Sobolev Training for Implicit Neural Representations with Approximated Image Derivatives

Wentao Yuan[1,2] Qingtian Zhu[1] Xiangyue Liu[1] Yikang Ding[1]
Haotian Zhang[1⋆] Chi Zhang[1]

[1]Megvii Research [2]Peking University

Abstract. Recently, Implicit Neural Representations (INRs) parameterized by neural networks have emerged as a powerful and promising tool to represent different kinds of signals due to its continuous, differentiable properties, showing superiorities to classical discretized representations. However, the training of neural networks for INRs only utilizes input-output pairs, and the derivatives of the target output with respect to the input, which can be accessed in some cases, are usually ignored. In this paper, we propose a training paradigm for INRs whose target output is image pixels, to encode image derivatives in addition to image values in the neural network. Specifically, we use finite differences to approximate image derivatives. We show how the training paradigm can be leveraged to solve typical INRs problems, i.e., image regression and inverse rendering, and demonstrate this training paradigm can improve the data-efficiency and generalization capabilities of INRs. The code of our method is available at https://github.com/megvii-research/Sobolev_INRs.

Keywords: implicit neural representations, finite differences, Sobolev training

1 Introduction

A promising recent direction in computer vision and computer graphics is encoding complex signals implicitly by aid of multi-layer perceptron (MLP) named Implicit Neural Representations (INRs) [23,16,28,20,19,26,27,30]. The MLPs (more specifically, coordinate-based MLPs) take low-dimensional coordinates as inputs and are trained to output a representation of shape, density, and/or colors at each input location.

Compared to classical alternatives, i.e., discrete grid-based representation, INRs offer two main benefits. (a) Due to the fact that MLPs are continuous functions, they are significantly more memory efficient than discrete grid-based representations and theoretically, signals parameterized by MLPs can be presented in arbitrary resolution. (b) INRs are naturally differentiable, so gradient

This work is done by the first four authors as interns at Megvii Research.

⋆ Corresponding author (zhanghaotian@megvii.com).

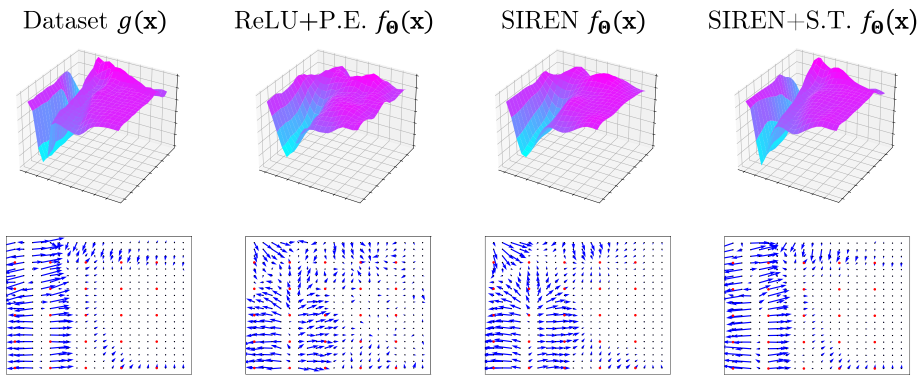


Fig. 1: Visualized approximation results of different training schemes and different activation functions. P.E. stands for positional encoding [19,31]; S.T. stands for Sobolev training. Top: dataset g (the target function to approximate) as well as approximated functions f_{Θ} (functions parameterized by neural networks) represented by MLPs within an image patch of 20×20 pixels. Bottom: vector fields of the first-order derivatives, where red points are training samples (in a total proportion of 6.25%) and remaining black points are unseen samples. When activated by sine functions and trained with additional supervision on derivatives, the network demonstrates the best approximation of both function values and derivatives.

descent (GD) can be applied to optimize MLPs using modern auto-differentiation tools, e.g., PyTorch [25], TensorFlow [1], JAX [5].

However, as will be discussed below, the training paradigm adopted by most of the INRs only utilizes input-output pairs, which brings about bad generalization capabilities on unseen coordinates. In other words, if trained with low-resolution training data, MLPs will yield heavily degraded predictions when the target outputs are high-resolution. Based on this observation, an intuitive idea is that we can take advantage of the differentiability of INRs to improve the generalization capabilities. In this paper, we propose a training paradigm for INRs whose training data and the desired results are both images, with both supervision on values and derivatives enforced. Considering that the ground truth representation (the "continuous" image) is agnostic, finite differences are applied to approximate numerical first-order derivatives on images.

Further, we study some important peripheral problems relevant to the proposed training paradigm, e.g., the choice of activation functions, the number of layers in MLPs and the choice of image filters.

We find that in practice, most of the recent INRs build on ReLU-based MLPs [19,31,8] fail to represent well the derivatives of target outputs, even if the corresponding supervision is provided. For this reason, we use periodic activation functions [22,26], i.e., sine function, to improve MLPs' convergence property of derivatives. As is shown in Fig. 1, INRs trained with traditional value-based schemes fail to explicitly constraint the behavior at unseen coordinates, leading to poor generalization both in the image field and the vector field of the first-order derivatives. Besides, with sine-activated MLPs trained with both value loss and derivative loss, INRs are able to generalize well even if the training samples only take a very limited proportion (6.25% in Fig. 1).

We show how the training paradigm can be adapted to specific problems, namely direct image regression [26,31] and indirect inverse rendering [19,31]. Experiments results demonstrate that our training paradigm can improve the data-efficiency and generalization capabilities of INRs. The main contributions are summarized below.

- We propose a training paradigm for INRs whose training data as well as target outputs are image pixels. With the paradigm, image values and image derivatives are encoded into the INRs. Concretely, we use finite differences to approximate the derivatives of the ground truth signal. To the best of our knowledge, supervising network derivatives in INRs is minimally explored.
- We study the factor of activation functions in INRs when applying the proposed training paradigm. By experiments, We use sine functions for better convergence property of derivatives.
- Experiment results on two typical INRs problems demonstrate that with the proposed paradigm we are able to train better INRs - representations that require fewer training samples and generalise better on unseen inputs.

2 Related Work

2.1 Implicit Neural Representations

Implicit Neural Representations (INRs), which usually represent an object as a multi-layer perceptron model that maps low-dimensional coordinates to signal values, have drawn a lot of attention recently. Since this representation is continuous and can capture fine details of signals, it has been widely applied to novel view synthesis [19,15,14,24,36,35,32], shape representation [9,23,17,6,3,11,38,34], and multi-view 3D reconstruction [16,21,7,39]. As a milestone, Mildenhall et al. [19] propose a novel view synthesis method that learns an implicit representation for a specific 3D scene by using a set of multi-view calibrated images. Recent work from Park et al. [23] puts forward to learn a Signed Distance Function (SDF) to represent the shape of objects, and different SDFs can be inferred with different input latent codes. However, the (approximated) ground truth derivative information of the target signal has rarely been utilized to train INRs.

2.2 Derivative Supervision

Derivative information for neural networks has also been exploited in some previous work. Hornik [12] proves the universal approximation theorems for neural networks in Sobolev spaces - a vector space of functions equipped with a

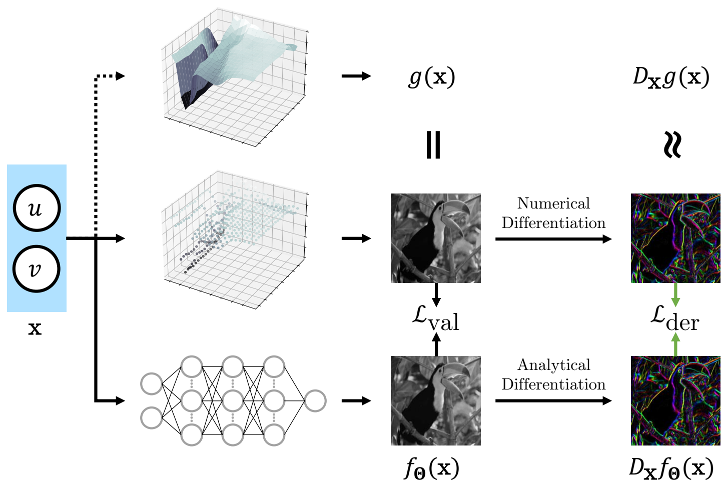


Fig. 2: An overview of the proposed training paradigm for INRs. Top: the underlying continuous signal g and its derivatives $D_{\mathbf{x}}g$. Middle: the pixels of image $\mathbf{I}$, which can be considered as discrete but exact sampling of g and its derivatives are obtained by numerical approximation at image space. Bottom: the MLP denoted as f_{Θ} used as a general approximator whose derivatives are analytical thanks to auto-differentiation tools. In the proposed training paradigm, we enforce the supervision on both values and derivatives.

norm that is a combination of ℓ_p-norms of the function together with its derivatives up to a given order. Hornik [12] shows that neural networks with non-constant, bounded, continuous activation functions, with continuous derivatives up to order K are universal approximators in the Sobolev spaces of order K. Further, Czarnecki et al. [10] prove ReLU-based MLPs are universal approximators for function values and first-order derivatives theoretically and propose Sobolev training for neural networks. However, only the scenarios with analytical derivatives are covered and the ReLU-based MLPs show relatively poor convergence properties in practice. Gropp et al. [11] propose an *Eikonal term* which is a first-order derivative related penalty term to regularize INRs. The Eikonal term is different from our explicit derivative supervision with ground truth derivatives approximation. Sitzmann et al. [26] verify that INRs with periodic activation functions can fit derivatives robustly, while the supervision on both values and derivatives and the property of Sobolev training in INRs remain unexplored.

3 Methodology

3.1 Formulation

Considering a continuous target signal $g : \mathbb{R}^D \to \mathbb{R}^C$, which is sampled discretely at $\{\mathbf{x}_i\}_{i=1}^N$, the goal of INRs is to approximate g with a neural network (typically a MLP) $f_{\boldsymbol{\Theta}} : \mathbb{R}^D \rightarrow \mathbb{R}^C$, parameterized by weights $\boldsymbol{\Theta}$. The inputs to the MLP are typically low dimensional coordinates $\mathbf{x} \in \mathbb{R}^D$, e.g., $D = 2$ for image coordinates, and the corresponding outputs are signal values, e.g., RGB colors. Training pairs of most recent INRs are denoted as $\{(\mathbf{x}_i, g(\mathbf{x}_i))\}_{i=1}^N$, where N is the number of total training samples.

By measuring the distance between predictions of the MLP $\{f_{\boldsymbol{\Theta}}(\mathbf{x}_i)\}_{i=1}^N$ and ground truth signal values $\{g(\mathbf{x}_i)\}_{i=1}^N$ with certain metrics, we actually obtain an optimization objective for tuning $\boldsymbol{\Theta}$. Benefiting from the natural differentiability of MLPs, we are able to minimize the error using gradient descent (GD) implemented by auto differentiation frameworks.

MLPs are known as a powerful approximator of functions and with their continuous property, we are able to interpolate the discrete signal up to an arbitrarily higher resolution. However, given the traditional training scheme, the resulting ability to interpolate at unseen coordinates is usually not satisfactory. In this paper, we alleviate this problem by leveraging the supervision on derivatives, which is to say, we optimize $\boldsymbol{\Theta}$ by supervising not only signal values, but also derivatives at given coordinates.

The overall training paradigm is illustrated in Fig. 2. We therefore construct a training dataset $\{(\mathbf{x}_i, g(\mathbf{x}_i), D_{\mathbf{x}} g(\mathbf{x}_i))\}_{i=1}^N$ for INRs. Since the analytical expression of g is unknown, its derivatives require to be approximated with finite differences, which will be elaborated in Sec. 3.2. Considering that the task is a standard regression task, we apply ℓ-2 error as the loss function. With the training set, the optimization objective evolves to a combination of both supervision on signal values and first-order derivatives:

$$\begin{aligned}\boldsymbol{\Theta} &= \arg\min_{\boldsymbol{\Theta}} \frac{1}{N} \sum_{i=1}^{N} [\mathcal{L}_{\text{val}}(g(\mathbf{x}_i), f_{\boldsymbol{\Theta}}(\mathbf{x}_i)) + \lambda \mathcal{L}_{\text{der}}(D_{\mathbf{x}} g(\mathbf{x}_i), D_{\mathbf{x}} f_{\boldsymbol{\Theta}}(\mathbf{x}_i))] \\ &= \arg\min_{\boldsymbol{\Theta}} \frac{1}{N} \sum_{i=1}^{N} (\| g(\mathbf{x}_i) - f_{\boldsymbol{\Theta}}(\mathbf{x}_i) \|_2^2 + \lambda \| D_{\mathbf{x}} g(\mathbf{x}_i) - D_{\mathbf{x}} f_{\boldsymbol{\Theta}}(\mathbf{x}_i) \|_2^2).\end{aligned} \tag{1}$$

We additionally extend this paradigm to enable both pre-processing of $\mathbf{x}$ and post-processing of $f_{\boldsymbol{\Theta}}(\mathbf{x})$, for instances that INRs are used to be an intermediate representation where the inputs to the MLP are calculated from $\mathbf{x}$ and the outputs of the MLP require further processing to obtain the final results. To formulate, we parameterize the MLP as $f_{\boldsymbol{\Theta}}$, and denote the pre-processing and post-processing as p and q respectively. The target signal g is therefore approximated as $q(f_{\boldsymbol{\Theta}}(p(\mathbf{x})))$. Assuming that both p and q are differentiable and non-parametric, we can integrate these two processes into $f_{\boldsymbol{\Theta}}$ and follow the aforementioned training paradigm. We will carry out experiments on tasks satisfying this configuration in Sec. 4.2. Specially, if p and q are identical mappings, the case degenerates to the original form.

Though can be applied under more circumstances, in this paper, we set the scope of target signals to $g : \mathbb{R}^2 \rightarrow \mathbb{R}^3$, that $\{\mathbf{x}_i\}_{i=1}^N$ are image coordinates and $\{g(\mathbf{x}_i)\}_{i=1}^N$ are pixel values (colors). The loss supervision on derivatives $\mathcal{L}_{\text{der}}$ in Eq. (1) is composed of partial derivatives w.r.t. u and v respectively.

3.2 Approximate Derivatives with Finite Differences

To enable Sobolev training on image coordinates, the first-order derivative of the target signal g is supposed to be known as a precondition. However, the only information accessible at this stage is discrete sampled values $\{g(\mathbf{x}_i)\}_{i=1}^N$ so we need to approximate derivatives from these values with finite differences, or namely numerical differentiation (Fig. 2). Since we only consider cases that $\mathbf{x} \in \mathbb{R}^2$, the partial derivatives are

$$\begin{aligned} D_u g(u,v) =& \frac{g(u+h,v) - g(u-h,v)}{2h}, \\ D_v g(u,v) =& \frac{g(u,v+h) - g(u,v-h)}{2h}. \end{aligned} \tag{2}$$

In this paper, we mainly apply the Sobel operator to obtain first-order derivatives of images. It defines a template that convolves the image $\mathbf{I}$ to obtain the partial derivative w.r.t. u while the transposed template for the partial derivative w.r.t. v:

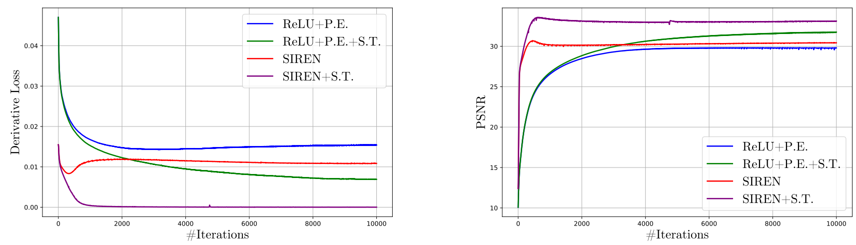
$$D_u = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * \mathbf{I}, D_v = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * \mathbf{I}, \tag{3}$$

where $*$ denotes the 2D convolution operation. We also study the choice of image filters for derivatives in Sec. 4.3, the conclusion of which is that enforcing the supervision of derivatives matters, regardless of the specific choice of filters.

3.3 Activation Functions

Activation functions are usually non-linear functions applied after each fully-connected layer. ReLU (Rectified Linear Units) [2] is a universally applied choice for its simple design, strong biological motivations and mathematical justifications. Due to the fact that the ReLU function are piecewise linear and its second derivative is zero everywhere, ReLU can not fitting derivatives well in practice even the positional encoding is applied, just as shown in Fig. 3a. On the other hand, [22,26] attempt to explore the potential of the sine function as activation functions due to its periodicity, boundedness, arbitrary-order differentiability. The superior convergence property of sine has been proved in [26] when supervising the derivative only. In our setting, we try to replace ReLU with sine for better convergence property in Sobolev space. As can be indicated from Fig. 3a, sine indeed converges to smaller derivative difference (close to 0) than ReLU when the training in Sobolev space is enabled.

However, studying the properties of activation functions in-depth is a complicated problem and in this paper, we mainly study the training paradigm applied to MLPs with activation functions of (a) ReLU, whose first-order derivative is piece-wise smooth, and (b) sine, whose arbitrary-order derivative is itself (with phase shift). We also study other activation functions in the Supplementary Material.



(a) (b)

Fig. 3: Convergence situations of $\mathcal{L}_{\text{der}}$ (a) and PSNR (b) w.r.t. iterations.

4 Experiments

In this section, we study the effectiveness of the proposed training paradigm on specific tasks that satisfy the preconditions. We sort tasks involving INRs into two categories, namely direct and indirect ones. Direct tasks are those where the supervision labels are in the same space as the network outputs, where p and q defined in Sec. 3.1 are both identical mappings. While in indirect tasks, the network outputs are processed afterwards so that the observations in the same space as the supervision labels are produced. We give an example of direct and indirect tasks respectively in Sec. 4.1 and Sec. 4.2.

4.1 Direct: Image Regression

A typical direct task is image regression. Given a RGB image $\mathbf{I}$, a MLP is appointed to regress the mapping from image coordinates to RGB colors $g: \mathbb{R}^2 \to \mathbb{R}^3$. This task is fundamental regarding demonstrating the representation ability of INRs.

Implementation

Dataset We construct a dataset based on the Set 5 dataset [4] to demonstrate the great interpolatability of MLPs when trained with additional supervision on derivatives. We split the pixels within an image into two groups, namely training samples and evaluation samples, by performing nearest-neighbor downscaling by a factor of 4. This is to say, for an image patch of 4×4, we have 1 training sample and the remaining 15 pixels as evaluation samples.

Network Architecture For network architecture, we implement a fully-connected MLP with 4 activated linear layers with 256 hidden units each and 1 output linear layer. For the ReLU-based implementation, we follow the conclusion in [37,31] and apply positional encoding (P.E.) to the inputs (additional 20 channels).

Table 1: Quantitative results on the task of image regression on Set 5 dataset [4]. When trained with additional supervision on derivatives and activated by sine and ReLU functions, the network achieves the best and second-best performance respectively in terms of both PSNR and SSIM. We also provided interpolated results by bilinear and bicubic interpolation. best second-best

Method	**Mean**	*Baby*	*Bird*	*Butterfly*	*Head*	*Woman*
PSNR ↑						
Bilinear	24.14	26.98	24.88	18.68	28.04	22.10
Bicubic	23.63	26.52	24.46	18.23	27.43	21.52
ReLU+P.E. [19]	26.59	29.80	28.18	20.54	29.47	24.95
ReLU+P.E.+S.T.	28.63	31.93	31.22	21.93	31.11	26.97
SIREN [26]	26.22	30.48	28.07	20.73	28.22	23.62
SIREN+S.T.	30.28	33.36	33.34	24.42	31.64	28.62
SSIM ↑						
Bilinear	0.716	0.778	0.754	0.636	0.674	0.736
Bicubic	0.705	0.773	0.746	0.625	0.655	0.727
ReLU+P.E. [19]	0.740	0.792	0.808	0.636	0.690	0.775
ReLU+P.E.+S.T.	0.809	0.867	0.888	0.708	0.757	0.827
SIREN [26]	0.722	0.849	0.804	0.698	0.635	0.625
SIREN+S.T.	0.890	0.918	0.955	0.870	0.795	0.910

Training & Evaluation We train the network with all training samples with Adam optimizer for 50k iterations, at a learning rate of 10^{-4}. Following the common practice in image super resolution [13,29,33], we measure PSNR and SSIM on the Y channel and ignore 4 pixels from the image border. For PSNR, we only take the evaluation samples into account.

Results The quantitative scores are shown in Tab. 1, where we compare the proposed training paradigm with the previous value-based training paradigm, and with different activation functions. When trained with $\mathcal{L}_{\text{val}}$ alone, ReLU and sine (SIREN [26]) functions lead to comparable results; while with additional supervision on image derivatives, a huge performance improvement has been gained with both activations in both metrics. It is worth noting that in all experiments, INRs demonstrate greater interpolatability than rule-based interpolation methods, i.e., bilinear and bicubic interpolation. Fig. 4 gives some example regions where our method produces clearer images and sharper edges and also approximates better the derivatives. It can be observed that sine-based MLPs outperform ReLU-based ones when Sobolev training is enabled. Even so, Sobolev training still improves the results of ReLU-based INRs by a large margin. As additional evidences, Fig. 3b shows the growth of PSNR w.r.t. the number of iterations, where the combination of Sobolev training and sine leads to the fastest convergence and the best performance.

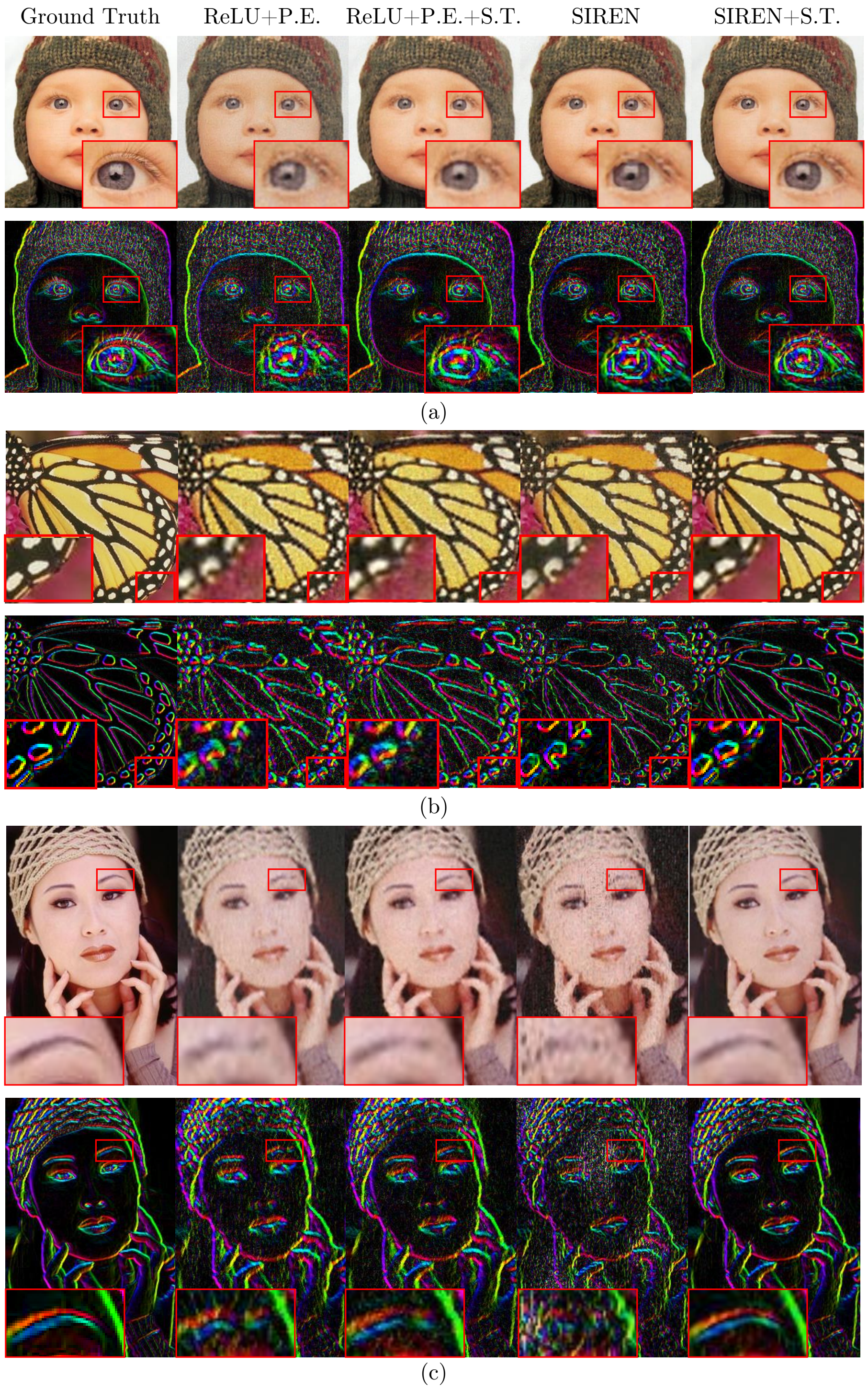


Fig. 4: Results of regressed images and derivatives of the set *Baby* (a), *Butterfly* (b) and *Woman* (c). When sine functions are adopted for activating linear layers and Sobolev training is enabled, the MLP yields the best visual effects, especially at high-frequency sharp edges, while ReLU-based MLPs fail to regress these details.

Table 2: Quantitative results on the task of inverse rendering on LLFF dataset [18]. When trained with additional supervision on derivatives and activated by sine functions, the network achieves the best performance in terms of both PSNR and SSIM. best second-best

Method	**Mean**	*Fern*	*Flower*	*Fortress*	*Horns*	*Leaves*	*Orchids*	*Room*	*T-Rex*
				PSNR ↑					
ReLU+P.E. [19]	23.42	21.82	25.03	27.89	24.75	18.14	18.92	27.22	23.62
ReLU+P.E.+S.T.	23.74	22.99	25.27	27.90	24.50	19.30	19.40	27.10	23.49
SIREN[26]+P.E.	23.38	21.61	25.30	27.60	24.73	18.22	18.99	26.89	23.69
SIREN+P.E.+S.T.	24.38	23.63	25.99	28.30	25.24	20.02	19.74	27.76	24.40
				SSIM ↑					
ReLU+P.E. [19]	0.706	0.651	0.755	0.769	0.729	0.535	0.544	0.877	0.791
ReLU+P.E.+S.T.	0.708	0.680	0.757	0.762	0.711	0.559	0.554	0.878	0.768
SIREN [26]+P.E.	0.682	0.596	0.739	0.763	0.720	0.495	0.525	0.851	0.768
SIREN+P.E.+S.T.	0.751	0.721	0.800	0.811	0.756	0.629	0.592	0.893	0.807

4.2 Indirect: Inverse Rendering

INRs are recently attention-drawing for their great power demonstrated when representing 3D scenes. NeRF [19], which is a typical example of the kind, leverages a MLP $f_{\boldsymbol{\Theta}} : \mathbb{R}^5 \rightarrow \mathbb{R}^4$, to regress the mapping from (x, y, z, θ, ϕ) to (r, g, b, σ), where x, y, z are 3D coordinates and θ, ϕ are the view directions while the outputs are emitted colors $\mathbf{c} = (r, g, b)$ and volume density σ. In the pipeline of NeRF, the 3D sampling points (x, y, z), as well as view directions (θ, ϕ), of the MLP are determined from 2D image coordinates and corresponding calibrated camera parameters. After obtaining (r, g, b, σ) for each sampling point, a numerical approximation of volume rendering is applied to obtain the integral colors. Considering that both pre-processing and post-processing are differentiable, we can adapt the extended formulation of proposed training paradigm mentioned in Sec. 3.1 to perform inverse rendering of scenes.

Implementation

Dataset We carry out experiments on LLFF dataset [18]. Similarly, we down-sample the training images to 189×252 while the resolution of target images to render is 756×1008.

Network Architecture We implement a simplified version of NeRF [19] $(x, y, z) \rightarrow (r, g, b, \sigma)$, removing the skip connection, the strategy of hierarchical sampling and view dependence. For network architecture, we implement a fully-connected MLP with 8 activated linear layers with 256 hidden units each and 1 output layer. We apply positional encoding (additional 60 channels) to both sine-activated and ReLU-activated MLPs.

Table 3: Results for comparisons on the task of inverse rendering on LLFF dataset [18] when the training resolution is larger. It is worth noting that even only trained with 1/16 samples, the MLP trained with the proposed paradigm is able to outperform the ReLU-activated network on both metrics in the set of *Flower*.

$H \times W$	**Method**	**Mean**	*Fern*	*Flower*	*Fortress*	*Horns*	*Leaves*	*Orchids*	*Room*	*T-Rex*
PSNR ↑										
756×1008	ReLU+P.E.	**24.69**	24.26	25.92	**28.58**	25.32	20.39	**19.99**	**28.54**	24.52
378×504	ReLU+P.E.	24.55	23.93	25.90	28.46	25.22	20.20	19.97	28.30	24.42
378×504	SIREN+P.E.+S.T.	24.67	**24.27**	**26.16**	28.37	**25.37**	**20.64**	19.94	27.99	**24.61**
189×252	SIREN+P.E.+S.T.	24.38	23.63	25.99	28.30	25.24	20.02	19.74	27.76	24.40
SSIM ↑										
756×1008	ReLU+P.E.	0.755	0.744	0.788	0.792	0.746	0.646	0.606	**0.900**	**0.816**
378×504	ReLU+P.E.	0.750	0.734	0.785	0.790	0.744	0.637	0.603	0.897	0.813
378×504	SIREN+P.E.+S.T.	**0.766**	**0.753**	**0.805**	**0.815**	**0.762**	**0.666**	**0.614**	0.899	0.815
189×252	SIREN+P.E.+S.T.	0.751	0.721	0.800	0.811	0.756	0.629	0.596	0.893	0.807

Training & Evaluation At the stage of training, we train the network with a batch size of 128 with Adam optimizer for 400k iterations, at a learning rate of 5×10^{-4}. For evaluation metrics, we report PSNR and SSIM on all channels (RGB) of the rendered images.

Results Tab. 2 demonstrates the PSNR and SSIM on all scenes of LLFF dataset [18] and Fig. 5 provides some results of novel view synthesis. The conclusions drawn are similar to the task of image regression.

Data-efficiency To provide more insights of the data-efficiency brought about by the proposed training paradigm, we also conduct experiments with different image resolution, namely different amount of training samples under different training paradigms. For the valued-based paradigm, we train MLPs with images of resolution 378×504 and 756×1008 respectively and render novel views of 756×1008. For the proposed paradigm, we additionally train MLPs with images of 378×504 and render novel views of 756×1008.

The quantitative comparisons demonstrating the data-efficiency of Sobolev training are shown in Tab. 3. The following two points can be observed and indicated. (a) Even with only 1/16 training samples, the Sobolev trained MLP sometimes exceeds the ReLU-activated MLP with the value-based training paradigm, e.g., on *Flower* in terms of PSNR and SSIM, on *Fortress* and *Horns* in terms of SSIM. (b) The Sobolev trained MLP using images of 378×504 gets similar PSNR and even higher SSIM compared to the ReLU-activated MLP trained with the value-based paradigm using images of 756×1008. Note that the former case is trained only with 1/4 samples as the latter.

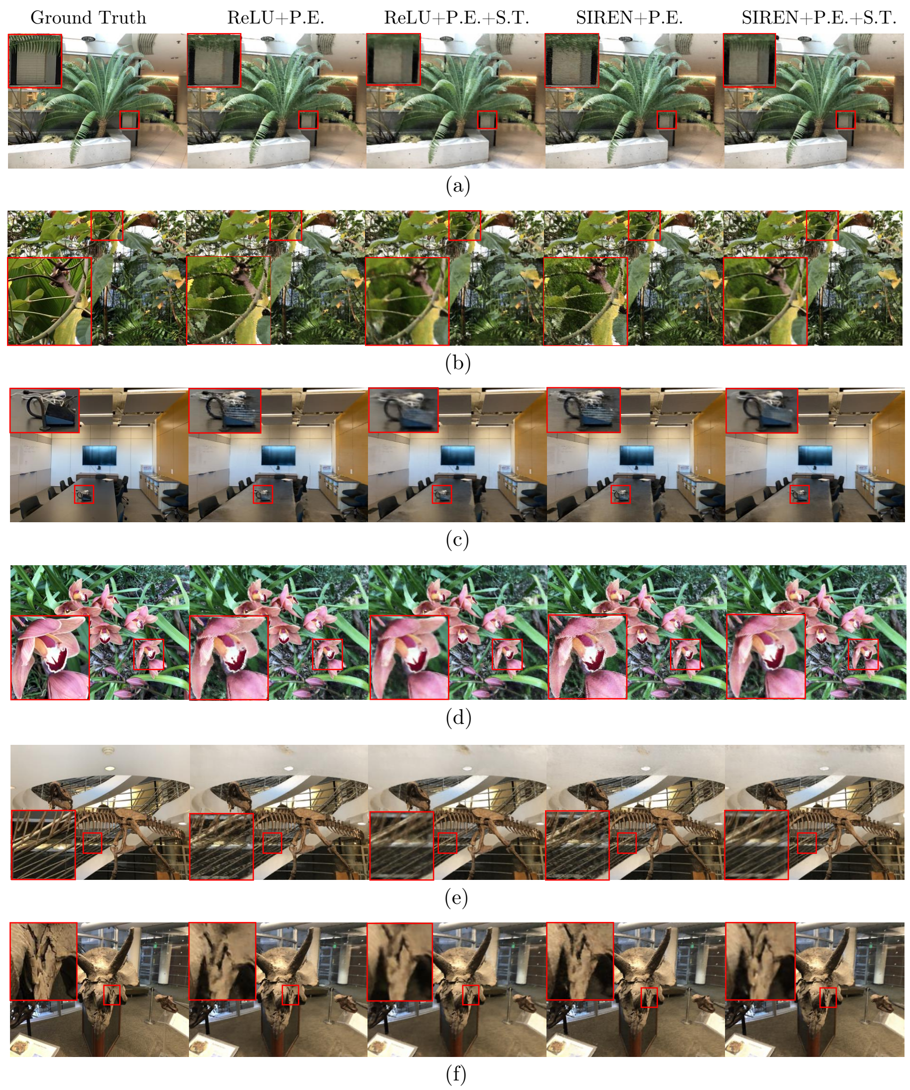


Fig. 5: Results of novel view synthesis by INRs in the task of inverse rendering. Top to bottom: *Fern* (a), *Leaves* (b), *Room* (c), *Orchids* (d), *T-Rex* (e), *Horns* (f). Supervision on derivatives, applied with sine-activated MLPs, helps to render clear images and sharp edges at unseen views.

4.3 Ablation Study

Image Derivative Filters We compare the network performance with another image filter as finite differences approximating derivatives, namely vanilla image

Table 4: Quantitative comparison on the choice of image filters when approximating derivatives of g. Experiments are carried out with MLPs activated by sine functions. PSNR and SSIM stand for mean values over all scenes.

Task	Filter	PSNR	SSIM
Image Regression	Vanilla	30.24	0.886
	Sobel	**30.28**	**0.890**
Inverse Rendering	Vanilla	24.28	0.746
	Sobel	**24.38**	**0.751**

Table 5: Quantitative comparison on the number of layers in MLPs under the task of inverse rendering. PSNR and SSIM stand for mean values over all scenes.

#layers	**Method**	PSNR	SSIM
4	ReLU+P.E.	23.29	0.685
	SIREN+P.E.	23.17	0.654
	ReLU+P.E.+S.T.	23.35	0.689
	SIREN+P.E.+S.T.	23.66	0.697
8	ReLU+P.E.	23.42	0.706
	SIREN+P.E.	23.38	0.682
	ReLU+P.E.+S.T.	23.74	0.708
	SIREN+P.E.+S.T.	**24.38**	**0.751**

derivative filter whose templates are

$$T_u = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, T_v = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}. \tag{4}$$

It considers the same neighborhood as Sobel operator but removes the spatial smoothing operation included in Sobel operator. The quantitative comparisons are shown in Tab. 4.

Though the results with Sobel operator are slightly better than the vanilla image derivative filter, the performance gap is relatively small, indicating the specific choice of filters is not crucial to the performance.

Network Capacity A simple intuition towards the fitting ability, including both values and derivatives, of a neural network is that the more parameters a network possesses, the more complex functions it can represent. We present the quantitative results obtained with networks with different number of layers (4 and 8 respectively) in the task of inverse rendering in Tab. 5. Other experiment settings are aligned with Sec. 4.2. It can be indicated that when the layer number increases to 8 from 4, the representation ability of INRs is stronger so it can obtain more performance gain from derivatives.

5 Discussions

5.1 Future Work

To restate, the preconditions of the proposed training paradigm are that (a) the raw dataset is in the form of $\{(\mathbf{x}_i, g(\mathbf{x}_i))\}_{i=1}^N$, where $\{\mathbf{x}_i\}_{i=1}^N$ are spatial coordinates; (b) the derivatives of g can be obtained or approximated with limited $\{g(\mathbf{x}_i)\}$; (c) $\{\mathbf{x}_i\}$ do not necessarily be the direct inputs of the MLP, and the target values $\{g(\mathbf{x}_i)\}$ do not necessarily be the direct outputs if and only if the pre-processing p is differentiable w.r.t. to $\mathbf{x}$ and the post-processing q is differentiable w.r.t. $f_{\boldsymbol{\Theta}}(p(\mathbf{x}))$.

Theoretically, referring to the formulation in Sec. 3.1, the training paradigm can further generalize to more tasks, e.g., audio regression (please refer to the Supplementary Material) and video regression, as long as their partial derivatives can be obtained by numerical approximation. Take the task of video regression $(u, v, t) \rightarrow (r, g, b)$ as an example, whose partial derivatives w.r.t. u and v are the same on images. We can approximate the partial derivative w.r.t. t by subtracting consecutive frames.

5.2 Limitations

We summarize the limitations of the proposed training paradigm as below.

- The computation of analytical derivatives is computationally expensive and occupies a considerable amount of memory. For example, both PyTorch [25] and TensorFlow [1] cannot calculate derivatives for each sample in a batch separately, leading to redundant computation.
- Though the proposed training paradigm can be applied to various INRs-based tasks, the scope of application is still limited. Since the derivatives are approximated with finite differences, the distribution of $\{\mathbf{x}_i\}$ is supposed to be as uniform as possible at its domain, which is not always satisfied.

6 Conclusion

In this paper, we put forward a novel training paradigm for INRs in Sobolev space, where supervision on both signal values and first-order derivatives is enforced for network training. The formulation of the task is generalized to any circumstance that the inputs are 2D image coordinates and the outputs are image values (e.g., RGB) as long as the pre-processing is differentiable w.r.t. the coordinates and the post-processing is differentiable w.r.t. the MLP's outputs. We obtain approximated image derivatives by the Sobel operator.

Experiments are carried out on the task of image regression (direct) and inverse rendering (indirect) and results show that the training paradigm brings remarkable improvement to the quality of reconstructed images, especially when applied together with a MLP whose activation functions are sine.

In addition, we study some important and interesting peripheral problems relevant to the proposed training paradigm and give insights based on our observations.

Acknowledgements The authors would like to thank Zhongtian Zheng from Peking University for his valuable advice.

References

1. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al.: {TensorFlow}: a system for {Large-Scale} machine learning. In: 12th USENIX symposium on operating systems design and implementation (OSDI 16). pp. 265–283 (2016) 2, 14
2. Agarap, A.F.: Deep learning using rectified linear units (relu). arXiv preprint arXiv:1803.08375 (2018) 6
3. Atzmon, M., Lipman, Y.: Sal: Sign agnostic learning of shapes from raw data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2565–2574 (2020) 3
4. Bevilacqua, M., Roumy, A., Guillemot, C., Alberi-Morel, M.L.: Low-complexity single-image super-resolution based on nonnegative neighbor embedding (2012) 7, 8
5. Bradbury, J., Frostig, R., Hawkins, P., Johnson, M.J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., Zhang, Q.: JAX: composable transformations of Python+NumPy programs (2018), http://github.com/google/jax 2
6. Chabra, R., Lenssen, J.E., Ilg, E., Schmidt, T., Straub, J., Lovegrove, S., Newcombe, R.: Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In: European Conference on Computer Vision. pp. 608–625. Springer (2020) 3
7. Chen, A., Xu, Z., Zhao, F., Zhang, X., Xiang, F., Yu, J., Su, H.: Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 14124–14133 (2021) 3
8. Chen, Y., Liu, S., Wang, X.: Learning continuous image representation with local implicit image function. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8628–8638 (2021) 2
9. Chen, Z., Zhang, H.: Learning implicit fields for generative shape modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5939–5948 (2019) 3
10. Czarnecki, W.M., Osindero, S., Jaderberg, M., Swirszcz, G., Pascanu, R.: Sobolev training for neural networks. Advances in Neural Information Processing Systems **30** (2017) 4
11. Gropp, A., Yariv, L., Haim, N., Atzmon, M., Lipman, Y.: Implicit geometric regularization for learning shapes. In: International Conference on Machine Learning. pp. 3789–3799. PMLR (2020) 3, 4
12. Hornik, K.: Approximation capabilities of multilayer feedforward networks. Neural networks **4**(2), 251–257 (1991) 3, 4
13. Jo, Y., Kim, S.J.: Practical single-image super-resolution using look-up table. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 691–700 (2021) 8
14. Lin, C.H., Ma, W.C., Torralba, A., Lucey, S.: Barf: Bundle-adjusting neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5741–5751 (2021) 3
15. Martin-Brualla, R., Radwan, N., Sajjadi, M.S., Barron, J.T., Dosovitskiy, A., Duckworth, D.: Nerf in the wild: Neural radiance fields for unconstrained photo collections. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7206–7215. IEEE Computer Society (2021) 3

16. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4460–4470 (2019) 1, 3
17. Michalkiewicz, M., Pontes, J.K., Jack, D., Baktashmotlagh, M., Eriksson, A.: Implicit surface representations as layers in neural networks. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 4742–4751. IEEE Computer Society (2019) 3
18. Mildenhall, B., Srinivasan, P.P., Ortiz-Cayon, R., Kalantari, N.K., Ramamoorthi, R., Ng, R., Kar, A.: Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. ACM Transactions on Graphics (TOG) **38**(4), 1–14 (2019) 10, 11
19. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: European conference on computer vision. pp. 405–421. Springer (2020) 1, 2, 3, 8, 10
20. Niemeyer, M., Mescheder, L., Oechsle, M., Geiger, A.: Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3504–3515 (2020) 1
21. Oechsle, M., Peng, S., Geiger, A.: Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 5589–5599 (2021) 3
22. Parascandolo, G., Huttunen, H., Virtanen, T.: Taming the waves: sine as activation function in deep neural networks (2017) 2, 6
23. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 165–174 (2019) 1, 3
24. Park, K., Sinha, U., Barron, J.T., Bouaziz, S., Goldman, D.B., Seitz, S.M., Martin-Brualla, R.: Nerfies: Deformable neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5865–5874 (2021) 3
25. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems **32** (2019) 2, 14
26. Sitzmann, V., Martel, J., Bergman, A., Lindell, D., Wetzstein, G.: Implicit neural representations with periodic activation functions. Advances in Neural Information Processing Systems **33**, 7462–7473 (2020) 1, 2, 3, 4, 6, 8, 10
27. Sitzmann, V., Rezchikov, S., Freeman, B., Tenenbaum, J., Durand, F.: Light field networks: Neural scene representations with single-evaluation rendering. Advances in Neural Information Processing Systems **34**, 19313–19325 (2021) 1
28. Sitzmann, V., Zollhöfer, M., Wetzstein, G.: Scene representation networks: Continuous 3d-structure-aware neural scene representations. Advances in Neural Information Processing Systems **32** (2019) 1
29. Soh, J.W., Park, G.Y., Jo, J., Cho, N.I.: Natural and realistic single image super-resolution with explicit natural manifold discrimination. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8122–8131 (2019) 8

30. Suhail, M., Esteves, C., Sigal, L., Makadia, A.: Light field neural rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8269–8279 (2022) 1
31. Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., Ng, R.: Fourier features let networks learn high frequency functions in low dimensional domains. Advances in Neural Information Processing Systems **33**, 7537–7547 (2020) 2, 3, 7
32. Tewari, A., Fried, O., Thies, J., Sitzmann, V., Lombardi, S., Xu, Z., Simon, T., Nießner, M., Tretschk, E., Liu, L., et al.: Advances in neural rendering. In: ACM SIGGRAPH 2021. pp. 1–320. ACM (2021) 3
33. Yang, W., Zhang, X., Tian, Y., Wang, W., Xue, J.H., Liao, Q.: Deep learning for single image super-resolution: A brief review. IEEE Transactions on Multimedia **21**(12), 3106–3121 (2019) 8
34. Yariv, L., Kasten, Y., Moran, D., Galun, M., Atzmon, M., Ronen, B., Lipman, Y.: Multiview neural surface reconstruction by disentangling geometry and appearance. Advances in Neural Information Processing Systems **33**, 2492–2502 (2020) 3
35. Yu, A., Li, R., Tancik, M., Li, H., Ng, R., Kanazawa, A.: Plenoctrees for real-time rendering of neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5752–5761 (2021) 3
36. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: pixelnerf: Neural radiance fields from one or few images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4578–4587 (2021) 3
37. Yüce, G., Ortiz-Jiménez, G., Besbinar, B., Frossard, P.: A structured dictionary perspective on implicit neural representations. arXiv preprint arXiv:2112.01917 (2021) 7
38. Zhang, J., Yang, G., Tulsiani, S., Ramanan, D.: Ners: Neural reflectance surfaces for sparse-view 3d reconstruction in the wild. Advances in Neural Information Processing Systems **34** (2021) 3
39. Zhang, J., Yao, Y., Quan, L.: Learning signed distance field for multi-view surface reconstruction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6525–6534 (2021) 3