

# Spiking NeRF: Making Bio-inspired Neural Networks See through the Real World

Xingting Yao<sup>1,2\*</sup>, Qinghao Hu<sup>1\*</sup>, Tielong Liu<sup>1,2</sup>,  
 Zitao Mo<sup>1</sup>, Zeyu Zhu<sup>1,2</sup>, Zhengyang Zhuge<sup>1</sup>, Jian Cheng<sup>1†</sup>

<sup>1</sup>Institute of Automation, Chinese Academy of Sciences

<sup>2</sup>School of Future Technology, University of Chinese Academy of Sciences

{yaoxingting2020, huqinghao2014, jian.cheng}@ia.ac.cn,

## Abstract

Spiking neuron networks (SNNs) have been thriving on numerous tasks to leverage their promising energy efficiency and exploit their potentialities as biologically plausible intelligence. Meanwhile, the Neural Radiance Fields (NeRF) render high-quality 3D scenes with massive energy consumption, and few works delve into the energy-saving solution with a bio-inspired approach. In this paper, we propose spiking NeRF (SpikingNeRF), which aligns the radiance ray with the temporal dimension of SNN, to naturally accommodate the SNN to the reconstruction of Radiance Fields. Thus, the computation turns into a spike-based, multiplication-free manner, reducing the energy consumption. In SpikingNeRF, each sampled point on the ray is matched onto a particular time step, and represented in a hybrid manner where the voxel grids are maintained as well. Based on the voxel grids, sampled points are determined whether to be masked for better training and inference. However, this operation also incurs irregular temporal length. We propose the temporal condensing-and-padding (TCP) strategy to tackle the masked samples to maintain regular temporal length, i.e., regular tensors, for hardware-friendly computation. Extensive experiments on a variety of datasets demonstrate that our method reduces the 76.74% energy consumption on average and obtains comparable synthesis quality with the ANN baseline.

## 1. Introduction

Spiking neural network (SNN) is considered the third generation of the neural network, and its bionic modeling encourages much research attention to explore the prospective biological intelligence that features multi-task supporting as the human brain does[19, 26]. While much dedication has

\*Equal contribution.

†Corresponding author.

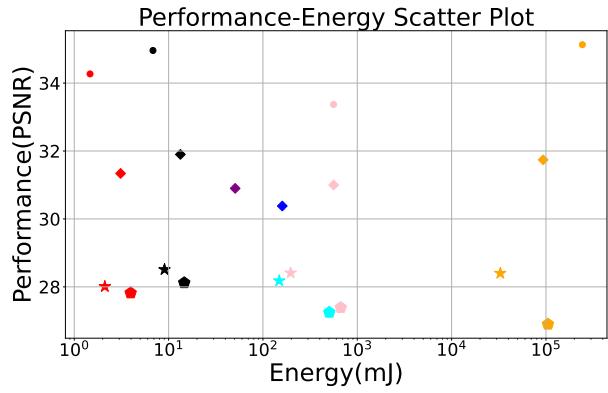


Figure 1. Comparison of our SpikingNeRF with other NeRF-based works on synthesis quality and model rendering energy. Different colors represent different works, our work is denoted in red. Different testing datasets are denoted in different node shapes.

been devoted to SNN research, the gap between the expectation of SNN boosting a wider range of intelligent tasks and the fact of artificial neural networks (ANN) dominating deep learning in the majority of tasks still exists.

Recently, more research interests have been invested to shrink the gap and acquired notable achievements in various tasks, including image classification[37], object detection[34], graph prediction[40], natural language processing[39], etc. Despite multi-task supporting, SNN research is also thriving in performance lifting and energy efficiency exploration at the same time. However, we have not yet witnessed the establishment of SNN in the real 3D reconstruction task with an advanced performance. To this end, this naturally raises an issue: *could bio-inspired spiking neural networks reconstruct the real 3D scene with an advanced quality at low energy consumption?* In this paper, we investigate rendering the neural radiance fields with a spiking approach to answer this question.

We propose spiking NeRF (SpikingNeRF) to reconstruct

a volumetric scene representation from a set of images. Learning from the huge success of NeRF[23] and its follow-ups[2, 3, 24, 25, 27], we utilize the voxel grids and the spiking multilayer perceptron (sMLP) to jointly model the volumetric representation. In such hybrid representations, voxel grids explicitly store the volumetric parameters, and the spiking multilayer perceptron (sMLP) implicitly transforms the parameters into volumetric information, *i.e.*, the density and color information of the radiance fields. After the whole model is established, we follow the classic ray-sample accumulation method of radiance fields to render the colors of novel scenes[21]. To accelerate the training and inference, we use the voxel grids to predefine borders and free space, and mask those samples out of borders, in free space or with a low-density value as proposed in [27].

Inspired by the imaging process of the primate fovea in the retina that accumulates the intensity flow over time to stimulate the photoreceptor cell[20, 29], we associate the accumulation process of rendering with the temporal accumulation process of SNNs, which ultimately stimulates the spiking neurons to fire. Guided by the above insight, we align the ray with the temporal dimension of the sMLP, and match each sampled point on the ray to a time step one-to-one during rendering. Therefore, different from the original rendering process where all sampled points are queried separately to retrieve the corresponding volumetric information, sampled points are queried continuously along each ray in SpikingNeRF, and the geometric consecutiveness of the ray is thus transformed into the temporal continuity of the SNN. As a result, SpikingNeRF seizes the nature of both worlds to make the NeRF rendering in a spiking manner.

However, the number of sampled points on different rays varies due to the aforementioned mask operation, which causes the temporal lengths of different rays to become irregular. Thus, the querying of the color information can hardly be parallelized by the sMLP on graphics processing units (GPUs), severely hindering the rendering process. To solve this issue, we first investigate the temporal padding (TP) method to attain the regular temporal length in a querying batch, *i.e.*, a regular-shaped tensor, thus ensuring parallelism. Furthermore, we propose the temporal condensing-and-padding (TCP), which totally removes the temporal effect of masked points on the sMLP, to fully constrain the tensor size and condense the data distribution, which is hardware-friendly to some domain-specific accelerators. Our theoretical analysis and empirical experiments prove that TCP can maintain the energy merits and good performance of SpikingNeRF.

Additionally, we discuss the querying direction of SpikingNeRF during rendering since SNNs have to process temporal information in a particular direction while the accumulation process of original rendering does not. Contrary to intuition, the experimental results indicate the direction

of camera ray is better for SNNs.

To sum up, our main contributions are as follows:

- We propose SpikingNeRF that aligns the radiance rays in NeRF with the temporal dimension of SNNs. To the best of our knowledge, this is the first work to accommodate the spiking neural network to reconstruct real 3D scenes.
- We propose TP and TCP to solve the irregular temporal length, ensuring the training and inference parallelism on GPUs.
- Our experiments demonstrate the effectiveness of SpikingNeRF on four mainstream inward-facing 3D reconstruction tasks. For example, SpikingNeRF can improve  $4.35 \times$  rendering energy efficiency with 0.3 PSNR drop on BlendedMVS.

## 2. Related Work

**NeRF-based 3D Reconstruction.** Different from the traditional 3D reconstruction methods that mainly rely on the explicit and discrete volumetric representations, NeRF[23] utilizes a coordinate neural network to implicitly represent the 3D radiance field, and synthesizes novel views by accumulating the density and color information along the view-dependent rays with a ray tracing algorithm[12]. With this new paradigm of 3D reconstruction, NeRF achieves huge success in improving the quality of novel view synthesis. The followup works further enhance the rendering quality[1, 5, 28], and many others focus on accelerating the training[6, 9, 27] or rendering process[16, 25, 27, 33]. While, we concentrate on exploring the potential integration of the spike-based low-energy communication and the NeRF-based high-quality 3D synthesis, seeking ways to energy efficient real 3D reconstruction.

**Fast NeRF Synthesis.** The accumulation process of rendering in NeRF[23] requires a huge number of MLP querying, which incurs heavy flop-operation and memory-access burdens, delaying the synthesis speed. Recent studies further combine the traditional explicit volumetric representations, *e.g.*, voxels[11, 18, 27] and MPIS[30], with MLP-dependent implicit representations to obtain more efficient volumetric representations. Thus, the redundant MLP queries for points in free space can be avoided. In SpikRF, we adopt the voxel grids to mask the irrelevant points with low density, and discard unimportant points with low weight, thus reducing the synthesis overhead.

**Spiking Neural Networks.** With the high sparsity and multiplication-free operation, SNNs outstrip ANNs in the competition of energy efficiency[4, 14, 15], but fall behind in the performance lifting. Therefore, great efforts have been dedicated to making SNNs deeper[8, 36], converge faster[7, 31], and eventually high-performance[37].

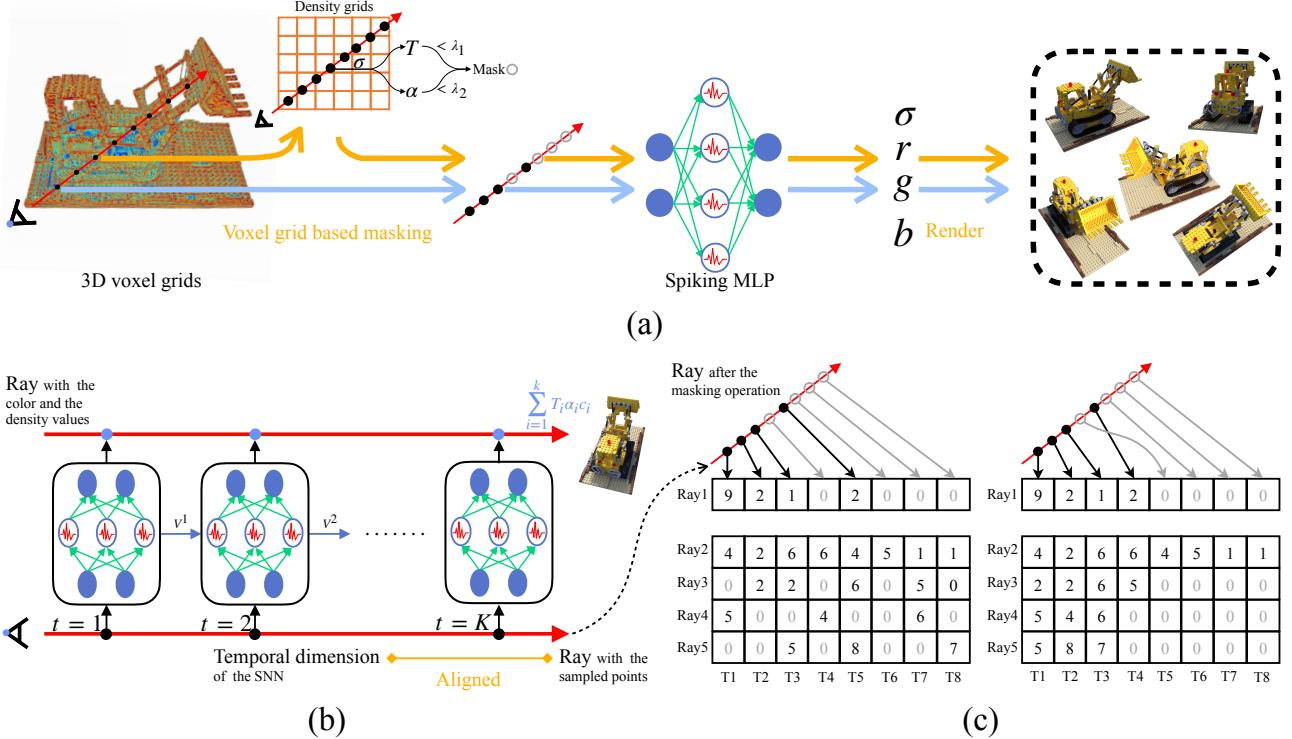


Figure 2. Overview of the proposed SpikingNeRF. (a) The rendering process of SpikingNeRF. The whole 3D volumetric parameters are stored in voxel grids. The irrelevant and unimportant samples are masked before the sMLP querying. The expected scenes are rendered with the volumetric information yielded by the sMLP. (b) Alignment between the temporal dimension and the ray. The sMLP queries each sampled point step-by-step to yield the volumetric information. (c) Proposed temporal padding (left) and temporal condensing-and-padding (right) methods. For simplification, the channel length of the volumetric parameters is set to 0

With the merits of energy efficiency and advanced performance, recent SNN research sheds light on exploring versatile SNNs, *e.g.*, Spikeformer[37], Spiking GCN[40], SpikeGPT[39]. In this paper, we seize the analogous nature of both NeRF and SNNs to make bio-inspired spiking neural networks reconstruct the real 3D scene with an advanced quality at low energy consumption.

**SNNs in 3D Reconstruction.** Applying SNNs to 3D Reconstruction has been researched with very limited efforts. As far as we know, only two works exist in this scope. EVSNN [38] is the first attempt to develop a deep SNN for image reconstruction task, which achieves comparable performance to ANN. E2P [22] strives to solve the event-to-polarization problem and uses SNN to achieve a better polarization reconstruction performance. Unfortunately, both of them focus solely on the reconstruction of event-based images with traditional methods, neglecting the rich RGB world. While we are the first to explore the reconstruction of the real RGB world with SNNs.

### 3. Preliminaries

**Neural Radiance Field.** To reconstruct the scene for the given view, NeRF[23] first utilizes an MLP, which takes in the location coordinate  $\mathbf{p} \in \mathbb{R}^3$  and the view direction  $\mathbf{v} \in \mathbb{R}^2$  and yields the density  $\sigma \in \mathbb{R}$  and the color  $\mathbf{c} \in \mathbb{R}^3$ , to implicitly maintain continuous volumetric representations:

$$\mathbf{e}, \sigma = \text{MLP}_\theta(\mathbf{p}), \quad (1)$$

$$\mathbf{c} = \text{MLP}_\gamma(\mathbf{e}, \mathbf{v}), \quad (2)$$

where  $\theta$  and  $\gamma$  denote the parameters of the separate two parts of the MLP, and  $\mathbf{e}$  is the embedded features. Next, NeRF renders the pixel of the expected scene by casting a ray  $\mathbf{r}$  from the camera origin point to the direction of the pixel, then sampling  $K$  points along the ray. Through querying the MLP as in Eq.1-2  $K$  times,  $K$  color values and  $K$  density values can be retrieved. Finally, following the principles of the discrete volume rendering proposed in

[21], the expected pixel RGB  $\hat{C}(\mathbf{r})$  can be rendered:

$$\alpha = 1 - \exp(-\sigma_i \delta_i), \quad T_i = \prod_{j=1}^{i-1} (1 - \alpha_j), \quad (3)$$

$$\hat{C}(\mathbf{r}) \approx \sum_{i=1}^K T_i \alpha_i \mathbf{c}_i, \quad (4)$$

where  $\mathbf{c}_i$  and  $\sigma_i$  denotes the color and the density values of the  $i$ -th point respectively, and  $\delta_i$  is the distance between the adjacent point  $i$  and  $i + 1$ .

After rendering all the pixels, the expected scene is reconstructed. With the ground-truth pixel color  $C(\mathbf{r})$ , the parameters of the MLP can be trained end-to-end by minimizing the MSE loss:

$$\mathcal{L} = \frac{1}{|\mathcal{R}|} \sum_{\mathbf{r} \in \mathcal{R}} \|\hat{C}(\mathbf{r}) - C(\mathbf{r})\|_2^2, \quad (5)$$

where  $\mathcal{R}$  is the mini-batch containing the sampled rays.

**Hybrid Volumetric Representation.** The number of sampled points  $K$  in Eq.4 is usually big, leading to the heavy MLP querying burden as displayed in Eq.1-2. To alleviate this problem, voxel grid representation is utilized to contain the volumetric parameters directly, *e.g.*, the embedded feature  $e$  and the density  $\sigma$  in Eq.1, as the values of the voxel grid. Thus, querying the MLP in Eq.1 is substituted to querying the voxel grids and operating the interpolation, which is much easier:

$$\sigma = \text{act}(\text{interp}(\mathbf{p}, \mathbf{V}_\sigma)), \quad \mathbf{e} = \text{interp}(\mathbf{p}, \mathbf{V}_f), \quad (6)$$

where  $\mathbf{V}_\sigma$  and  $\mathbf{V}_f$  are the voxel grids related to the volumetric density and features, respectively. “interp” denotes the interpolation operation, and “act” refers to the activation function, *e.g.*, ReLU or the shifted softplus[1].

Furthermore, those irrelevant points with low density or unimportant points with low weight can be **masked** through predefined thresholds  $\lambda$ , then Eq.4 turns into:

$$\hat{C}(\mathbf{r}) \approx \sum_{i \in A} T_i \alpha_i \mathbf{c}_i, \quad A \triangleq \{i : T_i > \lambda_1, \alpha_i > \lambda_2\}. \quad (7)$$

Thus, the queries of the MLP for sampled points in Eq.2 are significantly reduced.

**Spiking Neuron.** The spiking neuron is the most fundamental unit in spiking neural networks, which essentially differs SNNs from ANNs. The modeling of spiking neuron commonly adopts the leaky integrate-and-fire (LIF) model:

$$\mathbf{U}^t = \mathbf{V}^{t-1} + \frac{1}{\tau} (\mathbf{X}^t - \mathbf{V}^{t-1} + V_{reset}), \quad (8)$$

$$\mathbf{S}^t = \mathbb{H}(\mathbf{U}^t - V_{th}), \quad (9)$$

$$\mathbf{V}^t = \mathbf{U}^t \odot (1 - \mathbf{S}^t) + V_{reset} \mathbf{S}^t. \quad (10)$$

Here,  $\odot$  denotes the Hadamard product.  $\mathbf{U}^t$  is the intermediate membrane potential at time-step  $t$  and can be updated through Eq.8, where  $\mathbf{V}^{t-1}$  is the actual membrane potential at time-step  $t - 1$  and  $\mathbf{X}^t$  denotes the input vector at time-step  $t$ , *e.g.*, the activation vector from the previous layer in MLPs. The output spike vector  $\mathbf{S}^t$  is given by the Heaviside step function  $\mathbb{H}(\cdot)$  in Eq.9, indicating that a spike is fired when the membrane potential exceeds the potential threshold  $V_{th}$ . Dependent on whether the spike is fired at time-step  $t$ , the membrane potential  $\mathbf{V}^t$  is set to  $\mathbf{U}^t$  or the reset potential  $V_{reset}$  through Eq.10.

Since the Heaviside step function  $\mathbb{H}(\cdot)$  is not differentiable, the surrogate gradient method is utilized to solve this issue, which is defined as :

$$\frac{d\mathbb{H}(x)}{dx} = \frac{1}{1 + \exp(-\alpha x)}, \quad (11)$$

where  $\alpha$  is a predefined hyperparameter. Thus, spiking neural networks can be optimized end-to-end.

## 4. Methodology

### 4.1. Alignment between SNN and NeRF

We first consider the MLP querying process in the ANN philosophy. For an expected scene to reconstruct, the volumetric parameters of sampled points, *e.g.*,  $e$  and  $v$  in Eq.2, are packed as the input data with the shape of  $[batch, c_e]$  or  $[batch, c_v]$ , where  $batch$  represents the sample index and  $c$  is the channel index of the volumetric parameters. Thus, the MLP can query these data and output the corresponding color information in parallel. However, from the geometric view, the input data should be packed as  $[ray, pts, c]$ , where  $ray$  is the ray index and the  $pts$  is the index of the sampled points.

Obviously, the ANN-based MLP querying process can not reflect such geometric relations between the ray and the sampled points. Then, we consider the computation modality of SNNs. As illustrated in Eq.8-10, SNNs naturally entail the temporal dimension to process the sequential signals. This means a spiking MLP naturally accepts the input data with the shape of  $[batch, time, c]$ , where  $time$  is the temporal index. Therefore, we can reshape the volumetric parameters back to  $[ray, pts, c]$ , and intuitively match each sample along the ray to the corresponding time step:

$$\begin{aligned} \text{Input}_{MLP} &:= [batch, c] \\ &\Rightarrow [ray, pts, c] \\ &\Rightarrow [batch, time, c] := \text{Input}_{sMLP}, \end{aligned} \quad (12)$$

which is also illustrated in Fig.2(b). Such an alignment does not require any input data pre-process such as encoding[10] or duplication[37] as prior arts commonly do.

Table 1. Comparisons between the proposed TCP and TP.

	Synthetic-NeRF		Synthetic-NSVF		Blendedmvs		Tanks&Temples	
	TCP	TP	TCP	TP	TCP	TP	TCP	TP
PSNR↑	31.33	31.34	34.33	34.27	27.80	27.82	28.01	28.01
SSIM↑	0.949	0.949	0.970	0.970	0.912	0.912	0.892	0.892
LPIP <sub>Vgg</sub> ↓	0.068	0.068	0.039	0.040	0.103	0.103	0.174	0.0173
LPIP <sub>Alex</sub> ↓	0.039	0.039	0.021	0.021	0.065	0.065	0.142	0.142
Flops (M)	246.4	246.4	126.7	116.9	313.9	313.9	168.2	168.1
Add ops (M)	310.0	309.6	162.0	148.0	399.5	398.6	197.9	196.7
Energy (mJ)	3.10	3.10	1.60	1.47	3.95	3.96	2.12	2.11

Table 2. Comparison with the ANN-based NeRF Counterpart.

	Synthetic-NeRF		Synthetic-NSVF		Blendedmvs		Tanks&Temples	
	SpikingNeRF	Baseline	SpikingNeRF	Baseline	SpikingNeRF	Baseline	SpikingNeRF	Baseline
PSNR↑	31.33	31.90	34.33	34.96	27.80	28.12	28.01	28.30
SSIM↑	0.949	0.956	0.970	0.975	0.912	0.922	0.892	0.910
LPIP <sub>Vgg</sub> ↓	0.068	0.0539	0.039	0.033	0.103	0.101	0.0174	0.157
LPIP <sub>Alex</sub> ↓	0.039	0.0352	0.021	0.019	0.065	0.074	0.142	0.150
Flops (M)	246.4		126.7		313.9		168.2	
Add ops (M)	310.0	-	162.0	-	399.5	-	197.9	-
Energy (mJ)	3.10	13.43	1.60	6.90	3.95	17.10	2.12	9.16

## 4.2. TCP

However, one small, puzzling cloud remained on the horizon, which is the mask operation as illustrated in Sec.3. Such mask operation improves the rendering speed and quality by curtailing the computation cost for redundant samples and eliminating the obstruction of occluding samples. Nevertheless, it also causes the number of queried samples on different rays to be irregular, which indicates the reshape operation of Eq.12 is unfeasible for packed data, *i.e.*, a tensor, on GPUs.

To ensure the computation parallelism on GPUs, we first propose to remain the indices of those masked samples but discard their volumetric parameters. As illustrated in Fig.2(c) Left, we arrange both unmasked and masked samples sequentially to the corresponding *ray*-indexed vector. Since the parameters of the masked samples are discarded, we pad zeros to the vacant tensor elements. Such that, a regular-shaped input tensor is built. Thus, spiking MLP can process those valid samples step-by-step. We refer to this simple approach as the temporal padding (TP) method.

However, TP does not handle those masked samples effectively because those padded zeros will still get involved in the following computation and cause the membrane potential of sMLP to decay, implicitly affecting the outcomes of the unmasked samples from the posterior segment of the ray. Even for a sophisticated hardware accelerator that can skip those zeros, the sparse data structure still causes computation inefficiency such as imbalanced workload[35].

To solve this issue, we design the temporal condensing-and-padding (TCP) scheme, which is illustrated in Fig.2(c) Right. Different from TP, TCP completely discards the parameters and indices of the masked samples, and adjacently rearranges the unmasked samples to the corresponding *ray* vector. For the vacant tensor elements, zeros are filled as TP does. Consequently, valid data is condensed to the left side of the tensor. Notably, the *ray* dimension can be sorted according to the valid data number to further increase the density, but we leave that for future work. For now, TCP has fully eliminated the impact of the masked samples and increased the data density to some extent.

Our comparative experiments indicate TCP and TP achieve the same-level synthesis quality for SpikingNeRF, but TCP has an edge for a denser data structure and more efficient memory utilization. Therefore, we choose TCP as our main proposed method for the following study.

## 5. Experiments

In this section, we demonstrate the effectiveness of our proposed SpikingNeRF. We first build the codes on the voxel grid based NeRF implementation DVGO[27]<sup>1</sup>. Secondly, we compare the proposed TP and TCP with the original DVGO implementation, which is also referred to as the ANN baseline. Finally, we visualize the rendered scenes from TCP-based SpikingNeRF to directly display the synthesis results.

<sup>1</sup><https://github.com/sunset1995/DirectVoxGO>

## 5.1. Experiments Settings

We conduct experiments on the renowned Synthetic-NeRF[23], Synthetic-NSVF[17], Blend-edMVS[32], and Tanks&Temples datasets[13].

## 5.2. Comparisons

**Comparisons between TCP and TP.** The results are listed in Table 1.

**Comparisons between SpikingNeRF and the ANN Counterpart.** We use the original DVGO as the ANN Counterpart for comparisons. The results are listed in Table 2.

**Visualization.** We visualize the rendered scenes from SpikingNeRF and the ANN baseline. The results are stored at <https://github.com/Ikarosy/images>. Each row displays the two scenes from the same dataset. From top to bottom place Synthetic-NSVF, Synthetic-NeRF, and Blend-edMVS by order.

## References

- [1] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. <sup>2, 4</sup>
- [2] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision*, pages 333–350. Springer, 2022. <sup>2</sup>
- [3] Zhiqin Chen, Thomas Funkhouser, Peter Hedman, and Andrea Tagliasacchi. Mobilenerf: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16569–16578, 2023. <sup>2</sup>
- [4] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. Loihi: A neuromorphic manycore processor with on-chip learning. *Ieee Micro*, 38(1):82–99, 2018. <sup>2</sup>
- [5] Boyang Deng, Jonathan T Barron, and Pratul P Srinivasan. Jaxnerf: an efficient jax implementation of nerf, 2020. URL <https://github.com/google-research/google-research/tree/master/jaxnerf>. <sup>2</sup>
- [6] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12882–12891, 2022. <sup>2</sup>
- [7] Shikuang Deng, Yuhang Li, Shanghang Zhang, and Shi Gu. Temporal efficient training of spiking neural network via gradient re-weighting. In *International Conference on Learning Representations*, 2021. <sup>2</sup>
- [8] Wei Fang, Zhaofei Yu, Yanqi Chen, Tiejun Huang, Timothée Masquelier, and Yonghong Tian. Deep residual learning in spiking neural networks. *Advances in Neural Information Processing Systems*, 34, 2021. <sup>2</sup>
- [9] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, 2022. <sup>2</sup>
- [10] Isha Garg, Sayeed Shafayet Chowdhury, and Kaushik Roy. Dct-snn: Using dct to distribute spatial information over time for low-latency spiking neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4671–4680, 2021. <sup>4</sup>
- [11] Peter Hedman, Pratul P Srinivasan, Ben Mildenhall, Jonathan T Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5875–5884, 2021. <sup>2</sup>
- [12] James T Kajiya and Brian P Von Herzen. Ray tracing volume densities. *ACM SIGGRAPH computer graphics*, 18(3):165–174, 1984. <sup>2</sup>
- [13] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017. <sup>6</sup>
- [14] Jeong-Jun Lee, Wenrui Zhang, and Peng Li. Parallel time batching: Systolic-array acceleration of sparse spiking neural computation. In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 317–330. IEEE, 2022. <sup>2</sup>
- [15] Ziru Li, Bonan Yan, and Hai Li. Resipe: Reram-based single-spiking processing-in-memory engine. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2020. <sup>2</sup>
- [16] David B Lindell, Julien NP Martel, and Gordon Wetzstein. Autoint: Automatic integration for fast neural volume rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14556–14565, 2021. <sup>2</sup>
- [17] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020. <sup>6</sup>
- [18] Yuan Liu, Sida Peng, Lingjie Liu, Qianqian Wang, Peng Wang, Christian Theobalt, Xiaowei Zhou, and Wenping Wang. Neural rays for occlusion-aware image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7824–7833, 2022. <sup>2</sup>
- [19] Wolfgang Maass. Networks of spiking neurons: the third generation of neural network models. *Neural networks*, 10(9):1659–1671, 1997. <sup>1</sup>

- [20] Richard H Masland. The neuronal organization of the retina. *Neuron*, 76(2):266–280, 2012. 2
- [21] Nelson Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995. 2, 4
- [22] Haiyang Mei, Zuowen Wang, Xin Yang, Xiaopeng Wei, and Tobi Delbruck. Deep polarization reconstruction with pdavis events. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22149–22158, 2023. 3
- [23] 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. *Communications of the ACM*, 65(1):99–106, 2021. 2, 3, 6
- [24] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022. 2
- [25] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14335–14345, 2021. 2
- [26] Kaushik Roy, Akhilesh Jaiswal, and Priyadarshini Panda. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, 2019. 1
- [27] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022. 2, 5
- [28] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33:7537–7547, 2020. 2
- [29] Heinz Wässle. Parallel processing in the mammalian retina. *Nature Reviews Neuroscience*, 5(10):747–757, 2004. 2
- [30] Suttisak Wizadwongsu, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8534–8543, 2021. 2
- [31] Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, Yuan Xie, and Luting Shi. Direct training for spiking neural networks: Faster, larger, better. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1311–1318, 2019. 2
- [32] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1790–1799, 2020. 6
- [33] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5752–5761, 2021. 2
- [34] Jiqing Zhang, Bo Dong, Haiwei Zhang, Jianchuan Ding, Felix Heide, Baocai Yin, and Xin Yang. Spiking transformers for event-based single object tracking. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 8801–8810, 2022. 1
- [35] Zhekai Zhang, Hanrui Wang, Song Han, and William J Dally. Sparc: Efficient architecture for sparse matrix multiplication. In *2020 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 261–274. IEEE, 2020. 5
- [36] Hanle Zheng, Yujie Wu, Lei Deng, Yifan Hu, and Guoqi Li. Going deeper with directly-trained larger spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 11062–11070, 2021. 2
- [37] Zhaojun Zhou, Yuesheng Zhu, Chao He, Yaowei Wang, Shuicheng Yan, Yonghong Tian, and Li Yuan. Spikformer: When spiking neural network meets transformer. *arXiv preprint arXiv:2209.15425*, 2022. 1, 2, 3, 4
- [38] Lin Zhu, Xiao Wang, Yi Chang, Jianing Li, Tiejun Huang, and Yonghong Tian. Event-based video reconstruction via potential-assisted spiking neural network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3594–3604, 2022. 3
- [39] Rui-Jie Zhu, Qihang Zhao, and Jason K Eshraghian. Spikegpt: Generative pre-trained language model with spiking neural networks. *arXiv preprint arXiv:2302.13939*, 2023. 1, 3
- [40] Zulun Zhu, Jiaying Peng, Jintang Li, Liang Chen, Qi Yu, and Siqiang Luo. Spiking graph convolutional networks. *arXiv preprint arXiv:2205.02767*, 2022. 1, 3