Supplementary Material for Efficient Neural Radiance Fields

Tao Hu[1] Shu Liu [2] Yilun Chen [1] Tiancheng Shen [1] Jiaya Jia[1,2]
[1] The Chinese University of Hong Kong [2] SmartMore
{taohu,ylchen,tcshen,leojia}@cse.cuhk.edu.hk , sliu@smartmore.com

1. Technical Details

We supplement the detailed network structures and basic technologies in this paper.

1.1. Positional Encoding

Instead of directly inputing location **x** and direction **r** to MLPs, the original NeRF pre-process it by positional encoding [2] for better representing the high frequency details in scenes. As shown in Equ. 1.

$$\gamma(x) = (\sin(2^0\pi x), \cos(2^0\pi x), ..., \sin(2^{L-1}\pi x), \cos(2^{L-1}\pi x)) \quad (1)$$

We employ the same positional encoding and L for **x** while omitting **r** because of introducing the Spherical Harmonics model.

1.2. Spherical Harmonics

Original NeRF adopts an implicit function to predict the colors in different directions. However, the Spherical Harmonics model [] is a widely used model to model Lambertian surfaces and glossy surfaces. Followed Nex [3] and PlenOctree [4], Our EfficientNeRF also utilize the same technology as an explicit function to predict the colors.

Specifically, the MLPs will output the sphefical harmonics coefficient $k \in \mathbb{R}^{3\times(D+1)^2}$, then convert to RGB color by

$$c(d;k) = S(\sum_{l=0}^{L}\sum_{m=-l}^{l} k_l^m Y_l^m(d)) \quad (2)$$

where S is the sigmod function to normalize the color to range $[0, 1]$.

1.3. Network Structure

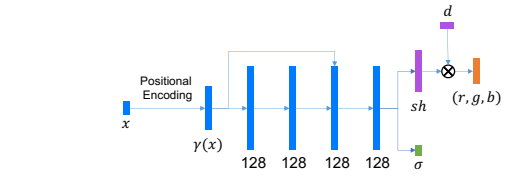
The network structure of Coarse MLP and Fine MLP are shown in Fig. 1a and Fig. 1b, respectively. First, the position **x** each sampled point are pre-processed by positional encoding to $\gamma(\mathbf{x})$. Then, $\gamma(\mathbf{x})$ are sent to coarseMLP or fine MLP to predict the 3D attributes, including color parameters sh and density σ. Finally, the color parameters sh will be convert to RGB through Equ.2. The only difference between coarse MLP and fine MLP is the width and depth of linear layers.

1.4. Implement NerfTree

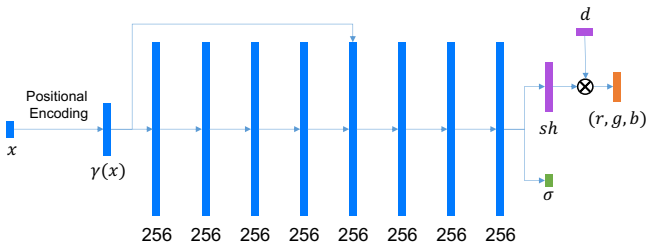
Our NerfTree consists of Coarse Dense Voxels V_c and Fine Sparse Voxels V_f. In addition, there are an edge $E \in \mathbb{R}^{D_c^3\times 1}$ linking the valid points in V_c to corresponding local voxels in V_f. We implement NerfTree in CUDA, the speed and performance comparison is illustrated in Tab. 1.

Inference	Speed (FPS)	PSNR (↑)
Coarse and Fine MLPs	0.18	**31.71**
NerfTree	**238.46**	31.68

Table 1. Comparison between different testing model.



(a) Lightweight Coarse MLP.



(b) Standard Fine MLP.

Figure 1. The network structure of our lightweight coarse MLP and Fine MLP

2. Additional Results

In this section, we perform more experiments to demonstrate the characteristic of our proposed EfficientNeRF.

2.1. Default Density : ε

Before training, we initialize the default density in Momentum Density Voxels V_σ as ε. These experiments will explore the influence of the different values of ε. As shown in Tab. 2. We found that the larger value of ε, the more accurate the synthesized results because of the number of samples, while the slower training speed. Therefore we choose 1.0 as the default density value to consider both training speed and accuracy.

ε	Training Speed	PSNR (↑)
0.1	**0.018s / iter**	31.54
1.0	0.021s / iter	31.68
10.0	0.057 s / iter	**31.75**

Table 2. The influence of different value of default density ε. The larger value of ε, the more accurate synthesized results, while the slower training speed.

2.2. Pivotal Threashold: ϵ

Pivotal threashold ϵ determines the minimal contribution w_i of pivotal samples, as shown in Equ. 3.

$$0 \leq w_i \leq 1$$
$$\sum_{i=1}^{N} w_i = 1 \tag{3}$$

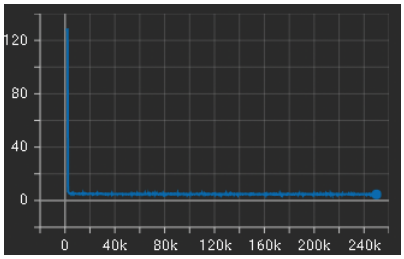
We conduct these experiments to demonstrate the influence of different pivotal thresholds: ϵ, the results are listed in Tab. 3. As the decrease of ϵ, the training speed rises while the accuracy gets saturated. Thus $\epsilon = 1 \times 10^{-4}$ is a good choice.

ϵ	Training Speed	PSNR (↑)
1×10^{-2}	**0.018s / iter**	31.27
1×10^{-4}	0.021 s / iter	31.68
1×10^{-6}	0.029 s / iter	**31.71**

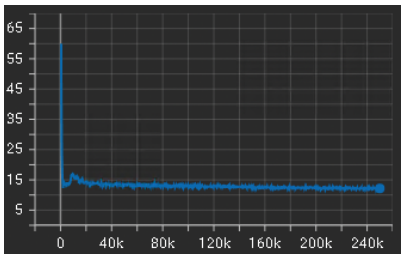
Table 3. The influence of different value of pivotal threashold ϵ.

2.3. Dynamic Number Samples

In the beginning, since all queried densities are greater than zero, the number of coarse sampling is N_c. As the training goes on, more and more invalid samples appeared, thus the number of valid and pivotal samples rapidly decreased. Finally, the number of samples at the coarse and fine stage converges to a relatively fixed value. As illustrated in Fig. 2 and Fig. 3, the horizontal axis is training iterations and the vertical axis is the number of sampling during the coarse or fine stage.

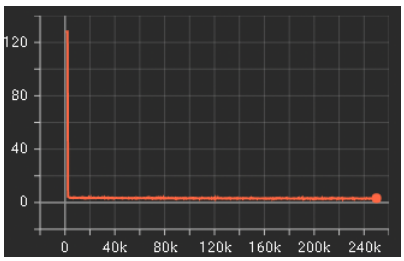


(a) The dynamic number of valid samples of Lego scene.

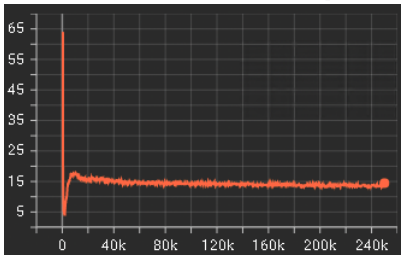


(b) The dynamic number of pivotal samples of Lego scene.

Figure 2. The dynamic number of coarse and fine sampling in Lego scene.



(a) The dynamic number of valid samples.



(b) The dynamic number of pivotal samples.

Figure 3. The dynamic number of coarse and fine sampling of Mic scene.

3. Visualization

We present more visualization results (including predicted depth) of different sceene in Fig. 4, Fig. 5, Fig. 6, and Fig. 6 to demonstrate the effectiveness of our Efficient-NeRF.

Figure 4. Qualitative results with state-of-the-art methods on the Realistic Synthetic dataset [2].

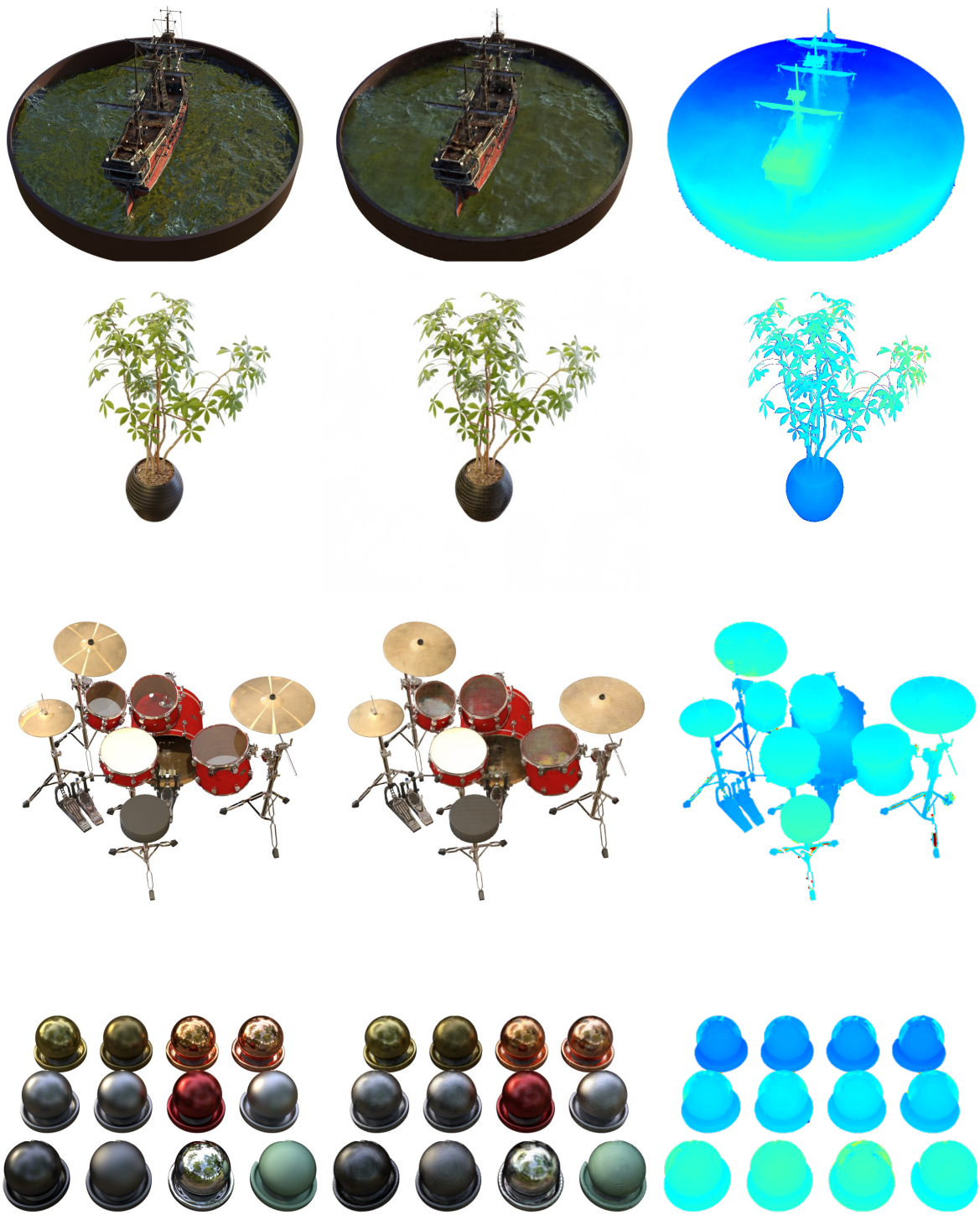


Figure 5. Qualitative results with state-of-the-art methods on the Realistic Synthetic dataset [2].

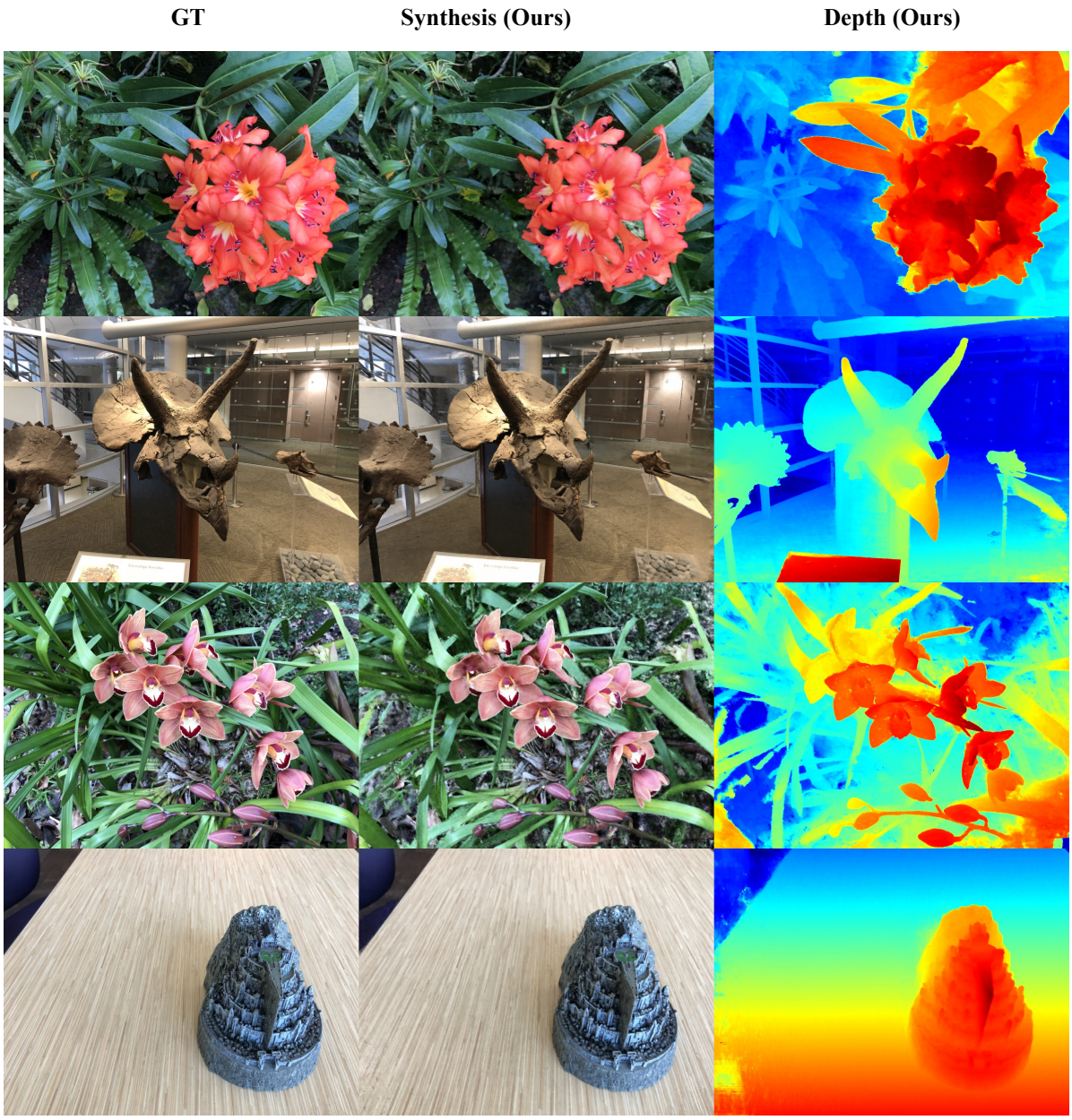


Figure 6. Qualitative results with state-of-the-art methods on the Real Forward-Facing dataset [1].

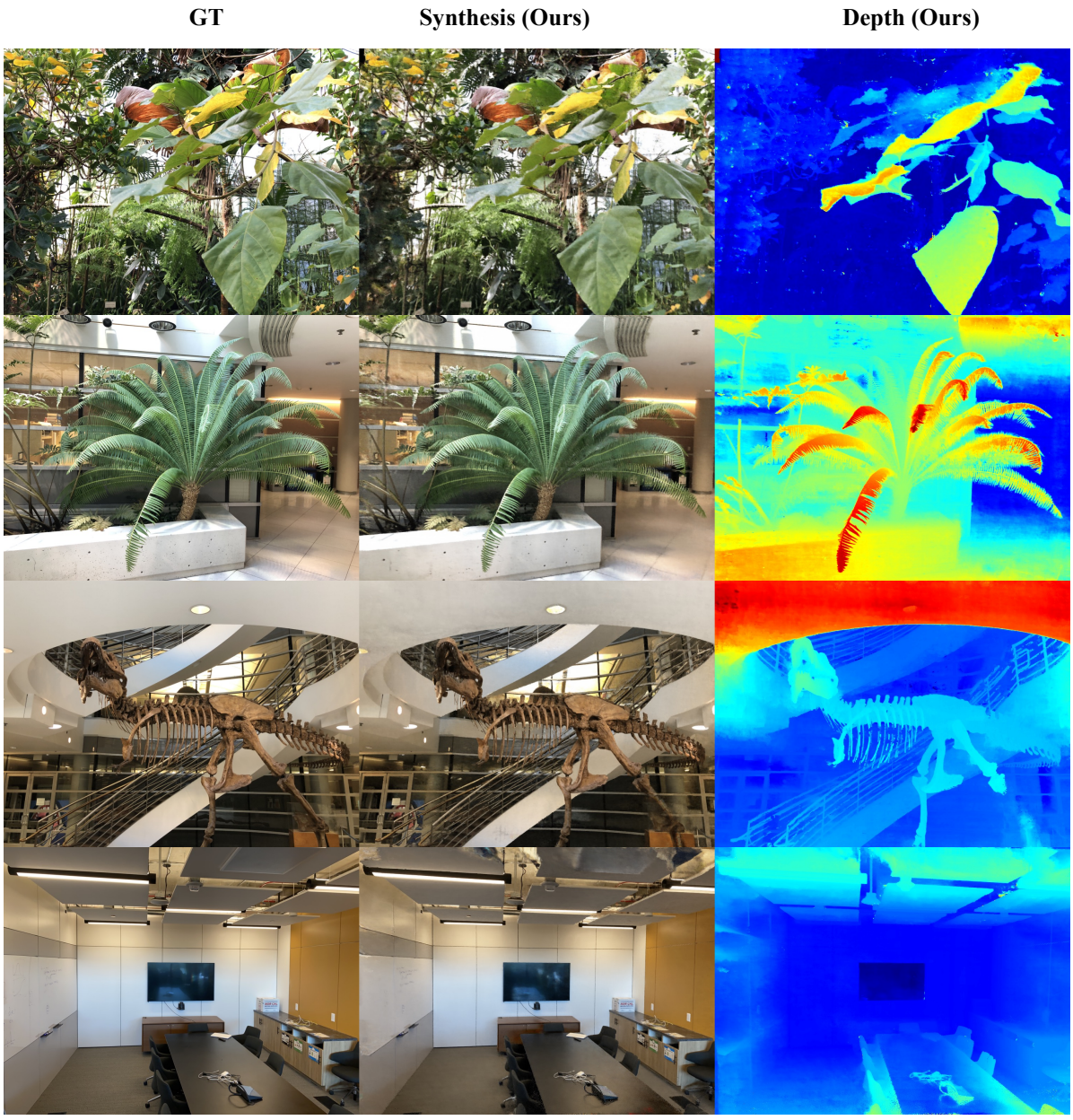


Figure 7. Qualitative results with state-of-the-art methods on the Real Forward-Facing dataset [1].

References

[1] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: practical view synthesis with prescriptive sampling guidelines. *ACM Trans. Graph.*, 2019. 5, 6

[2] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *ECCV*, volume 12346, pages 405–421, 2020. 1, 3, 4

[3] Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. In *CVPR*, June 2021. 1

[4] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *arXiv*, 2021. 1