Basis Encoded Polynomial Neural Fields for Subband Decomposition

Anonymous Author(s)
Affiliation
Address
email

Abstract

Neural fields have emerged as a new paradigm for representing signals, thanks to their ability to represent signals compactly while being easy to optimize. In most applications, however, neural fields are treated like a black box, which precludes many signal manipulation tasks. In this paper, we propose a new class of neural fields called basis-encoded polynomial neural fields (PNFs). The key advantage of a PNF is that it can represent a signal as a composition of a number of manipulable and interpretable components without losing the merits of neural fields representation. We develop a general theoretical framework to analyze and design PNFs. We use this framework to design Fourier PNFs, which match state-of-the-art performance in signal representation tasks that use neural fields. In addition, we empirically demonstrate that Fourier PNFs enable signal manipulation applications such as texture transfer and scale-space interpolation.

1 Introduction

Neural fields are neural networks that take as input spatial coordinates and output a function of the coordinates, such as image colors [10, 53], 3D signed distance functions [3, 43], or radiance fields [37]. Recent works have shown that such representations are compact [14, 31, 50], allow sampling at arbitrary locations [53, 58], and are easy to optimize within the deep learning framework. These advantages enabled their success in many spatial visual computing applications including novel view synthesis [37, 39, 41, 52, 63] and 3D reconstruction [16, 24, 44, 52, 53]. Most recent neural field based methods, however, treat the network as a black box. As such, one can only obtain information by querying it with spatial coordinates. This precludes the applications where we want to change the signal represented in a neural field. For example, it is difficult to remove high frequency noise or change the stationary texture of an image.

One way to enable signal manipulation is to decompose a signal into a number of interpretable components and use these components for manipulation. A general approach that works across a wide range of signals is to decompose them into frequency subbands [1, 8, 51]. Such decompositions are studied in the traditional signal processing literature, e.g., the use of Fourier or Wavelet transforms of the spatial signal. These transformations, however, usually assume the signal is densely sampled in a regular grid. As a result, it is non-trivial to generalize these approaches to irregular data (e.g., point clouds). Another shortcoming of these transforms is that they require a lot of terms to represent a signal faithfully, which makes it difficult to scale to signals of more than two dimensions, such as in the case of light fields. Interestingly, these are the very problems that are solved by neural fields: they can represent irregular signals compactly and scale easily to higher dimensions. This leads to the

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

central question of this paper: can we incorporate the interpretability and controllability of classical signal processing pipelines to neural fields?

Our goal is to design a new class of neural fields that allow for precise subband decomposition and manipulation as required by the downstream tasks. To achieve that, our network needs to have the ability to control the output of different parts of the network to output signal that is limited by desirable subbands. At the same time, we want the network to inherit the usual advantages of neural fields, namely, being compact, expressive, and easy to optimize. The most relevant prior works with related aims are Multiplicative Filter Network (MFN) [18] and BACON [28]. While MFNs enjoy the advantages of neural fields and are easy to analyze, they do not use this property to control the network's output for subband decomposition. BACON [28] extends the MFN architecture to enforce that its outputs are upper-band limited, but it lacks the ability to provide subband control beyond upper band limits. This hinders BACON's applicability to tasks that requires more precise control of subbands, such as manipulating stationary textures (shown in Sec. 4.2).

To address these issues, we propose a novel class of neural fields called **basis-encoded polynomial neural fields (PNFs)**. PNF is a polynomial neural network [11] evaluated using a set of basis functions. PNFs are compact, easy to optimize, and can be sampled at arbitrary locations as with general neural fields. Moreover, PNFs enjoy interpretability and controllability of signal processing methods. We provide a theoretical framework to analyze the output of PNFs. We use this framework to design the Fourier PNF, whose outputs can be localized in the frequency domain with both upper and lower band limits, along with orientation specificity. To the best of our knowledge, this is the first neural field architecture that can achieve such a fine-grained decomposition of a signal. Empirically, we demonstrate that Fourier PNFs can achieve subband decomposition while reaching state-of-the-art performance for signal representation tasks. Finally, we demonstrate the utility of Fourier PNFs in signal manipulation tasks such as texture transfer and scale-space interpolation.

2 Related Works

Fourier and Wavelet Transforms. Transforms such as Fourier and Wavelet Transforms are at the core of image processing, signal processing, and computer vision pipelines (see for example [57]). Of particular relevance are the foundational signal decomposition frameworks of Image Pyramids such as the Laplacian [8] and Steerable [51] pyramids. Such representations are interpretable, and enjoy many desirable properties such as translation and rotation equivariance, as well as orientation control. However, these representations, usually assume the signal is densely sampled in a regular grid, and so it is non-trivial to generalize these approaches to irregular data. Further, these transforms require a lot of terms to represent a signal faithfully.

Neural Fields. Neural Fields are used to represent a wide variety of signals such as images [56], 3D shapes [3, 9, 13, 22, 35, 43, 63], 3D scenes [3, 16, 22, 24, 35, 43, 44, 52, 52, 53] and can be used for many tasks such as novel view synthesis [5, 6, 7, 19, 25, 27, 29, 30, 32, 37, 38, 39, 41, 42, 45, 49, 52, 54, 62, 63, 64, 65, 66]. However, neural fields typically operate as black boxes, hindering one's ability to control their outputs. Recent work attempted to provide such control with the ability to represent signals globally [34, 43, 53, 58] or locally [9, 10, 24, 33, 48]. In the Fourier domain, Random Fourier Features [58] and SIREN [53] design a neural field biased towards low/high frequencies. MINER provides an internal representation via a Laplacian pyramid based on neural fields. Mip-NeRF [5] band limits the *input* provided to an MLP. However, these approaches do not band-limit the output of the network as a function of the input. Recently, BACON [28] proposed a neural field representation enabling upper-band-limited guarantees. This representation is based on multiplicative filter networks (MFNs) [17]. While BACON is limited to Fourier basis functions, our framework handles a more general set of basis functions and network topologies. Unlike BACON and MFN, our framework can band-limit a signal also from below, thus enabling decomposition of a signal's frequency bands.

Polynomial Neural Networks. Our work falls into the class of polynomial networks that are realized through multiplicative interactions [4, 23, 47, 55]. Our work extends polynomial neural networks by

operating on different basis functions. Further, our work allows for a subband decomposition which can be used for signal manipulation tasks.

3 Method

We first provide a definition for basis-encoded polynomial neural fields and a basic theoretical framework to analyze their outputs in terms of subbands (Sec. 3.1). We then use this framework to design Fourier PNFs, a novel neural fields architecture to represent signals as a composition of fine-grain subbands in frequency spaces (Sec. 3.3).

3.1 Basis Encoded Polynomial Neural Fields

Recall that we would like to maintain the merits of the neural field representation while adding the ability to partition it into analyzable components. As with all neural networks, to guarantee expressivity, we want to base our neural fields on function compositions [46]. At the same time, we want our neural fields to be interpretable in terms of a set of basis functions that have known properties, as in the signal processing literature. We propose the following class of neural fields:

Definition 3.1 (PNF)**.** *Let $\mathcal{B}$ be a basis for the vector space of functions for $\mathbb{R}^n \to \mathbb{R}$. A* Polynomial neural field of basis *$\mathcal{B}$ is a neural network $f = g_L \circ \cdots \circ g_1 \circ \gamma$, where $\forall i, g_i$ are finite degree multivariate polynomials, and $\gamma : \mathbb{R}^n \to \mathbb{R}^d$ is a d-dimensional feature encoding using basis $\mathcal{B}$: $\forall i, \gamma_i \in \mathcal{B}$.*

This definition allows a rich design space that subsumes several prior works. For example, MFN [18] and BACON [28] can be instantiated by setting g_i to be either the linear layer or a masked multiplication layer. Similarly, if we set the basis to be $\mathcal{B} = \{x^n\}_{n\in\mathbb{N}}$, then we can show architectures proposed in Π-Net [12] are a subclass of PNF. This rich design space can potentially allow us to tailor the architecture to the application of interests, as demonstrated later in Sec. 3.3 and Sec. 4.3. Moreover, such rich space also contains many expressive neural networks, as shown by both the prior works [11, 12, 18, 28] and by our experiment in Sec. 4.1.

Furthermore, as long as the span of the basis is closed under multiplication, PNF yields a linear combination of basis functions and is thus easy to analyze:

Theorem 1. *Let F be a polynomial neural fields with basis $\mathcal{B}$. $\forall b_1, b_2 \in \mathcal{B}, b_1(x)b_2(x) = \sum_{i\in I} a_i b_i(x), |I| < \infty$,, then the output of F is a finite linear sum of the basis functions from $\mathcal{B}$.*

Many commonly used bases, such as Fourier, Gabor, and spherical harmonics, all satisfy this property. Please refer to the supplementary for proofs for a variety of different bases.

3.2 Controllable Subband Decomposition

Theorem 1 is not enough to control or manipulate the signal represented by the network, because any single neuron in the network may potentially be working with an arbitrary set of basis functions. This is a problem if we want to manipulate the signal by, for example, discarding the contributions of some basis elements (e.g., high-frequency components in the Fourier basis): which neurons do we discard?

To allow better manipulation, we use the notion of *subbands* from traditional frequency domain analysis [61]. In the most general sense, a subband is simply a *subset* of the basis. Manipulations such as smoothing or sharpening can then be done by discarding or enhancing the contributions of one or more subbands. One way to instantiate these ideas with a PNF is to represent the signal as a sum of different PFNs, each of which is limited to only a specific subband; then one can manipulate these component PFNs.

Definition 3.2. *A PNF F of basis $\mathcal{B}$ is limited by subband $S \subset \mathcal{B}$ if F is in the span of S.*

The key challenge is to construct a subband limited PNF for certain subbands. To this end, we need to understand how the subband-limited PNF transforms under different network operations. Fortunately,

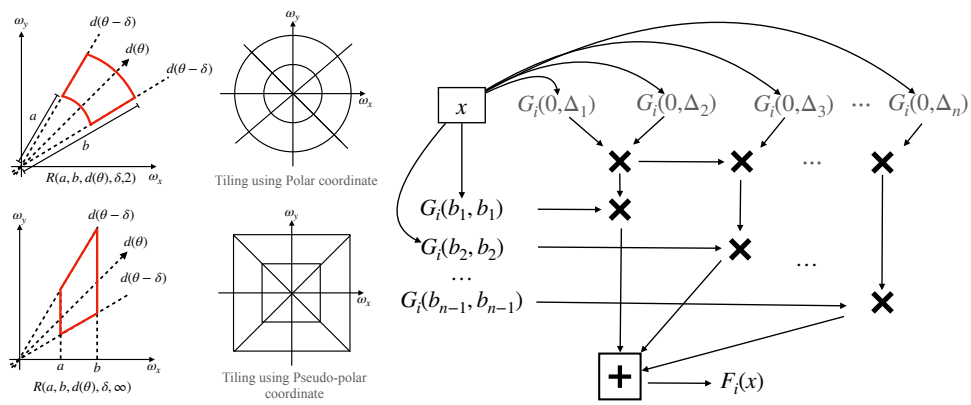


Figure 1: Left figure shows instances of two types of Fourier subbands and how these subbands tile the space. The right figure illustrate the architecture of F_i(Eq. (4)), a part of Fourier-PNF operates within a controllable set of subband. The output of Fourier PNF is an ensemble of these networks.

for PNFs we only need to study two types of operations: multiplication and addition. To this end, we need the notion of a *PNF-controllable set of subbands*:

Definition 3.3 (PNF-controllable Set of Subbands)**.** *$\mathcal{S} = \{S_\theta | S_\theta \subset \mathcal{B}\}_\theta$ is a PNF-Controllable Set of Subbands for basis $\mathcal{B}$ if (1) $S_{\theta_1} \cup S_{\theta_2} \in \mathcal{S}$ and (2) there exists a binary function $\otimes : \mathcal{S} \times \mathcal{S} \to \mathcal{S}$ such that if $b_1 \in S_{\theta_1}, b_2 \in S_{\theta_2} \implies b_1 b_2 \in S_{\theta_1} \otimes S_{\theta_2}$.*

Intuitively, a PNF-controllable set of subbands lends some *predictability* to what happens if two PNFs, limited to different subbands, are put together into a larger PNF:

Theorem 2. *Let $\mathcal{S}$ be a PNF-controllable set of subbands of basis $\mathcal{B}$ with binary relation $\otimes$. Suppose F_1 and F_2 are polynomial neural fields of basis $\mathcal{B}$ that maps $\mathbb{R}^n$ to $\mathbb{R}^m$. Furthermore, suppose F_1 and F_2 are subband limited by $R_1 \in \mathcal{S}$ and $R_2 \in \mathcal{S}$. Then we have the following:*

1. *$W_1 F_1(x) + W_2 F_2(x)$ is a PNF of $\mathcal{B}$ limited by subband $R_1 \cup R_2$ with $W_1, W_2 \in \mathbb{R}^{m \times n}$;*
2. *$F_1(x)^T W F_2(x)$ is a PNF of $\mathcal{B}$ limited by subband $R_1 \otimes R_2$ with $W \in \mathbb{R}^{m \times m}$.*

Many structures used in the signal processing literature, such as the Steerable Pyramid, use subbands that are PNF-controllable [8, 51]. Please refer to the supplementary for the derivations of PNF-controllable sets of subbands. In the following sections, we will focus on using Fourier bases and show how to build PNFs that instantiate subband manipulation efficiently.

3.3 Fourier PNF

In this section, we demonstrate how to build a PNF that can decompose a signal into frequency subbands in the following three steps. First, we identify a PNF-controllable sets of subband for the Fourier basis. Second, we choose a finite collection of subbands for the PNF to output, and organize them into controllable sets. The final step is to instantiate the PNF compactly using Theorem 2.

3.3.1 Controllable Subband Decomposition for Fourier Space

The Fourier basis can be written as: $b_{\boldsymbol{\omega}}(\mathbf{x}) = \exp(i\boldsymbol{\omega}^T \mathbf{x})$, where $\boldsymbol{\omega}, \mathbf{x} \in \mathbb{R}^d$ and i is the imaginary unit. It's easy to see that the fourier basis is complete under multiplication, which satisfies the condition for Theorem 1: $\exp(i\boldsymbol{\omega}_1^T \mathbf{x}) \exp(i\boldsymbol{\omega}_2^T \mathbf{x}) = \exp(i(\boldsymbol{\omega}_1 + \boldsymbol{\omega}_2)^T \mathbf{x})$.

Now we need to divide frequency space of $\boldsymbol{\omega} \in \mathbb{R}^d$ into subbands that are easy to manipulate and meaningful for downstream tasks. We define the subband following Simoncelli and Freeman [51]. Formally, frequency space is decomposed into following FAN-like regions:

$$R_F(\alpha, \beta, \mathbf{d}, \gamma, p) = \left\{ \boldsymbol{\omega} | \alpha \leq \|\boldsymbol{\omega}\|_p \leq \beta, \|\mathbf{d}\| = 1, \boldsymbol{\omega}^T \mathbf{d} < \gamma \|\boldsymbol{\omega}\| \right\}, \tag{1}$$

where p describe which norm do we choose to describe the frequency bands. Intuitively, α defines the lower band limits and β defines the upper band limits. The vector $\mathbf{d}$ defines the orientation of the subband and γ defines the angular width of the subband.

This definition of subband allow us to organize them into controllable sets: $\{R_F(\alpha, \beta, \mathbf{d}, \gamma, 2) | \forall 0 \leq \alpha \leq \beta\}$. These sets of subband are controllable because:

Theorem 3. *If $\boldsymbol{\omega}_1 \in R(\alpha_1, \beta_1, \mathbf{d}, \gamma, 2) = R_1$, and $\boldsymbol{\omega}_2 \in R(\alpha_2, \beta_2, \mathbf{d}, \gamma, 2) = R_2$. Moreover, assume that $\gamma < \frac{\pi}{4}$. If $b_{\boldsymbol{\omega}_i} b_{\boldsymbol{\omega}_i} = b_{\boldsymbol{\omega}_3}$, then $\boldsymbol{\omega}_3 \in R(\alpha_1 + \alpha_2, \beta_1 + \beta_2, \mathbf{d}, \gamma) = R_3$.*

This theorem captures the intuition that multiplication of two waves of similar orientations creates high frequency waves at that orientation. Please refer to the supplementary for visualization of how two 2D subbands interfacts under multiplications.

3.3.2 Subband Tiling

Next, we need to choose a finite sets of subbands that we want in the outputs. Using Theorem 3, we can tile the space using non overlapping fans

$$\mathcal{T}_{circ} = \{S_{ij} = R_F(b_i, b_{i+1}, \mathbf{d}(\theta_j), \delta, 2) | b_1 \leq \cdots \leq b_n, \theta_j = j\delta, \delta = \frac{\pi}{m}, 1 \leq j \leq 2m\}, \quad (2)$$

where $\mathbf{d}(\theta) = [\sin(\theta), \cos(\theta)]^T$ denotes unit vector rotate with angle θ, $\delta = \frac{\pi}{m}$, and m sufficiently small to allow the application of Theorem 3.

For 2D images, the region of interest is $[-N, N]^2$ where N is the Nyquist sampling rate [40]. To tile this rectangular region well without introducing unnecessary high-frequency details, we will use a pesudo-polar coordinate grids, which divide the space into vertical or horizontal sub-regions and tile those subregions according to l-∞ norm. Formally, the 2D pesudo polar coordinate tiling can be written as following:

$$\mathcal{T}_{rect} = \{S_{ij} = R_F(b_i, b_{i+1}, \mathbf{d}(\theta_j), \delta, \infty) | b_1 \leq \cdots \leq b_n, \theta_j = j\delta, 1 \leq j \leq 2m, j \neq m\}. \quad (3)$$

Note that we exclude certain regions to avoid having a subband to include orientation at $\frac{\pi}{4}$ and $\frac{3\pi}{4}$ in order for Theorem 3 to generalize to such tiling. We will show detail derivation in the supplementary.

This formulation allow us to organize information in a variety of meaningful ways. For example, the $\mathcal{T}$ can be organized into a set of different cones $\{S_{ij}\}_{j=1}^{2m}$, which corresponds to different orientation of the signal. Alternatively, we can also organize $\mathcal{T}$ as different rings $\{S_{ij}\}_{i=1}^{n}$, which corresponds to the decomposition of an image into the Laplacian Pyramid.

3.3.3 Network architecture

We now have decided on the set of subbands we want to produce by PNF. We want to design the final Fourier PNF as an ensemble of subband limited PNFs: $F(\mathbf{x}) = \sum_j \sum_i O_{ij} F_{ij}(\mathbf{x})$, where $F_{ij}(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^h$ is a PNF that's limited with subband S_{ij} defined in Eq. (3), and $O_{ij} \in \mathbb{R}^{h \times m}$ aggregates the output signals together. One way to achieve this is to naively define F_{ij} as a two layers PNF the feature encoding layer to include only the basis functions in the subband S_{ij} followed by a linear layer. However, such an approach fails to achieve good performance as it creates an ensemble of shallow networks. Alternatively, we leverage Theorem 2 to factorize F:

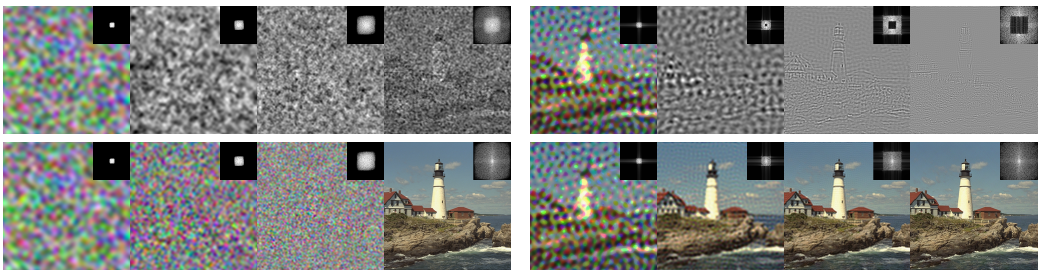
$$F(\mathbf{x}) = \textstyle\sum_j F_j(\mathbf{x}), F_j(\mathbf{x}) = G_j(\mathbf{x}, b_i, b_i) W_{jn} Z_{j,n}(\mathbf{x}), \quad (4)$$

$$Z_{j,1}(\mathbf{x}) = G_j(\mathbf{x}, 0, \Delta_1), Z_{j,k}(\mathbf{x}) = G_j(\mathbf{x}, 0, \Delta_k) W_i Z_{j,k-1}(\mathbf{x}), \quad (5)$$

where $G_j(\mathbf{x}, a, b)$ is subband limited in $R_F(a, b, d(\theta_j), \delta, \infty)$ and $\Delta_k = b_k - b_{k-1}$. This network architecture of F_i is illustrated in Fig. 1. We instantiate this architecture by setting $G_j(\mathbf{x}, a, b)$ into a linear transform of basis sampled from the subband to be limited:

$$G_j(\mathbf{x}, a, b) = W_i \gamma_j(\mathbf{x}), \gamma_j \in R_F(a, b, d(\theta_j), \delta, \infty)^d, W_i \in \mathbb{R}^{h \times d}, \quad (6)$$

where h and d is the dimension for the output and the feature encoding. We will provide more implementation details in the supplementary materials.



BACON PNF

Figure 2: BACON: The bottom row shows the output of each layer which is upper band limited. The top row (columns 2-4) shows the difference between the output of a given layer and the one before it. PNF: The top row shows the output of each layer which is both upper and lower band limited. The bottom row (columns 2-4) shows the addition of the output at a given layer and the one before it.

Table 1: Image Fitting on the DIV2K dataset. BAC stands for BACON.

	PSNR	SSIM
RFF	28.72	0.834
SIREN	29.22	0.866
BAC	28.67	0.838
BAC-L	29.44	0.871
BAC-M	29.44	0.871
PNF	**29.47**	**0.874**

Table 2: 3D shape fitting. CD is Chamfer Distance ($\times 10^6$); BAC is BACON

	CD	F-score
SIREN	9.00	99.76%
BAC	2.60	99.84%
BAC-L	2.60	99.85%
BAC-M	2.61	99.85%
PNF	**2.25**	**99.97%**

Table 3: NeRF Fitting for 64^2 resolution. Top rows: BACON; Bottom rows: PNF;

	300 epochs		500 epochs	
	PSNR	SSIM	PSNR	SSIM
1x	28.07	0.936	28.51	0.932
Avg	30.04	**0.936**	30.78	0.935
1x	**29.89**	**0.937**	**29.33**	**0.950**
Avg	**31.29**	0.934	**30.84**	**0.940**

4 Results

We demonstrate the applicability of our framework along three different axes: (*Expressivity*) For a number of different signal types, we demonstrate our ability to fit a given signal. We observe that our method enjoys faster convergence in terms of the number of iterations. (*Interpretability*) We visually demonstrate our learned PNF, whose outputs are localized in the frequency domain with both upper band-limit and lower-band limit. (*Decomposition*) Finally, we demonstrate the ability to control a PNF on the tasks of texture transfer and scale-space representation, based on our subband decomposition. For all presented experiments, full training details and additional qualitative and quantitative results are provided in the supplementary.

4.1 Expressivity

We demonstrate that a PNF is capable of representing signals of different modalities including images, 3D signed distance fields, and radiance fields. We compare our Fourier PNF with state-of-the-art neural field representations such as BACON [28], SIREN [53], and Random Fourier Features [58].

Images Following BACON [28], we train a PNF and the baselines to fit images from the DIV2K [2] dataset. During training, images are downsampled to 256^2. All networks are trained for 5000 iterations. At test time, we sample the fields at 512^2 and compare with the original resolution images.

Different from other networks, BACON is supervised with the training image in all output layers. For fair comparison, we also include two BACON variants, which are only supervised either at the last output layer ("BAC-L") or using the average of all output layers ("BAC-M"). We report the PSNR and Structural Similarity (SSIM) scores of these methods in Tab. 1. Fourier PNF improves on the performance of the previous state-of-the-arts.

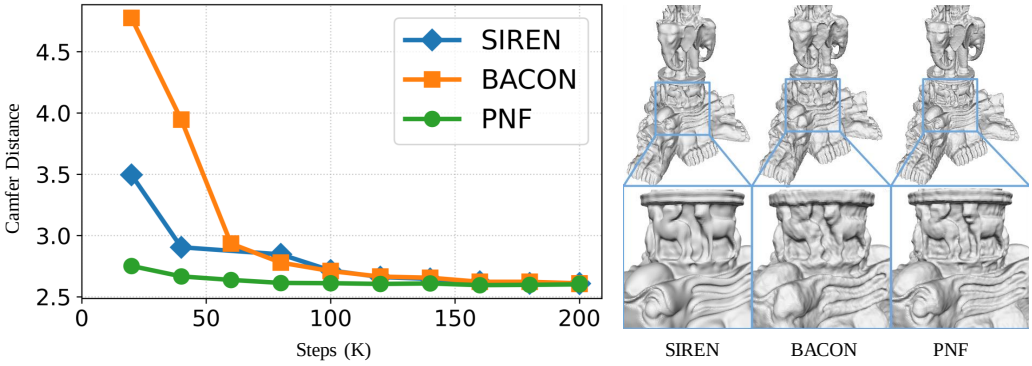


Figure 3: LHS: Convergence time in terms of number of steps/iterations for the *Thai Statue* model [59]. The x-axis shows the number of steps (K) and the y-axis shows the validation Chamfer Distance ($\times 10^6$). RHS: Qualitative comparison for the *Thai Statue* model [59].

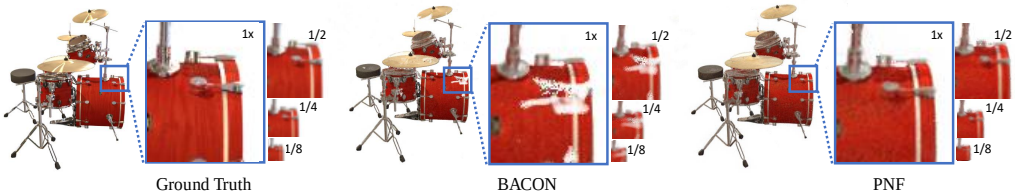


Figure 4: Qualitative comparison of neural radiance fields for the *Drums* scene given in [5].

In addition to its expressivity, PNF is also able to localize signals in different regions. Specifically, Fourier PNF decomposes an image into subbands in the freuqency domain. For example, Fig. 2 shows that the output branches of the Fourier PNF correspond to specific frequency bands. Note that such decomposition forms for all layers while training is only performed for the last layer of the PNF. Such subband control is reminiscent of the traditional signal processing analogue of Laplacian Pyramids [8]. Note that each layer of the PNF is guaranteed to be both lower and upper frequency band limted, while this is not achievable by BACON. For comparison Fig. 2 shows the corresponding result of training BACON only for the last layer. In Sec. 4.2 and Sec. 4.3 we demonstrate how to leverage such fine-grain localization for texture transfer and scale space interpolation.

3D Signed Distance Field One advantage of neural fields is their ability to represent irregular data such as 3D point cloud or signed distance fields with high fidelity. We demonstrate that PNFs can represent 3D shapes via signed distance fields expressively. We follow BACON's experimental setting to fit a range of 3D shapes from the Stanford 3D scanning repository [59] (a slightly different normalization is used, see supplementary for details). During training, we sample $10k$ oriented points from the ground truth surface and perturb them with noise to compute an estimate of SDF as ground truth. All models are trained with reconstruction loss for the same number of iterations. A quantitative comparison of the different methods is reported in Tab. 2. We observe that PNF achieves slightly higher fitting quality than other methods. Next, we investigate convergence behavior in Fig. 3 (Left) on the *Thai Statue* model [59]. Here, we observe that PNFs converges much more quickly than SIREN and and BACON while achieving comparable quality. In Fig. 3 (Right) we provide a qualitative comparison of our result, BACON, and SIREN.

Neural Radiance Field In the context of neural radiance fields, we show that PNFs can represent signals in higher dimensions compactly and faithfully. We follow BACON's setting and train a PNF to model the radiance field of a set of Blender scenes [37], see supplementary for details. Specifically, the PNF outputs a 4D vector of RGB and a density values. These values are used by a volumetric

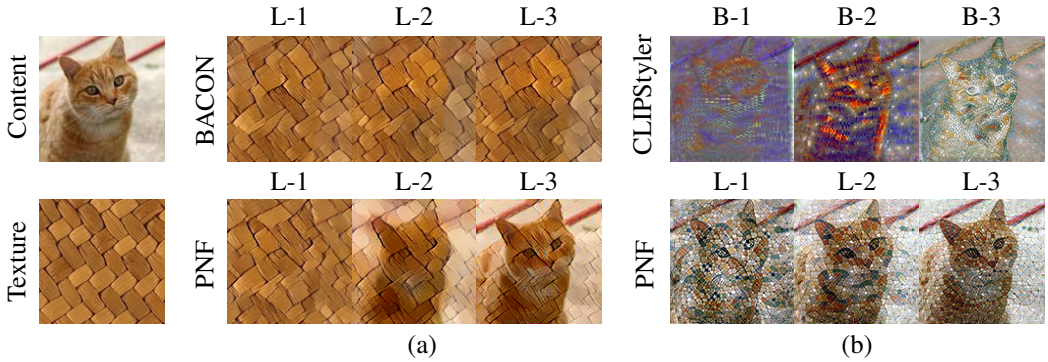


Figure 5: Texture transfer. (a). We optimize specific layers of the neural field. L-1 (layers 1-4), L-2 (layers 2-4), L-3 (layers 3-4). The texture contains stationary texture (in high frequency) and structured texture (in low frequency). As opposed to BACON, our method can isolate the stationary texture. (b) We consider the text based texture transfer objectives of CLIPStyler [26] for the cat content image. For PNF, we consider the text prompt "Mosaic" and apply the same layer-based optimization as in (a). For comparison, we apply CLIPStyler of "Low frequency mosaic" (B-1), "High frequency mosaic" (B-3) and "Mosaic" (B-3).

renderer proposed of NeRF [37] to produce an image, which is supervised with reconstruction loss. At test time, we use the same volumetric renderer to produce images from different camera poses and evaluate them with the known ground truth images. The results, in comparison to BACON, are presented in Tab. 3. A Qualitative comparison is also given in Fig. 4. Similarly to SDF, we are able to match or improve BACON's performance using about 60% of the total number of iterations. This can be potentially attributed to PNF's ability to disentangle coarse signal from finer one; and so at higher layers, BACON needs to relearn low-frequency details in the image while the PNF does not.

4.2 Texture Transfer

Traditional approaches for image manipulation assume the input and output images are represented using a regular grid [15, 21]. Recent work has opted to use neural fields instead, for example allowing fine detailed texturing of 3D meshes [36]. Our formulation allows for an additional layer of control. In particular, due to our subband decompositionality, one can restrict the manipulation to particular subbands. The manipulation can then be driven by optimizing the PNF weights corresponding to those subbands, using various loss objectives.

We demonstrate this in the setting of texture transfer. We consider a Fourier PNF with four layers of the following frequency ranges: (1) $[0, 8]$, (2) $[4, 16]$, (3) $[12, 32]$ and (4) $[28, 64]$. We consider a *content image* C of 128^2 resolution. In the first stage, we train a network to fit C as in Sec. 4.1. In the second stage, we optimize only the parameters of specifics layers. To optimize these parameters, we query the network on a 128^2 image grid producing image I. We then consider two sets of objectives: (a) Content and style loss objectives as given in [20]. (b), Text-based texture manipulation objectives as given by CLIPStyler [26]. See further details in the supplementary.

In Fig. 5, we illustrate the result of optimizing layers 1 to 4, 2 to 4 or 3 to 4, corresponding to frequency ranges [0, 64], [4, 64] and [12, 64] respectively. We consider a texture which contains both stationary texture in the high frequency range and structured texture in the low frequency range. As opposed to BACON, our method can isolate the stationary texture. As can be seen in Fig. 5(b), for text based manipulation, our method generates texture that is in the correct frequency range. For comparison, we consider CLIPStyler [26], with text prompts of "Low frequency mosaic" (B-1), "High frequency mosaic" (B-3) and "Mosaic" (B-3), resulting in non-realistic texturing. As can be seen, simply specifying the frequency in the text does not result in a satisfactory result.

4.3 Scale-space Representation

In many visual computing applications such as volumetric rendering, one is usually required to aggregate information from the neural fields using operations such as Gaussian convolution [5]. It is useful to model signals as a function of both the spatial coordinates and the scale: $f(\mathbf{x}, \Sigma) = \mathbb{E}_{\mathbf{x}\sim\mathcal{N}(\mathbf{x},\Sigma)}[g(\mathbf{x})]$, where g is the assumed ground truth signal. Existing works try to approximate the scale-space by using a black-box MLP with intergrated positional encoding, which computes the analytical Gaussian convolved Fourier basis functions [5, 60]. While such approaches demonstrate success in volumetric rendering applications which requires integrating different scales, they depend on supervision in multiple scales, possibly because their ability to interpolate correctly between scales is hindered by the black-box MLP.

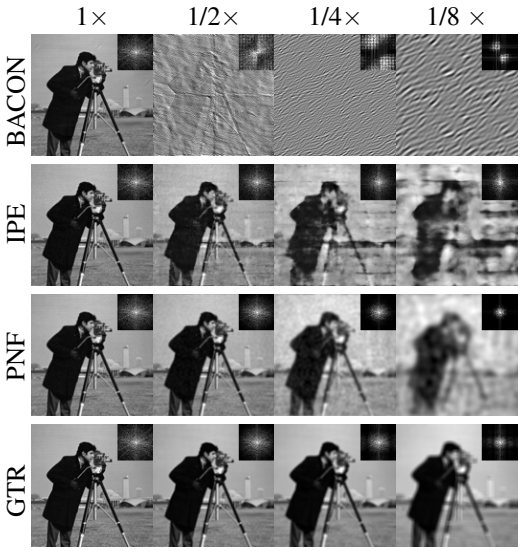


Figure 6: Scale-space representation. Networks are trained on full resolution image (1x) and test on the other reolutions. GTR is produced by applying Gaussian smoothing on (1x) image.

In this section we want to demonstrate the PNF's ability to better model this scale space with limited supervision. Suppose the signal of interest can be represented by Fouier bases as $g(\mathbf{x}) = \sum_n \alpha_n \exp(\omega_i^T \mathbf{x})$, then we know analytically the Gaussian convolved version should be $f(\mathbf{x}, \Sigma) = \sum_n \alpha_n \exp(\omega_i^T \Sigma \omega_i) \exp(i\omega_i^T \mathbf{x})$. If we assume that our Fourier PNF can learn the ground truth representation well, then one potential way to achieve this is setting $\gamma(\mathbf{x}, \Sigma)_n = \exp(-\frac{1}{2}\omega_n^T \Sigma \omega_n) \exp(-i\omega_n^T \mathbf{x})$. We also multiply the output of F_i (Eq. (4)) with a correction term addressing the error arises from missing the interference terms in the form of $\exp(-\frac{1}{2}\omega_i^T \Sigma \omega_j)$. We show in the supplementary how to derive and approximate these missing terms using Fourier PNF.

We test the effectiveness of Fourier PNF to learn the scale-space of a 2D image. In this application, the network is trained with the image signal in full resolution (finest scale). At test time, the network is asked to produce image with different scales and compared to the ground truth Gaussian smoothed image. We compared our method with IPE [5] as well as BACON with IPE as filter function. The results are shown in Fig. 6. Our model can represent a signal reasonably well when testing with a lower resolution while other methods degrade more quickly.

Limitations Currently, the activation memory of PNF scales linearly in the number of subbands, and so interpretability and decompositiality gained by PNF comes at a "cost" of a larger memory footprint. Further, it is also non trivial to tile higher dimensional space with controllabel subbands.

5 Conclusion

We proposed a novel class of neural fields (PNFs) that are compact and easy to optimize and enjoy the interpretability and controllability of signal processing methods. We provided a theoretical framework to analyse the output of a PNF and design a Fourier PNF, whose outputs can be decomposed in a fine-grained manner in the frequency domain with user-specified lower and upper limits on the frequencies. We demonstrated that PNFs matches state-of-the-art performance for signal representation tasks. We then demonstrated the use of PNF's subband decomposition in the settings of texture transfer and scale-space representations. As future work, the ability to generalize our representation to represent multiple higher-dimensional signals (such as multiple images) can enable applications in recognition and generation, where one can leverage our decomposeable architecture to impose a prior or regularize specific subbands to improve generalization.

References

[1] Edward H Adelson, Charles H Anderson, James R Bergen, Peter J Burt, and Joan M Ogden. Pyramid methods in image processing. *RCA engineer*, 29(6):33–41, 1984. 1

[2] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 126–135, 2017. 6

[3] Matan Atzmon and Yaron Lipman. SAL: Sign agnostic learning of shapes from raw data. In *Proc. CVPR*, 2020. 1, 2

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 2

[5] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. 2, 7, 9

[6] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. Nerd: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12684–12694, 2021. 2

[7] Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan Barron, and Hendrik Lensch. Neural-pil: Neural pre-integrated lighting for reflectance decomposition. *Advances in Neural Information Processing Systems*, 34, 2021. 2

[8] Peter J Burt and Edward H Adelson. The laplacian pyramid as a compact image code. In *Readings in computer vision*, pages 671–679. Elsevier, 1987. 1, 2, 4, 7

[9] Rohan Chabra, Jan Eric Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local SDF priors for detailed 3D reconstruction. In *Proc. ECCV*, 2020. 2

[10] Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8628–8638, 2021. 1, 2

[11] Grigorios Chrysos, Stylianos Moschoglou, Yannis Panagakis, and Stefanos Zafeiriou. Polygan: High-order polynomial generators. *arXiv preprint arXiv:1908.06571*, 2019. 2, 3

[12] Grigorios G Chrysos, Stylianos Moschoglou, Giorgos Bouritsas, Yannis Panagakis, Jiankang Deng, and Stefanos Zafeiriou. P-nets: Deep polynomial neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7325–7335, 2020. 3

[13] Thomas Davies, Derek Nowrouzezahrai, and Alec Jacobson. On the effectiveness of weight-encoded neural implicit 3D shapes. *arXiv preprint arXiv:2009.09808*, 2020. 2

[14] Emilien Dupont, Adam Goliński, Milad Alizadeh, Yee Whye Teh, and Arnaud Doucet. Coin: Compression with implicit neural representations. *arXiv preprint arXiv:2103.03123*, 2021. 1

[15] Alexei A Efros and William T Freeman. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 341–346, 2001. 8

[16] S. M. Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S. Morcos, Marta Garnelo, Avraham Ruderman, Andrei A. Rusu, Ivo Danihelka, Karol Gregor, et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018. 1, 2

[17] Rizal Fathony, Anit Kumar Sahu, Devin Willmott, and J Zico Kolter. Multiplicative filter networks. In *Proc. ICLR*, 2020. 2

[18] Rizal Fathony, Anit Kumar Sahu, Devin Willmott, and J. Zico Kolter. Multiplicative filter networks. In *ICLR*, 2021. 2, 3

[19] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. FastNeRF: High-fidelity neural rendering at 200fps. In *Proc. ICCV*, 2021. 2

[20] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015. 8

[21] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016. 8

[22] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *Proc. ICML*, 2020. 2

[23] Siddhant M Jayakumar, Wojciech M Czarnecki, Jacob Menick, Jonathan Schwarz, Jack Rae, Simon Osindero, Yee Whye Teh, Tim Harley, and Razvan Pascanu. Multiplicative interactions and where to find them. 2020. 2

[24] Chiyu Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, and Thomas Funkhouser. Local implicit grid representations for 3D scenes. In *Proc. CVPR*, 2020. 1, 2

[25] Yue Jiang, Dantong Ji, Zhizhong Han, and Matthias Zwicker. SDFDiff: Differentiable rendering of signed distance fields for 3D shape optimization. In *Proc. CVPR*, 2020. 2

[26] Gihyun Kwon and Jong Chul Ye. Clipstyler: Image style transfer with a single text condition. *arXiv preprint arXiv:2112.00374*, 2021. 8

[27] David B. Lindell, Julien N. P. Martel, and Gordon Wetzstein. AutoInt: Automatic integration for fast neural volume rendering. In *Proc. CVPR*, 2021. 2

[28] David B. Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. Bacon: Band-limited coordinate networks for multiscale scene representation. *CVPR*, 2022. 2, 3, 6

[29] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *NeurIPS*, 2020. 2

[30] Shaohui Liu, Yinda Zhang, Songyou Peng, Boxin Shi, Marc Pollefeys, and Zhaopeng Cui. DIST: Rendering deep implicit signed distance function with differentiable sphere tracing. In *Proc. CVPR*, 2020. 2

[31] Julien N. P. Martel, David B. Lindell, Connor Z. Lin, Eric R. Chan, Marco Monteiro, and Gordon Wetzstein. ACORN: Adaptive coordinate networks for neural scene representation. *ACM Trans. Graph. (SIGGRAPH)*, 40(4), 2021. 1

[32] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *Proc. CVPR*, 2021. 2

[33] Ishit Mehta, Michaël Gharbi, Connelly Barnes, Eli Shechtman, Ravi Ramamoorthi, and Manmohan Chandraker. Modulated periodic activations for generalizable local functional representations. In *Proc. ICCV*, 2021. 2

[34] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *Proc. CVPR*, 2019. 2

[35] Mateusz Michalkiewicz, Jhony K. Pontes, Dominic Jack, Mahsa Baktashmotlagh, and Anders Eriksson. Implicit surface representations as layers in neural networks. In *Proc. ICCV*, 2019. 2

[36] Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. Text2mesh: Text-driven neural stylization for meshes. *arXiv preprint arXiv:2112.03221*, 2021. 8

[37] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proc. ECCV*, 2020. 1, 2, 7, 8

[38] Thomas Neff, Pascal Stadlbauer, Mathias Parger, Andreas Kurz, Joerg H. Mueller, Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, and Markus Steinberger. DONeRF: Towards Real-Time Rendering of Compact Neural Radiance Fields using Depth Oracle Networks. *Computer Graphics Forum*, 40(4), 2021. 2

[39] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3D representations without 3D supervision. In *Proc. CVPR*, 2020. 1, 2

[40] Harry Nyquist. Certain topics in telegraph transmission theory. *Transactions of the American Institute of Electrical Engineers*, 47(2):617–644, 1928. 5

[41] Michael Oechsle, Lars Mescheder, Michael Niemeyer, Thilo Strauss, and Andreas Geiger. Texture fields: Learning texture representations in function space. In *Proc. ICCV*, 2019. 1, 2

[42] Michael Oechsle, Songyou Peng, and Andreas Geiger. UNISURF: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proc. ICCV*, 2021. 2

[43] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proc. CVPR*, 2019. 1, 2

[44] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *Proc. ECCV*, 2020. 1, 2

[45] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural radiance fields for dynamic scenes. In *Proc. CVPR*, 2021. 2

[46] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. Survey of expressivity in deep neural networks. *arXiv preprint arXiv:1611.08083*, 2016. 3

[47] Scott Reed, Kihyuk Sohn, Yuting Zhang, and Honglak Lee. Learning to disentangle factors of variation with manifold interaction. In *International conference on machine learning*, pages 1431–1439. PMLR, 2014. 2

[48] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. KiloNeRF: Speeding up neural radiance fields with thousands of tiny mlps. In *Proc. ICCV*, 2021. 2

[49] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. PIFu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *Proc. ICCV*, 2019. 2

[50] Vishwanath Saragadam, Jasper Tan, Guha Balakrishnan, Richard G Baraniuk, and Ashok Veeraraghavan. Miner: Multiscale implicit neural representations. *arXiv preprint arXiv:2202.03532*, 2022. 1

[51] Eero P Simoncelli and William T Freeman. The steerable pyramid: A flexible architecture for multi-scale derivative computation. In *Proceedings., International Conference on Image Processing*, volume 3, pages 444–447. IEEE, 1995. 1, 2, 4

[52] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3D-structure-aware neural scene representations. In *Proc. NeurIPS*, 2019. 1, 2

[53] Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Proc. NeurIPS*, 2020. 1, 2, 6

[54] Pratul P. Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T. Barron. NeRV: Neural reflectance and visibility fields for relighting and view synthesis. In *CVPR*, 2021. 2

[55] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015. 2

[56] Kenneth O Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic programming and evolvable machines*, 8(2):131–162, 2007. 2

[57] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010. 2

[58] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *Proc. NeurIPS*, 2020. 1, 2, 6

[59] Stanford University. 3D scanning repository. http://graphics.stanford.edu/data/3Dscanrep/, 1996. 7

[60] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd E. Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. *ArXiv*, abs/2112.03907, 2021. 9

[61] Martin Vetterli and Jelena Kovacevic. Wavelets and subband coding. In *Prentice Hall Signal Processing Series*, 1995. 3

[62] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *Proc. NeurIPS*, 2021. 2

[63] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Ronen Basri, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. In *Proc. NeurIPS*, 2020. 1, 2

[64] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. In *Proc. NeurIPS*, 2021. 2

[65] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *Proc. CVPR*, 2021. 2

[66] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 2

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] All contributions accounted for in Sec. 3 and Sec. 4.
   (b) Did you describe the limitations of your work? [Yes] See end of Sec. 4.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See supplementary.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Sec. 3 and supplementary
   (b) Did you include complete proofs of all theoretical results? [Yes] See Sec. 3 and supplementary
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Data used is publicly available and cited. Code and instructions for experimental results provided in supplementary.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Full details in Sec. 4 and supplementary
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See supplementary.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See supplementary.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes] See supplementary.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] Code is provided in the supplementary.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] See supplementary.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] See supplementary.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]