPU and the model size. We can see that methods without anti-aliasing design, *i.e.*, NeRF, TensoRF, Instant-NGP, and 3DGS, perform poorly in the multi-scale setting, which is the consequence of the discrete sampling in the continuous physical space. Importantly, our Rip-NeRF consistently outperforms all the other cutting-edge methods, even the strong Tri-MipRF and Zip-NeRF baselines, in terms of PSNR and LPIPS metrics. Though Zip-NeRF performs slightly better in terms of the SSIM metric, it requires almost double training time (4.5h *vs.*2.6h) and four times of model size (592 MB *vs.*160 MB). Notably, the variant of our method that reduces the training iterations from $120k$ to $25k$, Rip-NeRF$_{25\text{k}}$, also achieves comparable results to Zip-NeRF, while requiring only 11.76% training time (4.5h *vs.*32min). Additionally, the GPU memory consumption during training of Zip-NeRF is about 80 GB, which is unfeasible for some consumer-level GPUs, *e.g.*the NVIDIA GeForce RTX 4090 etc. , while that of Rip-NeRF is only 20 GB. And, the rendering speed of our Rip-NeRF is about 3 FPS in this experimental setting, which outperforms Zip-NeRF (0.25 FPS). Admittedly, Zip-NeRF is designed for the more general unbounded scenes and we believe its performance can be further optimized for bounded objects. Nevertheless, our work demonstrates the effectiveness of area-sampling for high-fidelity anti-aliased rendering. The effectiveness, efficiency, and compactness of our Rip-NeRF are attributed to our Ripmap-Encoded Platonic solid representation, which enables efficient anisotropic area-sampling of the 3D space. In contrast, the multi-sampling mechanism in Zip-NeRF is effective but not efficient enough, which inherently demands a large number of samples to featurize a single area, putting it in a dilemma between the rendering quality and efficiency of computation and storage.

Qualitative results. To qualitatively evaluate the performance of our Rip-NeRF, we compare it with Tri-MipRF [Hu et al. 2023] and Zip-NeRF [Barron et al. 2023], since they are the most relevant methods to our approach and perform well quantitatively in the multi-scale setting. We compared the full-resolution renderings of the three methods in Fig. 6 and the first row of Fig. 1 (teaser), where we can see that our Rip-NeRF achieves the best rendering quality, particularly in regions with challenging appearance and geometry, such as the anisotropic specular highlights on the gong in the "drums" scene, the slender rope on the "ship" scene, and the periodic grids on the "mic" scene. Besides evaluating full-resolution renderings, we also present the renderings of the three methods at 1/2, 1/4, and 1/8 resolutions in Fig. 7, to demonstrate the effectiveness of our Rip-NeRF in anti-aliasing and preserving fine details. We can see that in scenarios with lower resolution, our Rip-NeRF method exhibits superior performance compared to the other two methods. Notably, the periodic features on the microphone, which appear blurred in images rendered by Tri-MipRF and exhibit aliasing ("jaggies") in those by Zip-NeRF, are rendered with higher fidelity in our results. These qualitative findings further corroborate the efficacy of Rip-NeRF in producing high-fidelity, anti-aliasing images, demonstrating its robustness across various resolution settings.

	PSNR ↑	SSIM ↑	LPIPS ↓
SRN [Sitzmann et al. 2019]	22.26	0.846	0.170
LLFF [Mildenhall et al. 2019]	24.88	0.911	0.114
Neural Volumes [Lombardi et al. 2019]	26.05	0.893	0.160
NeRF [Mildenhall et al. 2020]	31.74	0.953	0.050
DVGO [Sun et al. 2022]	31.95	0.957	0.053
TensoRF [Chen et al. 2022]	33.14	0.963	0.047
Instant-NGP [Müller et al. 2022]	33.18	0.963	0.045
3DGS [Kerbl et al. 2023]	34.44	0.975	0.028
Mip-NeRF [Barron et al. 2021]	33.09	0.961	0.043
Tri-MipRF [Hu et al. 2023]	33.90	0.964	0.050
Zip-NeRF [Barron et al. 2023]	34.76	0.977	0.032
Rip-NeRF$_{25k}$	34.92	0.969	0.042
Rip-NeRF (Ours)	35.44	0.973	0.037

Table 2: Quantitative performance on the single-scale Blender dataset [Mildenhall et al. 2020]. We compared our Rip-NeRF, and its variant, Rip-NeRF$_{25k}$, against several representative methods. The best, second-best, and third-best results are marked in red, orange, and yellow, respectively.

4.3 Evaluation on the Single-scale Blender Dataset

The single-scale Blender dataset [Mildenhall et al. 2020] renders the image at a fixed resolution and distance, which is consistent with the assumption of methods without anti-aliasing design, *e.g.*, NeRF [Mildenhall et al. 2020], TensoRF [Chen et al. 2022], Instant-NGP [Müller et al. 2022], and 3DGS [Kerbl et al. 2023]. Even though this scenario is not our primary focus, we still compared our Rip-NeRF and Rip-NeRF$_{25k}$ on this dataset against several representative methods. The quantitative results of this comparison are detailed in Tab. 2, where we can observe that our Rip-NeRF and Rip-NeRF$_{25k}$ perform the best and second-best, respectively, in terms of the PSNR metric, while achieving comparable results to the Zip-NeRF and 3DGS in terms of the SSIM and LPIPS metrics. It demonstrates that our Rip-NeRF is also effective in the single-scale setting, even though it is not optimized for this scenario.

4.4 Evaluation on Real-world Captures

To further verify the practicality of our approach, we applied our Rip-NeRF to a newly captured real-world dataset. We captured four challenging objects that contain fine periodic structures using an iPhone. We applied Structure-from-Motion (SfM) to the image sequence to estimate camera parameters and then employed image segmentation to separate the targets from the background scene. We also applied a rigid transform and rescaling to the poses reconstructed by SfM, aiming to center and scale the target object in a manner similar to that used in the Blender dataset. Each captured scene consists of 400 to 420 images with a resolution of 3840 × 2160, down-sampled with factors of 2, 4, 8 and 16 (instead of 1, 2, 4, and 8 in Multi-scale Blender) due to its relatively high original resolution. For reconstruction, we uniformly sampled 50% of the images, reserving the remaining portion for evaluation. This setup makes the number of images and resolution of two splits similar to the previous Multi-scale Blender Dataset. However, noisy camera poses, motion blur, and defocus blur imply greater challenges to reconstruction than synthetic data.

We compared our Rip-NeRF with Tri-MipRF [Hu et al. 2023] and Zip-NeRF [Barron et al. 2023] since they show strong capabilities in the previous experiments. Three example results are shown in Fig. 8, where we can clearly observe that our Rip-NeRF renders more accurate intricate structures and appearance details. Additionally, quantitative results of the whole dataset in Tab. 3 and the PSNR/SSIM values displayed below each image further affirm the effectiveness of our approach for real-world captures.

4.5 Ablation study

In our ablation studies, we evaluated key components of our method. Rip-NeRF w/o PSP utilized orthogonal triplanes with anisotropic area sampling to assess the impact of excluding Platonic Solid Projection. For examining the role of anisotropic area sampling, Rip-NeRF w/o RE employed isotropic mipmaps, adapting the Tri-MipRF [Hu et al. 2023] framework with 10 unparalleled planes from an icosahedron, using nvdiffrast [Laine et al. 2020] for mipmap construction. The two ablation experiments are both trained for 25000 iterations.

The quantitative results from our experiments on the multi-scale Blender dataset are detailed in Table 4. Our findings reveal that the independent application of each component—Platonic Solid Projection (PSP) and Ripmap Encoding (RE) —yields only modest improvements. Specifically, employing PSP alone results in a mere 0.37% increase in average PSNR, while using solely RE slightly lowers all three metrics. However, when PSP and RE are combined in our Rip-NeRF$_{25k}$ model, they produce a synergistic effect: an average 3.44% increase in PSNR, with SSIM and LPIPS significantly surpassing those of Tri-MipRF [Hu et al. 2023]. This synergy suggests that the whole is greater than the sum of its parts. While PSP alone improves alignment of the ellipsoid footprint with the 2D grid, it still relies on square query areas. Ripmap Encoding with only 3 planes are not capable of modeling the nuance of different Gaussians as well.

In our ablation study focusing on the choice of Platonic Solid, we experimented with cubes (PS3), tetrahedrons (PS4), and dodecahedron (PS6). In this ablation experiment, we conducted experiment by increasing grid resolution for PS3, PS4, and PS6 to keep the total parameter count the same as our full method (160MB). From PS3 to our full model (PS10), the training time increases 28%. Interestingly, adding planes does not always correlate with performance improvements. For instance, despite the addition of one plane in PS4 compared to PS3, there was a decrease in PSNR, SSIM, and LPIPS. Conversely, applying more planes to the tri-mip encoding method yielded minimal enhancement, suggesting that ripmaps overcome model capacity limitations. This is evidenced by the fact that isotropic spheres project similarly onto different planes, whereas anisotropic 3D Gaussian footprints vary across planes.

4.6 Limitations

Despite the excellent performance on bounded datasets, our representation still faces challenges for unbounded scenes. We suppose two possible reasons that cause difficulties in 2D-style representations. First, non-vaguely-convex shapes lead to information from self-occluded locations being projected onto the same 2D area. Second, the space warping mechanism proposed in Mip-NeRF360

	PSNR↑					SSIM↑					LPIPS↓				
	lego	*flask*	*mic*	*filter*	*Avg*	*lego*	*flask*	*mic*	*filter*	*Avg*	*lego*	*flask*	*mic*	*filter*	*Avg*
Zip-NeRF [Barron et al. 2023]	35.95	37.79	39.15	38.45	37.84	0.991	0.992	0.997	0.991	0.993	0.014	0.012	0.008	0.021	0.014
Tri-MipRF [Hu et al. 2023]	37.19	37.38	40.30	36.58	38.09	0.991	0.991	0.996	0.985	0.991	0.012	0.013	0.006	0.022	0.013
Rip-NeRF (Ours)	37.85	38.46	42.07	37.17	38.89	0.993	0.993	0.998	0.986	0.992	0.010	0.010	0.004	0.020	0.011

Table 3: Quantitative results on our newly captured real-world dataset. The highest performance is marked in red.

		PSNR↑					SSIM↑					LPIPS↓				
	Train ↓	Full Res.	1/2 Res.	1/4 Res.	1/8 Res.	Avg.	Full Res.	1/2 Res.	1/4 Res.	1/8 Res.	Avg.	Full Res.	1/2 Res.	1/4 Res.	1/8 Res.	Avg.
Tri-MipRF [Hu et al. 2023]	5.5 mins	33.57	35.21	35.96	36.46	35.30	0.962	0.975	0.982	0.987	0.976	0.052	0.029	0.019	0.013	0.028
Rip-NeRF, PS3 (w/o PSP)	25.0 mins	33.32	34.65	35.25	35.92	34.79	0.959	0.972	0.979	0.985	0.974	0.056	0.032	0.021	0.014	0.031
Rip-NeRF, PS4	25.5 mins	32.76	34.14	34.63	34.63	34.06	0.950	0.961	0.964	0.970	0.961	0.066	0.042	0.031	0.028	0.041
Rip-NeRF, PS6	26.5 mins	33.86	35.37	36.34	36.78	35.59	0.963	0.974	0.982	0.986	0.976	0.051	0.029	0.019	0.013	0.028
Rip-NeRF w/o RE	8.5 mins	33.64	35.26	36.13	36.68	35.43	0.962	0.974	0.981	0.987	0.976	0.052	0.030	0.020	0.013	0.029
Rip-NeRF$_{25k}$	32.0 mins	34.30	35.94	36.92	37.47	36.16	0.966	0.978	0.984	0.989	0.979	0.045	0.025	0.016	0.011	0.024
Rip-NeRF (Ours)	2.6 hrs	35.30	37.01	38.07	38.54	37.23	0.973	0.983	0.988	0.991	0.984	0.037	0.019	0.011	0.008	0.019

Table 4: Quantitative Comparison of Rip-NeRF and its ablations on the Multi-Scale Blender Dataset [Barron et al. 2021]. The best, second-best, and third-best results are marked in red, orange, and yellow, respectively.

[Barron et al. 2022] encourages more locations along a non-linear curve, compared to a straight line, to be projected onto the same 2D area, which is difficult for the explicit 2D feature grid to characterize. To address these challenges, perhaps a more advanced 3D to 2D mapping function is required to be explored.

5 CONCLUSION

In this work, we present a Ripmap-Encoded Platonic Solid representation for neural radiance fields, named Rip-NeRF. Our Rip-NeRF can render high-fidelity anti-aliasing images while maintaining efficiency, enabled by the proposed Platonic Solid Projection and Ripmap Encoding. The Platonic Solid Projection factorizes the 3D space onto the unparalleled faces of a certain Platonic solid, such that the anisotropic 3D areas can be projected onto planes with distinguishable characterization. And the Ripmap Encoding enables featurizing the projected anisotropic areas both precisely and efficiently by the anisotropic area-sampling. These two components work together for precisely and efficiently featurizing anisotropic 3D areas. It achieves state-of-the-art rendering quality on both synthetic datasets and real-world captures, particularly excelling in the fine details of structures and textures, which verifies the effectiveness of the proposed Platonic Solid Projection and Ripmap Encoding.

REFERENCES

Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. 2021. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *ICCV*.

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. 2022. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *CVPR*.

Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. 2023. Zip-NeRF: Anti-Aliased Grid-Based Neural Radiance Fields. In *ICCV*.

Yash Belhe, Michaël Gharbi, Matthew Fisher, Iliyan Georgiev, Ravi Ramamoorthi, and Tzu-Mao Li. 2023. Discontinuity-Aware 2D Neural Fields. *ACM Transactions on Graphics (TOG)* 42, 6 (2023), 1–11.

Mojtaba Bemana, Karol Myszkowski, Jeppe Revall Frisvad, Hans-Peter Seidel, and Tobias Ritschel. 2022. Eikonal fields for refractive novel-view synthesis. In *ACM SIGGRAPH 2022 Conference Proceedings*. 1–9.

Ang Cao and Justin Johnson. 2023. Hexplane: A fast representation for dynamic scenes. In *CVPR*.

Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. 2022. Efficient geometry-aware 3D generative adversarial networks. In *CVPR*.

Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. 2022. Tensorf: Tensorial radiance fields. In *ECCV*.

Anpei Chen, Zexiang Xu, Xinyue Wei, Siyu Tang, Hao Su, and Andreas Geiger. 2023b. Dictionary fields: Learning a neural basis decomposition. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–12.

Xiaoxue Chen, Junchen Liu, Hao Zhao, Guyue Zhou, and Ya-Qin Zhang. 2023a. Nerrf: 3d reconstruction and view synthesis for transparent and specular objects with neural refractive-reflective fields. *arXiv preprint arXiv:2309.13039* (2023).

Zheng Dong, Ke Xu, Yaoan Gao, Qilin Sun, Hujun Bao, Weiwei Xu, and Rynson WH Lau. 2023. SAILOR: Synergizing Radiance and Occupancy Fields for Live Human Performance Capture. *ACM Transactions on Graphics (TOG)* 42, 6 (2023), 1–15.

Hao-Bin Duan, Miao Wang, Jin-Chuan Shi, Xu-Chuan Chen, and Yan-Pei Cao. 2023. BakedAvatar: Baking Neural Fields for Real-Time Head Avatar Synthesis. *ACM Transactions on Graphics (TOG)* 42, 6 (2023), 1–17.

Daniel Duckworth, Peter Hedman, Christian Reiser, Peter Zhizhin, Jean-François Thibert, Mario Lučić, Richard Szeliski, and Jonathan T Barron. 2023. SMERF: Streamable Memory Efficient Radiance Fields for Real-Time Large-Scene Exploration. *arXiv preprint arXiv:2312.07541* (2023).

Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. 2023. K-planes: Explicit radiance fields in space, time, and appearance. In *CVPR*.

Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. 2022. Plenoxels: Radiance fields without neural networks. In *CVPR*.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*.

Bingchen Gong, Yuehao Wang, Xiaoguang Han, and Qi Dou. 2023. SeamlessNeRF: Stitching Part NeRFs with Gradient Propagation. In *SIGGRAPH Asia 2023 Conference Papers*. 1–10.

Kunal Gupta, Milos Hasan, Zexiang Xu, Fujun Luan, Kalyan Sunkavalli, Xin Sun, Manmohan Chandraker, and Sai Bi. 2023. MCNeRF: Monte Carlo Rendering and Denoising for Real-Time NeRFs. In *SIGGRAPH Asia 2023 Conference Papers*. 1–11.

Paul S. Heckbert. 1986. Survey of Texture Mapping. *IEEE Computer Graphics and Applications* 6, 11 (1986), 56–67. https://doi.org/10.1109/MCG.1986.276672

Wenbo Hu, Yuling Wang, Lin Ma, Bangbang Yang, Lin Gao, Xiao Liu, and Yuewen Ma. 2023. Tri-MipRF: Tri-Mip Representation for Efficient Anti-Aliasing Neural Radiance Fields. In *ICCV*.

Yi-Hua Huang, Yan-Pei Cao, Yu-Kun Lai, Ying Shan, and Lin Gao. 2023. NeRF-texture: Texture synthesis with neural radiance fields. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–10.

Mustafa Işık, Martin Rünz, Markos Georgopoulos, Taras Khakhulin, Jonathan Starck, Lourdes Agapito, and Matthias Nießner. 2023. Humanrf: High-fidelity neural radiance fields for humans in motion. *arXiv preprint arXiv:2305.06356* (2023).

Kaiwen Jiang, Shu-Yu Chen, Hongbo Fu, and Lin Gao. 2023. NeRFFaceLighting: Implicit and Disentangled Face Lighting Representation Leveraging Generative Prior in Neural Radiance Fields. *ACM Transactions on Graphics* 42, 3 (2023), 1–18.

Kaiwen Jiang, Shu-Yu Chen, Feng-Lin Liu, Hongbo Fu, and Lin Gao. 2022. NeRFFaceEditing: Disentangled face editing in neural radiance fields. In *SIGGRAPH Asia 2022 Conference Papers*. 1–9.

Animesh Karnewar, Tobias Ritschel, Oliver Wang, and Niloy Mitra. 2022. Relu fields: The little non-linearity that could. In *ACM SIGGRAPH 2022 Conference Proceedings*.

Benjamin Keinert, Matthias Innmann, Michael Sänger, and Marc Stamminger. 2015. Spherical fibonacci mapping. *ACM Transactions on Graphics (TOG)* 34, 6 (2015),

1–7.
Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics* 42, 4 (July 2023). https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/
Tobias Kirschstein, Shenhan Qian, Simon Giebenhain, Tim Walter, and Matthias Nießner. 2023. NeRSemble: Multi-view Radiance Field Reconstruction of Human Heads. *arXiv preprint arXiv:2305.03027* (2023).
Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. 2020. Modular Primitives for High-Performance Differentiable Rendering. *TOG* 39, 6 (2020).
Yixing Lao, Xiaogang Xu, Xihui Liu, Hengshuang Zhao, et al. 2024. CorresNeRF: Image Correspondence Priors for Neural Radiance Fields. *Advances in Neural Information Processing Systems* 36 (2024).
Ruilong Li, Matthew Tancik, and Angjoo Kanazawa. 2022. NerfAcc: A General NeRF Accleration Toolbox. *arXiv preprint arXiv:2210.04847* (2022).
Zilu Li, Guandao Yang, Xi Deng, Christopher De Sa, Bharath Hariharan, and Steve Marschner. 2023. Neural Caches for Monte Carlo Partial Differential Equation Solvers. In *SIGGRAPH Asia 2023 Conference Papers*. 1–10.
Gao Lin, Liu Feng-Lin, Chen Shu-Yu, Jiang Kaiwen, Li Chunpeng, Yukun Lai, and Fu Hongbo. 2023. SketchFaceNeRF: Sketch-based facial generation and editing in neural radiance fields. *ACM Transactions on Graphics* (2023).
Haotong Lin, Sida Peng, Zhen Xu, Yunzhi Yan, Qing Shuai, Hujun Bao, and Xiaowei Zhou. 2022. Efficient neural radiance fields for interactive free-viewpoint video. In *SIGGRAPH Asia 2022 Conference Papers*. 1–9.
Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. 2019. Neural Volumes: Learning Dynamic Renderable Volumes from Images. *TOG* 38, 4 (2019), 65:1–65:14.
Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.
Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. 2021. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *CVPR*.
Tom McReynolds, David Blythe, Brad Grantham, and Scott Nelson. 1998. Advanced graphics programming techniques using OpenGL. *Computer Graphics* (1998), 95–145.
Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. 2019. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *TOG* 38, 4 (2019), 1–14.
Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *ECCV*.
Thomas Müller. 2021. *tiny-cuda-nn*. https://github.com/NVlabs/tiny-cuda-nn
Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *TOG* 41, 4 (2022), 1–15.
Keunhong Park, Philipp Henzler, Ben Mildenhall, Jonathan T Barron, and Ricardo Martin-Brualla. 2023a. CamP: Camera Preconditioning for Neural Radiance Fields. *SIGGRAPH Asia* (2023).
Keunhong Park, Philipp Henzler, Ben Mildenhall, Jonathan T Barron, and Ricardo Martin-Brualla. 2023b. CamP: Camera preconditioning for neural radiance fields. *ACM Transactions on Graphics (TOG)* 42, 6 (2023), 1–11.
Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M Seitz. 2021. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *arXiv preprint arXiv:2106.13228* (2021).
Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. PyTorch: An imperative style, high-performance deep learning library. In *NeurIPS*.
Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. 2020. Convolutional occupancy networks. In *ECCV*.
Ziqiao Peng, Wentao Hu, Yue Shi, Xiangyu Zhu, Xiaomei Zhang, Hao Zhao, Jun He, Hongyan Liu, and Zhaoxin Fan. 2023. SyncTalk: The Devil is in the Synchronization for Talking Head Synthesis. *arXiv preprint arXiv:2311.17590* (2023).
Christian Reiser, Rick Szeliski, Dor Verbin, Pratul Srinivasan, Ben Mildenhall, Andreas Geiger, Jon Barron, and Peter Hedman. 2023. Merf: Memory-efficient radiance fields for real-time view synthesis in unbounded scenes. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–12.
Zixi Shu, Ran Yi, Yuqi Meng, Yutong Wu, and Lizhuang Ma. 2023. RT-Octree: Accelerate PlenOctree Rendering with Batched Regular Tracking and Neural Denoising for Real-time Neural Radiance Fields. In *SIGGRAPH Asia 2023 Conference Papers*. 1–11.
Vincent Sitzmann, Michael Zollhoefer, and Gordon Wetzstein. 2019. Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations. *NeurIPS*.
Nagabhushan Somraj, Adithyan Karanayil, and Rajiv Soundararajan. 2023. SimpleNeRF: Regularizing Sparse Input Neural Radiance Fields with Simpler Solutions. In *SIGGRAPH Asia 2023 Conference Papers*. 1–11.
Nagabhushan Somraj and Rajiv Soundararajan. 2023. ViP-NeRF: Visibility Prior for Sparse Input Neural Radiance Fields. *arXiv preprint arXiv:2305.00041* (2023).
Cheng Sun, Min Sun, and Hwann-Tzong Chen. 2022. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *CVPR*.
Towaki Takikawa, Alex Evans, Jonathan Tremblay, Thomas Müller, Morgan McGuire, Alec Jacobson, and Sanja Fidler. 2022. Variable bitrate neural fields. In *ACM SIGGRAPH 2022 Conference Proceedings*. 1–9.
Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, et al. 2023. Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–12.
Alex Trevithick, Matthew Chan, Michael Stengel, Eric Chan, Chao Liu, Zhiding Yu, Sameh Khamis, Manmohan Chandraker, Ravi Ramamoorthi, and Koki Nagano. 2023. Real-time radiance fields for single-image portrait view synthesis. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–15.
Daoye Wang, Prashanth Chandran, Gaspard Zoss, Derek Bradley, and Paulo Gotardo. 2022. Morf: Morphable radiance fields for multiview neural head modeling. In *ACM SIGGRAPH 2022 Conference Proceedings*. 1–9.
Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. 2021. NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction. *NeurIPS* (2021).
Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *TIP* 13, 4 (2004), 600–612.
Yuxi Wei, Zi Wang, Yifan Lu, Chenxin Xu, Changxing Liu, Hao Zhao, Siheng Chen, and Yanfeng Wang. 2024. Editable Scene Simulation for Autonomous Driving via Collaborative LLM-Agents. *arXiv preprint arXiv:2402.05746* (2024).
Kin-Ming Wong and Tien-Tsin Wong. 2018. Spherical blue noise. In *Proceedings of the 26th Pacific Conference on Computer Graphics and Applications: Short Papers*. 5–8.
Jiangkai Wu, Liming Liu, Yunpeng Tan, Quanlu Jia, Haodan Zhang, and Xinggong Zhang. 2023b. ActRay: Online Active Ray Sampling for Radiance Fields. In *SIGGRAPH Asia 2023 Conference Papers*. 1–10.
Tong Wu, Jia-Mu Sun, Yu-Kun Lai, and Lin Gao. 2023c. De-nerf: Decoupled neural radiance fields for view-consistent appearance editing and high-frequency environmental relighting. In *ACM SIGGRAPH 2023 conference proceedings*. 1–11.
Xiuchao Wu, Jiamin Xu, Xin Zhang, Hujun Bao, Qixing Huang, Yujun Shen, James Tompkin, and Weiwei Xu. 2023d. ScaNeRF: Scalable Bundle-Adjusting Neural Radiance Fields for Large-Scale Scene Rendering. *ACM Transactions on Graphics (TOG)* 42, 6 (2023), 1–18.
Zirui Wu, Tianyu Liu, Liyi Luo, Zhide Zhong, Jianteng Chen, Hongmin Xiao, Chao Hou, Haozhe Lou, Yuantao Chen, Runyi Yang, et al. 2023a. Mars: An instance-aware, modular and realistic simulator for autonomous driving. In *CAAI International Conference on Artificial Intelligence*. Springer, 3–15.
Yuelang Xu, Hongwen Zhang, Lizhen Wang, Xiaochen Zhao, Han Huang, Guojun Qi, and Yebin Liu. 2023. LatentAvatar: Learning Latent Expression Code for Expressive Neural Head Avatar. *arXiv preprint arXiv:2305.01190* (2023).
Han Yan, Celong Liu, Chao Ma, and Xing Mei. 2023. PlenVDB: Memory Efficient VDB-Based Radiance Fields for Fast Training and Rendering. In *CVPR*.
Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. 2021. Plenoctrees for real-time rendering of neural radiance fields. In *ICCV*.
Shiran Yuan and Hao Zhao. 2023. SlimmeRF: Slimmable Radiance Fields. *arXiv preprint arXiv:2312.10034* (2023).
Chong Zeng, Guojun Chen, Yue Dong, Pieter Peers, Hongzhi Wu, and Xin Tong. 2023. Relighting Neural Radiance Fields with Shadow and Highlight Hints. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–11.
Jingbo Zhang, Xiaoyu Li, Ziyu Wan, Can Wang, and Jing Liao. 2022b. Fdnerf: Few-shot dynamic neural radiance fields for face reconstruction and expression editing. In *SIGGRAPH Asia 2022 Conference Papers*. 1–9.
Qiang Zhang, Seung-Hwan Baek, Szymon Rusinkiewicz, and Felix Heide. 2022a. Differentiable point-based radiance fields for efficient view synthesis. In *SIGGRAPH Asia 2022 Conference Papers*. 1–12.
Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*.
Zerong Zheng, Xiaochen Zhao, Hongwen Zhang, Boning Liu, and Yebin Liu. 2023. AvatarReX: Real-time Expressive Full-body Avatars. *arXiv preprint arXiv:2305.04789* (2023).
Qiang Zhou, Weize Li, Lihan Jiang, Guoliang Wang, Guyue Zhou, Shanghang Zhang, and Hao Zhao. 2024. Pad: A dataset and benchmark for pose-agnostic anomaly detection. *Advances in Neural Information Processing Systems* 36 (2024).
Zhenxin Zhu, Yuantao Chen, Zirui Wu, Chao Hou, Yongliang Shi, Chuxuan Li, Pengfei Li, Hao Zhao, and Guyue Zhou. 2023. Latitude: Robotic global localization with truncated dynamic low-pass filter in city-scale nerf. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 8326–8332.
Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. 2022. Nice-slam: Neural implicit scalable encoding for slam. In *CVPR*.
Jingyu Zhuang, Chen Wang, Liang Lin, Lingjie Liu, and Guanbin Li. 2023. Dreameditor: Text-driven 3d scene editing with neural fields. In *SIGGRAPH Asia 2023 Conference Papers*. 1–10.

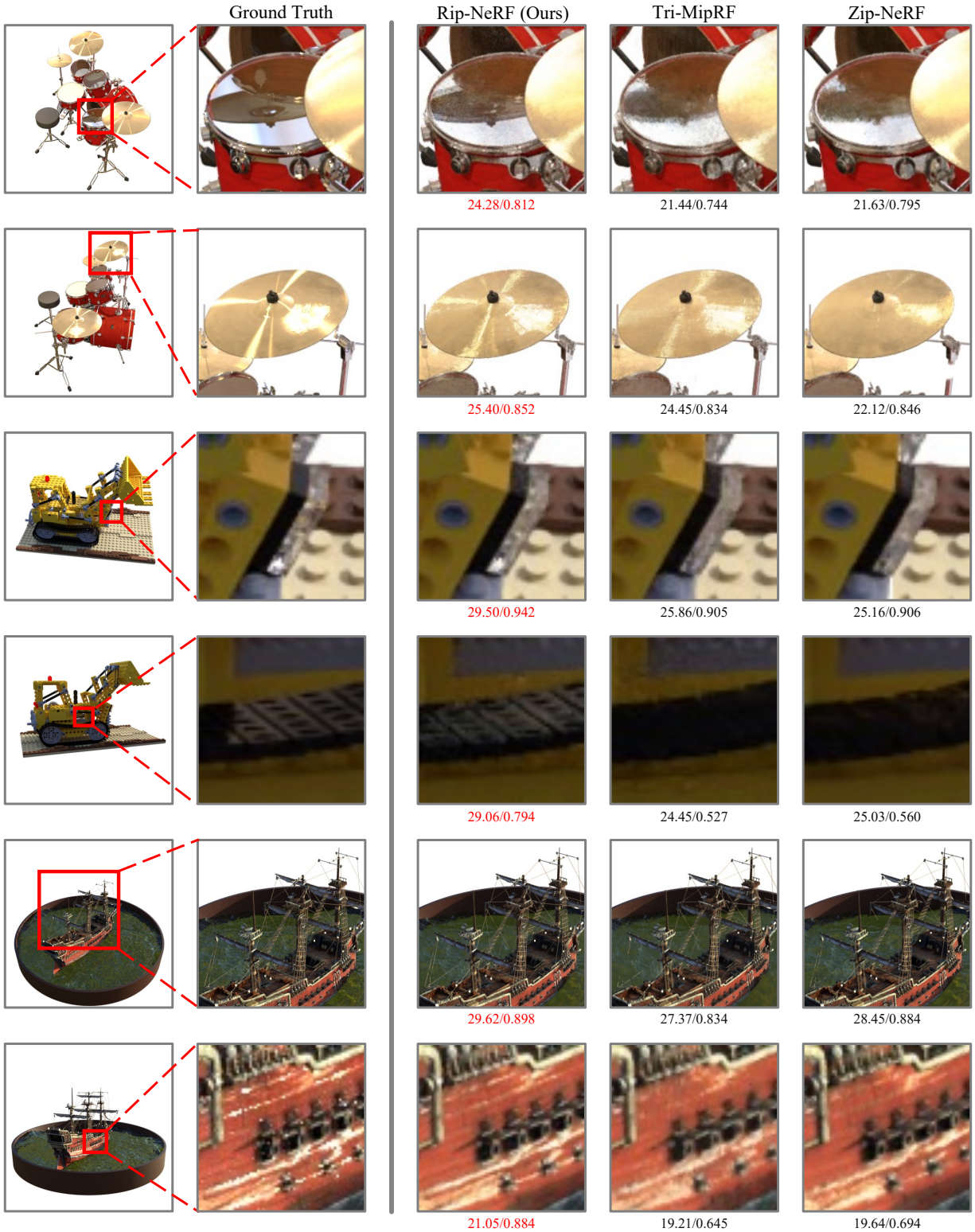


Figure 6: Qualitative comparison of the full-resolution (close-up views) renderings on the multi-scale Blender dataset. PSNR/SSIM values are shown at the bottom of each result.

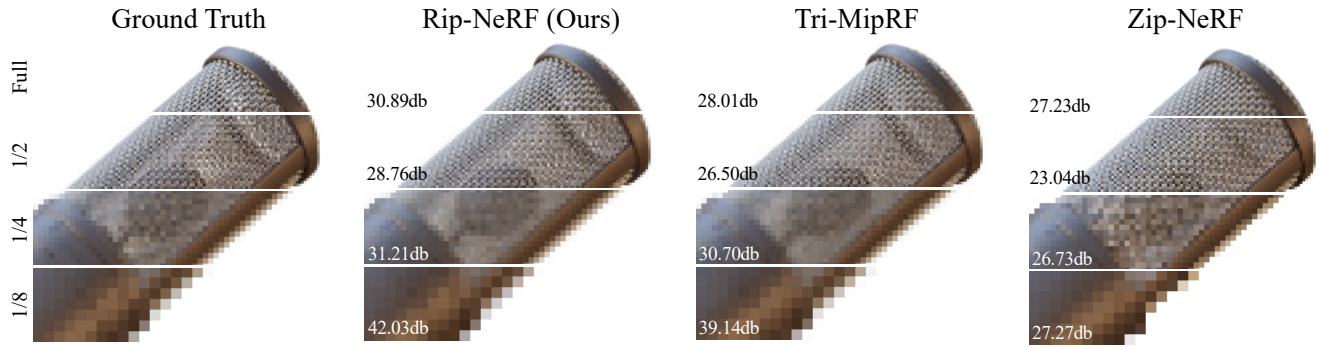


Figure 7: Qualitative comparison of the multi-resolution evaluation on the mic scene from the multi-scale Blender dataset. PSNR values are shown in the bottom right corners of each result.

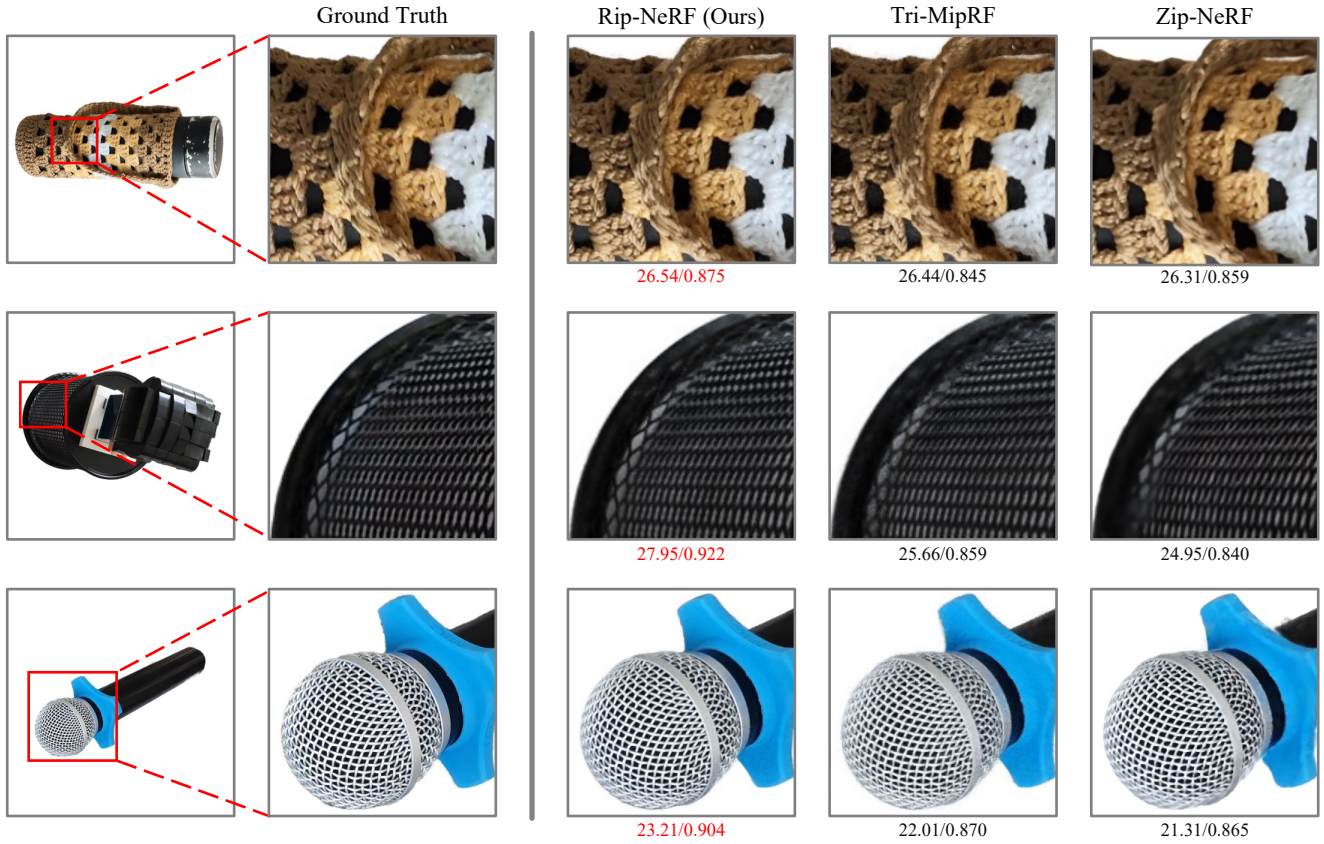


Figure 8: Qualitative results of our Rip-NeRF, Tri-MipRF [Hu et al. 2023], and Zip-NeRF [Barron et al. 2023] on real-world captures. PSNR/SSIM values are shown at the bottom of each result.

A CONE CASTING

To characterize an anisotropic sub-volume, we follow the procedure introduced in Mip-NeRF [Barron et al. 2021] to model the conical frustum as a 3D Gaussian $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$, where $\boldsymbol{\mu}$ and Σ are the mean and covariance, respectively. Given a conical frustum defined by its near and far t values $[t_0, t_1]$ along the ray direction $\mathbf{d}$ originating from the camera center $\mathbf{o}$, and the cone radius $\dot{r}$ at the image plane, we can compute $\boldsymbol{\mu}$ and Σ as follows:

First, we calculate the mean μ_t and variance σ_t^2 of the frustum along the ray direction, as well as the variance σ_r^2 perpendicular to the ray direction:

$$\mu_t = \frac{3\left(t_1^4 - t_0^4\right)}{4\left(t_1^3 - t_0^3\right)},\ \sigma_t^2 = \frac{3\left(t_1^5 - t_0^5\right)}{5\left(t_1^3 - t_0^3\right)} - \mu_t^2,\ \sigma_r^2 = \dot{r}^2\left(\frac{3\left(t_1^5 - t_0^5\right)}{20\left(t_1^3 - t_0^3\right)}\right). \tag{6}$$

Then, we transform these quantities from the local coordinate frame of the conical frustum to the world coordinate frame, obtaining the mean $\boldsymbol{\mu}$ and covariance Σ of the 3D Gaussian:

$$\boldsymbol{\mu} = \mathbf{o} + \mu_t \mathbf{d}, \quad \Sigma = \sigma_t^2\left(\mathbf{d}\mathbf{d}^{\mathrm{T}}\right) + \sigma_r^2\left(\mathbf{I} - \frac{\mathbf{d}\mathbf{d}^{\mathrm{T}}}{\|\mathbf{d}\|_2^2}\right), \tag{7}$$

where $\mathbf{I}$ is the identity matrix. The resulting 3D Gaussian $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$ approximates the geometry of the conical frustum and serves as the input for the subsequent Platonic Solid Projection and Ripmap Encoding steps in Rip-NeRF.

B TETRA-LINEAR INTERPOLATION

The tetra-linear interpolation process in Ripmap Encoding involves querying the feature values from the surrounding vertices in the 4D space defined by the position $(p_x,\ p_y)$ and level $(l_x,\ l_y)$. Given a query point $(p_x,\ p_y,\ l_x,\ l_y)$, we first identify the 16 neighboring vertices in the 4D space. Let $(p_x^i,\ p_y^j,\ l_x^k,\ l_y^m)$ denote the neighboring vertex, where $i, j, k, m \in \{0, 1\}$ represent the binary indices in each dimension. The feature value at the query point is then interpolated using the weighted sum of the feature values at these neighboring vertices:

$$\boldsymbol{f} = \sum_{i=0}^{1}\sum_{j=0}^{1}\sum_{k=0}^{1}\sum_{m=0}^{1} w_{ijkm} \cdot \boldsymbol{f}_{ijkm}, \tag{8}$$

where $\boldsymbol{f}_{ijkm} = \mathcal{R}^{l_x^k,\, l_y^m}(p_x^i,\ p_y^j)$ represents the feature value at the neighboring vertex $(p_x^i,\ p_y^j,\ l_x^k,\ l_y^m)$, and w_{ijkm} is the interpolation weight calculated based on the distances between the query point and the neighboring vertex in each dimension:

$$w_{ijkm} = \prod_{\substack{d \in \{x,y\} \\ t \in \{p,l\}}} (1 - |t_d - t_d^{ij}|), \tag{9}$$

where t_d^{ij} represents the neighboring vertex coordinates along dimension d, with i and j being the binary indices for position and level, respectively. By performing this tetra-linear interpolation, we obtain a smooth and continuous feature representation $\boldsymbol{f}$ for the anisotropic 2D Gaussian at the query point $(p_x,\ p_y,\ l_x,\ l_y)$ in the Ripmap Encoding. This interpolation process allows for efficient querying of the Ripmap Encoding while considering the anisotropic nature of the Gaussian footprint.

C IMPLEMENTATION DETAILS

C.1 Tiny MLP

Our tiny MLP nonlinearly maps the Ripmap-Encoding feature vector $\mathbf{f}$ and the view direction $\mathbf{d}$ to the density τ and color c of the sampled sphere $\mathcal{S}$, aligning with Tri-MipRF [Hu et al. 2023] for consistency. The dimension of $\mathbf{f}$ is 160, considering the ripmaps $\mathcal{R}$ shape of $512 \times 512 \times 16$ and the use of 10 unparalleled icosahedron faces. The first two MLP layers process $\mathbf{f}$ to yield τ and a 15-dimensional geometric feature $\mathbf{f}_{\text{geo}}$. The view direction $\mathbf{d}$, encoded via spherical harmonics, is combined with $\mathbf{f}_{\text{geo}}$ in the final three layers to estimate the view-dependent color c, similar to [Müller et al. 2022]. The MLP's width is set to 128, using ReLU activations (except for the output layer of τ, where a truncated exponential function is used, following [Müller et al. 2022]). This efficient MLP, implemented with tiny-cuda-nn, is optimized for fused and half-precision operations.

C.2 Sampling Strategy

Sample points are selected within a sphere of radius r, predefined for each scene. For the nerf-synthetic dataset scenes r is set to 1.5. We adopt the OccupancyGrid from nerfacc [Li et al. 2022] as our sampler, setting density to zero for points outside the bounding sphere. The occupancy threshold was 0.005 for all experiments.

C.3 Optimization

Optimizable parameters in Rip-NeRF include the tiny MLP's model weights and the ripmaps $\mathcal{R}$. Model weights are initialized following [Glorot and Bengio 2010], while ripmaps $\mathcal{R}$ start from a uniform distribution over $[-0.01, 0.01]$. We use AdamW [Loshchilov and Hutter 2019] for optimization, scaling the base learning rate for $\mathcal{R}$ by 10x due to its direct scene representation role. The base learning rate is 2×10^{-3}, reduced by 0.6x at steps 60K, 90K, 100K, and 108K, over a total of 120K iterations. Following [Müller et al. 2022], we dynamically adjust the batch size to maintain approximately 256K spheres per batch.

D DETAILED QUANTITATIVE RESULTS

For a more detailed quantitative per-scene analysis, Table 5 and Table 6 showcases our Rip-NeRF against representative baseline methods training on the multi-scale and single-scale blender dataset.

E MORE QUALITATIVE RESULTS

	PSNR ↑								
	chair	*drums*	*ficus*	*hotdog*	*lego*	*materials*	*mic*	*ship*	*Average*
NeRF w/o $\mathcal{L}_{\text{area}}$	29.92	23.27	27.15	32.00	27.75	26.30	28.40	26.46	27.66
NeRF [Mildenhall et al. 2020]	33.39	25.87	30.37	35.64	31.65	30.18	32.60	30.09	31.23
Mip-NeRF [Barron et al. 2021]	37.14	27.02	33.19	39.31	35.74	32.56	38.04	33.08	34.51
TensoRF [Chen et al. 2022]	32.47	25.37	31.16	34.96	31.73	28.53	31.48	29.08	30.60
Instant-NGP [Müller et al. 2022]	32.95	26.43	30.41	35.87	31.83	29.31	32.58	30.23	31.20
Tri-MipRF [Hu et al. 2023]	38.36	28.66	34.29	40.02	36.57	32.22	38.46	33.80	35.30
Zip-NeRF [Barron et al. 2023]	39.52	29.46	35.54	41.64	37.25	34.39	39.69	36.08	36.70
3DGS [Kerbl et al. 2023]	33.04	25.52	29.21	35.48	29.66	27.25	31.15	29.12	30.05
Rip-NeRF, PS3 (w/o PSP)	37.64	28.39	33.58	39.45	35.98	31.91	38.11	33.20	34.79
Rip-NeRF, PS4	37.73	28.56	34.58	38.85	34.50	31.66	38.29	28.28	34.06
Rip-NeRF, PS6	38.80	29.07	35.27	40.25	36.27	32.20	39.32	33.55	35.59
Rip-NeRF, w/o PE	38.99	28.92	35.21	39.97	35.87	32.32	38.92	33.24	35.43
Rip-NeRF$_{25k}$	39.36	29.40	35.89	40.82	36.94	32.66	40.07	34.13	36.16
Rip-NeRF (ours)	40.08	29.88	36.61	41.98	38.36	33.79	40.94	36.18	37.23

	SSIM ↑								
	chair	*drums*	*ficus*	*hotdog*	*lego*	*materials*	*mic*	*ship*	*Average*
NeRF w/o $\mathcal{L}_{\text{area}}$	0.944	0.891	0.942	0.959	0.926	0.934	0.958	0.861	0.927
NeRF [Mildenhall et al. 2020]	0.971	0.932	0.971	0.979	0.965	0.967	0.980	0.900	0.958
MipNeRF [Barron et al. 2021]	0.988	0.945	0.984	0.988	0.984	0.977	0.993	0.922	0.973
TensoRF [Chen et al. 2022]	0.967	0.930	0.974	0.977	0.967	0.957	0.978	0.895	0.956
Instant-ngp [Müller et al. 2022]	0.971	0.940	0.973	0.979	0.966	0.959	0.981	0.904	0.959
Tri-MipRF [Hu et al. 2023]	0.991	0.957	0.986	0.990	0.987	0.972	0.993	0.936	0.976
Zip-NeRF [Barron et al. 2023]	0.994	0.968	0.991	0.993	0.990	0.984	0.995	0.966	0.985
3DGS [Kerbl et al. 2023]	0.979	0.944	0.970	0.984	0.967	0.959	0.979	0.922	0.963
Rip-NeRF, PS3 (w/o PSP)	0.989	0.955	0.984	0.988	0.984	0.969	0.992	0.928	0.974
Rip-NeRF, PS4	0.984	0.951	0.987	0.983	0.976	0.953	0.988	0.868	0.961
Rip-NeRF, PS6	0.992	0.959	0.989	0.990	0.984	0.972	0.994	0.930	0.976
Rip-NeRF, w/o PE	0.992	0.959	0.989	0.989	0.983	0.973	0.993	0.929	0.976
Rip-NeRF$_{25k}$	0.993	0.963	0.990	0.991	0.987	0.976	0.995	0.939	0.979
Rip-NeRF (ours)	0.994	0.966	0.992	0.993	0.991	0.980	0.996	0.960	0.984

	LPIPS ↓								
	chair	*drums*	*ficus*	*hotdog*	*lego*	*materials*	*mic*	*ship*	*Average*
NeRF w/o $\mathcal{L}_{\text{area}}$	0.035	0.069	0.032	0.028	0.041	0.045	0.031	0.095	0.052
NeRF [Mildenhall et al. 2020]	0.028	0.059	0.026	0.024	0.035	0.033	0.025	0.085	0.044
MipNeRF [Barron et al. 2021]	0.011	0.044	0.014	0.012	0.013	0.019	0.007	0.062	0.026
TensoRF [Chen et al. 2022]	0.042	0.075	0.032	0.035	0.036	0.063	0.040	0.112	0.054
Instant-ngp [Müller et al. 2022]	0.035	0.066	0.029	0.028	0.040	0.051	0.032	0.095	0.047
Tri-MipRF [Hu et al. 2023]	0.013	0.049	0.017	0.016	0.014	0.036	0.010	0.071	0.028
Zip-NeRF [Barron et al. 2023]	0.017	0.038	0.010	0.013	0.013	0.019	0.008	0.049	0.021
3DGS [Kerbl et al. 2023]	0.024	0.056	0.029	0.020	0.037	0.039	0.024	0.082	0.039
Rip-NeRF, PS3 (w/o PSP)	0.015	0.051	0.020	0.019	0.016	0.039	0.012	0.077	0.031
Rip-NeRF, PS4	0.015	0.052	0.016	0.023	0.028	0.042	0.010	0.146	0.042
Rip-NeRF, PS6	0.012	0.046	0.013	0.015	0.017	0.036	0.009	0.077	0.028
Rip-NeRF, w/o PE	0.011	0.046	0.013	0.017	0.018	0.034	0.009	0.081	0.029
Rip-NeRF$_{25k}$	0.010	0.041	0.011	0.013	0.013	0.029	0.007	0.069	0.024
Rip-NeRF (ours)	0.009	0.037	0.010	0.010	0.009	0.024	0.006	0.047	0.019

Table 5: Quantitative Per-Scene Results on the Multi-Scale Blender Dataset: This table presents the arithmetic mean of each metric, averaged over the four scales of the dataset for individual scenes. Performance rankings are highlighted with color coding: the best, second-best, and third-best results are in red, orange, and yellow, respectively.

	PSNR ↑								
	chair	*drums*	*ficus*	*hotdog*	*lego*	*materials*	*mic*	*ship*	*Average*
SRN [Sitzmann et al. 2019]	29.96	17.18	20.73	26.81	20.85	18.09	26.85	20.60	22.26
LLFF [Mildenhall et al. 2019]	28.72	21.13	21.79	31.41	24.54	20.72	27.48	23.22	24.88
Neural Volumes [Lombardi et al. 2019]	28.33	22.58	24.79	30.71	26.08	24.22	27.78	23.93	26.05
NeRF [Mildenhall et al. 2020]	34.17	25.08	30.39	36.82	33.31	30.03	34.78	29.30	31.74
Mip-NeRF [Barron et al. 2021]	35.14	25.48	33.29	37.48	35.70	30.71	36.51	30.41	33.09
Mip-NeRF 360 [Barron et al. 2022]	35.65	25.60	33.19	37.71	36.10	29.90	36.52	31.26	33.24
DVGO [Sun et al. 2022]	34.09	25.44	32.78	36.74	34.64	29.57	33.20	29.13	31.95
TensoRF [Chen et al. 2022]	35.76	26.01	33.99	37.41	36.46	30.12	34.61	30.77	33.14
Instant-NGP [Müller et al. 2022]	35.00	26.02	33.51	37.40	36.39	29.78	36.22	31.10	33.18
Tri-MipRF [Hu et al. 2023]	36.53	26.73	34.93	38.46	36.61	30.63	37.56	29.74	33.90
Zip-NeRF [Barron et al. 2023]	36.19	27.37	36.08	39.18	36.60	32.63	36.74	33.27	34.76
3DGS [Kerbl et al. 2023]	36.88	26.80	36.09	38.72	36.82	30.99	36.69	32.52	34.44
Rip-NeRF$_{25k}$	37.31	27.46	36.45	39.04	37.72	31.24	39.28	30.88	34.92
Rip-NeRF (ours)	37.58	27.76	36.87	39.77	38.29	31.80	39.91	31.56	35.44

	SSIM ↑								
	chair	*drums*	*ficus*	*hotdog*	*lego*	*materials*	*mic*	*ship*	*Average*
SRN [Sitzmann et al. 2019]	0.910	0.766	0.849	0.923	0.809	0.808	0.947	0.757	0.846
LLFF [Mildenhall et al. 2019]	0.948	0.890	0.896	0.965	0.911	0.890	0.964	0.823	0.911
Neural Volumes [Lombardi et al. 2019]	0.916	0.873	0.910	0.944	0.880	0.888	0.946	0.784	0.893
NeRF [Mildenhall et al. 2020]	0.975	0.925	0.967	0.979	0.968	0.953	0.987	0.869	0.953
Mip-NeRF [Barron et al. 2021]	0.981	0.932	0.980	0.982	0.978	0.959	0.991	0.882	0.961
Mip-NeRF 360 [Barron et al. 2022]	0.983	0.931	0.979	0.982	0.980	0.949	0.991	0.893	0.961
DVGO [Sun et al. 2022]	0.977	0.930	0.978	0.980	0.976	0.951	0.983	0.879	0.957
TensoRF [Chen et al. 2022]	0.985	0.937	0.982	0.982	0.983	0.952	0.988	0.895	0.963
Instant-NGP [Müller et al. 2022]	0.979	0.937	0.981	0.982	0.982	0.951	0.990	0.896	0.963
Tri-MipRF [Hu et al. 2023]	0.987	0.940	0.984	0.984	0.983	0.952	0.992	0.886	0.964
Zip-NeRF [Barron et al. 2023]	0.988	0.957	0.990	0.988	0.985	0.974	0.993	0.945	0.977
3DGS [Kerbl et al. 2023]	0.990	0.961	0.991	0.988	0.986	0.969	0.993	0.918	0.975
Rip-NeRF$_{25k}$	0.989	0.948	0.989	0.986	0.985	0.960	0.995	0.898	0.969
Rip-NeRF (ours)	0.990	0.950	0.990	0.988	0.987	0.964	0.995	0.918	0.973

	LPIPS ↓								
	chair	*drums*	*ficus*	*hotdog*	*lego*	*materials*	*mic*	*ship*	*Average*
SRN [Sitzmann et al. 2019]	0.106	0.267	0.149	0.100	0.200	0.174	0.063	0.299	0.170
LLFF [Mildenhall et al. 2019]	0.064	0.126	0.130	0.061	0.110	0.117	0.084	0.218	0.114
Neural Volumes [Lombardi et al. 2019]	0.109	0.214	0.162	0.109	0.175	0.130	0.107	0.276	0.160
NeRF [Mildenhall et al. 2020]	0.026	0.071	0.032	0.030	0.031	0.047	0.012	0.150	0.050
Mip-NeRF [Barron et al. 2021]	0.021	0.065	0.020	0.027	0.021	0.040	0.009	0.138	0.043
Mip-NeRF 360 [Barron et al. 2022]	0.018	0.069	0.022	0.024	0.018	0.053	0.011	0.135	0.042
DVGO [Sun et al. 2022]	0.027	0.077	0.024	0.034	0.028	0.058	0.017	0.161	0.053
TensoRF [Chen et al. 2022]	0.022	0.073	0.022	0.032	0.018	0.058	0.015	0.138	0.047
Instant-NGP [Müller et al. 2022]	0.022	0.071	0.023	0.027	0.017	0.060	0.010	0.132	0.045
Tri-MipRF [Hu et al. 2023]	0.021	0.076	0.025	0.031	0.019	0.067	0.012	0.148	0.050
Zip-NeRF [Barron et al. 2023]	0.016	0.048	0.012	0.019	0.017	0.036	0.007	0.099	0.032
3DGS [Kerbl et al. 2023]	0.010	0.035	0.009	0.018	0.013	0.029	0.005	0.104	0.028
Rip-NeRF$_{25k}$	0.017	0.064	0.016	0.027	0.016	0.053	0.008	0.135	0.042
Rip-NeRF (ours)	0.016	0.062	0.015	0.022	0.014	0.046	0.006	0.111	0.037

Table 6: Quantitative Per-Scene Results on the Single-Scale Blender Dataset: This table presents the arithmetic mean of each metric, averaged over the four scales of the dataset for individual scenes. Performance rankings are highlighted with color coding: the best, second-best, and third-best results are in red, orange, and yellow, respectively.

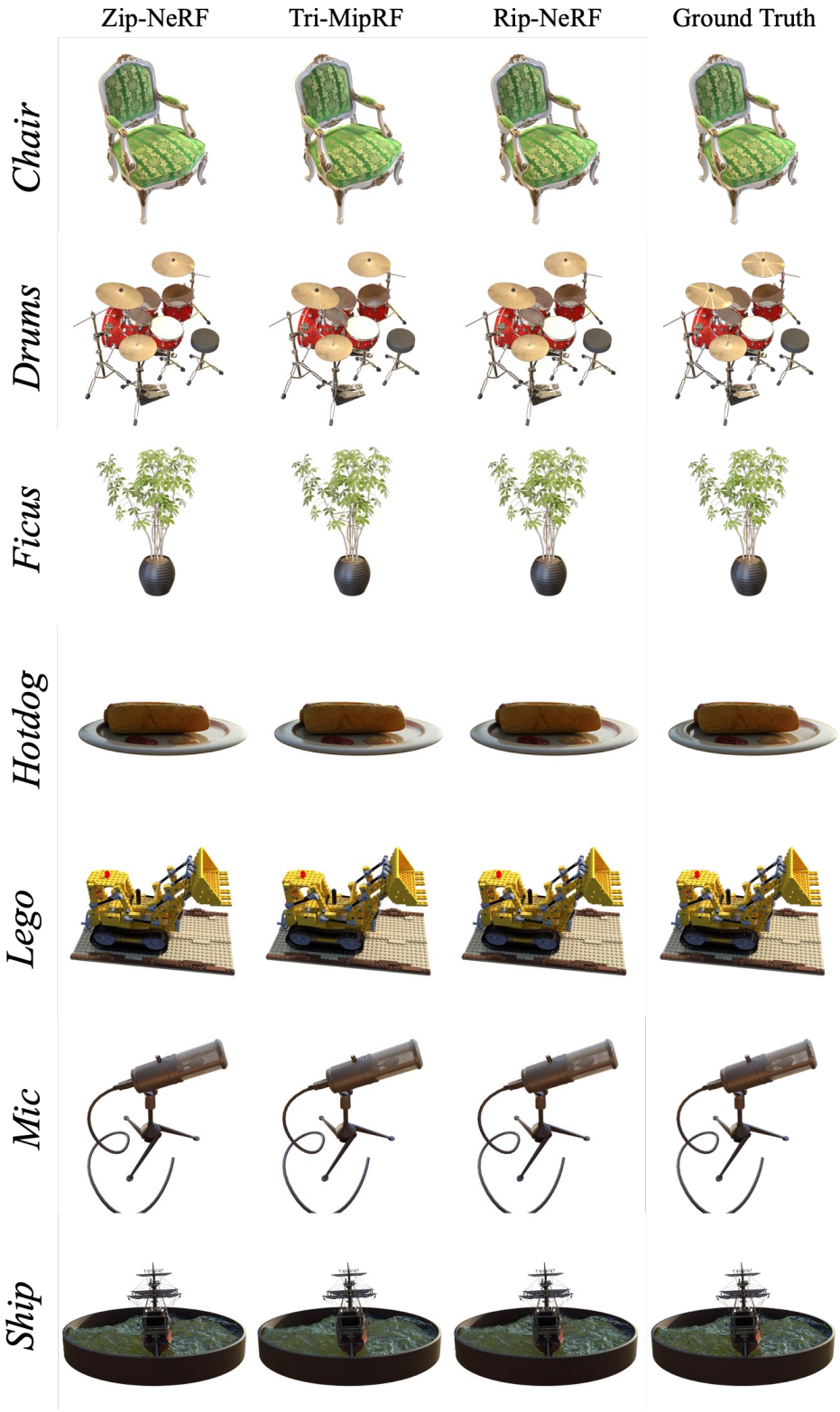


Figure 9: More qualitative rendering results of Zip-NeRF, Tri-MipRF, and our Rip-NeRF on the multi-scale Blender dataset.