BANF: Band-limited Neural Fields for Levels of Detail Reconstruction

Akhmedkhan (Ahan) Shabanov[1], Shrisudhan Govindarajan[1], Cody Reading[1], Lily Goli[2], Daniel Rebain[3], Kwang Moo Yi[3], Andrea Tagliasacchi[1,2,4]

[1]Simon Fraser University, [2]University of Toronto [3]University of British Columbia, [4]Google DeepMind

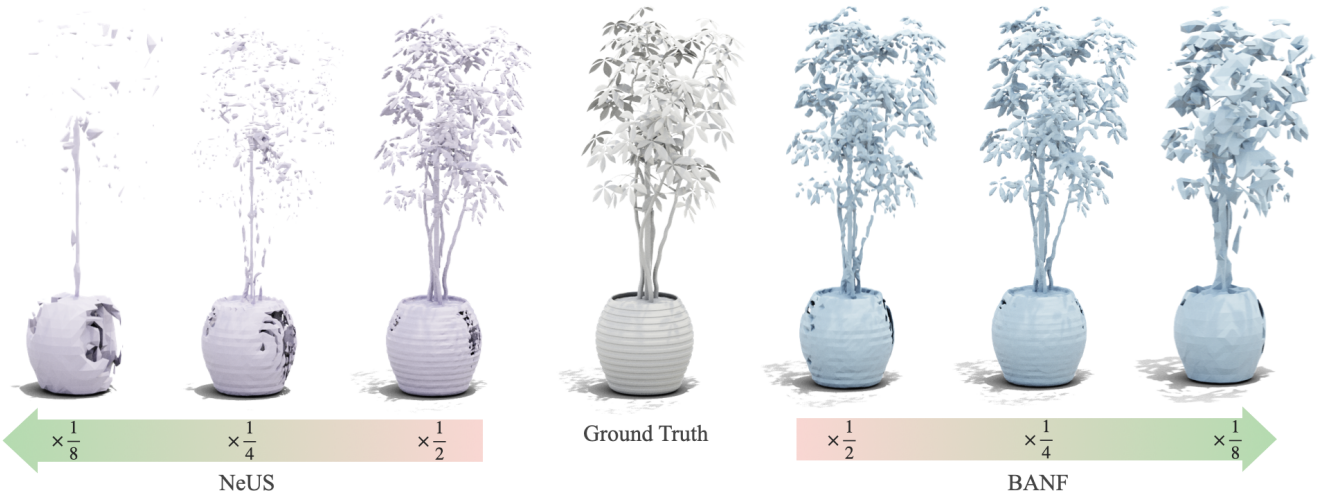


Figure 1. We introduce BANF, a method for band-limited frequency decomposition in neural fields. Our minimal yet impactful enhancements to the training process achieves anti-aliased coarse-to-fine signal reconstruction, seamlessly integrating with commonly used mesh extraction techniques [25], significantly surpassing prior methods in quality. The reconstruction artifacts due to view-dependent effects (i.e. vase bowl not being smooth) are due to the reconstruction baseline we employ, and is orthogonal to our method.

Abstract

Largely due to their implicit nature, neural fields lack a direct mechanism for filtering, as Fourier analysis from discrete signal processing is not directly applicable to these representations. Effective filtering of neural fields is critical to enable level-of-detail processing in downstream applications, and support operations that involve sampling the field on regular grids (e.g. marching cubes). Existing methods that attempt to decompose neural fields in the frequency domain either resort to heuristics or require extensive modifications to the neural field architecture. We show that via a simple modification, one can obtain neural fields that are low-pass filtered, and in turn show how this can be exploited to obtain a frequency decomposition of the entire signal. We demonstrate the validity of our technique by investigating level-of-detail reconstruction, and showing how coarser representations can be computed effectively. Source code is available at https://theialab.github.io/banf

1. Introduction

Recent research enabled neural rendering to accurately model 3D scenes from multi-view images alone. Many neural rendering techniques employ *neural fields* as the underlying 3D representation – coordinate neural networks that can represent a variety of signals including images [27, 37], surface manifolds [32, 42, 48], and volumetric densities [29]. These representations are now used in a plethora of applications that include robotics [1], semantic understanding [20], scene capture [22], editing [14], and generative modeling from natural language [33].

While neural fields are most commonly trained to represent the *entire* spectrum of signals, it is an established practice in classical signal processing to decompose the signal into separate frequency bands due to the many advantages it brings [2]. For example, signal decomposition allows 3D objects to be represented at multiple levels of detail, which finds application in reducing the computational cost of rendering or physics-based simulations [26]. In classical signal processing, frequency decomposition can be studied via the

Fourier transform [31]. Signals can be decomposed through band-pass filtering, and later recombined through linear superposition. One could apply these classical techniques by first *sampling* neural fields on a regular lattice, but the Nyquist theorem tells us that signals ought to be properly filtered before being sampled so to avoid aliasing [31]. This is particularly critical for methods that target surface reconstruction [22], as these techniques often rely on marching cubes to extract manifolds from fields [25], and this process involves sampling neural fields on *regular* grids; see Fig. 1 for an example.

While monte-carlo filtering of neural fields would make anti-aliased sampling possible [9], a critical question is whether filtering could be realized *directly* at training time to minimize its impact on inference, rather than *a-posteriori* as distillation [13]. This can be achieved when neural fields are implemented as fully-connected networks (MLPs). One can build over the theory of multiplicative filter networks [11] to realize signal decomposition for neural fields [10, 23, 46]. However, due to *significant* gains in training performance, most recent methods do not implement neural fields as large MLP networks, but rather via hybrid models combining interpolated feature grids with small MLPs mapping features to the field co-domain – this renders the theory of multiplicative filter networks to hybrid neural fields not applicable. One exception is Wu et al. [43], but while they introduce a pyramid-like architecture, their main outcome is a compressed representation.

Interestingly, early research in neural fields exploited the duality of time and frequency scaling to realize (explicit) multi-scale training architectures [24, 39]. [1] However, these methods require the multi-resolution structure of the signal to either be specified *a-priori* [39], or by monitoring the training process via heuristics [24]. Conversely, our technique can be applied to any neural field, without any modification to the underlying architecture, and without any assumption about the training data.

Our core insight is that regularly *sampling* a field, and then *interpolating* this field with a band-limited kernel can be seen as a low-pass filtering operation. We can then build band-pass signals by compositing these filters, which allows us to derive a suitable coarse-to-fine hierarchical training scheme. We demonstrate the validity of our method across domains (1D, 2D, 3D), and most importantly on 3D representations trained from 2D observations. We show how filtered representations can be extracted at any of the intermediate scales at a fidelity that is superior to that of reasonable baselines. Further, these representations can be aggregated, matching the performance of representations that were directly trained at high-resolution only.

[1] Interestingly, as we will see later, bi-linear interpolation does execute a form of filtering, but this filtering affects the field feature space, rather than, as we desire, in the field co-domain.

2. Related work

Our work is built on techniques in signal processing and neural fields. We further review how signals in neural fields can be spatially decomposed, frequency decomposed by positional encoding, and frequency decomposed by network design.

Frequency decomposition by signal processing. In traditional signal processing, signals are often converted to the frequency domain with the Fourier Transform [31]. In the frequency domain, lower frequencies represent smoother portions of the signal while high frequencies represent finer details. The frequency domain allows the signal to be separated into sub-bands to capture different levels of detail (LODs) [17, 31]. Unfortunately, there is no analytical Fourier Transform equivalent for signals represented as neural fields. A signal can be decomposed into coarse-to-fine LODs by convolution with appropriate kernels. However, since neural fields are continuous functions, convolutions must be executed as a monte-carlo integration [9] whose result can be distilled in a secondary neural field [13]. Conversely, we propose a simple method to directly generate low-pass and band-pass signals during field optimization, with a minimal modification in the training procedure.

Neural fields. As a form of signal representation, neural fields have recently gained substantial importance [45]. Neural fields can effectively represent many signals, including 2D images [27, 37], and 3D geometry in the form of Signed Distance Functions [32] and Occupancy Fields [28]. Arguably the most well-known example of neural fields is neural radiance fields, enabling novel view synthesis through volumetric [29] or surface [42] rendering. Classical neural fields [29, 32] use a multilayer perceptron (MLP) to map an input coordinate to a signal value. Non-neural variants that optimize explicit feature grids later emerged [12, 18], sacrificing slightly worse reconstructions for faster training speed. Hybrid models combining explicit representations (grid [38], hash table [30], K-Planes [7]) with a small MLP finally emerged, achieving both fast training speed and high quality reconstruction. Although there are many variants of neural field architectures, our sampling-based strategy can be applied on top of *any* architecture.

Multi-scale fields by spatial decomposition. Neural fields can be trained with spatial decomposition to produce coarse-to-fine LODs, where varying levels of detail are modeled with multi-resolution grids such as octrees [39] or multi-resolution tri-planes [49]. Adaptive data structures allow for increased resolution in high-frequency regions, which can be identified for further subdivision using indicators such as high local training error [27] or high density [24]. Saragadam et al. [35] suggest a memory-efficient

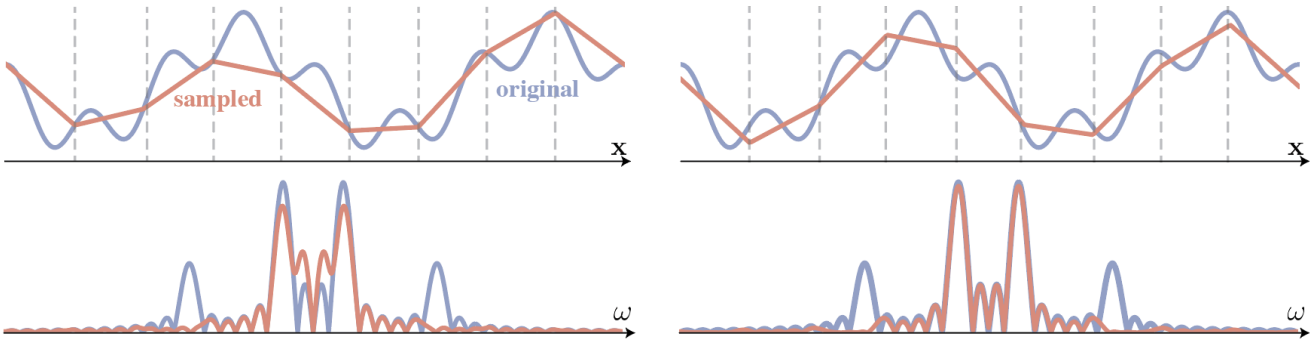


Figure 2. **Filtering by optimization** – We compare two approaches to generate a low-pass version of a neural field's signal. Left shows training a neural field to a 1D signal, and sampling the result a-posteriori on a regular lattice. On the right, our approach trains the neural field in a way that is "sampling aware", as the field is expressed as the (linear) interpolation of sampled field values. In other words, our method executes low-pass filtering *during* optimization. At the bottom we visualize the signal's spectra, where the baseline's reconstruction is clearly affected by aliasing, while our reconstruction is clearly low-pass filtered.

approach using separate MLPs for data patches when multi-scale supervision available. Such methods are dependent on *complex* and *heuristic-based* spatial subdivision strategies. Moreover, spatial decomposition methods neglect their effect on the frequency domain, resulting in LODs that are not explicitly frequency band-limited. Our method does not suffer from these limitations, as we provide a *simple* training scheme directly providing a frequency decomposition.

Frequency decomposition by positional encoding. Tancik et al. [40] showed that mapping coordinates to Fourier features overcomes spectral bias in MLPs [34], facilitating learning of high-frequency details in neural fields. Further, Mildenhall et al. [29] proposed positional encoding, a special case of Fourier feature encoding, can greatly boosts reconstruction quality for novel view synthesis. Barron et al. [4] extended positional encoding to *integrated* positional encoding to enable *anti-aliased* multi-scale neural rendering. Hu et al. [16] show the effectiveness of integrated positional encoding when applied to the multi-scale hybrid tri-plane representation. These methods necessitate *explicit multi-scale supervision* through downsampling images [4, 44] and are primarily designed for filtering in 2D screen space —- for novel-view synthesis applications. Our approach in contrast has broader applicability, as it can be employed on any neural field, and eliminates the need for a downsampled supervision signal.

Frequency decomposition by network design. Neural fields can decompose the frequency of the signal by specific neural network design. Multiplicative filter networks (MFNs) [11] introduce an architecture that outputs a linear combination of sinusoidal bases, enabling explicit frequency control. BACON [23] utilizes a multi-layer MFN for signal reconstruction, providing frequency upper-bound estimates at each layer. Polynomial Neural Fields [46] refines this approach with a more complex architecture, facilitating precise frequency sub-band decomposition, and Shekarforoush et al. [36] enables progressive training of finer level-of-detail via the introduction of skip connections. These methods have an inherent limitation: due to their specialized network design, they are not applicable to mainstream *hybrid* neural field representations. Some other methods perform joint spatial and frequency decomposition by applying Fourier feature encodings on local grid features [10, 43]. Despite their hybrid representation, similarly to MFNs, these methods rely on *specialized* architectures. Our approach can be applied atop *any* neural field and *independently* of its underlying architecture, therefore allowing to maintain the advantages of (mainstream) *hybrid* neural fields, while adding the ability to perform frequency decomposition.

3. Method

Neural fields $f(\mathbf{x};\boldsymbol{\theta})$ represent signals with neural networks, by randomly sampling field locations $\mathbf{x}$, and optimizing the parameters $\boldsymbol{\theta}$ to reproduce a given ground truth value $f(\mathbf{x})$:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x}} \|f(\mathbf{x};\boldsymbol{\theta}) - f(\mathbf{x})\|_2^2 \tag{1}$$

Thanks to the commutativity of the Fourier transform and linear operators [31], we can represent *any* signal, as the superposition of band-limited signals:

$$f(\mathbf{x}) = f_0(\mathbf{x}) + \cdots + f_K(\mathbf{x}) \tag{2}$$

where the discrete set of frequencies is denoted as $\{\omega_k\}$, and $\omega_0{=}0$, $\omega_{K+1}{=}\infty$, and $f_k(\mathbf{x}){=}f_{[\omega_k,\omega_{k+1}]}(\mathbf{x})$ is the band-pass filtered $f(\mathbf{x})$, with frequency band $[\omega_k, \omega_{k+1}]$. In our work, we seek to train a neural field with the same characteristics, so that the field decomposes as:

$$f(\mathbf{x};\{\boldsymbol{\theta}_k\}) = f_0(\mathbf{x};\boldsymbol{\theta}_0) + \cdots + f_K(\mathbf{x};\boldsymbol{\theta}_K). \tag{3}$$

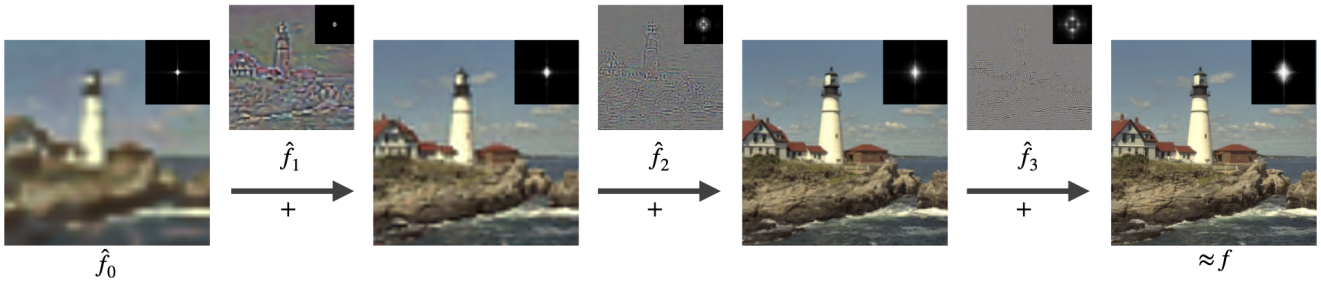


Figure 3. **Cascaded training** – We visualize our neural field decomposition where each level produces a signal with frequency band $[\boldsymbol{\omega}_i, \boldsymbol{\omega}_{i+1}]$. The decomposition is generated by a cascaded optimization akin to the generation of Laplacian pyramids in signal processing. Analogously, as the decomposition is quasi-orthogonal, we can generate the signal at any level through superposition. Image taken from the Kodak dataset [8].

Note that $\boldsymbol{\theta}_k$ *only* represents the fraction of the signal with spectra in the range $[\omega_k, \omega_{k+1}]$. Further, we will *not* assume a filtered version of the ground truth signal $f_k(\mathbf{x})$ to be readily available, as this is an unreasonable request for inverse problems; e.g. we cannot ask for a 3D representation to be filtered, as we only have 2D information about the 3D scene at hand. In theory, this could be achieved if a filtered version of the signal could be derived by convolving with a low-pass filter κ_{ω_k} with cut-off frequency ω_k:

$$f_{[0,\omega_k]}(\mathbf{x}) = \kappa_{\omega_k}(\mathbf{x}) \circledast f(\mathbf{x}) \tag{4}$$

from which we can obtain the desired *band-pass* signal: [2]

$$f_k(\mathbf{x}) = \kappa_{\omega_{k+1}}(\mathbf{x}) \circledast f(\mathbf{x}) - \kappa_{\omega_k}(\mathbf{x}) \circledast f(\mathbf{x}) \tag{5}$$

In what follows, we will first show that (4) can be achieved via *optimization* rather than explicit convolution (Sec. 3.1), and how (5) leads to a convenient cascaded training scheme (Sec. 3.2).

3.1. Filtering via optimization

Denote with $f[t]$ a discrete signal with uniform sampling of period T. From classical signal processing [31], such a discrete signal can be *reconstructed* as a continuous signal by applying a discrete-continuous convolution with a reconstruction kernel $\hat{\kappa}(\mathbf{x})$:

$$\hat{f}(\mathbf{x}) = \hat{\kappa}(\mathbf{x}) \circledast \sum_t f[t] \cdot \delta(\mathbf{x} - \mathbf{x}_t) \tag{6}$$

where $\{\mathbf{x}_t\}$ are the sampling locations. Via the Convolution Theorem, it can be shown that $\hat{f}(\mathbf{x})$ has frequency that is upper-bounded by $\omega \leq \frac{2\pi}{T}$, and use notation $\hat{f}_\omega(\mathbf{x})$ to indicate this upper bound.

We now replace the discrete signal $f[t]$ in (6) with a neural field sampled on a uniform lattice $\{\mathbf{x}_t\}$:

$$\hat{f}_\omega(\mathbf{x}; \boldsymbol{\theta}) = \kappa_\omega(\mathbf{x}) \circledast \sum_t f(\mathbf{x}_t; \boldsymbol{\theta}) \cdot \delta(\mathbf{x} - \mathbf{x}_t) \tag{7}$$

[2] Note the above resembles the filters employed to construct Laplacian pyramids via repeated applications of the so-called "mexican hat" kernel $\kappa_{\omega_{k+1}}(\mathbf{x}) - \kappa_{\omega_k}(\mathbf{x})$ [6].

We note that when $\kappa_\omega(\mathbf{x})$ is a triangle kernel $\Lambda_\omega(\mathbf{x})$ with local support equal to the period T, then the convolution above amounts to typical linear interpolation that is used to propagate features within voxels [39] that is found in many NeRF implementations [30].:

$$\hat{f}_\omega(\mathbf{x}; \boldsymbol{\theta}) = \Lambda_\omega(\mathbf{x}) \circledast \sum_t f(\mathbf{x}_t; \boldsymbol{\theta}) \cdot \delta(\mathbf{x} - \mathbf{x}_t) \tag{8}$$

$$\equiv \text{Interp}(\mathbf{x}; \{f(\mathbf{x}_t; \boldsymbol{\theta})\}) \tag{9}$$

Let us now incorporate (9) into the field training loop (1):

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x}} \|\hat{f}_\omega(\mathbf{x}; \boldsymbol{\theta}) - f(\mathbf{x})\|_2^2 \tag{10}$$

Under certain conditions (Appendix A), we can show that:

$$\hat{f}_\omega(\mathbf{x}; \boldsymbol{\theta}^*) \approx \kappa_\omega(\mathbf{x}) \circledast f(\mathbf{x}) \tag{11}$$

which shows that an optimized neural field $\hat{f}_\omega(\mathbf{x}; \boldsymbol{\theta}^*)$ can approximate a low-pass filtered signal $f(\mathbf{x})$ with frequency band $[0, \omega]$; see Figure 4.

We note that the choice of kernel $\kappa_\omega(\mathbf{x})$ gives us direct control over how a neural field behaves in the Fourier domain. Depending on the interpolation kernel chosen, we can opt for fast but less precise reconstruction using a *linear* kernel or slower yet highly accurate *sinc* interpolation; see Figure 5.

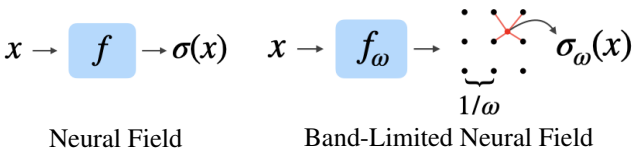


Figure 4. A standard neural field (left). Our band-limited neural field (right), which samples the outputs of a neural field f_w on a regular grid and interpolates to output a *band-limited* signal $\sigma_w(x)$. This enables direct control of the frequency bandwidth ω of the signal $\sigma_\omega(x)$.

	BANF-Linear			BANF-Sinc			BACON		
GT Filter	1/4×	1/2×	1×	1/4×	1/2×	1×	1/4×	1/2×	1×
Sinc	28.17	29.52	39.55	34.68	36.23	39.18	31.18	33.14	38.87
Linear	28.61	28.82		30.99	32.19		24.45	27.93	

Figure 5. **Interpolation Kernels** – By changing the interpolation kernel we can control the frequency cut in the Fourier domain.

3.2. Cascaded training

Recall our objective is to obtain a decomposition of the field like (3). With (10), we can optimize our first neural field $\hat{f}_0(\mathbf{x};\boldsymbol{\theta}_0)$ with frequency band $[0,\omega_1]$:

$$\boldsymbol{\theta}_0^* = \arg\min_{\boldsymbol{\theta}_0} \mathbb{E}_{\mathbf{x}} \|\hat{f}_0(\mathbf{x};\boldsymbol{\theta}_0) - f(\mathbf{x})\|_2^2 \tag{12}$$

We proceed with our next neural field $\hat{f}_1(\mathbf{x};\boldsymbol{\theta}_1)$ with frequency band $[\omega_1,\omega_2]$:

$$\boldsymbol{\theta}_1^* = \arg\min_{\boldsymbol{\theta}_1} \mathbb{E}_{\mathbf{x}} \|\hat{f}_1(\mathbf{x};\boldsymbol{\theta}_1) - (f(\mathbf{x}) - \hat{f}_0(\mathbf{x};\boldsymbol{\theta}_0^*))\|_2^2 \tag{13}$$

We note that $(f(\mathbf{x}) - \hat{f}_0(\mathbf{x};\boldsymbol{\theta}_0^*))$ is a high-pass filtered signal with frequency band $[\omega_1,+\infty]$, and as we have shown in (11), $\hat{f}_1(\mathbf{x};\boldsymbol{\theta}_1)$ will be a low-pass signal with frequency band $[0,\omega_2]$. Therefore, the optimization above will capture a representation whose frequency band $[\omega_1,\omega_2]$ is the intersection of these two ranges. The optimization scheme for subsequent levels can then be derived by induction:

$$\arg\min_{\boldsymbol{\theta}_k} \mathbb{E}_{\mathbf{x}} \|\hat{f}_k(\mathbf{x};\boldsymbol{\theta}_k) - (f(\mathbf{x}) - \Sigma_{l=0}^{k-1} \hat{f}_l(\mathbf{x};\boldsymbol{\theta}_l^*))\|_2^2 \tag{14}$$

Once all levels are trained, the signal can then be recomposed via superposition of frequency bands:

$$\hat{f}(\mathbf{x};\boldsymbol{\theta}^*) = \sum_k \hat{f}_k(\mathbf{x};\boldsymbol{\theta}_k^*) \approx f(\mathbf{x}) \tag{15}$$

This straightforward, yet powerful, idea can be applied to a variety of signal reconstruction tasks.

Implementation. To implement cascaded training, unless specified otherwise, we initialize all neural field parameters of all levels with random zero-mean weights of near-zero variance and zero bias. Further, the lowest resolution model $\hat{f}_0(\mathbf{x};\boldsymbol{\theta}_0)$ is trained gradually; following [15, 47], we initially train the model in a warm-up phase at $1/4\times$ and then $1/2\times$ of its intended final resolution. We empirically found that having this warm-up phase encourages smoothness and stabilizes optimization. We ablate our choice in supplementary

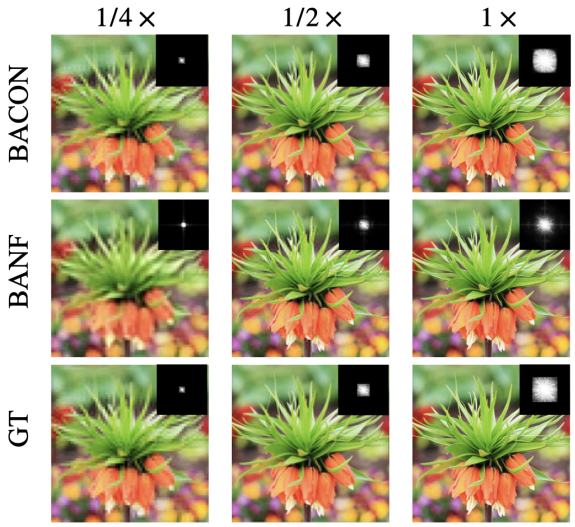


Figure 6. **Image fitting and filtering** – We visualize both multi-scale reconstruction and Fourier spectra of BANF compared against BACON and the ground truth image.

4. Experiments

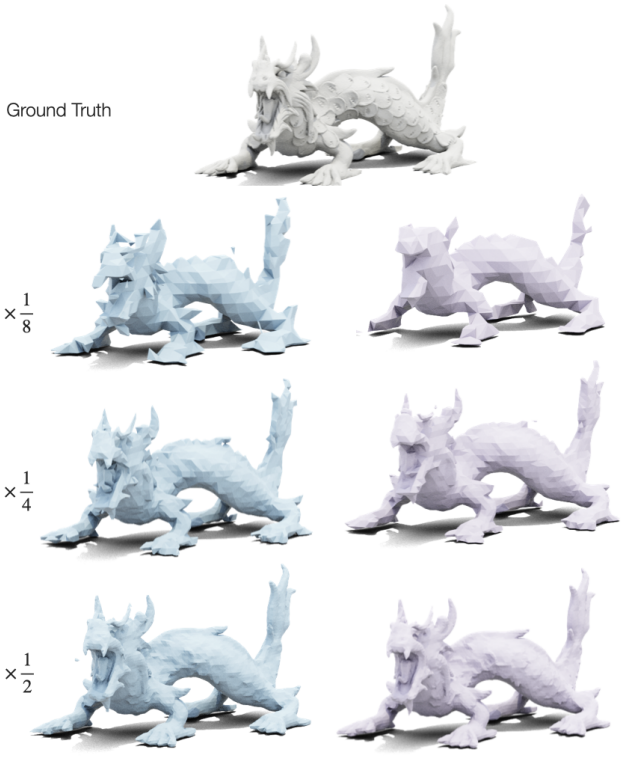
We demonstrate the efficacy of our method in frequency decomposition with three distinct applications: 2D image fitting (Sec. 4.1), 3D shape fitting with signed distance field supervision (Sec. 4.2), and 3D shape recovery from inverse rendering (Sec. 4.3). We then ablate our design choices in (Sec. 4.4). We utilize the linear interpolation kernel in all experiments unless otherwise specified.

4.1. Image filtering (2D)

We evaluate our method qualitatively by fitting a neural field to 2D images and reconstructing each image at four different frequency levels. We use an iNGP [30] backbone for images, with an MLP with 3 hidden layers, each with 32 neurons to convert deep features into color. The ground truth signal is a 2D image of resolution 256^2, for which we learn to represent it at resolution $r \in \{64^2, 128^2, 256^2\}$. We train the network with RMSProp [41] for 1K iterations with a learning rate of 2×10^{-3}. We use a batch size of 2^{16} randomly sampled points. We note that when sampling, we sample from a continuous 2D space with bilinear interpolation, as we are interested in the behavior of each method related to the underlying *continuous* signal. We use the mean squared error as the training objective.

We evaluate our method qualitatively on images from the DIV2K dataset [3], resampled to a resolution of 256^2 following [23]. We select BACON [23] as a representative baseline in 2D image decomposition using the Multiplicative Filter Network (MFN) [11] architecture. As a reference, we provide images decomposed directly with classical signal processing, which we refer to as ground truth.

As shown in Figure 6, our method successfully approx-



	Asian Dragon			Thai Statue		
method	32	64	128	32	64	128
iNGP	1.97	0.79	**0.32**	1.83	0.82	**0.38**
BANF	**1.23**	**0.65**	**0.32**	**1.41**	**0.68**	**0.40**

Figure 7. **Fitting SDFs** – We show qualitative results and report the Chamfer-L2 distance ($\times 10^2$) $\downarrow$. By filtering the signal during training, our method implements anti-aliasing, so that the extracted signal at coarser scales approximates better.

imates low-pass filtering, being similar to the one provided by BACON as well as the ground truth. We note that as we use linear interpolation, we do not obtain exact low-pass filtering, but rather an approximation (The Fourier transform of a triangle kernel is $sinc^2$, which is only an approximate low-pass filter). Yet, as shown, it approximates well and leads to a meaningful frequency decomposition of the signal. Also, note that there is visible Gibbs ringing in both BACON and the ground truth at the lower scales. Our results, on the other hand, while faithfully representing the details at lower frequency, do not suffer from these artifacts.

In Figure 5, we demonstrate the adaptability of our method to various interpolation kernels. We assess our method's performance using *sinc* (of order 6 with box window) and *linear* interpolation kernels with circular interpolation support, resulting in sharp frequency cut and alias-free results respectively. Additionally, we quantitatively compare the reconstructed output and ground truth filtered using either *linear* and *sinc* filters.

4.2. Signed Distance Fields (3D)

We now evaluate our method for learning Signed Distance Fields (SDF) at various levels of detail through *direct* 3D supervision. We assume that exact meshes are unavailable, and only implicit signed distances are, which is commonly the case for real-world captures. We demonstrate that our method can directly decompose frequencies.

Dataset. We evaluate with two commonly used meshes "Asian Dragon" and "Thai Statue". Both have intricate details that are of high frequency, making the SDF fitting task and, more specifically frequency decomposition, challenging. We first normalize the meshes to fit into a unit sphere. We then sample 500k points to get their respective SDF values, with 40% of the points on the surface, 40% near the surface, and the rest uniformly within the bounding box. We use the same points to train all models, and optimize via mean square error for the predicted SDF values.

Baseline. We compare our method against iNGP. We use a multi-resolution hash grid of resolutions 16 to 2048, for 100k iterations with a batch size of 10k points until full convergence. We then extract the meshes at the same resolutions as before using marching cubes [25]. We evaluate each method both qualitatively and quantitatively via Chamfer-L2 distance.

Implementation. We use the Adam optimizer [19] with a learning rate of 10^{-3} and the default training settings provided in iNGP. We learn the signal at four different resolutions $r \in \{32^3, 64^3, 128^3, 256^3\}$. As in Wang et al. [42], the coarsest level's MLP is specifically initialized to produce a sphere for stable training. We train with a batch size of 100k points. We train each level for 10k iterations, except for the coarsest level, which we found to converge already at 5k iterations.

Discussions. In Figure 7, we show quantitatively and qualitatively that our method better respects the Nyquist sampling theorem [31] due to the filtering that our method performs, resulting in higher quality approximations at coarse resolutions. As the resolution increases, the marching cubes sampling frequency exceeds the Nyquist rate, thus, as expected, both our method and vanilla iNGP achieve similar reconstruction quality.

4.3. Inverse rendering (3D from 2D)

We demonstrate the applicability of our method in a more complex scenario where only 2D images of a 3D signal are available. Here, we tackle the inverse problem of 3D surface reconstruction at *multiple* levels of detail. Note that as a uniformly sampled 3D signal is not available (only 2D observations are available) Fourier analysis cannot be used to create supervision.

Dataset and baseline. For the baseline we use vanilla

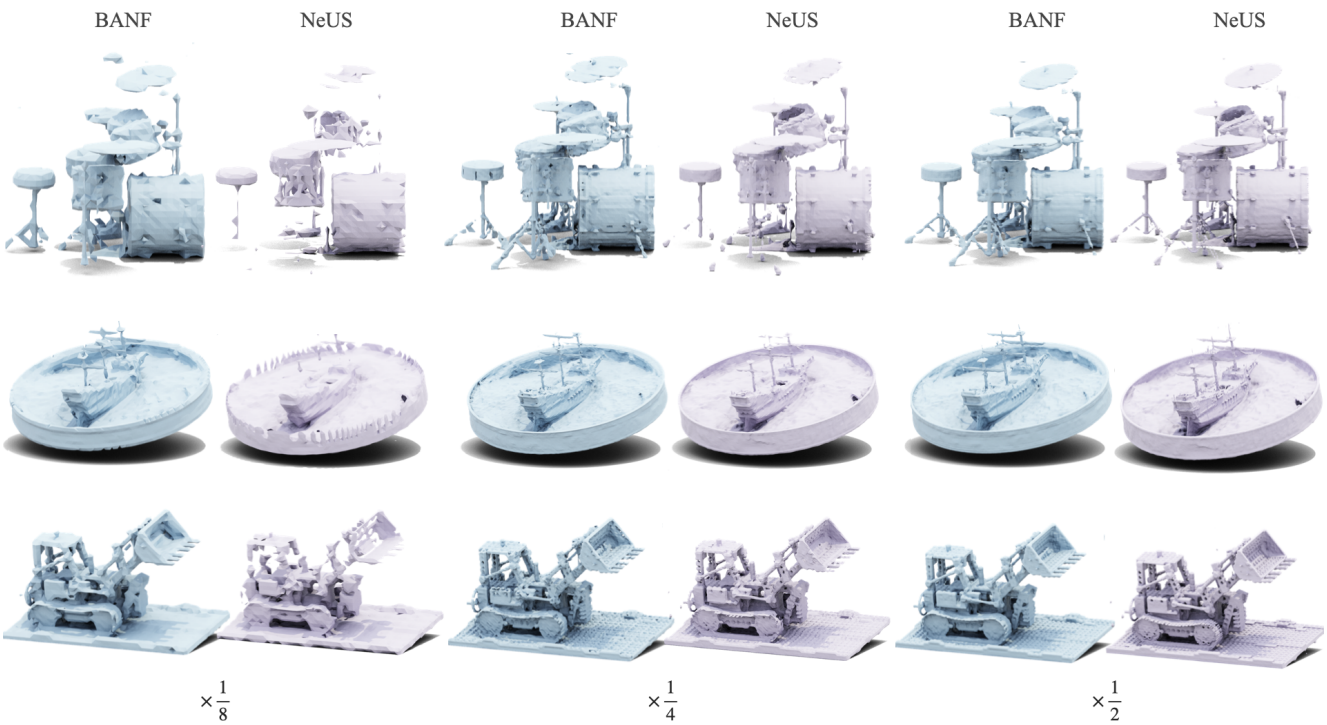


Figure 8. **Inverse rendering –** We integrate our technique into NeUS [42] for level-of-detail surface reconstruction. We show qualitative results on on the Synthetic NeRF Dataset [4]. Note how, *especially* at coarser scales, the baseline lacks approximation of fine-grained details (caused by aliasing, as the baseline lacks filtering). Quantitative results are provided in Table 1.

Scale	Method	Chair	Drums	Ficus	Hotdog	Lego	Material	Mic	Ship	Average
1/8x	NeUS	**6.05**	21.8	40.3	**8.82**	10.2	24.5	10.5	16.4	17.3
	BANF	7.79	**14.4**	**11.1**	15.0	**9.36**	**11.6**	**10.2**	**11.6**	**11.4**
1/4x	NeUS	**4.09**	11.41	18.9	**7.54**	**5.89**	17.0	**5.60**	13.2	10.5
	BANF	6.29	**8.66**	**7.16**	10.2	6.92	**9.44**	7.53	**9.32**	**8.19**
1/2x	NeUS	**3.94**	9.03	8.06	**7.17**	**4.74**	14.7	**4.32**	12.1	8.01
	BANF	5.92	**8.43**	**5.78**	10.2	6.89	**8.90**	6.66	**9.15**	**7.74**
1x	NeUS	**4.08**	8.45	5.75	**7.12**	**4.48**	13.7	**4.30**	11.8	**7.45**
	BANF	5.82	**8.31**	**5.36**	10.0	6.69	**8.68**	5.79	**8.95**	**7.45**

Table 1. **Inverse rendering** – Quantitative results for NeUS [42] LOD reconstruction on the Synthetic NeRF Dataset [4]. See qualitative results in Figure 8. Metric is Chamfer Distance ($\times 10^{-2}$) $\downarrow$.

NeUS [42] followed by marching cubes [25] to extract meshes. We evaluate our method on the (Synthetic) NeRF dataset [4] and the (real) MobileBrick dataset [21]. For MobileBrick, for both our method and NeUS, we use mask supervision and train without background modeling. We evaluate each method both qualitative, and quantitatively via the Chamfer-L2 distance.

Implementation. We extend the surface reconstruction baseline provided by NeUS [42] with our method. Following NeUS, we optimize the 3D surface field using volume rendering, guided by supervision from 2D images. Specifically, given a 3D position $\mathbf{x}\in\mathbb{R}^3$ and a viewing direction $\mathbf{v}\in\mathbb{S}^2$, we train neural fields for both signed distance $\hat{s}(\mathbf{x};\boldsymbol{\theta})$ and color $\hat{c}(\mathbf{x},\mathbf{v};\boldsymbol{\theta})$. We apply cascaded training (Sec. 3.2) only to the SDF MLP and directly train separate color MLPs at each level – this is due to the challenge in color MLP optimization with signal decomposition (details in Sec. 4.4). We use an iNGP [30] backbone with an MLP that predicts SDF values and a feature vector. The feature vector is then fed into the color MLP to output RGB values. We supervise training via classical photometric reconstruction error, along with Eikonal and Laplacian regularizers. When implementing our method, we observe a performance degradation when the color MLP relies solely on the output feature vector of the decomposed SDF MLP (see Sec. 4.4 for details). We thus additionally concatenate the features from the hashgrid to the inputs of the color MLP. We note that this minor architectural change does not lead to any significant changes in the performance of the baseline NeUS.

Scale	Method	Ben	Boat	Bridge	Cabin	Camera	Castle	Colos	Jeep	Bus	Motor	Satel	Shuttle	Avg.
1/4x	NeUS	20.4	59.5	9.25	5.52	5.40	26.6	15.5	3.67	10.0	3.54	7.08	19.3	15.5
	BANF	**10.0**	**6.40**	**4.61**	**4.64**	**2.78**	**17.4**	**12.9**	**2.96**	**5.48**	**2.73**	**5.27**	**13.5**	**7.39**
1/2x	NeUS	8.04	6.15	5.01	**3.95**	3.94	11.4	**9.90**	2.60	5.66	**2.15**	**4.17**	6.78	5.81
	BANF	**7.13**	**4.21**	**4.63**	4.42	**2.76**	**10.7**	10.0	**2.58**	**5.18**	2.37	4.56	**5.93**	**5.38**

Table 2. **Inverse rendering –** We show qualitative results on the MobileBrick dataset [21]. Metric is Chamfer Distance ($\times 10^{-3}$) $\downarrow$. Again, larger gains are observable at coarser scales, when signals are sampled at a frequency that is below the one required by Nyquist.

As done in Sec. 4.2, the coarsest SDF MLP head is initialized to produce a sphere. For initializing the color MLP heads, we first initialize the coarsest level randomly without any special treatment. For the subsequent levels, we bootstrap their training by initializing them with the previous level's (converged) weights. Our models are trained at four different resolutions $r \in \{64^3, 128^3, 256^3, 512^3\}$, trained in cascade for 20k iterations, with a batch size of 5k rays. To compute gradients of SDF that are used both as an input for the color MLP, and for computing the Eikonal and Laplacian regularizers, we use finite differences with the delta value set to $1/(4 \times r)$ for each level of our method, and $1/1024$ for the baseline model.

Discussions. As shown in Tables 1 and 2, our method significantly outperforms NeUS on average. In Figure 8, we visually see improvements on thin structures (see drum stool legs, ship's mast, and loader's bucket). We further observe improvement in anti-aliasing on the ship's rim.

4.4. Ablations

Other feature grid representations.. Our method is agnostic to the underlying representation. We test 3D reconstruction with grid and MLP-backed representations on 4 objects from Stanford dataset (we follow BACON, and use their preprocessed data and sampling scheme). Note how the representation *does not* heavily influence reconstruction quality (Chamfer-L2 $\downarrow \cdot 10^{-2}$):

HashGrid			Dense			MLP		
32	64	128	32	64	128	32	64	128
1.35	0.82	0.54	1.38	0.83	0.55	1.37	0.79	0.58

5. Conclusions

We introduce a way to train neural fields that enables frequency decomposition of the represented signals. Departing from heuristic- or architecture-based approaches, we realize this by a simple modification to the training process. Our approach is versatile, accommodating various neural field architectures (both fully neural and hybrid representations), and makes *no assumptions* about the training data, such as the availability of pre-filtered signals. We demonstrate its applicability across a number of neural fields workloads, and in particular by testing its effectiveness for anti-aliased level-of-detail reconstruction.

Limitations. In this work, we focused on uniformly sampled signals, and it would be interesting how to extend the method to "contracted" representations that are commonly used in NeRF to deal with unbounded signals. Additionally, in our current implementation, the method becomes highly memory-intensive at higher grid resolutions. This issue could potentially be addressed through the adoption of more efficient querying strategies and the CUDA-based implementation.

A. Appendix

Without loss of generalization to higher dimensions, let us consider 1D signals. Let us start by replacing $\mathbb{E}_{\mathbf{x}}$ in (10) with a sum operator, hence reducing our stochastic gradient descent optimization with least square optimization. Samples $\{\mathbf{x}_n\}$ are drawn uniformly with a period sufficient to satisfy the Nyquist theorem for the ground truth signal f:

$$\arg\min_{\boldsymbol{\theta}} \sum_{\{\mathbf{x}_n\}} \|\text{Interp}(\mathbf{x}_n; \{f(\mathbf{x}_t; \boldsymbol{\theta})\}) - f(\mathbf{x}_n)\|_2^2 \quad (16)$$

Let us rewrite this expression in matrix form by denoting $\{f(\mathbf{x}_n)\}$ as $\mathbf{b} \in \mathbb{R}^{N \times D}$, $\{f(\mathbf{x}_n; \boldsymbol{\theta})\}$ as $\mathbf{X} \in \mathbb{R}^{T \times D}$, and by storing in the *skinny* matrix $\mathbf{A} \in \mathbb{R}^{N \times T}$ the linear interpolation coefficients corresponding to the positions $\{\mathbf{x}_n\}$:

$$\arg\min_{\mathbf{X}} \|\mathbf{A}\mathbf{X} - \mathbf{b}\|_2^2 \quad (17)$$

Assuming the matrix $\mathbf{A}$ is invertible, the optimization above provides the closest point projection of $\mathbf{b}$ onto the range of $\mathbf{A}$ [5, Sec. 5-3]. Recalling $\mathbf{A}$ represents functions whose frequency is upper-bounded by ω, this implies the optimization projects our true signal onto the closest function with bounded spectra, which is equivalent to convolving the signal with a low-pass filter as in (11). Our approximations stem from the fact that *linear* interpolation is only an *approximation* of a low-pass filter (with a sinc^2 frequency spectra), and that our optimization is a stochastic gradient descent rather than a close-form solve.

Acknowledgments. We would like to thank David Fleet, David Lindell, Shayan Shekarforoush for their valuable feedback. In addition, we are grateful to the BIRS workshop on "generative 3D modeling" that made this discussion possible. Andrea Tagliasacchi is supported by the NSERC discovery grant [2023-05617], and by the SFU Visual Computing Research Chair.

References

[1] Michal Adamkiewicz, Timothy Chen, Adam Caccavale, Rachel Gardner, Preston Culbertson, Jeannette Bohg, and Mac Schwager. Vision-only robot navigation in a neural radiance world. *CoRR*, 2021. 1

[2] 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

[3] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. *CVPRW*, 2017. 5

[4] 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. *ICCV*, 2021. 3, 7

[5] Stephen Boyd. Introduction to linear dynamical systems (course reader for ee263). https://ee263.stanford.edu/archive/263lectures_slides.pdf, 2007. 8

[6] P. Burt and E. Adelson. The laplacian pyramid as a compact image code. *TCOM*, 1983. 4

[7] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. *ECCV*, 2022. 2

[8] Eastman Kodak Company. Kodak lossless true color image. http://r0k.us/graphics/kodak/, 1999. 4

[9] Boyang Deng, JP Lewis, Timothy Jeruzalski, Gerard Pons-Moll, Geoffrey Hinton, Mohammad Norouzi, and Andrea Tagliasacchi. Neural articulated shape approximation. *ECCV*, 2020. 2

[10] Yishun Dou, Zhong Zheng, Qiaoqiao Jin, and Bingbing Ni. Multiplicative fourier level of detail. *CVPR*, 2023. 2, 3

[11] Rizal Fathony, Anit Kumar Sahu, Devin Willmott, and J Zico Kolter. Multiplicative filter networks. *ICLR*, 2021. 2, 3, 5

[12] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. *CVPR*, 2022. 2

[13] Lily Goli, Daniel Rebain, Sara Sabour, Animesh Garg, and Andrea Tagliasacchi. nerf2nerf: Pairwise registration of neural radiance fields. *ICRA*, 2023. 2

[14] Ayaan Haque, Matthew Tancik, Alexei Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. *CVPR*, 2023. 1

[15] Amir Hertz, Or Perel, Raja Giryes, Olga Sorkine-Hornung, and Daniel Cohen-Or. Sape: Spatially-adaptive progressive encoding for neural optimization. *NeurIPS*, 2021. 5

[16] Wenbo Hu, Yuling Wang, Lin Ma, Bangbang Yang, Lin Gao, Xiao Liu, and Yuewen Ma. Tri-miprf: Tri-mip representation for efficient anti-aliasing neural radiance fields. *ICCV*, 2023. 3

[17] Anil K Jain. *Fundamentals of digital image processing*. Prentice-Hall, Inc., 1989. 2

[18] Animesh Karnewar, Tobias Ritschel, Oliver Wang, and Niloy J. Mitra. Relu fields: The little non-linearity that could. *SIGGRAPH*. 2

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015. 6

[20] Abhijit Kundu, Kyle Genova, Xiaoqi Yin, Alireza Fathi, Caroline Pantofaru, Leonidas Guibas, Andrea Tagliasacchi, Frank Dellaert, and Thomas Funkhouser. Panoptic Neural Fields: A Semantic Object-Aware Neural Scene Representation. *CVPR*, 2022. 1

[21] Kejie Li, Jia-Wang Bian, Robert Castle, Philip H.S. Torr, and Victor Adrian Prisacariu. Mobilebrick: Building lego for 3d reconstruction on mobile devices. *CVPR*, 2023. 7, 8

[22] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. *CVPR*, 2023. 1, 2

[23] 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, 5

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

[25] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM*, 1987. 1, 2, 6, 7

[26] David Luebke, Martin Reddy, Jonathan D. Cohen, Amitabh Varshney, Benjamin Watson, and Robert Huebner. *Level of Detail for 3D Graphics*. Morgan Kaufmann Publishers Inc., 2002. 1

[27] 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. *SIGGRAPH*, 2021. 1, 2

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

[29] 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. *ECCV*, 2020. 1, 2, 3

[30] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *SIGGRAPH*, 2022. 2, 4, 5, 7

[31] Sophocles J Orfanidis. *Introduction to Signal Processing*. Prentice-Hall, Inc., 1988. 2, 3, 4, 6

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

[33] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *ICLR*, 2023. 1

[34] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. *ICML*, 2019. 3

[35] Vishwanath Saragadam, Jasper Tan, Guha Balakrishnan, Richard G. Baraniuk, and Ashok Veeraraghavan. Miner: Multiscale implicit neural representations. *ECCV*, 2022. 2

[36] Shayan Shekarforoush, David B Lindell, David J Fleet, and Marcus A Brubaker. Residual multiplicative filter networks for multiscale reconstruction. *NeurIPS*, 2022. 3

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

[38] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. *CVPR*, 2022. 2

[39] Towaki Takikawa, Joey Litalien, Kangxue Yin, Karsten Kreis, Charles Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Neural geometric level of detail: Real-time rendering with implicit 3D shapes. *CVPR*, 2021. 2, 4

[40] 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. *NeurIPS*, 2020. 3

[41] Geoffrey Hinton Tijmen Tieleman. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 2012. 5

[42] 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. *NeurIPS*, 2021. 1, 2, 6, 7

[43] Zhijie Wu, Yuhe Jin, and Kwang Moo Yi. Neural fourier filter bank. *CVPR*, 2023. 2, 3

[44] Yuanbo Xiangli, Linning Xu, Xingang Pan, Nanxuan Zhao, Anyi Rao, Christian Theobalt, Bo Dai, and Dahua Lin. Bungeenerf: Progressive neural radiance field for extreme multi-scale scene rendering. *ECCV*, 2022. 3

[45] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. *CGF*, 2022. 2

[46] Guandao Yang, Sagie Benaim, Varun Jampani, Kyle Genova, Jonathan T. Barron, Thomas Funkhouser, Bharath Hariharan, and Serge Belongie. Polynomial neural fields for subband decomposition and manipulation. *NeurIPS*, 2022. 2, 3

[47] Jiawei Yang, Marco Pavone, and Yue Wang. Freenerf: Improving few-shot neural rendering with free frequency regularization. *CVPR*, 2023. 5

[48] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *NeurIPS*, 2021. 1

[49] Zhuang Yiyu, Zhang Qi, Feng Ying, Zhu Hao, Yao Yao, Li Xiaoyu, Cao Yan-Pei, Shan Ying, and Cao Xun. Anti-aliased neural implicit surfaces with encoding level of detail. *SIGGRAPH*, 2023. 2