Density Estimation-based Effective Sampling Strategy for Neural Rendering

Yunxiang He[1] and Xin Lou[1,2]
[1]School of Information Science and Technology, ShanghaiTech University
[2]Key Laboratory of Intelligent Perception Human-Machine Collaboration, Ministry of Education
Shanghai, China
Emails: {heyx1, louxin}@shanghaitech.edu.cn

***Abstract*—Expanding upon the foundational Neural Radiance Fields (NeRF) framework, neural rendering techniques have seen wide-ranging applications in 3D reconstruction and rendering. Despite the numerous attempts to accelerate the original NeRF, the ray marching process in many neural rendering algorithms remains a performance bottleneck, constraining their rendering speed. In this work, we introduce an innovative method for effective sampling based on density estimation to alleviate this bottleneck. The proposed method can effectively reduce the number of required occupancy grid access without compromising rendering quality. Evaluation and analysis results validate the effectiveness of the proposed approach, delivering notable improvements in rendering efficiency.**

***Index Terms*—Neural radiance fields (NeRF), neural rendering, kernel density estimation (KDE), ray marching**

I. Introduction

Synthesizing photorealistic images and videos is a fundamental topic in computer graphics. Conventional rendering techniques, including rasterization and ray tracing, have been the go-to tools for fabricating artificial images of scenes, utilizing geometric constructs (e.g., meshes or point clouds) and material properties as their primary input. Recently, innovation of Neural Radiance Fields (NeRF) [1] and its associated research have emerged, blending principles from classical computer graphics with machine learning approaches. Triggered by the seminal work of NeRF, the idea of neural volume rendering has been extended to many applications such as simultaneous localization and mapping (SLAM) [2], 3D content generation [3] and medical imaging [4].

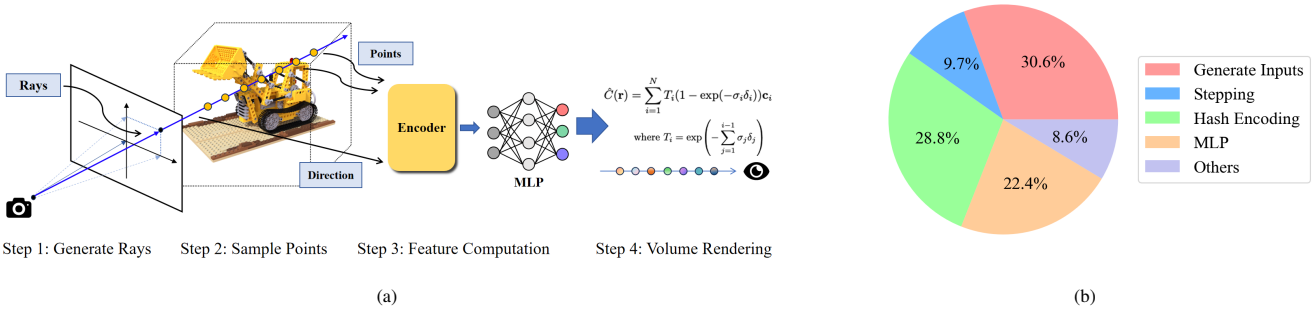
Fig. 1 illustrates the primary process of neural volume rendering, which can be divided into four key components:

1) Ray Generation: In this step, light rays associated with each pixel are generated based on the virtual camera's position and orientation.
2) Ray Sampling: This step involves generating sample points along the trajectories of the light rays.
3) Feature Computation: The sampled points undergo computations to determine their color and density in this step.
4) Volume Rendering: Based on the volume rendering formula, the color and density information from each sampled point is accumulated to produce the final color and density values for the corresponding pixel.

This work was supported by the Shanghai Rising-Star Program (21QC1401400) and Central Guided Local Science and Technology Foundation of China (YDZX20223100001001). Corresponding arthor: Xin Lou.

While NeRF-based methods have made significant advancements in neural rendering, they come with certain limitations, with the most critical one being the time overhead, in both reconstruction (training) and rendering (inference). To address the time overhead, a number of algorithm level optimizations have emerged to enhance the speed and efficiency of NeRF. Chen et al. [5] employ tensor decomposition to break down the 3D scene mesh into a combination of low-dimensional tensors. PlenOctrees [6] and Plenoxels [7] utilize explicit octree-based storage to enhance rendering speed, albeit at the expense of substantial storage demands. For illustration, an octree representation of a typical scene consumes over 1GB of storage. Mobile-NeRF [8] addresses storage and speed issues by baking NeRF into an explicit mesh and integrating it into the conventional rasterization pipeline, but it incurs a certain loss in quality. As one of the most efficient NeRF-based algorithms, the recent Instant-NGP [9] leverage occupancy grid for effective sampling point generation and multi-resolution hash encoding to reduce the size of neural network, i.e., Multilayer Perceptron (MLP). While the meticulously-engineered Instant-NGP can achieve real-time rendering for certain scenes (at a relatively low resolution), it experiences a decline in rendering speed when dealing with more intricate geometries. When delving into the Instant-NGP framework, it is found that the ray marching step during the sampling process can potentially become a bottleneck, attributable to the design of the occupancy grid. The main reason is twofold:

- Geometric Reconstruction Quality. NeRF, being a method that optimizes constraints based on RGB image data, generates its density distribution as a byproduct [10]. Consequently, this yields an optimized density grid that fails to accurately represent the object's density distribution in space. In fact, the density grid from Instang-NGP not only encompasses objects in space but also includes a significant amount of low-density, high-frequency noise. This, in turn, results in a multitude of invalid samples during the Ray Marching process, affecting not only efficiency but also potentially leading to undesirable artifacts.
- Parallel Efficiency. Our profiling analysis (Fig. 1(b)) in Instant-NGP [9] reveals that the Ray Marching step not



(a) (b)

Fig. 1: (a) Pipeline of typical NeRF-based neural rendering algorithms. (b) Time breakdown of Instant-NGP on NVIDIA 3090Ti.

only accounts for more than 30% of the overall computation time but also, due to its inherently sequential nature, prevents unrelated systems, such as interpolation, from compensating for each other's latency. Consequently, while the occupancy grid seems to enhance Ray Marching speed, its efficiency falls short of achieving the desired outcome.

To address the above mentioned drawbacks in ray marching within existing NeRF-based algorithms, we propose a density estimation-based method for effective sampling point generation. In particular, we first introduce a density filtering scheme to remove the extraneous high-frequency noise within the density grid, which allows the grid to faithfully represent the object's shape. Based on that, we further propose a density estimation method that leverages a Gaussian prior for cost-effective, coarse sampling to estimate the light density distribution. This approach minimizes the required number of sampling points for further feature computation without sacrificing the rendering quality. Experimental results verify the effectiveness of the proposed method, in terms of both rendering quality and computation efficiency.

II. Overview of the Proposed Method

The overview of the proposed density estimation-based effective sampling point generation method is illustrated in Fig. 2. The fundamental idea behind is that, in the inference process of neural volume rendering, the inclusion of accurate prior information from an object's geometric model can significantly enhance the efficiency of the Ray Marching process. But the occupancy grid training strategy employed in many NeRF algorithms, e.g., Instant-NGP, does not impose constraints on geometric information. As a result, while it essentially reconstructs the spatial occupancy of the object, it is also accompanied by a substantial amount of noise.

To take advantage of the prior information from the geometric, we propose to first apply filtering techniques to the grid to remove the high-frequency and low-amplitude noises. Once we have established an accurate object occupancy model, we further utilize the continuity constraints of the objects to estimate the density distribution curve of light through a low-density coarse sampling. This approach can significantly reduce the number of occupancy grid querying during the generation of effective ray sampling points.

III. Occupancy Grid Filtering

The presence of noise in the occupancy grid not only undermines the efficiency of ray marching but also causes artifact-related issues, as highlighted in [11]. When we view the reconstructed occupancy grid as a collection of 3D signals, addressing this noise problem becomes a matter of employing appropriate filtering techniques.

As shown in Fig. 3, simply increasing the threshold in the Instang-NGP may result in loss of scene details. This is due to the fact that, apart from the high-density occupied areas, low-density regions near the surface of objects in the scene often contain important texture details.

Therefore, we develop a filtering strategy that focuses on the low-density or high-frequency attributes of the noise in the occupancy grid. We utilize a density filter for the scene, implemented as a cascaded process comprising a Gaussian filter followed by a threshold filter. The Gaussian filter primarily addresses high-frequency components, while the subsequent threshold filter effectively eliminates low-density residuals. Specifically, the proposed density grid filtering can be formalized as follows:

$$\hat{O}(\mathbf{r}) = O(\mathbf{r}) \circledast \mathscr{G}_{3\times3\times3} \geq \mathscr{T}, \quad \mathscr{T} = 0.05,$$

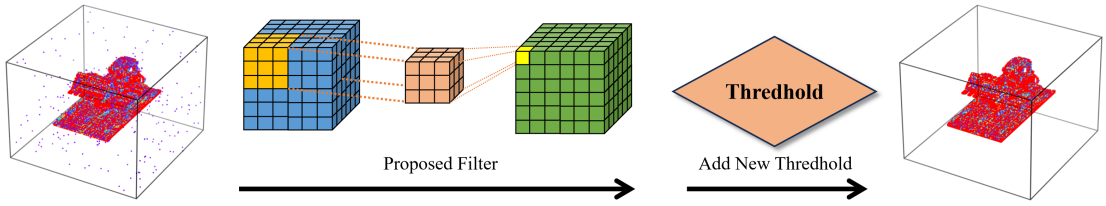
where $\mathscr{G}_{3\times3\times3}$ is a Gaussian filter, O represents occupied grids, and $\mathscr{T}$ is the threshold.

To showcase the performance, we use the Ficus scene with complex geometry as an example. As illustrated in Fig. 3, the Gaussian + threshold filtering approach enables us to effectively remove the excessive low-density, high-frequency artifacts while preserving essential object details (near the object surface). The filtered occupancy grid not only enables the following density estimation-based sample point generation, but also improves the efficiency the ray marching process. Our profiling results presented in Table I show that it can improve the efficiency of **Ray Marching Step** in Instang-NGP by 9.19% without any quality loss.

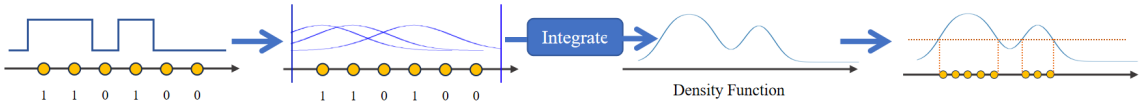
IV. Density Estimation-based Sampling Strategy

A. Overview

The overview of the proposed density estimation-based effective sampling point generation is illustrated in Fig. 2.



(a) Density Filtering.



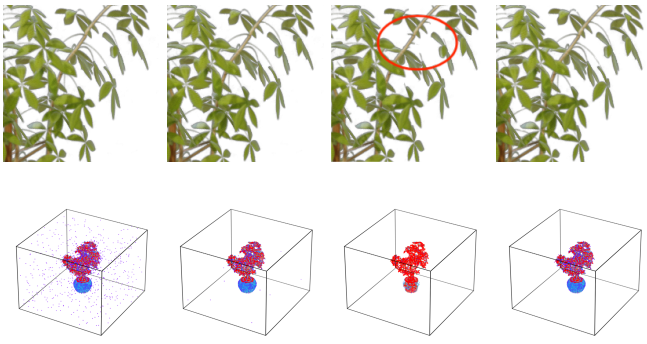
Step 1: Coarse Sampling

Step 2: Density Estimation

Step 3: Fine Sampling

(b) Density Estimation.

Fig. 2: Illustration of the proposed method. We first apply density filtering on the occupancy grid to establish an accurate object occupancy model. Based on that, we further utilize the continuity constraints of the objects to estimate the density distribution curve of light through a low-density coarse sampling.



(a) Threshold = 0.01 (b) Threshold = 0.02 (c) Threshold = 0.05 (d) Proposed Method

Fig. 3: Comparison of occupancy grid filtering using different methods and configurations. (a) The default setting in Instang-NGP with threshold=0.01. Results of (b) threshold=0.02 and (c) threshold=0.05. (d) Result of the proposed method.

TABLE I: RAY MARCHING TIME BEFORE AND AFTER OCCUPANCY GRID FILTERING

Scene	Instant-NGP (w.o. Filtering)	Proposed (with Filtering)	Acc. ratio (Percentage)
Chair	1.09	1.05	3.93%
Drums	1.48	1.42	4.68%
Ficus	1.51	1.21	25.36%
Hotdog	1.76	1.71	3.3%
Lego	1.57	1.55	1.44%
Materials	2.3	2.28	0.89%
Mic	1.11	0.86	29.14%
Ship	5.09	4.36	16.59%
Avg	1.6	1.47	9.19%

To estimate the density curve for the subsequent fine-grained sampling, an initial coarse sampling step is performed. The density curve is then obtained using probability density estimation methods. In this study, we utilize the widely adopted non-parametric technique called Kernel Density Estimation (KDE) for density estimation, as it does not assume any predefined data distribution. In particular, the following Gaussian function

$$K(u) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{u^2}{2})$$

is selected as the kernel function. It has been shown that Gaussian functions are a good prior for characterizing the distribution of objects in space [12]. The details of the density estimation method is described in Algorithm 1.

B. Coarse Sampling

We begin by conducting a low-density, uniform coarse sampling of the rays and employ the Occupancy Grid to determine the occupancy status of these sampled points. Unlike the coarse sampling in original NeRF, the goal of this stage is to gather occupancy information for the coarse sampling points along the rays through sparse interactions with the Occupancy Grid. To reduce unnecessary computations, these sampling points are not included in the final sampling step.

C. Kernel Density Estimation

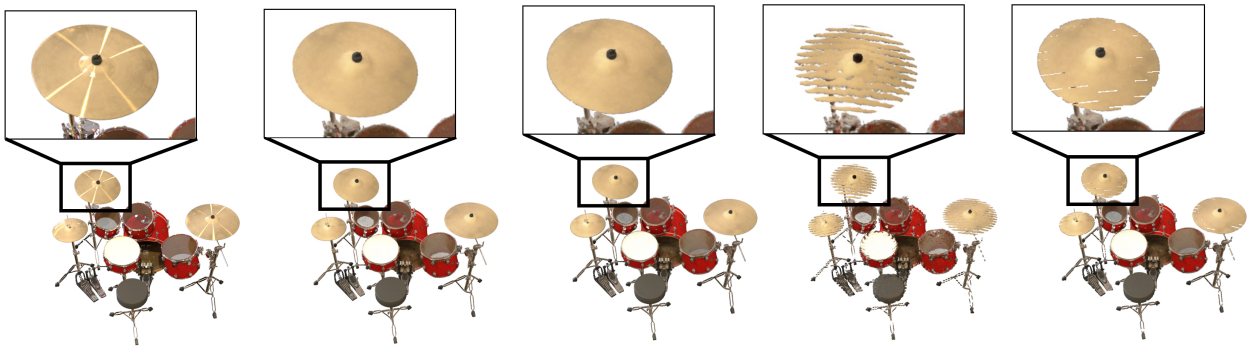
After obtaining occupancy information in the previous step, we employ Gaussian kernel density estimation to estimate the density profile along the curve. For the kernel density estimation function

$$\hat{f}_h(x) = \frac{1}{n} \sum_{i=1}^{n} K_h(x - x_i),$$

since the sampling points from the coarse sampling are uniformly distributed, we can discretize the Gaussian function and use a cost-effective accumulation method to compute the final density estimation as illustrated in Algorithm 1.

D. Fine Sampling

After obtaining the density profile, we select all points with an estimated density $\hat{f}(x) > \sigma$ using a specified threshold value σ. We then perform uniform sampling in the vicinity of these selected points, employing the same



(a) Ground Truth (b) Instant-NGP, Default (c) Ours, 192 Steps (d) Insant-NGP, 192 Steps (e) Ours, 96 Step

Fig. 4: Visual comparison of rendering results produced by Instant-NGP and our method under different configurations.

Algorithm 1 Discrete Density Estimation

```
function GEN-CURVE(NonZeroPointIndexs, Length)
    result ← [0,0,...,0]
    for all index in NonZeroPointIndexs do
        Offset ← index - ⌊Length / 2⌋
        for i in [0,1,...,Length − 1] do
            if i ≥ Offset and i < Offset + Length then
                result[i] += GaussianFunction[i − Offset]
```

sampling step as in the original Instant-NGP. As will be shown in the results, due to the scene's sparsity, we generate significantly fewer sample points per step compared to the Instant-NGP scheme, despite employing the same sampling step.

V. Evaluation and Analysis

This section presents the evaluation and analysis results to showcase the effectiveness of the proposed method. As a common practice, we use the peak signal-to-noise ratio (PSNR) to measure the rendering quality and use frames per second (FPS) under a certain resolution to measure the rendering speed. We use Instant-NGP as our benchmark for comparison, as it represents the state-of-the-art solution with a remarkable balance between speed, storage, and rendering quality. Other solutions are either too slow (TensoRF: ≤ 1 FPS) or consume excessive storage (Plenoxels: ≈ 2.59 GB).

In Fig. 4, we present a visual comparison of rendering results produced by both the original Instant-NGP and our proposed method. It is evident that, with 192 coarse sampling points, our proposed method yields results of nearly the same quality as the default settings of Instant-NGP, which uses 1024 steps. However, reducing the number of steps in Instant-NGP to match the 192 used in our method results in noticeable artifacts, as clearly visible in the zoomed-in region. Furthermore, for our proposed method, decreasing the coarse sampling points to 96 also introduces some artifacts. Therefore, 192 is the appropriate number of coarse sampling points for our proposed method.

Next, we present the results of our quantitative analysis in Table II. Instant-NGP relies on pure memory access to ascertain the validity of a point within an occupancy grid, whereas our proposed method necessitates several additional operations to estimate the curve. In addition to evaluating PSNR and FPS, we also estimate energy consumption per ray to assess efficiency. The results reveal that the 192-step version of our proposed method delivers rendering results with PSNR nearly identical to the default settings of Instant-NGP. Notably, our method achieves an equivalent FPS that is 1.5× higher, and the energy expended per ray is only 24.8%. This clearly demonstrates that the proposed method is significantly more efficient in terms of both speed and energy consumption.

TABLE II: COMPARISON BETWEEN INSTANT-NGP AND OUR METHOD UNDER DIFFERENT CONFIGURATIONS OF COARSE SAMPLE POINTS

Metrics \ Methods	I-NGP	Ours (192-step)	Ours (96-step)
Mem-Access (Per Ray)	1024	192	96
Addition (Per Ray)	0	3072	1536
Energy (pJ,Per Ray)	5120	1267.2	633.6
PSNR	31.15	31.02	29.65
Equivalent FPS	92	140	146

The energy data is from [13]. In the calculation, we assume the entire Occupancy Grid (256KB) can be stored in on-chip SRAM.

VI. Conclusion

To address the ray marching bottleneck present in NeRF-based algorithms, this work introduces a density estimation-based method for efficient sampling point generation. Initially, a density filtering approach is applied to eliminate high-frequency noise within the density grid, enabling more accurate representation of object shapes. Subsequently, a density estimation method is proposed, utilizing a Gaussian prior for economical, coarse sampling of the light density distribution. This strategy reduces the number of required sampling points for subsequent feature computation while maintaining rendering quality. Experimental results confirm the effectiveness of this method, demonstrating improvements in rendering quality and computational efficiency. The proposed method achieves an equivalent FPS that is 1.5× higher, and the energy expended per ray is only 24.8%.

REFERENCES

[1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis," in *European Conference on Computer Vision (ECCV)*, 2020.

[2] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys, "NICE-SLAM: Neural implicit scalable encoding for slam," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[3] J. Tang, "Stable-dreamfusion: Text-to-3D with Stable-diffusion," 2022, https://github.com/ashawkey/stable-dreamfusion.

[4] Y. Wang, Y. Long, S. H. Fan, and Q. Dou, "Neural Rendering for Stereo 3D Reconstruction of Deformable Tissues in Robotic Surgery," in *IEEE International Conference on Medical Image Computing and Computer-Assisted Intervention(MICCAI)*, 2022.

[5] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su, "TensoRF: Tensorial radiance fields," in *European Conference on Computer Vision (ECCV)*, 2022.

[6] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa, "PlenOctrees for real-time rendering of neural radiance fields," in *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[7] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, "Plenoxels: Radiance Fields without Neural Networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[8] Z. Chen, T. Funkhouser, P. Hedman, and A. Tagliasacchi, "Mobilenerf: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[9] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Transactions on Graphics (ToG)*, vol. 41, no. 4, pp. 102:1–102:15, Jul. 2022. [Online]. Available: https://doi.org/10.1145/3528223.3530127

[10] S. Li, C. Li, W. Zhu, B. Yu, Y. Zhao, C. Wan, H. You, H. Shi, and Y. Lin, "Instant-3D: Instant neural radiance field training towards on-device ar/vr 3d reconstruction," in *IEEE International Symposium on Computer Architecture (ISCA)*, 2023.

[11] S. Sabour, S. Vora, D. Duckworth, I. Krasin, D. J. Fleet, and A. Tagliasacchi, "RobustNeRF: Ignoring distractors with robust losses," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[12] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3D Gaussian Splatting for Real-time Radiance Field Rendering," *ACM Transactions on Graphics (ToG)*, vol. 42, no. 4, pp. 1–14, 2023.

[13] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally, "EIE: Efficient inference engine on compressed deep neural network," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 243–254, 2016.