Spiking NeRF: Representing the Real-World Geometry by a Discontinuous Representation

Zhanfeng Liao[1], **Qian Zheng**[1], **Yan Liu**[1], **Gang Pan**[1]

[1]Zhejiang University
{zhanfengliao, qianzheng, rliuyan, gpan}@zju.edu.cn

Abstract

A crucial reason for the success of existing NeRF-based methods is to build a neural density field for the geometry representation via multiple perceptron layers (MLPs). MLPs are continuous functions, however, real geometry or density field is frequently discontinuous at the interface between the air and the surface. Such a contrary brings the problem of unfaithful geometry representation. To this end, this paper proposes spiking NeRF, which leverages spiking neuron and a hybrid Artificial Neural Network (ANN)-Spiking Neural Network (SNN) framework to build a discontinuous density field for faithful geometry representation. Specifically, we first demonstrate the reason why continuous density fields will bring inaccuracy. Then, we propose to use the spiking neurons to build a discontinuous density field. We conduct comprehensive analysis for the problem of existing spiking neuron models and then provide the numerical relationship between the parameter of spiking neuron and the theoretical accuracy of geometry, Based on this, we propose a bounded spiking neuron to build the discontinuous density field. Our results achieve SOTA performance. Our code and data will be released to the public.

Introduction

3D reconstruction from RGB images is a challenging and complex task in computer vision (Mildenhall et al. 2021; Wang et al. 2021). Neural Radiance Fields (NeRF) (Mildenhall et al. 2021), a recently promising solution for novel view synthesis in an implicit manner (Shao et al. 2022), has also achieved very competitive results in 3D reconstruction (Shao et al. 2022; Oechsle, Peng, and Geiger 2021; Wang et al. 2021; Wang, Skorokhodov, and Wonka 2022, 2023).

One of the reasons for the success of NeRF is its ability to implicitly represent geometric information using a neural network based on conventional MLPs, which are continuous functions. However, geometric information in real world is discontinuous at the interface between the air and the surface, which is inconsistent with the computational representation in conventional MLPs. This inconsistency poses three problems. **Post-processing**. Existing methods require a post-processing approach (i.e., filtering with a threshold (Peng et al. 2021; Pumarola et al. 2021; Boss et al. 2021)) to extract discontinuous geometric representations from learned continuous fields. However, determining the optimal threshold requires empirical and tedious tuning, and many methods rely on manual selection of the threshold for different scenarios (Mildenhall et al. 2021; Peng et al. 2021; Pumarola et al. 2021; Boss et al. 2021) (see the left of Fig. 1). **Optimal threshold perturbation**. NeRF-based method optimize the network view by view, resulting in the optimal threshold of the surface to perturbate (see supplementary material for more details). However, existing methods use the same threshold to filter output values (Mildenhall et al. 2021; Peng et al. 2021; Pumarola et al. 2021; Boss et al. 2021), which cannot obtain the accurate geometric information because the optimal filtering thresholds under different views vary (see the middle of Fig. 1). ***Light density scenarios***[1]. The inconsistency can result in even greater errors in light density scenarios (see the right of Fig. 1). Because it is difficult for continuous functions like conventional MLPs to generate quite different densities (i.e., significantly different outputs) for points close to each other (i.e., similar inputs), resulting in the disappearance of the object.

Although there are some attempts to address the aforementioned problems, these methods are still based on the continuous representation and do not provide a fundamental solution to those problems. For example, some works replace the density field by other fields to avoid threshold selection (e.g., (Oechsle, Peng, and Geiger 2021; Wang et al. 2021; Fu et al. 2022; Darmon et al. 2022; Yariv et al. 2021; Wang, Skorokhodov, and Wonka 2022)). However, these methods cannot use the filter value to get the accurate geometric representation (see the left of Fig. 1). Moreover, they fail on light density scenarios (see the right of Fig. 1). Some works modify the input to improve its frequency or discretization, thereby mitigating the inconsistency between continuous and discontinuous representations. However, these methods still fail on light density scenarios (Wang, Skorokhodov, and Wonka 2022, 2023) (see the right of Fig. 1). Some works are designed for specific scenarios (e.g., (Guo et al. 2022; Levy et al. 2023)). However, these methods have poorer performance in general scenarios. Moreover, they mainly focus on novel

[1]These scenarios contain several rays, along which the density with non-zero values distribute within a very narrow range (e.g., thin objects) or with small values (e.g., semi-transparent objects). In real life, semi-transparent objects (e.g. windows, glasses) and thin objects (e.g. branches, axles, nets) can be seen everywhere.

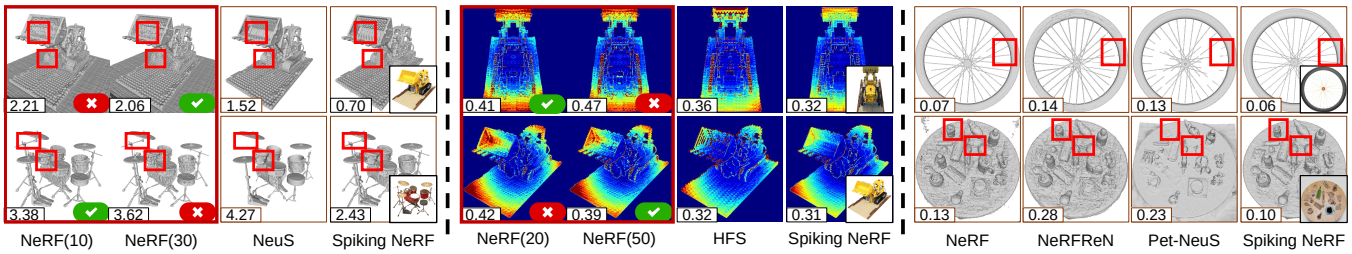

Figure 1: Left: The extracted surfaces from NeRF. Each row in the first big red box represents a surface extracted by a trained NeRF using different thresholds, indicating that the optimal thresholds corresponding to different scenarios are different. The tick represents the threshold is optimal. The cross represents the threshold is not optimal. Middle: The error maps from different views in the same scene. Each row in the second big red box represents the depth error map of a trained NeRF's surface extracted with different thresholds from different views. It can be seen that the optimal thresholds corresponding to different views are different. Right: The extracted surfaces from NeRF. These figures show that the inconsistency can result in even greater errors in light density scenarios. The image in the bottom right corner of each part represents the original image from the corresponding view. The number displayed in the bottom left corner of each image represents either the Chamfer distance (left and right) or the depth error (middle).

view synthesis, and the quality of the 3D reconstruction is low (see the right of Fig. 1).

Compared to the ANN transmitting continuous values, SNN transmits spikes which are discontinuous. The discontinuity of SNN is expected to represent the discontinuous geometric information in real world and fundamentally resolve the inconsistency between real-world representation and computational representation in conventional MLPs. Moreover, the threshold of spiking neurons in SNNs can serve as a filtering threshold through a parameter learning scheme (e.g., (Li et al. 2022; Zhang et al. 2022)), which avoids the issue of manually specifying threshold. Different from traditional cognitive applications (Li et al. 2022; Zhang et al. 2022; Han et al. 2023; Zou, Huang, and Wu 2022), using the spiking representation to model 3D geometry from a numerical perspective is a non-cognitive application (Aimone et al. 2022; Ren et al. 2023).

In this paper, we address the problems of continuous geometric representation in conventional NeRF by proposing spiking NeRF, which is based on a hybrid ANN-SNN framework to model the real-world 3D geometry in discontinuous representation. First, we build the relationship between the spiking threshold, maximum activation, and depth error for our discontinuous computational representation. Second, based on this relationship, we observe that when the spiking threshold is sufficiently large, the depth error is sufficiently small. However, the spiking threshold cannot be set sufficiently high in semi-transparent scenarios and cannot be set to infinity in practical implementation. We further discover a way to maintain a small error under a finite spiking threshold, which is to control the maximum activation. Last, based on our analysis, we propose Bounded Full-precision Integrate and Fire (B-FIF) spiking neuron to build a hybrid ANN-SNN framework. Meanwhile, we design a corresponding training pipline for the hybrid ANN-SNN framework and verify the effectiveness on mainstream datasets and light density scenarios. Our contributions can be summarized as follows:

- We build the relationship between the spiking threshold, maximum activation, and depth error. We constrain the bound of depth error by the spiking threshold and maximum activation. This bound can be leveraged to facilitate the real-world application for geometric estimation without knowing the ground truth.
- We propose B-FIF spiking neuron based on the aforementioned analysis and build a hybrid ANN-SNN framework. We design the corresponding training pipline and training strategy for the hybrid ANN-SNN framework. We verify the effectiveness on mainstream datasets and light density scenarios.

Related Work

Neural Implicit Representations

Recently, neural implicit functions have emerged as an effective representation of 3D geometry (Atzmon et al. 2019; Chen et al. 2018; Genova et al. 2019; Mescheder et al. 2019; Michalkiewicz et al. 2019; Niemeyer et al. 2019; Park et al. 2019; Peng et al. 2020; Saito et al. 2019; Xu et al. 2019) and appearance (Mildenhall et al. 2021; Liu et al. 2020a,b; Niemeyer et al. 2020; Oechsle et al. 2019, 2020; Saito et al. 2019; Schwarz et al. 2020; Sitzmann, Zollhöfer, and Wetzstein 2019) as they represent 3D content continuously and without discretization while simultaneously having a small memory footprint. Most of these methods require 3D supervision. However, several recent works (Mildenhall et al. 2021; Liu et al. 2019; Niemeyer et al. 2020; Sitzmann, Zollhöfer, and Wetzstein 2019; Yariv et al. 2020) demonstrated differentiable rendering for training directly from images. Some works use estimated depth information for surface rendering without pixel-accurate object masks (Verbin et al. 2022). Some works enhance the accuracy of geometric information by incorporating point cloud information and warp operations (Fu et al. 2022; Darmon et al. 2022). some works do not model the density field and use other fields to avoid threshold selection (Oechsle, Peng, and Geiger 2021; Wang et al. 2021; Fu

et al. 2022; Darmon et al. 2022; Yariv et al. 2021; Wang, Skorokhodov, and Wonka 2022). However, these works are still based on the continuous representation, the same threshold cannot accurately divide the surface. Different from previous methods, we model the 3D geometric information in a discontinuous representation by proposing a hybrid ANN-SNN framework.

Spiking Neural Networks in Computer Vision

Over the past few years, brain-inspired (Mainen and Sejnowski 1995; Li et al. 2021a) deep SNNs using spike trains (Marchisio et al. 2020) have gradually replaced ANNs in various tasks (Pfeiffer and Pfeil 2018; Davies et al. 2018), and it is extensively utilized to develop energy-efficient solutions for various tasks. Some works use SNNs to process video or event streams (Zhu et al. 2022; Gehrig et al. 2020). Some works exploit the discontinuity of network to enhance its robustness (Bagheri, Simeone, and Rajendran 2018; Sharmin et al. 2019, 2020). Some works leverage the neural parameter of SNNs to further improve the utilization of available information (Li et al. 2022; Zhang et al. 2022). Although previous works have demonstrated SNN applications on a wide range of tasks, they are still limited in their performance (Neftci, Mostafa, and Zenke 2019; Pascanu, Mikolov, and Bengio 2013; Li et al. 2021b; Fang et al. 2021). Meanwhile, there has been a growing interest in exploring the potential benefits of combining ANNs and SNNs (Kugele et al. 2021; Yang et al. 2019; Lee et al. 2020; Liu and Zhao 2022; Zhao et al. 2022). By combining ANNs and SNNs, better performance can be achieved while reducing the time step of SNNs. Different combination strategies have been explored for a variety of tasks. A group of work employs the strategy of processing the accumulated spike train of SNNs with ANNs (Kugele et al. 2021; Lee et al. 2020; Liu and Zhao 2022). In these works, the SNN is used as an efficient encoder of spatio-temporal data. The output of the SNN is accumulated to summarize the temporal dimension before the ANN processes the accumulated features (Kugele et al. 2021; Lee et al. 2020). A second line of work uses a strategy where the output of the independently operating SNN and ANN is fused (Lele et al. 2022; Yang et al. 2019; Zhao et al. 2022). In these works, the output of both networks are fused based on heuristics (Lele et al. 2022), temporal filtering (Yang et al. 2019), or accumulation based on the output of the ANN (Zhao et al. 2022). Different from the previous methods, we apply the hybrid ANN-SNN framework to a non-cognitive application.

Preliminary

Neural Radiance Fields

NeRF (Mildenhall et al. 2021) represents a scene as a continuous volumetric field, where the density $\sigma \in R^3$ and radiance $c \in R^3$ at any 3D position $\mathbf{x} \in R^3$ under viewing direction $\mathbf{d} \in R^2$ are modeled by a MLP $f_\theta : (\mathbf{x}, \mathbf{d}) \rightarrow (c, \sigma)$, with θ as learnable parameters. To render a pixel, the MLP first evaluates points sampled from the camera ray $r = \mathbf{o} + t\mathbf{d}$ to get their densities and radiance. The light starts at 0 and stops at a hypothetical large density T. Define C_p as the estimated pixel color and $\hat{C}_p$ as the ground truth. NeRF is optimized by minimizing the photometric loss:

$$L_{\text{rgb}} = \sum_{\text{p}} \left|\left| C_{\text{p}} - \hat{C}_{\text{p}} \right|\right|. \tag{1}$$

Here, p refers to each pixel as defined in (Guo et al. 2022). There are currently two main methods for depth estimation: one based on the integration (Zhang et al. 2021; Deng et al. 2022) and the other based on the threshold to get global shape (Wang et al. 2021; Ichnowski et al. 2021; Fu et al. 2022). Since we focus on a global geometric shape extracted from the network by threshold rather than a depth map dependent on perspective, this paper mainly follow the second method to define the depth.

Spiking Neuron

Integrate-and-fire model. We introduce the well-known Integrate-and-Fire (IF) neuron model (Leroy, Franco, and Boyer 2018) for SNNs. Given a membrane potential u_t at time step t, the membrane potential u_{t+1}^{pre} before firing at time step $t+1$ is updated as

$$u_{t+1}^{\text{pre}} = u_t + W x_{t+1}. \tag{2}$$

Here, the elements, W and x_{t+1}, represent respectively the weight and the output from previous layer at time step $t+1$. V_{th} is the firing threshold. The spiking neuron will fire a spike when u_{t+1}^{pre} exceeds the V_{th} (see Fig. 2), and then the membrane potential u_{t+1} at time step $t+1$ resets to 0. The spike output is given by

$$o_{t+1} = \begin{cases} 0 & u_{t+1}^{\text{pre}} < V_{\text{th}} \\ 1 & \text{otherwise}, \end{cases} \tag{3}$$

$$y_{t+1} = o_{t+1} \cdot V_{\text{th}}. \tag{4}$$

After firing, the spike output y_{t+1} will propagate to the next layer and become the input x_{t+1} of the next layer. Note that we omit the layer index for simplicity.

Full-precision integrate-and-fire model. However, the spikes convey less information than floating-point number, and networks using IF neurons do not perform well in regression tasks (Henkes, Eshraghian, and Wessels 2022; Iannella and Back 2001; Gehrig et al. 2020; Rançon et al. 2022; Kahana et al. 2022; Lu et al. 2021; Shrestha and Orchard 2018). It's difficult to ensure accuracy by directly using IF neurons for modeling. To obtain full-precision information, (Li et al. 2022) changes the Eqn. 4 to:

$$y_{t+1} = o_{t+1} \cdot u_{t+1}^{\text{pre}}. \tag{5}$$

By replacing the output V_{th} with u_{t+1}^{pre}, the full-precision information is reserved. Due to the full-precision information meeting our requirements, we will only consider one time step and omit t in following formulations.

Spiking NeRF

To make the density field discontinuous, a trivial way is to introduce spiking neurons in NeRF, which replaces the last activation layer of density network with spiking neurons (e.g., IF, FIF). However, on the one hand, as previously mentioned, using IF is difficult to ensure accuracy, and on the other hand, directly replacing the activation layer with FIF will result in a significant depth error when the spiking threshold is small (see ablation study for more details).

In this section, we first analyze what kind of spiking neurons can solve the problems of post-processing and optimal threshold perturbation, and alleviate the problem of light density scenarios by building the relationship between the spiking threshold, maximum activation, and depth error. Then, based on this relationship, we further propose our method.

Relationship between Parameters of Spiking Neuron and Depth Error

We define that d_{v} is the estimated depth and d is the accurate depth. T refers to the length of the sampling range and Δt refers to the sampling interval. V_{th} is the spiking threshold and V_{max} is the maximum density value. For a well-trained NeRF, the V_{th} is also the non zero minimum value of its density field, and the density value of the first point that a ray encounters with a non zero density is V_{th}. We build the relationship based on the following Proposition 1 (see supplementary material for proving).

Proposition 1. For a well-trained NeRF, we have:

$$|d - d_{\text{v}}| < \max((\Delta t - Te^{-V_{\text{max}}\Delta t})e^{-V_{\text{th}}\Delta t}, T(1 - e^{-V_{\text{max}}T})e^{-V_{\text{th}}\Delta t}). \quad (6)$$

Based on the relationship, when the V_{th} is sufficiently large, $e^{-V_{\text{th}}\Delta t}$ is sufficiently small, resulting in a sufficiently small error. Meanwhile, for a fixed V_{th}, a small V_{max} can decrease the error (see supplementary material for more details).

However, the V_{th} cannot be set sufficiently high in semi-transparent scenarios and cannot be set to infinity in practical implementation. Meanwhile, FIF does not have V_{max} constraint (see the right of Fig. 2), resulting in a large positive $(\Delta t - Te^{-V_{\text{max}}\Delta t})$. In this case, significant errors may occur in semi-transparent scenarios and practical implementation. Direct use of the FIF spiking neuron is not able to completely solve the above-mentioned problems. Therefore, V_{max} should have a constraint to decrease the error and we need a spiking neuron with a relatively small maximum activation.

B-FIF: Bounded full-precision integrate and fire spiking neuron. Based on previous analysis, the maximum activation close to the spiking threshold can ensure lower error. Therefore, we need to constrain the maximum activation of spiking neuron to decrease the error. In this paper, we use $\tanh()$ to constrain the maximum activation (see supplementary material for other bounded functions). The u^{pre} in Eq. 5 is reformulated as:

$$u^{\text{pre}} = k \cdot \tanh(u + Wx), \quad (7)$$

$$\sigma = y = o \cdot u^{\text{pre}}. \quad (8)$$

Here, k is a learnable parameter. By increasing k, it is ensured that the spiking neuron can have a larger spiking threshold. The proposed neuron constrains the maximum output value while ensuring accuracy as it is also a full-precision neuron. So the spiking neuron can ensure that the network is trained accurately and also decrease errors when the spiking threshold is relatively small as its maximum activation approaches its spiking threshold.

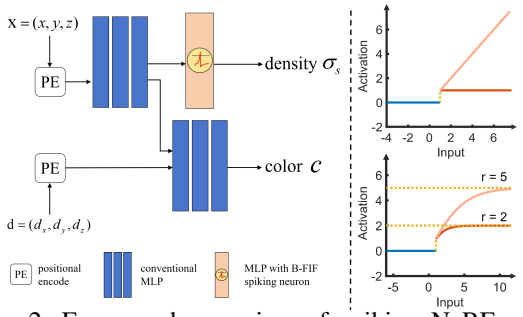

Figure 2: Framework overview of spiking NeRF and an illustration of different existing spiking neuron models and the proposed one. Left: The network structure of our approach. We use a NeRF model following (Mildenhall et al. 2021), but excluding the last activation layer of density network. Instead of using ReLU, we use B-FIF spiking neuron to make the density field discontinuous. Right top: the IF and FIF spiking neuron. Right bottom: the B-FIF spiking neuron with different r ($r = 2$ and 5). These curves show that B-FIF become more similar to FIF as the parameter r increases. And when the r is sufficiently large, the B-FIF spiking neuron degenerate to the FIF spiking neuron.

Hybrid ANN-SNN Framework

B-FIF implementation. Directly using the previously mentioned spiking neuron can lead to slow training. It is potentially caused by the derivative of the initial network output approaching 0 (see Fig. 2), leading to a smaller learning step size. Therefore, to better train the network, we introduce a learnable parameter r for the proposed neurons to increase the derivative of the initial network output (see Fig. 2), which can improve training.

$$u^{\text{pre}} = k \cdot r \cdot \tanh(\frac{u + Wx}{r}), \quad (9)$$

We set the initial value of r and k to 10 and 1 to maintain similarity to the original NeRF output (i.e., ReLU (Mildenhall et al. 2021; Azinović et al. 2022; Wang et al. 2021)) near point 0 (see Fig. 2). This setting ensures that the network maintains performance similar to NeRF during the initial training stage, which can ensure that the range of the derivative approaching 1 is expanded. In addition to above advantage, this initial strategy and the proposed neuron in Eq. 9 have another advantage that the network is trained with an increased penalty for small density regions that should not appear, and it is able to limit the density value that should not appear above the spiking threshold more easily during training.

It can be observed that when r is sufficiently large, B-FIF spiking neuron degenerates into FIF spiking neuron, and when the maximum activation approaches the threshold, B-FIF spiking neuron degenerates into IF spiking neuron (please find Fig. 2 for the comparison of different spiking neuron models).

Loss function. Based on Proposition 1, it is necessary to have a large spiking threshold to obtain accurate geometric information. The network can learn the spiking threshold during the training process. However, duo to the difficulty of

the network spontaneously pushing the spiking threshold to a larger value, the spiking threshold at the end of the training process may not necessarily be relatively large. Therefore, we propose L_{v} as a regularization term to increase the spiking threshold and ensure that the spiking threshold does not remain a small value after the training is completed. The L_{v} is as follows:

$$L_{\text{v}} = \frac{1}{V_{\text{th}}}. \tag{10}$$

Following (Verbin et al. 2022), we use a regularization term L_{g} to maintain the smoothness of the geometric representation in the initial stage, improving the network convergence.

$$L_{\text{g}} = \sum_{\text{p}} \sum_{i} w_i \max(-\mathbf{d}_{\text{p}} \nabla \sigma_i, 0)^2. \tag{11}$$

Here, p refers to each pixel as defined in (Guo et al. 2022). i refers to each sampling point and w_i refers to the weight at each sampling point as defined in (Verbin et al. 2022). However, in (Verbin et al. 2022), they specifically designed a MLP to predict the normal vector, which generated more parameters that needed to be learned. We directly use the normal vector calculated by the gradient of density on the input coordinates to calculate L_{g}, reducing the number of parameters that need to be learned while reducing network complexity.

Finally, We optimize the following loss function:

$$L = L_{\text{rgb}} + \lambda_1 L_{\text{v}} + \lambda_2 L_{\text{g}}. \tag{12}$$

Surrogate gradient. The non-differentiability of the firing function remains one of the most significant challenges in training SNNs (Neftci, Mostafa, and Zenke 2019). Direct training requires the use of the surrogate gradient (Pfeiffer and Pfeil 2018; Li et al. 2022, 2021c; Bagheri, Simeone, and Rajendran 2018). There are 2 kinds of surrogate gradient, the surrogate gradient for conventional spiking neurons and the surrogate gradient for full-precision spiking neurons. The proposed neuron is based on the full-precision spiking neuron, so we use the surrogate gradient for full-precision spiking neurons. However, there are multiple time steps in (Li et al. 2022), and the surrogate gradient of the spiking threshold is based on the number of spikes, while our method only uses one time step. Directly using the surrogate gradient in (Li et al. 2022) can lead to unstable training (see supplement material for more details). In this paper, we use the piecewise linear function similar to (Li et al. 2021c) as the surrogate gradient for V_{th} to avoid the gradient that is 0 almost everywhere. The surrogate gradient of B-FIF is as follows:

$$\frac{\partial y}{\partial u^{\text{pre}}} = o, \tag{13}$$

$$\frac{\partial y}{\partial V_{\text{th}}} = \lambda \max(0, \frac{k - |u^{\text{pre}} - V_{\text{th}}|}{k^2}). \tag{14}$$

Meanwhile, due to the use of full-precision spiking neurons, we only need one time step to meet requirements. So theoretically, our training complexity is similar to that of ANN.

Training strategy. During the initial stage of network training, the density field exhibits significant deviations. A larger spiking threshold will render the network untrainable because most of the membrane potential cannot reach the spiking threshold and is not in the range of surrogate gradient non-zero. So we initially set the spiking threshold to 0. Furthermore, we set λ_1 to 0.15 at first and continue to increase λ_1 as the training process progresses. To preserve true high-frequency geometric information, we initially set the λ_2 to 0.001 and gradually decrease the proportion of L_{g} in L as the training progresses by decreasing λ_2.

Experiments

Experimental Settings

Datasets. We evaluate our methods in 8 scenes from the Blender dataset (Mildenhall et al. 2021) and 6 scenes from Dex-NeRF (Ichnowski et al. 2021). Following most 3D reconstruction methods (Wang et al. 2021; Oechsle, Peng, and Geiger 2021; Wang, Skorokhodov, and Wonka 2022, 2023), we also evaluate our methods in 15 scenes from the DTU dataset (Jensen et al. 2014). These datasets cover different types of scenes or objects and are benchmarks for NeRF-based methods. Furthermore, we additionally use 2 self created scenes to show that our method is better than previous methods in special scenarios (see supplementary material for more details).

Implementation details. We sample 512 rays per batch and train our model for 200k iterations for 3 hours on a single NVIDIA RTX3080Ti GPU. Our network architecture and initialization scheme are similar to those of NeRF (Mildenhall et al. 2021) and we model the background using NeRF++ (Zhang et al. 2020) with the same settings as NeuS in the real scene dataset. More details of the network architecture and training parameters can be found in the supplementary material.

Metrics. Following most 3D reconstruction methods, we measure the reconstruction quality with the Chamfer distances (Wang et al. 2021; Wang, Skorokhodov, and Wonka 2022, 2023; Fu et al. 2022; Darmon et al. 2022).

Overall Performance

Comparison methods. We compared our method with 2 kinds of NeRF-based methods, conventional NeRF-based method and the SDF-based methods. For conventional NeRF-based method, we compared with NeRF (Mildenhall et al. 2021), MipNeRF (Barron et al. 2021) (referred to as "Mip" in the table), For SDF-based methods, we compare with NeuS (Wang et al. 2021), Hf-NeuS (Wang, Skorokhodov, and Wonka 2022) (referred to as "HFS" in the table) and PET-NeuS (Wang, Skorokhodov, and Wonka 2023) (referred to as "PET" in the table). PET-NeuS and Hf-NeuS are follow-up works on NeuS and achieve more detailed geometry reconstruction. We did not include IDR (Yariv et al. 2020), UNISURF (Oechsle, Peng, and Geiger 2021) or VolSDF (Yariv et al. 2021) as NeuS had shown superior performance on Chamfer distances. We report the Chamfer distances in Tab. 3, Tab. 2, and Tab. 1, and conduct the qualitative comparisons in Fig. 3.

Quantitative comparison. As shown in Tab. 1, these SDF-based methods failed in semi-transparent scenes, resulting in significant Chamfer distances. This indicates that the SDF-based method cannot reconstruct semi-transparent scenes. Meanwhile, as shown in Tab. 2, our

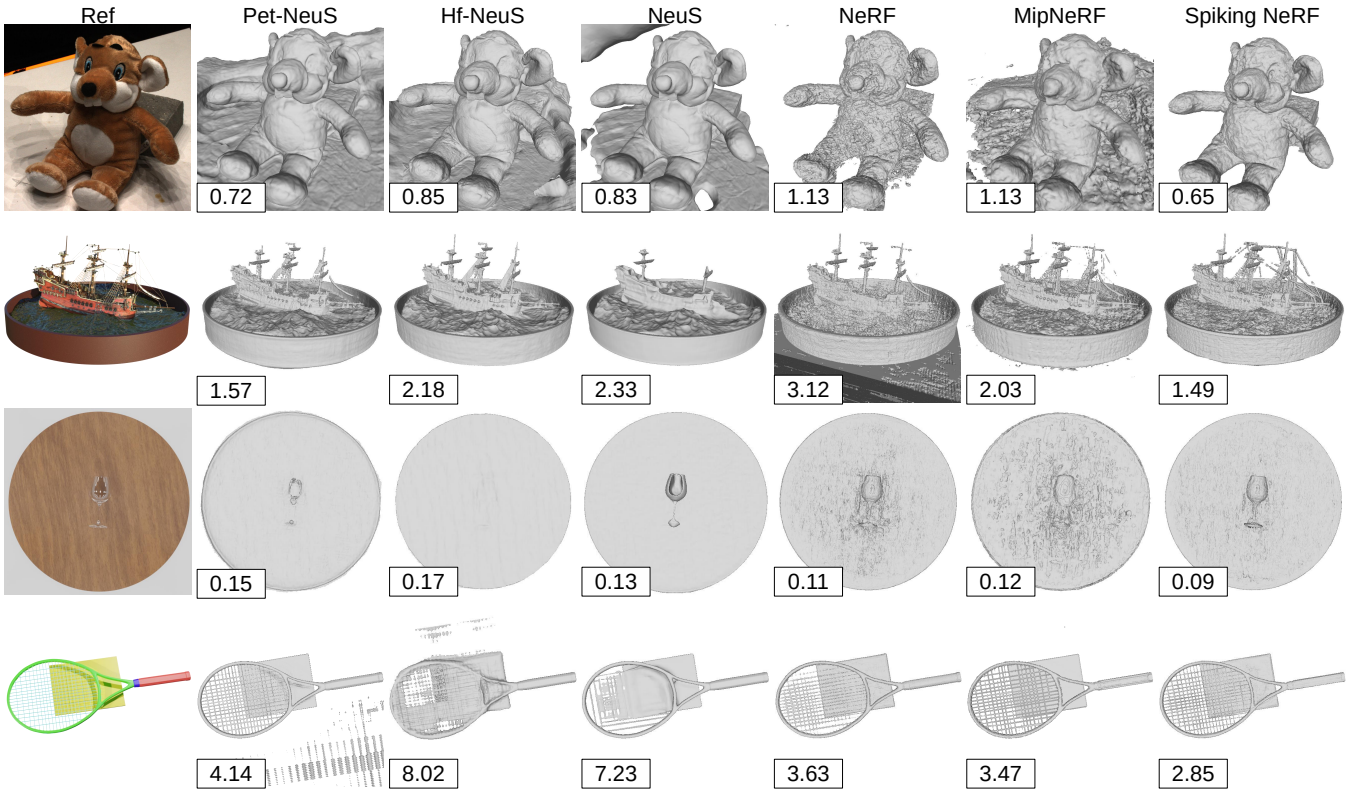

Figure 3: Visual Quality Comparisons on surface reconstruction on Blender dataset (Mildenhall et al. 2021), DTU dataset (Jensen et al. 2014), semi-transparent dataset (Ichnowski et al. 2021), and thin object dataset. We show the Chamfer distance in the bottom left corner of the image. The results of the 2nd and 4th rows are multiplied by 10^2.

average Chamfer distances are the lowest in Blender dataset (Mildenhall et al. 2021). This indicates that our method can obtain more accurate geometric information. It is rare (Ren et al. 2023) that the result of SNN is better than that of ANN. As shown in Tab. 3, we have a similar average Chamfer distances to SDF-based methods in the DTU dataset. While our method achieves leading results, some of our results are less competitive compared to some mainstream methods (e.g. Pet-NeuS, Hf-Neus). Because SDF-based methods adopt lower frequency position encoding and constrain the variation of output with input, they have good performance for low frequency and smooth scenes. However, when these premises are not met, their performance will decrease.

Since our method is based on conventional NeRF (Mildenhall et al. 2021) and is similar to NeRF, the robustness of our method closely resembles that of NeRF. The actual performance can be seen in Tab. 2 and Tab. 3 which show results on the general real scene (i.e. DTU dataset (Jensen et al. 2014)) and the general synthetic scene (i.e. Blender dataset (Mildenhall et al. 2021)). For conventional NeRF-based methods, even the results obtained by manually selecting the optimal threshold are worse than our methods. Because our method uses the hybrid ANN-SNN framework to model the 3D geometric information in a discontinuous representation, the reconstructed geometric information is more accurate.

Table 1: Quantitative Comparison on light scenarios (Ichnowski et al. 2021). We show the Chamfer distance $\times 10^{-2}$ for the reconstructed geometry on 6 scenes from the Dex-NeRF dataset and 2 scenes from our Blender dataset.

Method	wheel	tennis	mount	glass	turbine	clutter	pawn	pipe	Avg.
PET	13.23	4.14	15.94	15.32	15.34	23.28	14.31	14.23	14.47
HFS	30.41	8.02	15.35	17.24	11.53	17.23	15.78	12.57	16.01
NeuS	23.85	7.23	12.75	13.14	11.70	25.32	14.30	11.53	14.97
NeRF	7.47	3.63	10.22	11.53	10.06	13.15	12.04	10.12	9.77
Mip	11.31	3.47	10.30	12.74	10.25	12.36	11.67	**9.23**	10.16
Ours	**6.42**	**2.85**	**9.87**	**9.45**	**9.82**	**10.44**	**10.87**	10.32	**8.75**

Visual quality comparison. As shown in the 5th and 6th columns of Fig. 3, with conventional NeRF-based methods, determining the threshold that can accurately segment the surface requires enumeration attempts. We choose the optimal threshold manually for them. And due to the lack of surface constraints, the reconstruction results have many erroneous high-frequency information, and there is a loss of reconstruction accuracy for thin objects and semi-transparent scenes. As shown in the 2nd, 3rd and 4th columns of Fig. 3, with SDF-based methods, the reconstruction resulted in significant errors in high-frequency changes in the image, such as failing to reconstruct holes that should have been present and surfaces that should have been separate. Moreover, NeuS demonstrated excessive smoothness in 2nd row. And NeuS, HF-NeuS, and PET-NeuS fail in the reconstruction of semi transparent scenes. Even though HF-NeuS uses methods

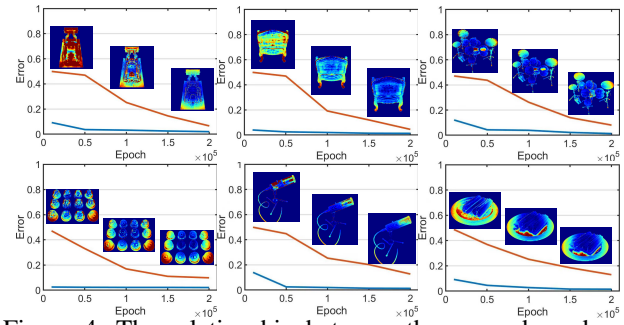

Figure 4: The relationship between the upper bound and the average depth error during training. We show six scenes from Blender dataset (Mildenhall et al. 2021). We randomly choose a view for displaying from each scene, and compute error and upper bound in epoch $=$ 10K, 50K, 100K, 150K and 200K (see supplementary material for more scenes). The red curve represents upper bound while the blue curve represents the average depth error during training. It can be seen that the average depth error decreases with the upper bound and the average depth error keeps being less than upper bound during training.

to reconstruct high-frequency information, it still fail in some thin high-frequency positions. Because our method uses the hybrid ANN-SNN framework to model the 3D geometric information in a discontinuous representation, the reconstructed geometric information is more accurate.

The practicality of Upper Bound of Proposition 1

To explore whether the bound is practical, we verify whether there is a correlation between the bound and the average depth error, and whether the bound has similar patterns across different scenarios. Because the Blender dataset provides depth information (Mildenhall et al. 2021), we conduct our experiments on it. As shown in Fig. 4, the average depth error is under the bound. Moreover, there is indeed correlation between the bound and the average depth error, and the bound has similar patterns across different scenarios. It means that the bound can be leveraged to facilitate the real-world application for geometric estimation without knowing the ground truth. However, note that there is still a relatively large gap between the bound and the average depth error. While we made a new attempt in the field of NeRF and this bound can be used to a certain extent, it is still not enough to be practical.

Ablation Study

Validation for proposition 1. When L_{v} is minimized, the spiking threshold will continue to increase, and when L_{v} is removed, the spiking threshold of the final network will remain at a lower value. To validate the proposition 1, we design an experiment without L_{v} and compare results with our method. It can be seen that there are many areas that should not exist above the reconstructed surface (see the 3rd column of Fig. 5), indicating that a relatively large spiking threshold is necessary, which is consistent with our analysis.

Effectiveness of B-FIF. To evaluate the effectiveness of the B-FIF, we design an experiment that replaced B-FIF spiking neuron with FIF spiking neuron which does not have a maximum activation constraint and compare the result with our method. It can be seen that the reconstructed surface has significant deviation (see the 2nd column of Fig. 5), indicating that a bound is necessary, which is consistent with our analysis.

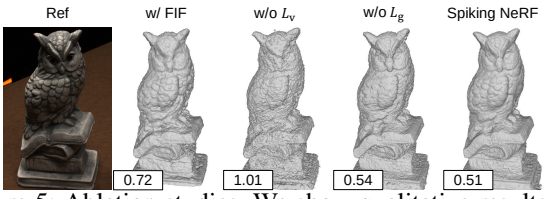

Figure 5: Ablation studies. We show qualitative results and report the quantitative metrics in Chamfer distance.

Table 2: Quantitative Comparison on Blender. We show the Chamfer distance $\times 10^{-2}$ for the reconstructed geometry on 8 scenes from the Blender dataset.

Method	lego	chair	mic	ficus	hotdog	drums	materials	ship	Avg.
PET	**0.58**	**0.65**	**0.59**	0.71	1.02	2.53	**1.05**	1.57	1.09
HFS	0.96	0.65	0.72	0.87	1.35	3.82	1.08	2.18	1.45
NeuS	1.52	0.70	0.85	1.67	1.40	4.27	1.08	2.33	1.73
NeRF	2.06	0.75	0.95	0.56	1.83	3.38	1.12	3.12	1.72
Mip	1.92	0.90	1.13	0.55	1.98	3.34	1.30	2.03	1.64
Ours	0.70	0.66	0.72	**0.54**	**0.94**	**2.43**	1.10	**1.49**	**1.07**

Table 3: Quantitative Comparison on DTU (Jensen et al. 2014). We show a quantitative comparison for the reconstructed geometry on 15 scans from the DTU dataset.

Method	24	37	40	55	63	65	69	83	97	105	106	110	114	118	122	Avg.
PET	**0.59**	**0.84**	**0.72**	0.40	**0.95**	0.84	0.72	1.42	**1.14**	0.72	0.53	**1.05**	0.44	0.59	0.68	**0.78**
HFS	0.83	1.37	0.78	0.47	1.11	**0.68**	**0.68**	1.20	1.17	0.85	0.57	1.27	0.38	0.54	0.55	0.83
NeuS	1.00	1.37	0.93	0.43	1.10	0.70	0.72	1.48	1.16	0.83	0.52	1.69	**0.35**	**0.49**	0.54	0.89
NeRF	1.90	1.63	1.75	0.60	2.03	1.07	1.47	1.70	1.95	1.13	0.79	2.33	0.87	1.05	0.86	1.41
Mip	2.15	1.83	1.65	1.61	2.90	1.79	1.52	1.93	2.19	1.32	1.79	2.52	1.38	1.49	1.21	1.82
Ours	0.84	1.20	1.02	**0.38**	1.15	0.72	0.69	**1.10**	1.19	**0.65**	**0.49**	1.60	0.49	0.55	**0.51**	0.83

Effectiveness of smoothness. To evaluate the effectiveness of the initial smoothing, we design an experiment without L_{g} and compare the result with our method. It can be seen that although the Chamfer distances are similar, the reconstructed surface contain wrong high-frequency information (see the 4th column of Fig. 5), indicating that training from wrong high-frequency surface is difficult and training from smooth surface is necessary.

Conclusion

We present a novel neural surface reconstruction method for reconstructing objects with high fidelity from 2D image inputs. NeRF and its variants use the volume rendering to produce a neural scene representation. However, extracting high-quality surfaces from this learned implicit representation is difficult because there are not a definite interface density. In our work, we propose to use a hybrid ANN-SNN framework to reconstruct density fields. We observe that the conventional volume rendering method causes inherent geometric errors for surface reconstruction, and propose a new spiking neuron to get more accurate surface reconstruction with adaptive spiking threshold.

Limitation. Since NeRF (Mildenhall et al. 2021) does not specifically consider scenes with high light and low brightness, it will struggle to accurately represent the geometry on these scenes (Verbin et al. 2022). Our method also did not specifically consider these issues, so it is highly likely to struggle to accurately represent the geometry on these scenes.

References

Aimone, J.; Date, P.; Fonseca-Guerra, G.; Hamilton, K.; Henke, K.; Kay, B.; Kenyon, G.; Kulkarni, S.; Mniszewski, S.; Parsa, M.; et al. 2022. A review of non-cognitive applications for neuromorphic computing. *Neuromorphic Computing and Engineering*.

Atzmon, M.; Haim, N.; Yariv, L.; Israelov, O.; Maron, H.; and Lipman, Y. 2019. Controlling neural level sets. *Advances in Neural Information Processing Systems*, 32.

Azinović, D.; Martin-Brualla, R.; Goldman, D. B.; Nießner, M.; and Thies, J. 2022. Neural rgb-d surface reconstruction. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 6290–6301.

Bagheri, A.; Simeone, O.; and Rajendran, B. 2018. Adversarial training for probabilistic spiking neural networks. In *2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 1–5. IEEE.

Barron, J. T.; Mildenhall, B.; Tancik, M.; Hedman, P.; Martin-Brualla, R.; and Srinivasan, P. P. 2021. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proc. International Conference on Computer Vision (ICCV)*, 5855–5864.

Boss, M.; Braun, R.; Jampani, V.; Barron, J. T.; Liu, C.; and Lensch, H. 2021. Nerd: Neural reflectance decomposition from image collections. In *Proc. International Conference on Computer Vision (ICCV)*, 12684–12694.

Chen, L.-C.; Zhu, Y.; Papandreou, G.; Schroff, F.; and Adam, H. 2018. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proc. European Conference on Computer Vision (ECCV)*, 801–818.

Darmon, F.; Bascle, B.; Devaux, J.-C.; Monasse, P.; and Aubry, M. 2022. Improving neural implicit surfaces geometry with patch warping. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 6260–6269.

Davies, M.; Srinivasa, N.; Lin, T.-H.; Chinya, G.; Cao, Y.; Choday, S. H.; Dimou, G.; Joshi, P.; Imam, N.; Jain, S.; et al. 2018. Loihi: A neuromorphic manycore processor with on-chip learning. *Ieee Micro*, 38(1): 82–99.

Deng, K.; Liu, A.; Zhu, J.-Y.; and Ramanan, D. 2022. Depth-supervised nerf: Fewer views and faster training for free. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 12882–12891.

Fang, W.; Yu, Z.; Chen, Y.; Huang, T.; Masquelier, T.; and Tian, Y. 2021. Deep residual learning in spiking neural networks. *Advances in Neural Information Processing Systems*, 34: 21056–21069.

Fu, Q.; Xu, Q.; Ong, Y. S.; and Tao, W. 2022. Geo-neus: Geometry-consistent neural implicit surfaces learning for multi-view reconstruction. *Advances in Neural Information Processing Systems*, 35: 3403–3416.

Gehrig, M.; Shrestha, S. B.; Mouritzen, D.; and Scaramuzza, D. 2020. Event-based angular velocity regression with spiking networks. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 4195–4202. IEEE.

Genova, K.; Cole, F.; Vlasic, D.; Sarna, A.; Freeman, W. T.; and Funkhouser, T. 2019. Learning shape templates with structured implicit functions. In *Proc. International Conference on Computer Vision (ICCV)*, 7154–7164.

Guo, Y.-C.; Kang, D.; Bao, L.; He, Y.; and Zhang, S.-H. 2022. Nerfren: Neural radiance fields with reflections. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 18409–18418.

Han, J.; Wang, Z.; Shen, J.; and Tang, H. 2023. Symmetric-threshold ReLU for Fast and Nearly Lossless ANN-SNN Conversion. *Machine Intelligence Research*, 20(3): 435–446.

Henkes, A.; Eshraghian, J. K.; and Wessels, H. 2022. Spiking neural network for nonlinear regression. *arXiv preprint arXiv:2210.03515*.

Iannella, N.; and Back, A. D. 2001. A spiking neural network architecture for nonlinear function approximation. *Neural networks*, 14(6-7): 933–939.

Ichnowski, J.; Avigal, Y.; Kerr, J.; and Goldberg, K. 2021. Dex-NeRF: Using a Neural Radiance Field to Grasp Transparent Objects. In *5th Annual Conference on Robot Learning*.

Jensen, R.; Dahl, A.; Vogiatzis, G.; Tola, E.; and Aanæs, H. 2014. Large scale multi-view stereopsis evaluation. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 406–413.

Kahana, A.; Zhang, Q.; Gleyzer, L.; and Karniadakis, G. E. 2022. Function regression using spiking deeponet. *arXiv preprint arXiv:2205.10130*.

Kugele, A.; Pfeil, T.; Pfeiffer, M.; and Chicca, E. 2021. Hybrid SNN-ANN: Energy-efficient classification and object detection for event-based vision. In *DAGM German Conference on Pattern Recognition*, 297–312. Springer.

Lee, C.; Kosta, A. K.; Zhu, A. Z.; Chaney, K.; Daniilidis, K.; and Roy, K. 2020. Spike-flownet: event-based optical flow estimation with energy-efficient hybrid neural networks. In *Proc. European Conference on Computer Vision (ECCV)*, 366–382. Springer.

Lele, A. S.; Fang, Y.; Anwar, A.; and Raychowdhury, A. 2022. Bio-mimetic high-speed target localization with fused frame and event vision for edge application. *Frontiers in Neuroscience*, 16: 1010302.

Leroy, V.; Franco, J.-S.; and Boyer, E. 2018. Shape reconstruction using volume sweeping and learned photoconsistency. In *Proc. European Conference on Computer Vision (ECCV)*, 781–796.

Levy, D.; Peleg, A.; Pearl, N.; Rosenbaum, D.; Akkaynak, D.; Korman, S.; and Treibitz, T. 2023. SeaThru-NeRF: Neural Radiance Fields in Scattering Media. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 56–65.

Li, S.; Zhang, Z.; Mao, R.; Xiao, J.; Chang, L.; and Zhou, J. 2021a. A fast and energy-efficient SNN processor with adaptive clock/event-driven computation scheme and online learning. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 68(4): 1543–1552.

Li, W.; Chen, H.; Guo, J.; Zhang, Z.; and Wang, Y. 2022. Brain-inspired multilayer perceptron with spiking neurons. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 783–793.

Li, Y.; Deng, S.; Dong, X.; Gong, R.; and Gu, S. 2021b. A free lunch from ANN: Towards efficient, accurate spiking neural networks calibration. In *Proc. International Conference on Machine Learning (ICML)*, 6316–6325. PMLR.

Li, Y.; Guo, Y.; Zhang, S.; Deng, S.; Hai, Y.; and Gu, S. 2021c. Differentiable spike: Rethinking gradient-descent for training spiking neural networks. *Advances in Neural Information Processing Systems*, 34: 23426–23439.

Liu, F.; and Zhao, R. 2022. Enhancing spiking neural networks with hybrid top-down attention. *Frontiers in Neuroscience*, 16: 949142.

Liu, L.; Gu, J.; Zaw Lin, K.; Chua, T.-S.; and Theobalt, C. 2020a. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33: 15651–15663.

Liu, S.; Saito, S.; Chen, W.; and Li, H. 2019. Learning to infer implicit surfaces without 3d supervision. *Advances in Neural Information Processing Systems*, 32.

Liu, S.; Zhang, Y.; Peng, S.; Shi, B.; Pollefeys, M.; and Cui, Z. 2020b. Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2019–2028.

Lu, L.; Jin, P.; Pang, G.; Zhang, Z.; and Karniadakis, G. E. 2021. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3): 218–229.

Mainen, Z. F.; and Sejnowski, T. J. 1995. Reliability of spike timing in neocortical neurons. *Science*, 268(5216): 1503–1506.

Marchisio, A.; Nanfa, G.; Khalid, F.; Hanif, M. A.; Martina, M.; and Shafique, M. 2020. Is spiking secure? a comparative study on the security vulnerabilities of spiking and deep neural networks. In *2020 International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

Mescheder, L.; Oechsle, M.; Niemeyer, M.; Nowozin, S.; and Geiger, A. 2019. Occupancy networks: Learning 3d reconstruction in function space. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 4460–4470.

Michalkiewicz, M.; Pontes, J. K.; Jack, D.; Baktashmotlagh, M.; and Eriksson, A. 2019. Implicit surface representations as layers in neural networks. In *Proc. International Conference on Computer Vision (ICCV)*, 4743–4752.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1): 99–106.

Neftci, E. O.; Mostafa, H.; and Zenke, F. 2019. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6): 51–63.

Niemeyer, M.; Mescheder, L.; Oechsle, M.; and Geiger, A. 2019. Occupancy flow: 4d reconstruction by learning particle dynamics. In *Proc. International Conference on Computer Vision (ICCV)*, 5379–5389.

Niemeyer, M.; Mescheder, L.; Oechsle, M.; and Geiger, A. 2020. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 3504–3515.

Oechsle, M.; Mescheder, L.; Niemeyer, M.; Strauss, T.; and Geiger, A. 2019. Texture fields: Learning texture representations in function space. In *Proc. International Conference on Computer Vision (ICCV)*, 4531–4540.

Oechsle, M.; Niemeyer, M.; Reiser, C.; Mescheder, L.; Strauss, T.; and Geiger, A. 2020. Learning implicit surface light fields. In *2020 International Conference on 3D Vision (3DV)*, 452–462. IEEE.

Oechsle, M.; Peng, S.; and Geiger, A. 2021. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proc. International Conference on Computer Vision (ICCV)*, 5589–5599.

Park, J. J.; Florence, P.; Straub, J.; Newcombe, R.; and Lovegrove, S. 2019. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 165–174.

Pascanu, R.; Mikolov, T.; and Bengio, Y. 2013. On the difficulty of training recurrent neural networks. In *Proc. International Conference on Machine Learning (ICML)*, 1310–1318. Pmlr.

Peng, S.; Niemeyer, M.; Mescheder, L.; Pollefeys, M.; and Geiger, A. 2020. Convolutional occupancy networks. In *Proc. European Conference on Computer Vision (ECCV)*, 523–540. Springer.

Peng, S.; Zhang, Y.; Xu, Y.; Wang, Q.; Shuai, Q.; Bao, H.; and Zhou, X. 2021. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 9054–9063.

Pfeiffer, M.; and Pfeil, T. 2018. Deep learning with spiking neurons: Opportunities and challenges. *Frontiers in neuroscience*, 12: 774.

Pumarola, A.; Corona, E.; Pons-Moll, G.; and Moreno-Noguer, F. 2021. D-nerf: Neural radiance fields for dynamic scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 10318–10327.

Rançon, U.; Cuadrado-Anibarro, J.; Cottereau, B. R.; and Masquelier, T. 2022. Stereospike: Depth learning with a spiking neural network. *IEEE Access*, 10: 127428–127439.

Ren, D.; Ma, Z.; Chen, Y.; Peng, W.; Liu, X.; Zhang, Y.; and Guo, Y. 2023. Spiking PointNet: Spiking Neural Networks for Point Clouds. In *Proc. Conference on Neural Information Processing Systems (NeurIPS)*.

Saito, S.; Huang, Z.; Natsume, R.; Morishima, S.; Kanazawa, A.; and Li, H. 2019. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In

Proc. International Conference on Computer Vision (ICCV), 2304–2314.

Schwarz, K.; Liao, Y.; Niemeyer, M.; and Geiger, A. 2020. Graf: Generative radiance fields for 3d-aware image synthesis. *Advances in Neural Information Processing Systems*, 33: 20154–20166.

Shao, R.; Zhang, H.; Zhang, H.; Chen, M.; Cao, Y.-P.; Yu, T.; and Liu, Y. 2022. Doublefield: Bridging the neural surface and radiance fields for high-fidelity human reconstruction and rendering. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 15872–15882.

Sharmin, S.; Panda, P.; Sarwar, S. S.; Lee, C.; Ponghiran, W.; and Roy, K. 2019. A comprehensive analysis on adversarial robustness of spiking neural networks. In *2019 International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

Sharmin, S.; Rathi, N.; Panda, P.; and Roy, K. 2020. Inherent adversarial robustness of deep spiking neural networks: Effects of discrete input encoding and non-linear activations. In *Proc. European Conference on Computer Vision (ECCV)*, 399–414. Springer.

Shrestha, S. B.; and Orchard, G. 2018. Slayer: Spike layer error reassignment in time. *Advances in neural information processing systems*, 31.

Sitzmann, V.; Zollhöfer, M.; and Wetzstein, G. 2019. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *Advances in Neural Information Processing Systems*, 32.

Verbin, D.; Hedman, P.; Mildenhall, B.; Zickler, T.; Barron, J. T.; and Srinivasan, P. P. 2022. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 5481–5490. IEEE.

Wang, P.; Liu, L.; Liu, Y.; Theobalt, C.; Komura, T.; and Wang, W. 2021. NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction. In *Proc. Conference on Neural Information Processing Systems (NeurIPS)*, 27171–27183. Curran Assoicates, Inc.

Wang, Y.; Skorokhodov, I.; and Wonka, P. 2022. Hf-neus: Improved surface reconstruction using high-frequency details. *Advances in Neural Information Processing Systems*, 35: 1966–1978.

Wang, Y.; Skorokhodov, I.; and Wonka, P. 2023. PET-NeuS: Positional Encoding Tri-Planes for Neural Surfaces. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 12598–12607.

Xu, Q.; Wang, W.; Ceylan, D.; Mech, R.; and Neumann, U. 2019. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. *Advances in neural information processing systems*, 32.

Yang, Z.; Wu, Y.; Wang, G.; Yang, Y.; Li, G.; Deng, L.; Zhu, J.; and Shi, L. 2019. Dashnet: A hybrid artificial and spiking neural network for high-speed object tracking. *arXiv preprint arXiv:1909.12942*.

Yariv, L.; Gu, J.; Kasten, Y.; and Lipman, Y. 2021. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems*, 34: 4805–4815.

Yariv, L.; Kasten, Y.; Moran, D.; Galun, M.; Atzmon, M.; Ronen, B.; and Lipman, Y. 2020. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems*, 33: 2492–2502.

Zhang, J.; Dong, B.; Zhang, H.; Ding, J.; Heide, F.; Yin, B.; and Yang, X. 2022. Spiking transformers for event-based single object tracking. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 8801–8810.

Zhang, K.; Riegler, G.; Snavely, N.; and Koltun, V. 2020. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*.

Zhang, X.; Srinivasan, P. P.; Deng, B.; Debevec, P.; Freeman, W. T.; and Barron, J. T. 2021. Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics (TOG)*, 40(6): 1–18.

Zhao, R.; Yang, Z.; Zheng, H.; Wu, Y.; Liu, F.; Wu, Z.; Li, L.; Chen, F.; Song, S.; Zhu, J.; et al. 2022. A framework for the general design and computation of hybrid neural networks. *Nature communications*, 13(1): 3427.

Zhu, L.; Wang, X.; Chang, Y.; Li, J.; Huang, T.; and Tian, Y. 2022. Event-based video reconstruction via potential-assisted spiking neural network. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 3594–3604.

Zou, X.-L.; Huang, T.-J.; and Wu, S. 2022. Towards a new paradigm for brain-inspired computer vision. *Machine Intelligence Research*, 19(5): 412–424.