

# Oblique-MERF: Revisiting and Improving MERF for Oblique Photography

XIAOYI ZENG, University of Science and Technology of China, China

KAIWEN SONG, University of Science and Technology of China, China

LEYUAN YANG, University of Science and Technology of China, China

BAILIN DENG, Cardiff University, United Kingdom

JUYONG ZHANG\*, University of Science and Technology of China, China

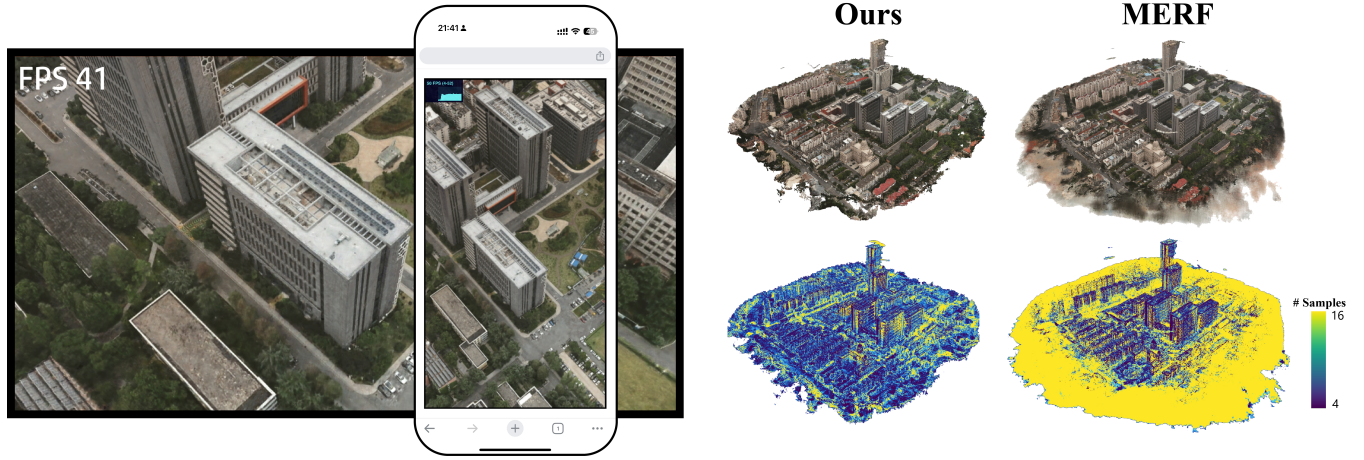


Fig. 1. Oblique-MERF enables real-time synthesis of novel views for oblique photography on diverse commodity devices, including tests on an NVIDIA GTX 1650 and an iPhone 14 Pro Max (left). We introduce a novel compact sampling representation that markedly reduces sampling points and emphasizes significant regions, enhancing rendering quality and increasing frame rates. We present the rendering output and the visualization of the number of samples (right).

Neural implicit fields have established a new paradigm for scene representation, with subsequent work achieving high-quality real-time rendering. However, reconstructing 3D scenes from oblique aerial photography presents unique challenges, such as varying spatial scale distributions and a constrained range of tilt angles, often resulting in high memory consumption and reduced rendering quality at extrapolated viewpoints. In this paper, we enhance MERF to accommodate these data characteristics by introducing an innovative adaptive occupancy plane optimized during the volume rendering process and a smoothness regularization term for view-dependent color to address these issues. Our approach, termed Oblique-MERF, surpasses state-of-the-art real-time methods by approximately 0.7 dB, reduces VRAM usage by about 40%, and achieves higher rendering frame rates with more realistic rendering outcomes across most viewpoints.

CCS Concepts: • **Computing methodologies** → *Neural networks; Volumetric models; Reconstruction*.

Additional Key Words and Phrases: Memory Efficient, Oblique Photograph, Neural Radiance Fields, Volumetric Representation, Real-Time Rendering.

## 1 INTRODUCTION

Reconstructing a 3D scene for high-fidelity rendering from freely chosen viewpoints has been a longstanding challenge in computer graphics. Neural Radiance Field (NeRF) [Mildenhall et al. 2020] accomplishes this via a novel implicit representation parameterized by multi-layer perceptrons (MLP). In the realm of large-scale scene reconstruction for oblique aerial photography datasets, various works

have sought to improve upon NeRF by employing strategies such as spatial partitioning [Mi and Xu 2023; Reiser et al. 2021; Tancik et al. 2022], advanced sampling methods [Turki et al. 2023; Wang et al. 2023], and efficient data structures [Müller et al. 2022; Sara Fridovich-Keil and Alex Yu et al. 2022; Sun et al. 2022; Yu et al. 2021]. Furthermore, recent works bake models into meshes with optimized textures [Chen et al. 2023; Tang et al. 2022; Yariv et al. 2023] or feature grids and planes [Hedman et al. 2021; Reiser et al. 2023], to enable interactive rendering frame rates on commercial devices.

However, these advancements have not been specifically tailored to the unique characteristics of oblique aerial photography data. When applied to such large-scale scenes, they incur high memory footprints and fail to deliver high-fidelity rendering from all perspectives. Specifically, oblique aerial photography datasets often cover vast areas, spanning hundreds of thousands of square meters yet only extending to a few hundred meters of vertical height. The commonly used cubic grid [Müller et al. 2022] has a high complexity of storage and struggles to adapt to such data, leading to inefficient use of storage and computational resources. Additionally, such datasets might lead to artifacts such as floaters due to inadequate constraints on the sampling space. Moreover, instead of a complete 360-degree view, these scenes are often captured from a constrained range of pitch angles relative to the ground and may exhibit abnormal high-lights and shadows in viewpoints not covered during training. In

\*Corresponding author (juyong@ustc.edu.cn).

Table 1. Difference among sampling representations.

	Memory efficient	Query speed	Loss aware
Occupancy Grid [Müller et al. 2022]	×	✓	×
Proposal MLP [Barron et al. 2022]	✓	×	✓
Occupancy Plane (Ours)	✓	✓	✓

this work, we enhance the performance and quality of current NeRF-based real-time rendering methods on oblique aerial photography datasets, focusing on two key aspects: the representation of the sampling space and the issue of color extrapolation.

Current prevalent methods, such as 3D grids or implicit proposal sampling networks [Barron et al. 2022; Müller et al. 2022], may face significant limitations when dealing with such vast scenes as shown in Tab. 1. On the one hand, the cubic storage complexity of grid-based representations limits their ability to represent sampling spaces at high resolutions; moreover, they often rely on optimization methods that are not inherently tied to rendering quality. These limitations can hinder the precise conveyance of occupancy information and lead to redundant sample points that slow down the rendering pipeline and affect the rendering quality. On the other hand, network-based representations are computationally expensive, as they require evaluating numerous candidate points to determine the final sample points. To overcome these drawbacks, we propose a novel sampling strategy that models the occupancy space as a sandwiched region between two height field surfaces conforming to the ground. Furthermore, we integrate this adaptive representation with volume rendering to ensure awareness of the photometric loss. Thanks to our explicit representation of the sampling space, features of the occupied region can be directly extracted for real-time rendering. This eliminates the need for tens of hours of baking.

In addition, we address the extrapolation issue arising from the limited range of view directions in the captured training set. In the setting of extrapolation novel view, the rendering results often exhibit aberrant colors due to the lack of supervision. As such anomalies primarily result from the non-smooth behavior of specular color with respect to varying viewpoints, we introduce a novel smoothing term suitable for oblique photography to regularize color dependence on observation directions. Compared to directly constraining the Lipschitz continuity in neural networks [Liu et al. 2022], our approach focuses on the change of the specular component with the viewing directions and exhibits stronger generalizability and robustness, offering a more natural handling of color variations related to viewpoints during rendering from most perspectives.

In summary, our primary contributions are:

- We propose a novel sampling strategy that seamlessly integrates training and baking processes by optimizing an explicit occupancy representation. The adaptive structure is aware of the photometric loss and the regularization term to convey occupancy information at the least memory cost accurately.
- Utilizing the smooth nature of specular reflections, we present a new regularization approach that significantly enhances the rendering quality of novel extrapolated viewpoints, providing smoother and more realistic visual outcomes.

- Extensive experiments confirm that our method enhances the PSNR by 0.7 dB, decreases VRAM usage by 40%, and boosts the frame rate by 35% and 300% for low-altitude and high-altitude viewpoints, respectively, compared to the baseline. Additionally, our technique for smoothing view-dependent color significantly improves PSNR by 0.52 dB in extrapolation scenarios.

## 2 RELATED WORK

Our work primarily focuses on large-scale scene reconstruction and real-time rendering. Below, we review domains related to our work, including large-scale scene representation, real-time rendering, sampling strategy, and regularization of the radiance field.

*3D Reconstruction for Oblique Photography.* To reconstruct large-scale scenes, a common approach involves using drones equipped with one or more cameras to capture high-altitude flight data across the scene, known as oblique photography. Traditional 3D modeling techniques include the use of Structure-from-Motion (SfM) pipelines [Agarwal et al. 2009; Frahm et al. 2010; Schönberger and Frahm 2016; Wu 2013] to estimate camera poses and obtain sparse point clouds, followed by surface reconstruction through dense multi-view stereo [Furukawa et al. 2010; Furukawa and Ponce 2010; Jin et al. 2005]. These methods rely on manual operations to acquire fine textures and geometry, and struggle to reconstruct view-dependent colors. With the advent of Neural Radiance Fields, neural rendering for novel view synthesis has been increasingly applied to large-scale scene reconstruction. The original NeRF [Mildenhall et al. 2020] represents the scene as an MLP, which maps positional encodings of spatial locations and directions to attributes such as color and volumetric density, and utilizes volume rendering principles for realistic rendering outcomes. Recently, there has been a notable increase in research efforts to adapt NeRF for large-scale scenes such as oblique Photography datasets. Some approaches [Mi and Xu 2023; Song and Zhang 2023; Tancik et al. 2022; Turki et al. 2022] adopt a divide-and-conquer strategy, segmenting the scene into chunks to perform parallel reconstruction, and then merging them to represent the entire scene holistically. GridNeRF [Xu et al. 2023] combines a multi-resolution ground feature plane representation with vanilla NeRF incorporating position-encoded inputs, enabling a collaborative learning process for rendering. Moreover, BungeNeRF [Xiangli et al. 2022] employs residual networks to learn multi-scale features, fitting scenes with dramatic changes in altitude.

*Real-Time Rendering.* Many methods accelerate rendering by reducing the volume of network queries [Kurz et al. 2022; Neff et al. 2021; Song et al. 2019] or by decomposing larger MLPs to facilitate parallel processing [Reiser et al. 2021]. Some works [Chen et al. 2022; Müller et al. 2022; Sara Fridovich-Keil and Alex Yu et al. 2022; Sun et al. 2022; Yu et al. 2021] reach it by introducing explicit structures, such as regular 3D voxel grids or octrees, storing features to replace partial or complete network. Other approaches seek to circumvent extensive network queries by baking models into explicit structures. For example, SNeRG [Hedman et al. 2021] extracts a sparse 3D voxel grid that stores density, diffuse color, and specular color. During rendering, each ray only needs to traverse a small MLP once. On the contrary, certain works [Božić et al. 2022; Chen et al. 2023; Guo et al.

2023; Liu et al. 2023; Tang et al. 2022; Wan et al. 2023; Yariv et al. 2023] utilize optimized surfaces with textures to represent scenes, integrating into modern computer graphics rendering pipelines. While achieving interactive frame rates on commercial devices, they fall short in rendering quality and memory efficiency for large-scale scenes. MERF [Reiser et al. 2023] employs a memory-efficient triplane combined with a sparse grid to represent the features of spatial points, achieving a significant reduction in memory consumption without compromising quality.

*Sampling in Rendering.* Various methods enhance rendering efficiency by refining the sampling strategy. NeRF [Mildenhall et al. 2020] and Mip-NeRF 360 [Barron et al. 2022] employ a coarse-to-fine strategy to concentrate on significant regions. DDNeRF [Dadon et al. 2023] adopts the Gaussian function instead of a piecewise-constant probability density function, achieving accurate density representation. DOnERF [Neff et al. 2021] and ENeRF [Lin et al. 2022] use depth information to reduce the number of sampling points. NeuSample [Fang et al. 2021] directly maps rays to sampling points through a single inference. In contrast to the network-based sampling method, Instant-NGP [Müller et al. 2022] skips empty space by explicit multi-resolution occupancy grids. While Adaptive Shells [Wang et al. 2023] and HybridNeRF [Turki et al. 2023] refine the sampling interval size across different spatial locations by optimizing the spatially-varying parameter, compressing the sampling area.

*Regularizations of Neural Fields.* In addition to these advancements, subsequent work on NeRF has introduced various regularization terms to enhance its performance. Indeed, in the realm of neural fields, numerous studies have incorporated regularization terms to encourage the smoothness of networks, such as penalties on the norms of Jacobians and Hessians [Drucker and Le Cun 1991; Moosavi-Dezfooli et al. 2019] or encouragement of smaller Lipschitz constants for weights [Liu et al. 2022]. In NeRF-related works, Plenoxels [Sara Fridovich-Keil and Alex Yu et al. 2022] proposes a total variation loss to minimize differences in features among adjacent voxels, and RegNeRF [Niemeyer et al. 2022] aims to encourage the continuity of volumetric density changes. Additionally, sparse regularization [Reiser et al. 2023; Yu et al. 2021] is employed in many real-time rendering schemes to eliminate irrelevant features.

### 3 BACKGROUND

NeRF [Mildenhall et al. 2020] employs an MLP to represent a scene as a continuous volumetric function  $\mathcal{F} : (\mathbf{p}, \mathbf{d}) \mapsto (\sigma, \mathbf{c})$ , mapping positional encodings of 3D points  $\mathbf{p} \in \mathbb{R}^3$  and normalized directions  $\mathbf{d} \in \mathbb{S}^2$  to volumetric density  $\tau(\mathbf{p})$  and color  $\mathbf{c}(\mathbf{p}, \mathbf{d})$ .

To render the corresponding color of a pixel, a ray  $\mathbf{r} = \mathbf{o} + t\mathbf{d}$  is first emitted from the origin  $\mathbf{o}$  along view directions  $\mathbf{d}$ , where dozens of points  $\{\mathbf{p}_i = \mathbf{o} + t_i\mathbf{d} \mid i = 1, \dots, n, t_i < t_{i+1}\}$  are sampled to estimate their volumetric density and features. These estimates are integrated through the numerical integral form of volumetric rendering to synthesize the final color [Max 1995]:

$$\hat{\mathbf{C}}(\mathbf{r}) = \sum_{i=1}^n \omega_i \mathbf{c}_i, \quad \omega_i = T_i \alpha_i, \quad (1)$$

where  $\alpha_i = (1 - e^{-\tau_i \delta_i})$  is the opacity of the sample  $\mathbf{p}_i$ , with  $\delta_i = t_{i+1} - t_i$  being the distance between adjacent samples.  $T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)$  is the accumulated transmittance from  $\mathbf{p}_1$  to  $\mathbf{p}_{i-1}$ . Subsequently, the MLP is optimized by minimizing the mean squared error between the predicted colors  $\{\hat{\mathbf{C}}(\mathbf{r})\}$  along rays emitted from the camera and the ground-truth colors from the input images.

MERF [Reiser et al. 2023] employs a low-resolution voxel grid  $\mathbf{V} \in \mathbb{R}^{L \times L \times L \times 8}$  and a high-resolution triplane  $\{\mathbf{P}_i \in \mathbb{R}^{R \times R \times 8} \mid i = x, y, z\}$  to represent the scene. For each sample  $\mathbf{p}$ , the features are obtained by adding the results of trilinear interpolation on the sparse grid and bilinear interpolation on the three planes, respectively:

$$\mathbf{t}(\mathbf{p}) = \mathbf{V}(\mathbf{p}) + \mathbf{P}_x(\mathbf{p}) + \mathbf{P}_y(\mathbf{p}) + \mathbf{P}_z(\mathbf{p}). \quad (2)$$

Similar to SNeRG [Hedman et al. 2021], to disentangle diffuse color and specular color, the feature is split into three components  $\mathbf{t}(\mathbf{p}) = [\tilde{\tau}, \tilde{\mathbf{c}}^d, \tilde{\mathbf{f}}]$ . And exponential  $\exp(\cdot)$  and sigmoid activation function  $\sigma(\cdot)$  are applied to obtain separately volumetric density  $\tau \in \mathbb{R}$ , diffuse color  $\mathbf{c}^d \in \mathbb{R}^3$ , and specular features  $\mathbf{f} \in \mathbb{R}^4$ :

$$\tau = \exp(\tilde{\tau}), \quad \mathbf{c}^d = \sigma(\tilde{\mathbf{c}}^d), \quad \mathbf{f} = \sigma(\tilde{\mathbf{f}}). \quad (3)$$

After alpha composition as in Eq. (1), the diffuse color and specular features are concatenated with the direction and fed into a deferred MLP  $\mathcal{G}$  to obtain the final color:

$$\hat{\mathbf{C}}(\mathbf{r}) = \sum_{i=1}^n \omega_i \mathbf{c}_i^d + \mathcal{G}(\mathbf{F}, \mathbf{d}), \quad \mathbf{F} = \left[ \sum_{i=1}^n \omega_i \mathbf{c}_i^d, \sum_{i=1}^n \omega_i \mathbf{f}_i \right]. \quad (4)$$

After the training, MERF performs a full-resolution rendering on all images to determine the areas where the opacity and weights exceed a predefined threshold. These areas are then stored as the occupied space, preserving corresponding volumetric density, diffuse colors, and specular colors. The information is utilized for subsequent real-time rendering on the web.

## 4 METHOD

Based on the model combining sparse grids and triplanes, we propose a method for high-quality, real-time rendering suitable for oblique photography. Section 4.1 introduces a novel explicit two-dimensional occupancy plane and its optimization to obtain a compact and detailed representation for efficient sampling. Section 4.2 proposes a novel regularization term to address the view extrapolation issue for oblique photography with limited tilt angles. The entire optimization process is presented in Section 4.3. Section 4.4 explains the rapid baking of scene features for real-time rendering from arbitrary viewpoints based on the proposed model.

### 4.1 Occupancy Plane

For reconstruction from aerially captured data, it is beneficial to align the  $xy$ -plane of the world coordinate system with the ground, so that the  $z$ -axis is perpendicular to the ground and points upward. With this configuration, we make two primary assumptions:

- (1) For any point on the  $xy$ -plane, elements in the scene are contained within a single continuous interval along the  $z$ -axis, i.e., no object exists isolated in mid-air;
- (2) As the  $z$ -value increases, the occupancy within the scene gradually becomes sparser, signifying a decrease in elements or structures at higher elevations.

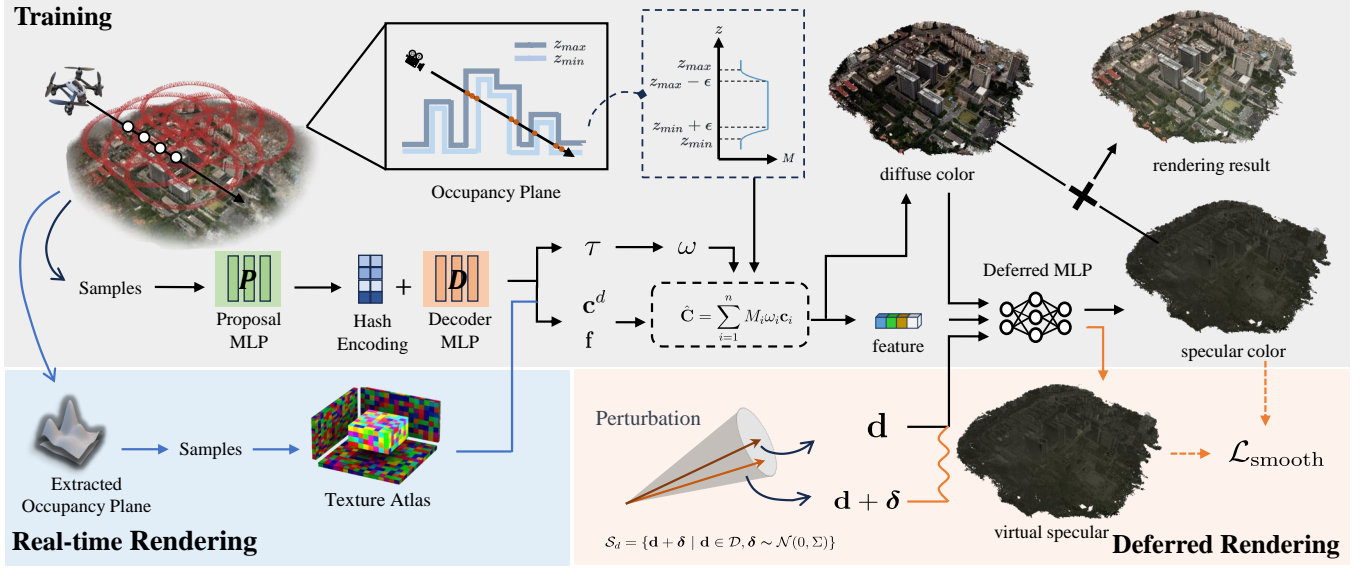


Fig. 2. Overview of our Oblique-MERF pipeline. During training, we introduce a 2D plane to represent the occupied space as a sandwiched region between two height field surfaces. For sampling points on rays, occupancy masks retrieved from the occupancy plane, are used as multipliers in the volume rendering process (section 4.1). Additionally, we incorporate a smoothness regularization for view-dependent color to minimize variations in specular color with viewing direction (section 4.2). Post-training, spatial occupancy information is directly extracted from the occupancy plane, and corresponding features are stored for real-time rendering (section 4.4).

Based on these assumptions, we represent the occupied space as a sandwiched region between two height field surfaces over the ground, corresponding to the lower and upper bounds of the  $z$  coordinates, respectively (see the inset figure). To parameterize the occupied space, we sample the  $xy$ -plane with a high-resolution  $M \times M$  grid, and store the heights  $z_{\min}(x, y)$  and  $z_{\max}(x, y)$  of the two surfaces at each sample grid point  $(x, y)$ , resulting in a representation  $\mathcal{P}_o \in \mathbb{R}^{M \times M \times 2}$ . We refer to this as the occupancy plane in the following. The heights  $z_{\min}(x, y)$  and  $z_{\max}(x, y)$  define a continuous occupied internal  $\mathcal{I}_{(x, y)} = [z_{\min}(x, y), z_{\max}(x, y)]$  for the  $z$  coordinates corresponding to each grid point  $(x, y)$ . We set the probability of occupancy outside this interval to be zero, so that we can exclude the points outside this interval when computing the color for a ray. To maximize memory efficiency, we would like to compress the interval as much as possible. Thus, we introduce a loss function term to penalize the span of the interval:

$$\mathcal{L}_{\text{occ}} = \sum_{(x, y) \in \mathcal{S}} (z_{\max}(x, y) - z_{\min}(x, y))^2, \quad (5)$$

where  $\mathcal{S}$  denotes the set of grid points for the occupancy plane.

However, simply compressing the occupancy intervals using  $\mathcal{L}_{\text{occ}}$  can affect the reconstruction quality if significant elements in the scene are left outside the sandwiched region. To avoid this issue, we integrate the occupancy plane into the volume rendering process, to ensure the final occupancy intervals are sufficient to represent the scene. Specifically, for the vertical line over a sample grid point  $(x, y)$

in the occupancy plane, we derive a differentiable occupancy function for the points along the line based on their  $z$  coordinates and the occupancy interval  $[z_{\min}(x, y), z_{\max}(x, y)]$  (below we ignore the arguments  $(x, y)$  for  $z_{\min}$  and  $z_{\max}$  to simplify the presentation):

$$M_{(x, y)}(z; \mathcal{P}_o) = \begin{cases} 1 & \text{if } z \in [z_{\min} + \epsilon, z_{\max} - \epsilon], \\ 0 & \text{if } z \in (-\infty, z_{\min}) \cup (z_{\max}, \infty), \\ (z - z_{\min})^q / \epsilon^q & \text{if } z \in [z_{\min}, z_{\min} + \epsilon], \\ (z_{\max} - z)^q / \epsilon^q & \text{if } z \in [z_{\max} - \epsilon, z_{\max}]. \end{cases} \quad (6)$$

Here we use a threshold  $\epsilon$  to introduce two buffer zones  $[z_{\min}, z_{\min} + \epsilon]$  and  $[z_{\max} - \epsilon, z_{\max}]$  near the endpoints of  $[z_{\min}, z_{\max}]$ , where the occupancy function transitions smoothly and monotonically from 0 to 1 using power functions. We set the exponent parameter  $q = 2$  in this paper. During volume rendering, for a sample point  $\mathbf{p}_i = (x_i, y_i, z_i)$  on a ray  $\mathbf{r}$ , we project it onto the  $XY$  plane and find the nearest sample grid point  $(\bar{x}_i, \bar{y}_i)$  to the projection  $(x_i, y_i)$ , and obtain a value for  $\mathbf{p}_i$  using the occupancy function at  $(\bar{x}_i, \bar{y}_i)$ :

$$M_{(x_i, y_i)}(z_i; \mathcal{P}_o) = M_{(\bar{x}_i, \bar{y}_i)}(z_i; \mathcal{P}_o). \quad (7)$$

The value is then combined with the weight from volume rendering, to determine the contribution of the feature at  $\mathbf{p}_i$  to the final color

$$\hat{\mathbf{C}}(\mathbf{r}) = \sum_{i=1}^n M_{(x_i, y_i)}(z_i; \mathcal{P}_o) \omega_i \mathbf{c}_i. \quad (8)$$

This is used to define a photometric loss  $\mathcal{L}_{\text{rgb}}$  that penalizes the deviation between the predicted color and the ground truth (see Eq. (12)). The occupancy value in (7) correlates the predicted color with the occupancy interval, such that the photometric loss suppresses further compression of the occupancy interval when its



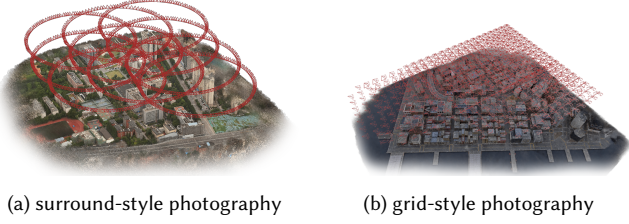


Fig. 3. Camera trajectories from two oblique photography methods.

endpoints with high weights, helping to achieve both rendering quality and memory efficiency. In addition, the occupancy value improves the efficiency of computation, as it can be utilized to filter out sampling points outside the occupied region. This significantly reduces the number of sampling points that need to be evaluated.

#### 4.2 Smoothness Regularization for View-dependent Color

Another challenge in large-scale scene reconstruction is the extrapolation issue of view-dependent colors. As shown in Fig. 3, when a drone flies through the entire scene, the conventional approach involves capturing images with limited pitch angles, either by orbital capture at a consistent angle relative to the ground plane or by employing a multi-camera system to photograph the scene in a grid pattern. The limited range of pitch angles in these photography methods often results in abnormal specular colors in the reconstructed scene, particularly when the scene is observed from a horizontal or upward perspective that is outside the range of view angles in the captured images (see Fig. 4b).

In our context, the deferred MLP  $\mathcal{G}$  that generates the color (see Eq. (4)) is supervised from a sparse set of input viewpoints. When rendering under extrapolated viewpoints, it tends to shift towards high-frequency components, leading to extreme highlights in the images. Our key observation is that adjacent viewpoints in real-world scenes often exhibit similar specular reflection colors, aligning with the local consistency seen in BRDF on smooth surfaces. We incorporate this prior into our model to encourage smoothness in view-dependent colors in unseen viewpoints. A typical approach is to impose constraints on the network parameters to achieve this continuity [Drucker and Le Cun 1991; Hoffman et al. 2019; Liu et al. 2022; Moosavi-Dezfooli et al. 2019]. For example, LipschitzMLP [Liu et al. 2022] achieves smoothness of the output by constraining the Lipschitz constant of the network. However, it enforces smoothness with respect to all inputs of the MLP (i.e., the viewing direction, the diffuse color, and the specular feature), whereas we only require smoothness with respect to the viewing direction; this could lead to over-constraints of the MLP parameters and may hinder the optimization process. To address this issue, we propose a regularization that enforces the smoothness of network output concerning the viewing direction only. Specifically, for a known viewing direction  $\mathbf{d} \in \mathcal{D}$  in the training set, we apply a small Gaussian perturbation to define a sampling space for the viewing direction.

$$\mathcal{S}_d = \{\mathbf{d} + \boldsymbol{\delta} \mid \mathbf{d} \in \mathcal{D}, \boldsymbol{\delta} \sim \mathcal{N}(\mathbf{0}, \Sigma)\}, \quad (9)$$

where  $\Sigma$  is a hyperparameter that determines the sampling range. Then, we introduce a loss to penalize large changes between the

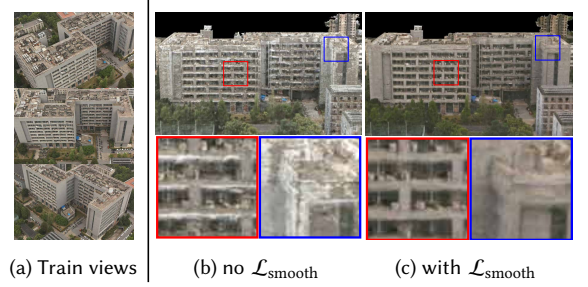


Fig. 4. In (a), we present training views captured around a building at limited tilt angles. (b) and (c) illustrate the real-time rendering results from a novel extrapolation viewpoint without and with  $\mathcal{L}_{\text{smooth}}$ , respectively. The introduction of the smoothness regularization yields renderings that are smoother and more consistent.

deferred MLP’s outputs for  $\mathbf{d}$  and the perturbed directions

$$\mathcal{L}_{\text{smooth}} = \sum_{\mathbf{d} \in \mathcal{D}} \sum_{\mathbf{s} \in \mathcal{S}_d} S(\mathbf{d}, \mathbf{s}) \|\mathcal{G}(\mathbf{F}, \mathbf{s}) - \mathcal{G}(\mathbf{F}, \mathbf{d})\|_2^2, \quad (10)$$

where  $S(\cdot, \cdot)$  denotes cosine similarity. In our experiments, this regularization term significantly enhances the robustness of the deferred MLP across interpolated and extrapolated viewpoints. It effectively mitigates the instability of view-dependent colors under new viewpoints while ensuring rendering quality and 3D consistency.

#### 4.3 Optimization

Our model is trained using a loss function written as a weighted sum:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{rgb}} + \lambda_2 \mathcal{L}_{\text{S3IM}} + \lambda_3 \mathcal{L}_{\text{distortion}} + \lambda_4 \mathcal{L}_{\text{interval}} + \lambda_5 \mathcal{L}_{\text{sparsity}} + \lambda_6 \mathcal{L}_{\text{entropy}} + \lambda_7 \mathcal{L}_{\text{occ}} + \lambda_8 \mathcal{L}_{\text{smooth}}. \quad (11)$$

Here  $\mathcal{L}_{\text{occ}}$  and  $\mathcal{L}_{\text{smooth}}$  are defined in Eq. (5) and Eq. (10), respectively.  $\mathcal{L}_{\text{rgb}}$  is a photometric loss that penalizes the disparity between the rendered images and the ground truth images, using the Charbonnier loss [Charbonnier et al. 1997] as a robust norm:

$$\mathcal{L}_{\text{rgb}} = \sum_{\mathbf{r} \in \mathcal{R}} \sqrt{\|\mathbf{C}(\mathbf{r}) - \hat{\mathbf{C}}(\mathbf{r})\|_2^2 + \epsilon_c}, \quad (12)$$

where  $\mathcal{R}$  is the set of training rays, and  $\epsilon_c = 10^{-6}$  is a parameter to ensure smoothness.  $\mathcal{L}_{\text{S3IM}}$  is the S3IM loss from [Xie et al. 2023] to enforce structural similarity between the rendered and input images. The interval loss  $\mathcal{L}_{\text{interval}}$  and the distortion loss  $\mathcal{L}_{\text{distortion}}$  are both adopted from [Barron et al. 2022]; the former aligns the weight distribution predictions of the proposal MLP and the NeRF MLP to rationalize the sample point distribution, and the latter reduces floater artifacts.  $\mathcal{L}_{\text{sparsity}}$  is a sparsity loss defined using randomly sampled points in the occupied space to encourage lower opacity:

$$\mathcal{L}_{\text{sparsity}} = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p}_i \in \mathcal{P}} \alpha_i. \quad (13)$$

This helps to address the issue of near-camera foggy artifacts resulting from inaccurate volume density estimation that often occurs in large scenes. Additionally, following [Kim et al. 2022], we randomly sample a number of rays  $\mathcal{R}$  from high altitude to the ground, calculate the opacity  $\alpha_i$  of sample points  $\mathbf{p}_i$  along each ray  $\mathbf{r}$ , and

combine it with the previous occupancy value to derive an entropy loss for the discrete density variable in the occupied space:

$$\mathcal{L}_{\text{entropy}} = -\frac{1}{|\mathcal{R}|} \sum_{\mathbf{r} \in \mathcal{R}} \sum_{\mathbf{p}_i \in \mathbf{r}} p(\mathbf{p}_i) \log(p(\mathbf{p}_i)), \quad (14)$$

where  $p(\mathbf{p}_i) = M_i \alpha_i / (\sum_{\mathbf{p}_i \in \mathbf{r}} M_i \alpha_i)$ . This encourages the occupancy probability of spatial points to approach either 0 or 1, which ensures consistency between training and real-time rendering.

During training, we initialize the weight for  $\mathcal{L}_{\text{occ}}$  to a small value and gradually increase it. In this way, the training first focuses on reconstruction of the scene, and then incrementally compresses the occupied space while maintaining the reconstruction quality. Further details can be found in the supplementary materials.

#### 4.4 Real-time Rendering

After training, we store the features identified as occupied by the occupancy plane for subsequent rendering. Existing approaches [Hedman et al. 2021; Reiser et al. 2023] construct a 3D voxel grid using the NeRF model for volume density evaluation or perform a full-resolution rendering of the training set for weight evaluation. This often requires dozens of hours for large-scale, high-resolution scenes. In contrast, we efficiently determine whether a voxel grid should store features based on the high-resolution occupancy plane obtained during training, which is completed in just a few minutes. Similar to SNeRG [Hedman et al. 2021] and MERF [Reiser et al. 2023], we store voxel grids as sparse blocks and the deferred MLP as floating-point arrays. For the 2D triplane, we employ a texture map with high resolution in the  $xy$  direction and low resolution in the  $z$  direction, according to the upper and lower bounds of the occupancy plane. This approach saves a significant amount of video memory during real-time rendering. To skip empty space efficiently, we employ 2D max pooling to obtain multiple low-resolution occupancy planes. All these features are encoded in PNG format.

After the baking process, we follow MERF [Reiser et al. 2023] for real-time rendering but diverge in ray marching. Unlike the multi-resolution binary grid used to determine the current point’s occupancy status, the occupancy plane directly stores the start and end positions of the point’s sampling interval in the  $z$  direction. As the ray traverses the blank area, it can efficiently reach the next grid point or the starting sampling position of the current grid point through bounding box intersection detection. Experimental results indicate that this sampling strategy is notably faster than previous approaches, especially when overlooking the entire scene.

## 5 EXPERIMENTS

We evaluate our method in terms of rendering quality, memory consumption, and real-time rendering performance. In Section 5.1, we compare our method with a series of offline novel view synthesis methods and some real-time rendering methods. In Section 5.2, we validate the effectiveness of our smoothness regularization. In Section 5.3, we compare the rendering quality, memory, and occupancy ratio under different resolutions of sparse feature grids.

### 5.1 Real-time rendering on oblique photography dataset

*Dataset and Experiment Settings.* We employ two distinct datasets for evaluation: the *Matrix City* [Li et al. 2023] and *Campus-Oblique*

Table 2. Quantitative results on *Matrix City* and *Campus-Oblique* datasets.

	<i>Campus-Oblique</i>			<i>Matrix City</i>		
	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$
InstantNGP	22.49	0.583	0.540	23.09	0.612	0.707
Nerfacto	22.07	0.599	0.361	23.40	0.674	0.443
MobileNeRF	20.49	0.409	0.529	21.48	0.526	0.544
BakedSDF	21.86	0.537	0.498	22.09	0.582	0.652
MERF	23.44	0.667	0.299	24.50	0.679	0.454
Ours	<b>24.14</b>	<b>0.694</b>	<b>0.270</b>	<b>25.18</b>	<b>0.714</b>	<b>0.406</b>

datasets. *Matrix City* is a synthetic dataset, captured in a grid-style format typical of classic oblique photography. *Campus-Oblique* is a real-world dataset captured by ourselves using a surround-style approach. It encompasses three distinct scenes on a university campus. Two of them cover an area of about 120,000 square meters and comprise over 1,000 images each. The third one is even more extensive, covering approximately 300,000 square meters and containing over 3,000 images. For each scene, we use 99% of the images for training, and the remaining ones for testing. Our training code and baseline MERF [Reiser et al. 2023] is built upon the nerfstudio framework [Tancik et al. 2023], augmented with the tiny-cuda-nn [Müller 2021] extension. Our real-time viewer is implemented as a JavaScript web application, utilizing GLSL for rendering. We conduct comprehensive comparisons with several offline and real-time methods to evaluate the performance of our method. For offline models, we compare with established methods like NerFacto [Tancik et al. 2023] and Instant-NGP [Müller et al. 2022]. For real-time models, we compare with MobileNeRF [Chen et al. 2023] and BakedSDF [Yariv et al. 2023]. We use MERF as the baseline for the proposed method and set the triplane resolution  $R$  to 2048, the sparse grid resolution  $L$  to 512 and the occupancy plane resolution  $M$  to 512. We evaluate the rendering quality using a set of established metrics: peak signal-to-noise ratio PSNR, SSIM [Wang et al. 2004], and LPIPS [Zhang et al. 2018]. Additionally, we use GPU memory usage (VRAM), frames per second (FPS), and on-disk storage (DISK) as metrics for the efficiency of real-time rendering methods.

*Results.* In Tab. 2, we conduct a quantitative comparison of rendering quality between our method and both offline and real-time rendering methods. Our approach not only matches offline methods in all metrics but also outperforms competing real-time rendering solutions. For real-world scenes, our method demonstrates superior rendering quality, thanks to the color network’s robustness enhanced by the proposed smoothness regularization term. As shown in Fig. 5, unlike other models that blur natural scene details, our method preserves clear and high-frequency details. Our sampling space regularization, which specifically addresses scene geometry orthogonal to the ground, allows for the depiction of crisp surface details, resulting in sharper geometry compared to the baseline.

We evaluate the real-time rendering performance of our method and compare it with MobileNeRF [Chen et al. 2023], BakedSDF [Yariv et al. 2023], and MERF [Reiser et al. 2023] in Tab. 3. The evaluation is carried out at 1920×1080 resolution on an NVIDIA RTX 1650. Despite the inherently lower frame rates of volumetric rendering against mesh rasterization, our method excels in DISK and VRAM

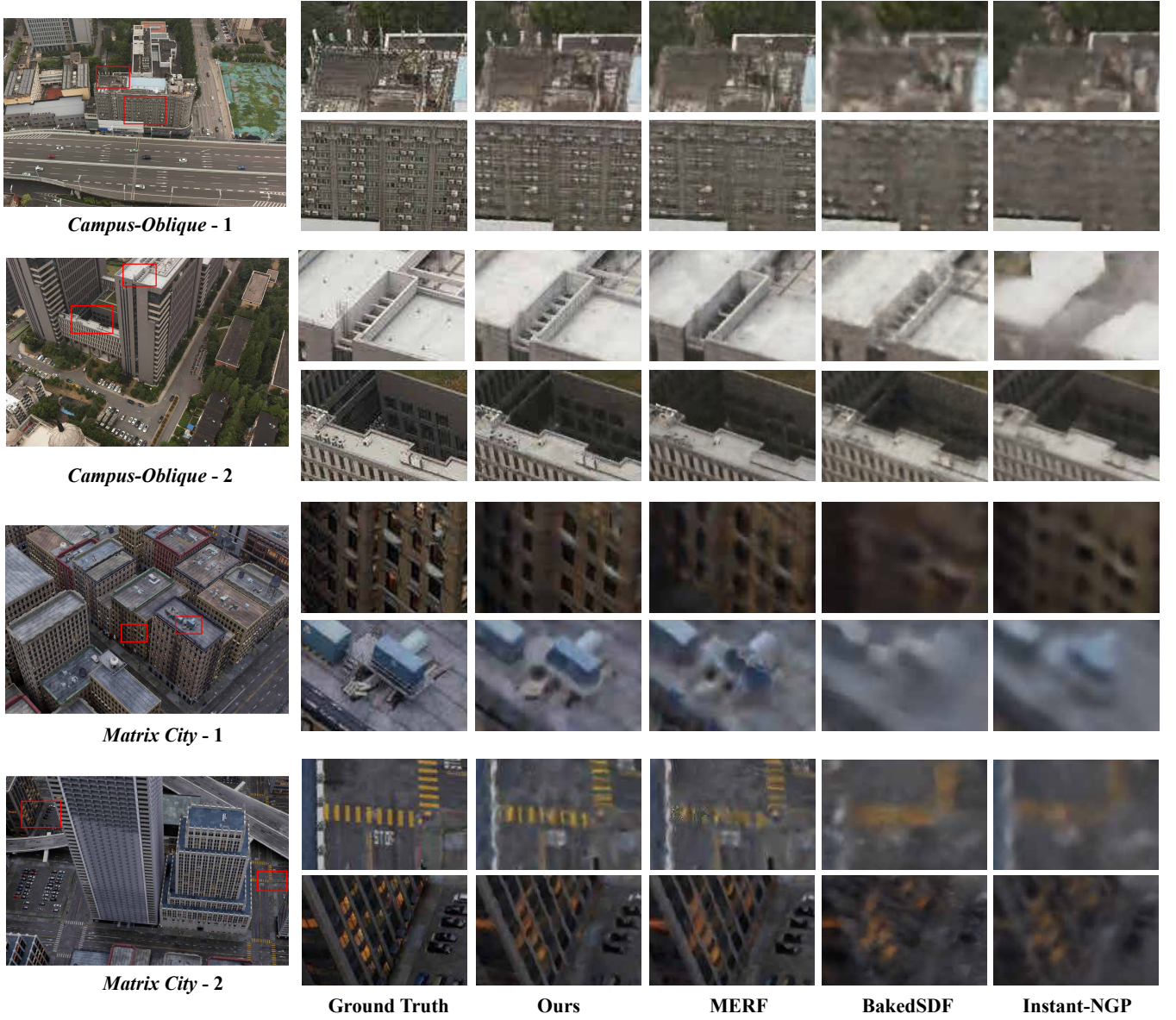


Fig. 5. Comparison on rendering quality for novel view between Oblique-MERF and other methods on the *Campus-Oblique* and *Matrix City* [Li et al. 2023] dataset.

efficiency, while delivering exceptional real-time rendering quality as shown in Tab. 2. Our compact occupancy space design yields requires less storage and improves frame rates. Remarkably, our sampling approach maintains consistent rendering speeds across scales, in contrast to MERF’s performance drop in large-scale scenes.

## 5.2 Color for extrapolation novel viewpoints

To validate the effectiveness of our smoothness regularization term for specular color introduced in Section 4.2, we employ a novel real-world dataset named *Campus-extra*. For the training set, we

Table 3. The performance for our model and other real-time methods.

	VRAM↓ (MB)	DISK↓ (MB)	FPS↑	
			high-altitude	low-altitude
MobileNeRF	1117.0	399.1	<b>63</b>	43
BakedSDF	595.0	509.0	58	<b>61</b>
MERF	185.9	97.7	8	31
Ours	<b>108.7</b>	<b>75.1</b>	32	42



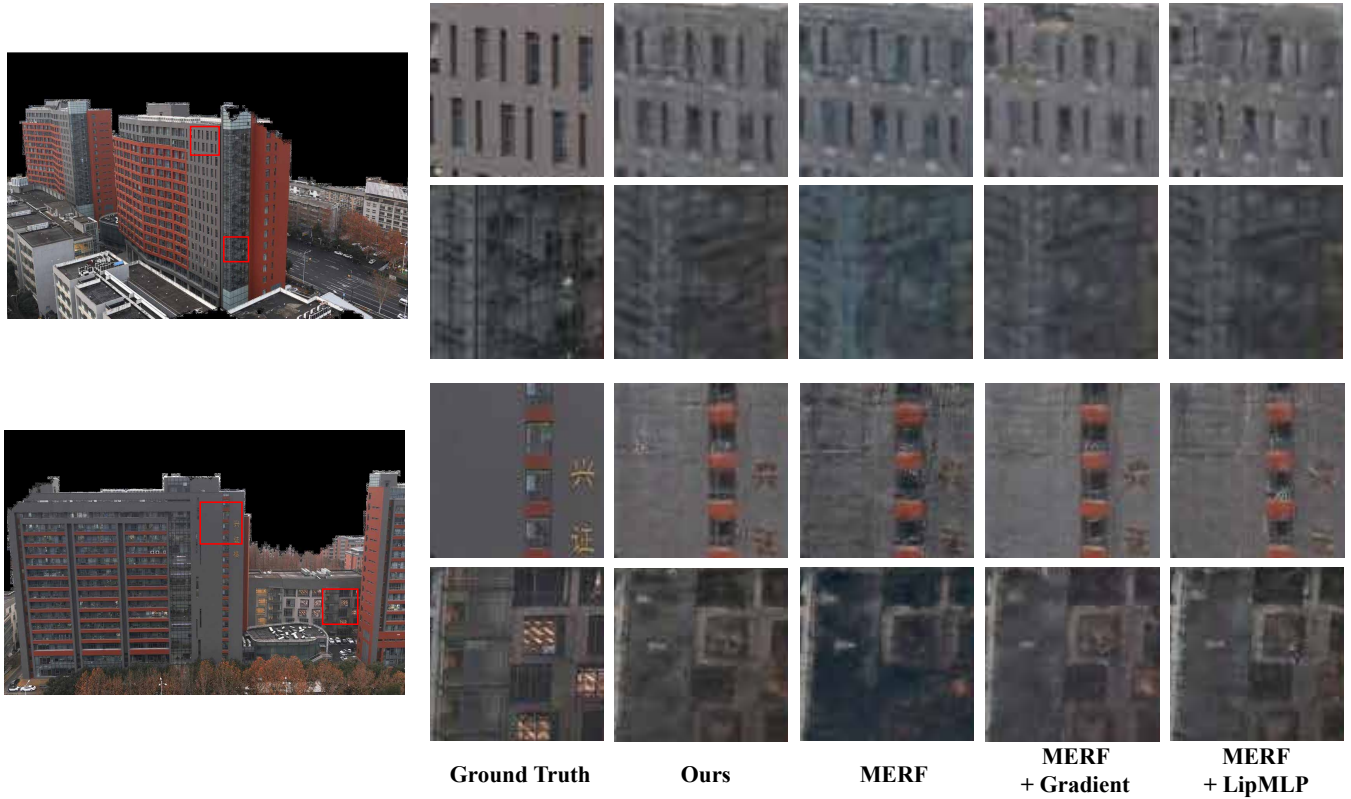


Fig. 6. Rendering quality comparison between Oblique-MERF and other MERF variants for test views on the *Campus-extra* dataset.

Table 4. Effectiveness of the Smoothness Regularization.

	Training views			Test views		
	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$
MERF	25.93	0.770	0.189	22.14	0.710	0.213
+LipMLP	25.86	0.768	0.188	22.42	0.729	0.201
+Gradient	25.97	0.772	0.187	<b>22.66</b>	0.732	0.199
Ours	<b>26.00</b>	<b>0.776</b>	<b>0.183</b>	<b>22.66</b>	<b>0.744</b>	<b>0.196</b>

follow the same technique as *Campus-Oblique* to capture 1486 images around a building, with a fixed tilt angle to the ground at approximately 60 degrees. For the test set, we simulate extrapolated viewpoints by capturing 140 images with tilt angles at around 25 degrees. To minimize the impact of unseen scenes, we adopt the NerfBusters [Warburg\* et al. 2023] protocol to mask test images to include only regions observed during training.

Tab. 4 showcases our method’s rendering performance, alongside a series of MERF variants under both interpolated and extrapolated viewpoints. In MERF+LipMLP, we replace the deferred MLP with a Lipschitz MLP and corresponding regularization [Liu et al. 2022]. In MERF+Gradient, we penalize the  $\ell_2$ -norm of the specular color’s gradient with respect to the viewing direction. Fig. 6 qualitatively demonstrates the results of these methods at test views. We find

that merely ensuring the network’s Lipschitz continuity slightly enhances the test view performance but can negatively impact the training views. This is potentially because it enforces smoothness on all network inputs including diffuse and specular features, which is redundant and hinders the optimization. Furthermore, applying regularization on the specular color’s gradient markedly improves test set performance but can disrupt some views’ optimization, occasionally resulting in undesirable artifacts. Compared to these approaches, our method consistently excels on both the training and test sets, achieving an approximate improvement of 0.5dB PSNR over the baseline and producing visually superior results.

### 5.3 Abalation Study on Occupancy Space

Tab. 5 compares our method with the baseline MERF in rendering quality, memory usage, and occupancy ratio (OR) on one scene of the *Campus-Oblique* dataset. To more clearly illustrate the compactness of the occupied space optimized by our sampling strategy, we exclusively employed sparse feature grids at resolutions from 512 to 2048, omitting the use of triplane. During the training process, occupancy planes matching the sparse feature grids’ resolution are utilized. After training, we extract a 3D occupancy grid (requiring roughly 2 to 10 minutes), contrasting it with the one obtained from MERF’s baking process (about 2 to 7 hours). Our approach consistently yields a lower OR (approximately 60% of MERF’s), enhancing



Table 5. Comparison of rendering quality and memory usage. VRAM capacity is denoted in megabytes (MB).

Spatial Res.	512 <sup>3</sup>			1024 <sup>3</sup>			2048 <sup>3</sup>		
	PSNR↑	VRAM↓	OR↓	PSNR↑	VRAM↓	OR↓	PSNR↑	VRAM↓	OR↓
MERF	21.83	94.8	2.54%	23.35	441.0	1.97%	24.18	2478.0	1.34%
Ours	<b>22.84</b>	<b>61.3</b>	<b>1.49%</b>	<b>24.54</b>	<b>322.5</b>	<b>1.27%</b>	<b>25.24</b>	<b>1605.0</b>	<b>0.89%</b>

memory efficiency and rendering quality across all tested resolutions. This demonstrates that the regularization term introduced in Section 4.1 effectively eliminates redundant sampling areas within objects or in the air, aligning closely with scene geometry.

## 6 CONCLUSION

We introduced Oblique-MERF, a compact and robust model optimized for real-time NeRFs in large-scale scenes, specially designed for oblique photography. Our key contribution is an innovative adaptive two-dimensional occupancy plane that is integrated with volume rendering and optimized during training. This approach ensures a balance between memory efficiency and rendering quality, while avoiding prolonged baking after training. We also introduced a smoothness regularization term for specular color relative to viewing directions, producing more natural rendering results for novel extrapolated viewpoints. Compared to existing real-time rendering techniques, Oblique-MERF delivers superior rendering quality and lower memory usage, achieving higher real-time frame rates.

Although our method improves upon the baseline in terms of sampling space and real-time rendering rates, it still performs volume rendering similar to MERF [Reiser et al. 2023]. Compared to methods based on mesh rasterization, it performs slightly worse in real-time frame rates and faces challenges on devices with weaker GPUs. Furthermore, scalability remains an issue to be addressed. Oblique photography often requires reconstructing larger scenes. While our proposed occupancy plane offers a more efficient representation than 3D grids, resolution limitations still exist. Adopting divide-and-conquer strategies, similar to Block-NeRF [Tancik et al. 2022] and Mega-NeRF [Turki et al. 2022], could be a viable solution for enhancing representation ability of our model. Additionally, our smoothness prior mitigates artifacts in extrapolated viewpoints to some extent but does not faithfully reproduce view-dependent colors in the scene. Integrating our method with physically-based rendering to accurately simulate the changes in specular color with viewing direction represents a future research direction.

## ACKNOWLEDGMENTS

This research was supported by the National Natural Science Foundation of China (No.62122071, No.62272433), the Youth Innovation Promotion Association CAS (No. 2018495) and the Fundamental Research Funds for the Central Universities (No. WK3470000021).

## REFERENCES

Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. 2009. Building Rome in a day. In *2009 IEEE 12th International Conference on Computer Vision*. 72–79. <https://doi.org/10.1109/ICCV.2009.5459148>

Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. 2022. Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields. In *IEEE Conf.*

*on Computer Vision and Pattern Recognition (CVPR)*.

Aljaž Božič, Denis Gladkov, Luke Doukakis, and Christoph Lassner. 2022. Neural Assets: Volumetric Object Capture and Rendering for Interactive Environments. *arXiv:2212.06125 [cs.CV]*

P. Charbonnier, L. Blanc-Feraud, G. Aubert, and M. Barlaud. 1997. Deterministic edge-preserving regularization in computed imaging. *Trans. Img. Proc.* 6, 2 (feb 1997), 298–311. <https://doi.org/10.1109/83.551699>

Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. 2022. TensorRF: Tensorial Radiance Fields. In *European Conference on Computer Vision (ECCV)*.

Zhiqin Chen, Thomas Funkhouser, Peter Hedman, and Andrea Tagliasacchi. 2023. MobileNeRF: Exploiting the Polygon Rasterization Pipeline for Efficient Neural Field Rendering on Mobile Architectures. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

David Dadon, Ohad Fried, and Yacov Hel-Or. 2023. DDNeRF: Depth Distribution Neural Radiance Fields. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 755–763.

H. Drucker and Y. Le Cun. 1991. Double backpropagation increasing generalization performance. In *IJCNN-91-Seattle International Joint Conference on Neural Networks*, Vol. ii. 145–150 vol.2. <https://doi.org/10.1109/IJCNN.1991.155328>

Jiemin Fang, Lingxi Xie, Xinggang Wang, Xiaopeng Zhang, Wenyu Liu, and Qi Tian. 2021. NeuSample: Neural Sample Field for Efficient View Synthesis. *arXiv:2111.15552 (2021)*.

Jan-Michael Frahm, Pierre Fite-Georgel, David Gallup, Tim Johnson, Rahul Raguram, Changchang Wu, Yi-Hung Jen, Enrique Dunn, Brian Clipp, Svetlana Lazebnik, and Marc Pollefeys. 2010. Building Rome on a cloudless day. In *Proceedings of the 11th European Conference on Computer Vision: Part IV (Heraklion, Crete, Greece) (ECCV'10)*. Springer-Verlag, Berlin, Heidelberg, 368–381.

Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski. 2010. Towards Internet-scale multi-view stereo. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 1434–1441. <https://doi.org/10.1109/CVPR.2010.5539802>

Yasutaka Furukawa and Jean Ponce. 2010. Accurate, Dense, and Robust Multiview Stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32, 8 (2010), 1362–1376. <https://doi.org/10.1109/TPAMI.2009.161>

Yuan-Chen Guo, Yan-Pei Cao, Chen Wang, Yu He, Ying Shan, and Song-Hai Zhang. 2023. VMesh: Hybrid Volume-Mesh Representation for Efficient View Synthesis. In *SIGGRAPH Asia 2023 Conference Papers (SA '23)*. Association for Computing Machinery, New York, NY, USA, Article 17, 11 pages.

Peter Hedman, Pratul P. Srinivasan, Ben Mildenhall, Jonathan T. Barron, and Paul Debevec. 2021. Baking Neural Radiance Fields for Real-Time View Synthesis. *International Conference on Computer Vision (ICCV)* (2021).

Judy Hoffman, Daniel A. Roberts, and Sho Yaida. 2019. Robust Learning with Jacobian Regularization. *arXiv:1908.02729 [stat.ML]*

Hailin Jin, Stefano Soatto, and Anthony J. Yezzi. 2005. Multi-View Stereo Reconstruction of Dense Shape and Complex Appearance. *International Journal of Computer Vision* 63 (2005), 175–189. <https://api.semanticscholar.org/CorpusID:2067330>

Mijeong Kim, Seonguk Seo, and Bohyung Han. 2022. InfoNeRF: Ray Entropy Minimization for Few-Shot Neural Volume Rendering. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Andreas Kurz, Thomas Neff, Zhaoyang Lv, Michael Zollhöfer, and Markus Steinberger. 2022. AdaNeRF: Adaptive Sampling for Real-time Rendering of Neural Radiance Fields. (2022).

Yixuan Li, Lihan Jiang, Lining Xu, Yuanbo Xiangli, Zhenzhi Wang, Dahua Lin, and Bo Dai. 2023. MatrixCity: A Large-scale City Dataset for City-scale Neural Rendering and Beyond. *arXiv e-prints* (2023), arXiv–2308.

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 Conference Proceedings*.

Hsueh-Ti Derek Liu, Francis Williams, Alec Jacobson, Sanja Fidler, and Or Litany. 2022. Learning Smooth Neural Functions via Lipschitz Regularization. In *SIGGRAPH '22: Special Interest Group on Computer Graphics and Interactive Techniques Conference, Vancouver, BC, Canada, August 7 - 11, 2022*. 31:1–31:13.

Jeffrey Yunfan Liu, Yun Chen, Ze Yang, Jingkan Wang, Sivabalan Manivasagam, and Raquel Urtasun. 2023. Real-Time Neural Rasterization for Large Scenes. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. 8382–8393. <https://doi.org/10.1109/ICCV51070.2023.00773>

N. Max. 1995. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics* 1, 2 (1995), 99–108. <https://doi.org/10.1109/2945.468400>

Zhenxing Mi and Dan Xu. 2023. Switch-NeRF: Learning Scene Decomposition with Mixture of Experts for Large-scale Neural Radiance Fields. In *International Conference on Learning Representations (ICLR)*. <https://openreview.net/forum?id=PQ2zolZqvm>

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 *The European Conference on Computer Vision (ECCV)*.

- Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. 2019. Robustness via Curvature Regularization, and Vice Versa. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. Computer Vision Foundation / IEEE, 9078–9086.
- 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. *ACM Trans. Graph.* 41, 4, Article 102 (July 2022), 15 pages. <https://doi.org/10.1145/3528223>. 3530127
- Thomas Neff, Pascal Stadlbauer, Mathias Parger, Andreas Kurz, Joerg H. Mueller, Chakravarthy R. Alla Chaitanya, Anton S. Kaplanyan, and Markus Steinberger. 2021. DONeRF: Towards Real-Time Rendering of Compact Neural Radiance Fields using Depth Oracle Networks. *Computer Graphics Forum* 40, 4 (2021). <https://doi.org/10.1111/cgf.14340>
- Michael Niemeyer, Jonathan T. Barron, Ben Mildenhall, Mehdi S. M. Sajjadi, Andreas Geiger, and Noha Radwan. 2022. RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.
- Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. 2021. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *International Conference on Computer Vision (ICCV)*. 14335–14345.
- Christian Reiser, Richard Szeliski, Dor Verbin, Pratul P. Srinivasan, Ben Mildenhall, Andreas Geiger, Jonathan T. Barron, and Peter Hedman. 2023. MERF: Memory-Efficient Radiance Fields for Real-time View Synthesis in Unbounded Scenes. *ACM Trans. Graph.* 42, 4 (2023), 89:1–89:12.
- Sara Fridovich-Keil and Alex Yu, Matthew Tancik, Qinzhong Chen, Benjamin Recht, and Angjoo Kanazawa. 2022. Plenoxels: Radiance Fields without Neural Networks. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.
- Johannes L. Schönberger and Jan-Michael Frahm. 2016. Structure-from-Motion Revisited. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 4104–4113. <https://doi.org/10.1109/CVPR.2016.445>
- Kaiwen Song and Juyong Zhang. 2023. City-on-Web: Real-time Neural Rendering of Large-scale Scenes on the Web. [arXiv:2312.16457 \[cs.CV\]](https://arxiv.org/abs/2312.16457)
- Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. AutoInt: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. ACM, 1161–1170.
- Cheng Sun, Min Sun, and Hwann-Tzong Chen. 2022. Direct Voxel Grid Optimization: Super-fast Convergence for Radiance Fields Reconstruction. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.
- Matthew Tancik, Vincent Casser, Xinchun Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P. Srinivasan, Jonathan T. Barron, and Henrik Kretschmar. 2022. Block-NeRF: Scalable Large Scene Neural View Synthesis. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 8248–8258.
- Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salehi, Abhik Ahuja, David McAllister, and Angjoo Kanazawa. 2023. Nerfstudio: A Modular Framework for Neural Radiance Field Development. In *ACM SIGGRAPH 2023 Conference Proceedings (SIGGRAPH '23)*.
- Jiaxiang Tang, Hang Zhou, Xiaokang Chen, Tianshu Hu, Errui Ding, Jingdong Wang, and Gang Zeng. 2022. Delicate Textured Mesh Recovery from NeRF via Adaptive Surface Refinement. *arXiv preprint arXiv:2303.02091* (2022).
- Haithem Turki, Vasu Agrawal, Samuel Rota Bulò, Lorenzo Porzi, Peter Kotschieder, Deva Ramanan, Michael Zollhöfer, and Christian Richardt. 2023. HybridNeRF: Efficient Neural Rendering via Adaptive Volumetric Surfaces. [arXiv:2312.03160 \[cs.CV\]](https://arxiv.org/abs/2312.03160)
- Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. 2022. Mega-NeRF: Scalable Construction of Large-Scale NeRFs for Virtual Fly-Throughs. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 12922–12931.
- Ziyu Wan, Christian Richardt, Aljaž Božič, Chao Li, Vijay Rengarajan, Seonghyeon Nam, Xiaoyu Xiang, Tuotuo Li, Bo Zhu, Rakesh Ranjan, and Jing Liao. 2023. Learning Neural Duplex Radiance Fields for Real-Time View Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 8307–8316.
- Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing* 13, 4 (2004), 600–612.
- Zian Wang, Tianchang Shen, Merlin Nimier-David, Nicholas Sharp, Jun Gao, Alexander Keller, Sanja Fidler, Thomas Müller, and Zan Gojcic. 2023. Adaptive Shells for Efficient Neural Radiance Field Rendering. *ACM Trans. Graph.* 42, 6 (2023).
- Frederik Warburg\*, Ethan Weber\*, Matthew Tancik, Aleksander Holyński, and Angjoo Kanazawa. 2023. Nerfbusters: Removing Ghostly Artifacts from Casually Captured NeRFs.
- Changchang Wu. 2013. Towards Linear-Time Incremental Structure from Motion. In *2013 International Conference on 3D Vision - 3DV 2013*. 127–134. <https://doi.org/10.1109/3DV.2013.25>
- Yuanbo Xiangli, Linning Xu, Xingang Pan, Nanxuan Zhao, Anyi Rao, Christian Theobalt, Bo Dai, and Dahua Lin. 2022. BungeeNeRF: Progressive Neural Radiance Field for Extreme Multi-scale Scene Rendering. In *The European Conference on Computer Vision (ECCV)*.
- Zeke Xie, Xindi Yang, Yujie Yang, Qi Sun, Yixiang Jiang, Haoran Wang, Yunfeng Cai, and Mingming Sun. 2023. S3IM: Stochastic Structural Similarity and Its Unreasonable Effectiveness for Neural Fields. In *International Conference on Computer Vision (ICCV)*.
- Linning Xu, Yuanbo Xiangli, Sida Peng, Xingang Pan, Nanxuan Zhao, Christian Theobalt, Bo Dai, and Dahua Lin. 2023. Grid-guided Neural Radiance Fields for Large Urban Scenes. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 8296–8306.
- Lior Yariv, Peter Hedman, Christian Reiser, Dor Verbin, Pratul P. Srinivasan, Richard Szeliski, Jonathan T. Barron, and Ben Mildenhall. 2023. BakedSDF: Meshing Neural SDFs for Real-Time View Synthesis. In *ACM SIGGRAPH 2023 Conference Proceedings, SIGGRAPH 2023, Los Angeles, CA, USA, August 6-10, 2023*. ACM, 46:1–46:9.
- Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. 2021. PlenOctrees for Real-time Rendering of Neural Radiance Fields. In *International Conference on Computer Vision (ICCV)*.
- 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*.