PREF: Phasorial Embedding Fields for Compact Neural Representations

Binbin Huang **Xinhao Yan** **Anpei Chen** **Shenghua Gao*** **Jingyi Yu ***

ShanghaiTech University
{huangbb, yanxh, chenap, gaoshh, yujingyi}@shanghaitech.edu.cn

Abstract

We present a phasorial embedding field *PREF* as a compact representation to facilitate neural signal modeling and reconstruction tasks. Pure multi-layer perceptron (MLP) based neural techniques are biased towards low frequency signals and have relied on deep layers or Fourier encoding to avoid losing details. PREF instead employs a compact and physically explainable encoding field based on the phasor formulation of the Fourier embedding space. We conduct a comprehensive theoretical analysis to demonstrate the advantages of PREF over the latest spatial embedding techniques. We then develop a highly efficient frequency learning framework using an approximated inverse Fourier transform scheme for PREF along with a novel Parseval regularizer. Extensive experiments show our compact PREF-based neural signal processing technique is on par with the state-of-the-art in 2D image completion, 3D SDF surface regression, and 5D radiance field reconstruction.

1 Introduction

Coordinate-based neural representations have emerged as a compelling alternative for modeling and processing signals (e.g., sound, image, video, geometry, etc.). Instead of using discrete primitives such as pixels or vertices, coordinate-based multi-layer perceptrons (MLPs) represent graphics images [25, 11, 7], shapes [1, 9, 24, 33], or even radiance fields [4, 18, 23, 26, 28, 32, 51] in terms of continuous functions that are memory efficient and amenable for solving inverse reconstruction problems via training. Earlier coordinate-based MLPs tend to be biassed towards low frequencies, whereas more recent implicit neural approaches have adopted a sinusoidal representation for better recovering high frequencies, by either transforming the input coordinates to the Fourier basis [26, 52, 50], or encoding the sinusoidal nonlinearity via deeper MLP architectures [37, 7].

Although effective in countering high frequency losses, such approaches are inefficient in training or inference, inherent to the MLP-based optimization strategy: brute-force mappings from low-dimension, low-frequency inputs onto high frequency target functions do not sufficiently consider the underlying characteristics of the mapping function. As a result, they have largely resorted to large MLPs (either wide in hidden dimensions or deep in layers) to reliably learn the corresponding function mapping. The downside of large MLPs is the long training time and slow inference speed.

To accelerate training, embedding-based MLPs [48, 39, 28, 3] have adopted an inverse optimization strategy. They jointly search for an optimal mapping network and the optimal inputs (i.e., the high dimension embedding volume). By replacing the input field with a high-dimension high-frequency embedding volume (whose size is generally determined by the volume resolution and kernel size),

*Corresponding authors.

Preprint. Under review.

they manage to bridge the gap between the low-dimension low-frequency coordinate inputs and the high-frequency outputs with a much smaller MLP in both width and depth. Successes of these approaches are illustrated by their accelerations by orders of magnitudes in both training and rendering [3, 28, 39, 48, 49].

However, as their embedding space is also represented in discrete forms, state-of-the-art techniques have relied on interpolations for querying high-dimensional embedded features. To maintain high efficiency, the most adopted schemes are still linear interpolation. Yet they present several major limitations:

- The highest recoverable frequency of the embedding fields is determined by the resolution of the volume. To preserve high frequencies, it is critical to discretize the volume at a very fine level. Fully frequency preservation hence requires demand extremely high memory consumption, prohibitive on even the most advanced GPUs.
- From a signal processing perspective, linear interpolation within the embedding volume not only leads to aliasing but also causes higher-order derivatives to vanish, hindering back-propagation and the overall convergence. Figure 1 shows a typical example.
- Spatially discretized embedding, compared to its frequency dual, provides limited insights on the signal. In particular, despite a large variety of Fourier signal processing tools for both editing and stylization, few are directly applicable to spatial embedding.

To address these limitations, we present PREF, a novel PhasoRial Embedding Field, that represents the Fourier transformed embedding with compact, complex-numbered phasors. We derive a comprehensive framework using PREF to approximate the Fourier embedding space so that each spatial coordinate's feature can be represented in terms of the sinusoidal features within a discrete phasor volume. PREF presents several key advantages over previous spatial embedding techniques. First, the nonlinear transform nature of the phasor avoids the vanishing derivative problem in linearly interpolated space embedding. Second, PREF is highly compact and does not require using high resolution volumes to preserve high frequency. In fact, it facilitates easy queries of the specific frequency of the input embedding. Finally, based on Fourier transforms, PREF manages to exploit many existing Fourier signal processing techniques such as differentiation, convolution, and Parseval's theorem to conduct traditionally expensive operations for the MLP. Overall, PREF provides a new embedding that benefits direct and inverse neural reconstructions on tasks ranging from 2D image completion to 3D point cloud processing and the 5D neural radiance field reconstruction.

To summarize, the contributions of our work include:

- A compact and physically explainable encoding field based on the phasor formulation called PREF. We conduct a comprehensive theoretical analysis and demonstrate the advantages of PREF over previous spatial embedding techniques.
- We show that PREF provides a robust and compact solution to MLP-based signal processing. It is compact and at the same time effectively preserves high frequency components and therefore details in signal reconstruction.
- We develop a highly efficient frequency learning framework using an approximated inverse Fourier transform scheme along with a novel Parseval regularizer. Comprehensive experiments demonstrate that PREF outperforms the state-of-the-art techniques in various signal reconstruction tasks in both accuracy and robustness.

2 Related Work

Our PREF framework is in line with renewed interest on adopting implicit neural networks to represent continuous signals from low dimensional input. In computer vision and graphics, they include 2D images with pixels [6], 3D surfaces in the form of occupancy fields [24, 29, 33] or signed distance fields (SDFs) [1, 9, 42, 47, 21, 40, 46, 43] , concrete 3D volumes with a density field [13, 34], 4D light fields [17, 44] and 5D plenoptic functions for the radiance fields [26, 51, 30, 19, 32, 47].

Encoding vs. Embedding.

To learn high frequencies, state-of-the-art implicit neural networks have adopted the Fourier encoding scheme by transforming the coordinates by periodic $\sin(x)$ and $\cos(x)$ functions or under Euler's

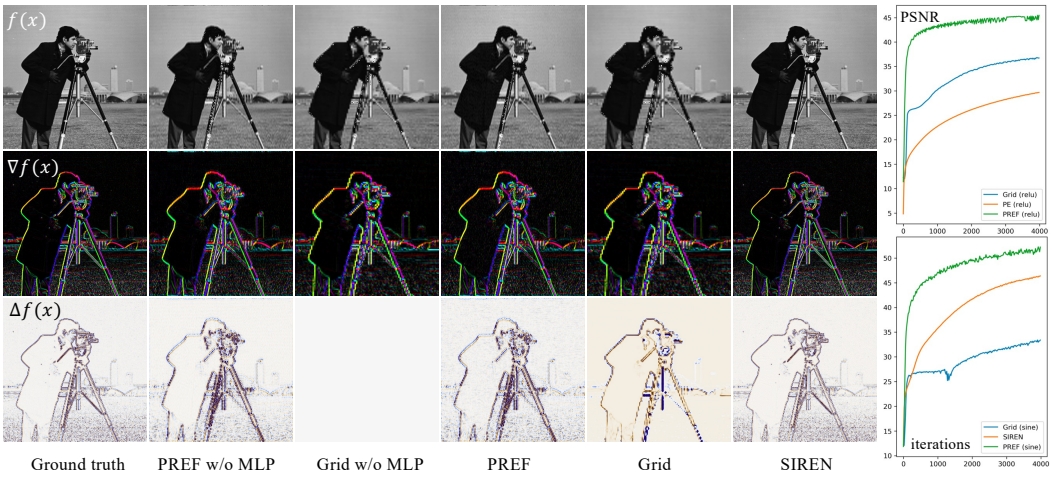


Figure 1: Comparison of spatial encoding and frequency encoding fitting a ground truth image (top left). We show that PREF fits the image well, correctly reflecting its first- and second-order derivatives. On the left plot, we show that PREF outperform explict (grid) representation and state-of-the art implicit neural networks under different activations.

formula e^{ix}. Under Fourier encoding, feature optimization through MLPs can be mapped to optimizing complex-valued matrices with complex-valued inputs in the linear layer. For example, Position Encoding (PE) and Fourier Feature Maps (FFM) both transform spatial coordinates to Fourier basis at earlier input layers [22, 41, 52] whereas SIREN [37] embeds the process in the deeper layers by using periodic activation functions.

Improving training and inference efficiency of MLP-based networks has also been explored from the embedding perspective with smart data structures. Various schemes [35, 10, 49, 48, 39, 28, 3] replace the deep MLP architecture with voxel-based representations, to trade memory for speed. Early approaches bake an MLP to an Octree along with a kilo sub-NeRF or 3D texture atlas for real-time rendering [35, 10]. These approaches rely on a pre-trained or half-trained MLP as prior and therefore still incur long per-scene training. Planoxel [48] and DVGO [39] directly optimize the density values on discretized voxels and employ an auxiliary shading function represented by either spherical harmonics (SH) or a shallow MLP network to account for view dependency. They achieve orders of magnitude acceleration over the original NeRF on training but incur very large memory footprint by storing per-voxel features. However, over parametrization can easily lead to noisy density estimation and subsequently inaccurate surface estimations and rendering. The seminal work of Instant-NGP [28] spatially groups features via embedding with a hash table to achieve unprecedentedly fast training. In a similar vein, TensoRF [5] employs highly efficient tensor decomposition via vertical projections.

It is worth noting that embedding-based techniques share many similarities to neural inversion [12] that aims to jointly optimize the inputs and network weights. In a nutshell, different from traditional feed-forward neural network optimization that attempts to refine network weights, neural inversion seeks to find inputs, often non-unique, that will produce the desired output response under the fixed set of weights. Classic examples include latent embedding optimization [36] in machine learning and GAN inversions [15, 45, 14, 15] that finds the optimal latent code to best match the target image for subsequent editing or stylization. In the context of embedding, the focus is to maintain a learnable high-order input embedding and then compute the feature via interpolation in the embedding space via schemes as simple as the nearest neighbor but no more complicated than linear interpolation. While efficiency in feature querying, they share a common limitation of *vanishing gradient*, that is, the piecewise constant first-order derivative and zero second-order derivative due to linear interpolation, leading to higher than usual errors than brute-force pure MLP implementations. We show that our PREF representation is both compact and effective in preserving high frequency details by overcoming the hurdle of vanishing derivatives.

3 Background and Notations

Our goal is to fit a continuous function $\Phi(\mathrm{x})$ parameterized with low dimensional inputs $\mathrm{x} \in \mathbb{R}^n$. Let $\mathcal{M}(;\theta)$ be an MLP with parameter θ, $\mathbf{P} \in \mathbb{C}^{k \times N^d}$ be a complex-valued (phasor) volume of dimensionality $n+1$. $\widetilde{\mathcal{T}}$ represents the inverse Fourier Transform. We use the MLP to approximate as $\Phi(\mathrm{x})$ as $\mathcal{M}(f(\mathrm{x});\theta)$, where $f(\mathrm{x}) \in \mathbb{R}^k$ can be computed as $f(\mathrm{x}) = \widetilde{\mathcal{T}}(\mathbf{P};\mathrm{x})$.

Our phasorial embedding field (PREF) resembles existing spatial embedding. However, spatial embedding uses a real-value volume rather than $\mathbf{P}$ and employs local interpolation or hash functions in place for $\widetilde{\mathcal{T}}$. Similar to spatial embedding though, PREF can also handles any continuous fields by the spatially-embedded MLPs, e.g., signed distance fields (SDFs) [40, 42, 47] or radiance fields (RFs) [5, 39, 48, 26]. In fact, as a frequency-based alternative, PREF explicitly associates frequencies with features: each volume entry represents the Fourier coefficients under the corresponding frequency.

Next, we show that under PREF, many neat properties of the Fourier transform translate to the complex phasor volume, facilitating much more efficient optimization and accessible manipulation. For simplicity, we carry out our derivations in 2D while high dimensional extensions can be similarly derived as shown in various applications.

We first present the inverse Fourier transform, and along the way, present the associate theorems, which we utilize to optimize our phasorial representation. Before proceeding, we explain our notation. Let $F(\cdot), F[\cdot]$ be the continue and discrete Fourier series, and $\mathbf{P}$ is a phasor volume that translates to phasorial embedding field $f(\mathrm{x})$.

Inverse Fourier Transform. Let $f(x, y)$ be a 2D continuous band-limited signal, its discrete inverse Fourier transform $\widetilde{\mathcal{T}}$ factorizes the signal into a Fourier series $u \in \mathbb{N}, v \in \mathbb{N}$ with corresponding coefficients:

$$f(x,y) = \widetilde{\mathcal{T}}(F) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F[u,v] e^{j2\pi(u\frac{x}{M} + v\frac{y}{N})} \tag{1}$$

Recall that $F[u,v] \in \mathbb{R}^{M \times N}$ corresponds to an equally-spaced matrix where each entry is a phasor with real and imaginary part of the corresponding frequency as:

$$F[u,v] = F_R[u,v] + jF_I[u,v] \tag{2}$$

The resulting phasor volume $\mathbf{P}$ can be viewed as a multi-channel version of the Fourier map F. Therefore, $\mathbf{P}$ inherits several nice properties of the Fourier transform, which is efficient for frequency-domain manipulations.

Theorem 1. *Let $F(u,v)$ be an absolutely continuous differentiable function, and $f(x,y)$ be its inverse Fourier transform, we have*

$$\frac{\partial f^n(x,y)}{\partial x^n} = \widetilde{\mathcal{T}}((j2\pi u)^n F[u,v]) \tag{3}$$

Theorem 2. *Let $F(u,v)$ be absolutely continuous differentiable function, and $f(x,y)$ be its inverse Fourier transform, we have*

$$\iint \|f(x,y)\|^2 dxdy = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} \|F(u,v)\|^2. \tag{4}$$

Beyond these, modeling a signal in a Fourier domain provides various unique properties; for example, the circular convolution theorem indicates that a convolution of two sequences can be obtained as the inverse transform of the product of the individual transforms.

A n$\ddot{a}$ive solution is to represent the signals based on the standard inverse Fourier transform and a dense phasorial volume grid describing the Fourier coefficients, then jointly update the volume grid and MLP with gradient descent. However, such a solution is inefficient since both model size and computation increase in $\mathcal{O}(N^3)$, where N is the resolution of the Fourier series. We next introduce a novel modeling, phasorial encoding field, for resolving the limitation in the n$\ddot{a}$ive solution.

4 Phasorial Embedding Fields

Our PREF is a de facto continuous feature field $f(\mathrm{x})$ transformed from a multi-channel multi-dimensional square Fourier (phasor) volume $\mathbf{P}$ (*e.g.*, $\mathbb{C}^{k\times N^2}$ for 2D tasks). In a nutshell, PREF employs the Fourier transform in Eq. 1 to map spatial coordinate x into k-channel feature $f(\mathrm{x}) \in \mathbb{R}^k$. The mapped results can be fed into an MLP to process task-specific fields $\Phi(\mathrm{x})$ such as SDFs or RFs. Figure 2 illustrate this. As each channel is independent, PREF can transform the spatial coordinates and update the phasor volume in parallel. Therefore, we simply omit k in the rest of the derivations for clarity. We start by defining a phasor volume of PREF and subsequently discuss how to efficiently extract features from PREF and then optimize the volume.

4.1 Phasor Volume Decomposition

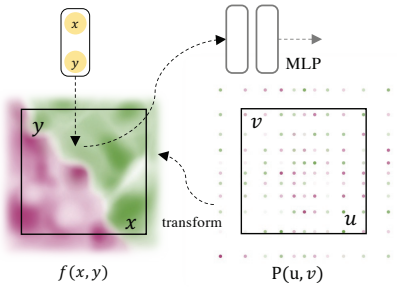


Figure 2: Pipeline of our PREF.

Brute-force representation of PREF with full frequencies is clearly too expensive in both computation and memory. Because many natural signals are band-limited and to reduce complexity, we instead use a sparse set of frequencies to encode PREF. The process is equivalent to selectively marking a large portion of the entries in matrix $F(u, v)$ as zero. In particular, we set out to factorize $\mathbf{P}[u, v] \in \mathbb{C}^{N\times N}$ by logarithmic sampling along each dimensions. This results in two thin matrices $\mathbf{P}^u = \mathbf{P}[u, k] \in \mathbb{C}^{N\times D}, k = [0, 1, 2, \cdots, 2^{D-2}]$ and $\mathbf{P}^v = \mathbf{P}[l, v] \in \mathbb{C}^{D\times N}, l = [0, 1, 2, \cdots, 2^{D-2}]$, with D being a small number. Consequently, such a transform simplifies to $f(x, y) = \widetilde{\mathcal{T}}(\mathbf{P}^v) + \widetilde{\mathcal{T}}(\mathbf{P}^u)$. The same formulations can be also easily extended to a higher dimensional signal with a similar logarithmic sampling along individual dimensions.

Recall recovering $f(x, y)$ corresponds to transforming the full volume $\mathbf{P}[u, v]$ with selectively marking specific entries as zero. For clarity, we use the same notation of full volume $\mathbf{P}[u, v]$ from Section 4.2 and to 4.3. A more detailed discussion on complexity is carried out in Section 4.4.

4.2 IFT Approximation

Recall that full numerical integration of the Fourier transform (i.e., Eq. 1) requires computing an expensive inner product between the entire phasor volume $\mathbf{P}[u, v]$ and the frequency-encoded volume $\exp\{j2\pi(ux + vy)\}$. It needs to be computed for every single point in $f(x, y)$, prohibitively expensive for any practical frequency learning scheme. To make the computational tractable, we observe that if all input coordinates are equally spaced, then the Fourier transform simplifies to the Discrete Fourier Transform (DFT) that can be explicitly evaluated using fast Fourier transform (FFT) methods, e.g., the Cooley–Tukey algorithm. To compute an arbitrary (off-grid) input coordinate (x, y), one possible solution is to first compute a map of equally-spaced coordinates $f(x^*, y^*)$ by 2D FFT and then perform bilinear interpolation from the 2D on-grid map given. However, such a scheme, *i.e.*, computing an entire dense grid with many unused vertices, provides a poor trade-off between complexity and accuracy.

We instead employ both FFT and numerical integration (NI) to achieve both high accuracy and relatively low complexity. Specifically, we first perform 1D FFT along one of the axes u to obtain a intermediate map $\mathbf{P}_v[x^*, v]$, with $x^* = [0, 1, \cdots, M-1]$.

$$f(x, y) \approx \underbrace{\sum_{v=0}^{D} e^{j2\pi v \frac{y}{N}} \underbrace{\left\langle \mathbf{P}_v[x^*, v] \right\rangle}_{\text{FFT+interpolation}}}_{\text{Numerical integration}} = \widetilde{\mathcal{T}}_*(\mathbf{P}) \tag{5}$$

where $\langle\cdot\rangle$ is a linear interpolation operation that corresponds to interpolating $\mathbf{P}_v(x, v)$ from the intermediate map $\mathbf{P}_v[x^*, v]$. Note that, the length d in the reduced dimension is extremely small (details described in 4.1). Therefore, per-sample numerical integration is very efficient, significantly reducing the training cost. Further, different from interpolation in the spatial domain [3, 28, 39, 48]

that results in vanishing high order gradient, such a frequency interpolation scheme, benefits from the periodic characteristic in the frequency domain, and manages to preserve gradients (see Fig 1).

4.3 Volume Regularization

Recall that many high dimensional signal reconstruction problems including NeRF are ill-posed. Therefore, PREF alone without additional priors may still produce reconstruction artifacts on such problems. In traditional signal processing, several regularization techniques such as the Lasso Regression (LR loss) and Total Variation regularizers (TV loss) have been imposed on natural signals to restrict the complexity (parsimony) of the reconstruction. Yet both LR and TV losses are designed for processing signals in the spatial domain and are not directly applicable to PREF. We therefore propose a novel Parseval regularizer as:

$$\begin{aligned}\mathcal{L}_{Reg}(\mathbf{P}) &= \|2\pi u\mathbf{P}[u,v]\|_2 + \|2\pi v\mathbf{P}[u,v]\|_2 \\ &= \sqrt{\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}\|2\pi u\mathbf{P}[u,v]\|^2} + \sqrt{\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}\|2\pi v\mathbf{P}[u,v]\|^2}\end{aligned} \tag{6}$$

We show that the Parseval regularizer behaves like anisotropic TV in the spatial domain.

Lemma 3. *Let $f(x,y)$ be integrable, and $\mathbf{P}(u,v)$ be its Fourier transform. The anisotropic TV loss of $f(x,y)$ can be represented by $\|2\pi u\mathbf{P}(u,v)\|_2 + \|2\pi v\mathbf{P}(u,v)\|_2$.*

Proof: Recall the TV loss can be computed as $\|\nabla_x f(x,y)\|_2 + \|\nabla_y f(x,y)\|_2$. Since $f(x,y)$ and $P(u,v)$ are Fourier pairs, we have Fourier transform preserves the energy of original quantity based on Parseval's theorem (theorem 2), *i.e.*,

$$\iint \|f(x,y)\|^2 dxdy = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\|P(u,v)\|^2 dudv. \tag{7}$$

According to theorem 1, $\nabla_x f(x)$ and $j2\pi uP(u,v)$ are also Fourier pairs. The integration derivative along axis x is defined as,

$$\iint \|\nabla_x f(x,y)\|^2 dxdy = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\|j2\pi uP(u,v)\|^2 dudv. \tag{8}$$

By taking square root on both sides, we have $\|\nabla_x f(x,y)\|_2 = \|2\pi uP(u,v)\|_2$. And $\|\nabla_x f(x,y)\|_2 = \|2\pi vP(u,v)\|_2$ can be derived with similar proof.

4.4 Complexity Analysis

Our PREF approximation scheme reduces the memory complexity by representing the feature space with a sparse set of frequencies, i.e., $O(kN^2D)$ to represent a k channel 3D feature volume. Similar to NGP, we can equip a shallow MLP module to PREF to achieve full spectral reconstruction.

Finally, to further improve efficiency, we adopt a two-step procedure for computing the PREF features for all input coordinates: (1) we conduct FFT to compute an off-grid intermediate map per batch of the input, and (2) we perform per sample numerical integration (NI) of the Fourier Transform. This indicates that the larger the batch, the lower the average complexity for the input samples. For a feature volume of size $O(k \times N^3)$ with X samples to be transformed and queried, our scheme reduces the complexity from $O(XkN^3)$ based on naive 3D NI to $O(kN^2logN + XkD^2)$ using 2D FFT and 1D NI. Notice that $X \gg N$ in a single training batch. Therefore, our implementation only incurs around 20% increase in computational cost over the spatially-embedded acceleration schemes. Yet PREF reduces the memory consumption, improves modeling/rendering quality via full spectral reconstruction, and provides convenient frequency manipulations.

5 Experiments

We evaluate our PREF with a number of natural signal processing tasks, including 2D image completion, signed distance field regression, and radiance field reconstruction. For each task, we tailor a solution based on our PREF.

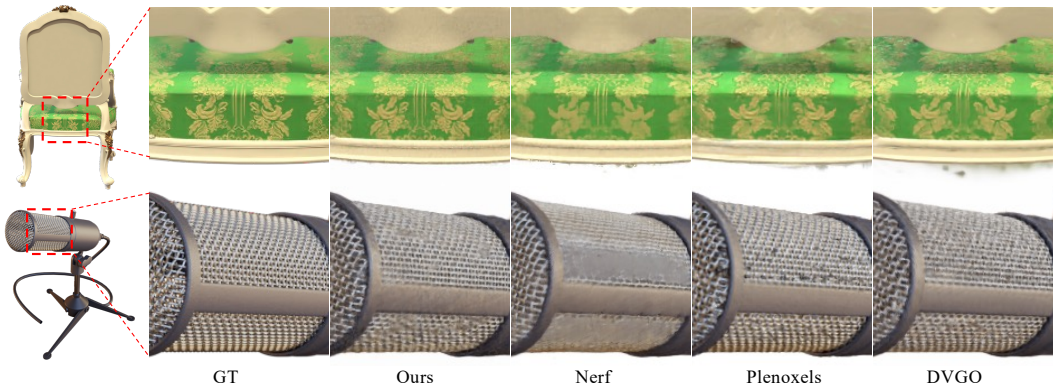


Figure 3: Quality results of our PREF and comparison methods (Nerf [22], Plenoxels [48] and DVGO [39]) on the synthetic NeRF scenes. Benefiting from our phasorial embedding, we manage to preserve details while rendering less outliers.

5.1 Baselines

We first analyze and compare our PREF with three standard backbones of coordinate-based MLPs, which also focus on boosting the capability or training efficiency.

Positional Encoding with Fourier Features. The Fourier features [41] aim to learn high-frequency functions in low dimensional problem domains. Fourier features transform low dimension coordinates p to $\gamma(p)$ before passing them into an MLP, where $\gamma(p)$ is defined as $\gamma(p) = [\dots, \cos(2^k p), \sin(2^k p), \dots]$ for $k = 0, ..., K-1$. Such a mapping function is deterministic and robust to hyperparameter K. However, the usage of exponential spaced on-axis frequency series is insufficient to cover the complete frequency domain. Therefore, PE bias toward axis-align signals [41].

Periodic Activation Functions. SIREN [37] proposes an alternative encoding: $\gamma(p) = \sin(\mathbf{W}p + b)$. A major advantage of SIREN is that it better preserves high-order gradients and subsequently supports network modulation by controlling the amplitude and phase of the active layer. A major challenge here is that the periodic activation contains a mass of local minimal. Thus it requires a careful network initialization for stable training.

Spatial Feature Grids. The latest acceleration schemes including DVGO [39], i-NGP [28], and TensoRF [5], use $\gamma(p) = \langle \mathcal{G} \rangle_p$ as a local transform. That is, TensoRF uses an orthogonal projection, DVGO uses linear interpolation, and i-NGP uses the hashing to map spatial coordinates into a learnable embedding space. These schemes greatly reduce the MLPs dependency and are highly efficient in training (e.g., a Pytorch implementation reduces NeRF training time from hours to minutes, and a CUDA implementation further reduces the time cost to seconds). Yet, as local interpolants, they struggle to maintain continuity while avoiding diminishing high-order gradients.

PREF. Our PREF employs $\gamma(p) = \widetilde{\mathcal{T}}(\mathbf{P}, p)$ based on a learnable embedding grid in the frequency domain. It continuously (after inverse Fourier transform) and globally encodes spatial coordinates into the sum of inner products. Here we advocate the "global" characteristics of PREF because each phasor impacts all spatial grids. Next, we will demonstrate that PREF can simultaneously achieve robustness via frequency decomposition and maintain the high-order gradients via periodic modeling. It is also highly efficient to conduct embedding-based learning equipped with shallow MLPs.

5.2 Applications

We evaluate our PREF on three neural reconstruction tasks, including image regression, SDF regression and radiance field reconstruction. The choice of phasor volume size and MLP varies with tasks, mainly for fair comparisons with the aforementioned baselines, which we detail in each task.

2D Image Regression and Reconstruction. 2D image regression aims to evaluate the capability of the representation supervised with all image pixels, and image reconstruction or image inpainting is

	Natural	Text
Dense Grid [39]	23.627 ± 4.137	27.446 ± 2.274
PE [41]	23.294 ± 3.245	26.852 ± 1.860
SIREN [37]	23.451 ± 3.414	26.808 ± 5.733
Ours	24.208 ± 3.607	28.238 ± 2.371

Table 1: 2D image results(mean $\pm$ standard deviation of PSNR)

Method	BatchSize	Steps	Time ↓	Size(MB)↓	PSNR↑	SSIM↑
SRN [38]	-	-	>10h	-	22.26	0.846
NeRF [26]	4096	300k	∼35h	5.0	31.01	0.947
SNeRG [10]	8192	250k	∼15h	1771.5	30.38	0.950
NSVF [19]	8192	150k	>48*h	-	31.75	0.950
PlenOctrees [49]	1024	200k	∼15h	1976.3	31.71	0.958
Plenoxels [48]	5000	128k	11.4m	778.1	31.71	0.958
DVGO [39]	5000	30k	15.0m	612.1	31.95	0.957
TensoRF [3]	4096	30k	17.4m	71.8	33.14	0.963
Ours	4096	30k	18.1m	34.4	32.08	0.952

Table 2: Application for fast radiance fields reconstruction. We compare our approach with both pure MLP models (SRN), positional encoding (NeRF, SNeRG, PlenOctrees) and grid based (Plenoxels, DVGO) approaches.

trained with partial pixels and predict the missing ones. We quantitatively evaluate the PSNR scores between the outputs and ground-truth images for both tasks.

We first conduct pilot experiments of frequency learning under the image regression task, as shown in Figure 1. The dense grid setting, corresponding to the 3rd and the 5th column, adopts a linear interpolation in a learnable $8 \times 100 \times 100$ feature grid and optional, followed by an MLP; Our PREF uses the same volume resolution as the dense grids and performs the standard inverse Fourier transform to predict the target 512×512 image. Our method significantly outperforms the dense grid setting in terms of the convergence speed and the reconstruction quality of pixels. We owe this improvement to the periodic nature of our Fourier decomposition that globally regularizes the whole signal instead of regionally as that in dense grids. We show that the representation of local parameterization with linear interpolation can hurt its generalization ability, consequently producing noisy results and losing its high-order gradients.

We further evaluate our PREF with the inpainting task, where we train our PREF with a regularly-spaced grid containing 75% missing pixels from each image in the Natural and Text dataset. In this experiment, our PREF consists of a phasor volume with the reduced dimension d = 8 and a two-layer MLP. Following [7], We report test error on an unobserved 25% pixels in Tab. 1. Reasonably, our approach also achieves qualitatively better reconstruction results.

SDF regression and editing. Next, we explore the capability of PREF for geometric representation [24, 31] and editing. We evaluate the 3D shape regression from given point clouds together with its SDF values. We adopt Armadillo and Gargoyle, two widely-used models. For each model, we normalize the training mesh into a bounded box of $[-1, 1]$, and sample $= 2^{18}$ points for training: $4/8N$ points on the surface, $3/8N$ points around the surface by adding a Gaussian noise on the surface point with scale $= 0.01$, and $1/8N$ points uniformly sampled within the bounding box. We report the IOU of the ground truth mesh and the regressed signed distance field, by discretizing them into two 128^3 volumes. We also report the Chamfer distance metric by sampling $30k$ surface points from the predicted mesh with marching cube [20]. We implement DVGO by ourselves and adopt the implementation of i-NGP[2] for comparisons. We include detailed parameter choices in supplementary. The quantitative evaluation in Tab.4 demonstrates that we achieve competitive results with existing STOAs, while remaining a more compact model.

[2]adopted from https://github.com/ashawkey/torch-ngp

		Armadillo		Gargoyle	
	Memory (MB)	IOU	Chamfer$-L_1$	IOU	Chamfer$-L_1$
i-NGP [28]	46.7	99.34	5.54e-6	99.42	1.03e-5
DVGO [39]	128.0	98.81	5.67e-6	97.99	1.19e-5
PE [41]	6.0	96.65	1.21e-5	80.46	1.03e-4
Ours	36.0	99.02	5.57e-6	99.05	1.06e-5

Table 3: SDF regression results. We compare our approach with both coordinate-based models (PE) and embedding-based models (i-NGP, DVGO).

Figure 4: SDF editing via phasorial embedding. After the sdf is regressed, PREF allows implicit surface editing by manipulate the phasorial embedding. This figure show the smoothing effect, by point-wise scaling the embedding with a Gauss function, *i.e.*, $g(\mathrm{k}) = \exp(-\mathrm{k}^T \mathrm{k}\sigma^2)$, where k is the frequency coordinate. Notice we perform smoothing in the neural fields, which is different to existing techniques like mesh smoothing.

Our trained SDF model also allows implicit surface editing such as Gauss smoothing, where we apply a point-wise multiplication of a Gauss on the trained phasorial embedding, as shown in Fig.4. This may be useful for surface denoising or texture removal.

Neural Radiance Field Reconstruction. Finally, we evaluate PREF on the popular radiance field reconstruction tasks. NeRF reconstruction attempts to recover scene geometrics and appearance given a set of multi-view input images with posed cameras. In this task, we evaluate the reconstruction quality of the novel views and evaluate the model size for the compactness and the efficiency with its training speed. In practice, we individually model the density and color, then jointly optimize them via a volume render scheme [26] supervised only with image color. However, this sometimes leads to overfitting to the training view and leading to floaters in empty space due to the strong model capability. We utilize a new Parseval term to regularize the embedding field, as described in sec. 4.3. We set the expected volume size as 256^3, roughly containing 36MB parameters. Please see supplementary for detailed information. During optimization, we apply a coarse-to-fine training scheme starting from the 128 low frequency series. We then progressively unlock the rest high frequency series to be learnable at step $[2000, 3000, 4000, 5500, 7000]$ to reach the expected frequencies 256. Tab. **??** shows the quantitative results; our model can be on par with the state-of-the-art radiance field reconstruction approaches while reaching compact modeling and a fast training process.

6 Conclusion

We have presented a novel neural approach for compact modeling that decompose a natural signal into Fourier series, and developed a new approximating inverse transform scheme for efficient reconstruction. PREF produces high quality images, shapes, and radiance fields from given limited data and outperforms recent works. Benefiting from our physical meaningful Fourier decomposition and fast transformation, we allow explicitly manipulating the learned embedding under different frequencies. One interesting future direction is to apply our representation to 3D-aware image generation, and PREF is potentially able to resolve the challenge of high-resolution rendering.

References

[1] M. Atzmon and Y. Lipman. Sal: Sign agnostic learning of shapes from raw data. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[2] P. Beatty, D. Nishimura, and J. Pauly. Rapid gridding reconstruction with a minimal oversampling ratio. *IEEE Transactions on Medical Imaging*, 24(6):799–808, 2005.

[3] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su. Tensorf: Tensorial radiance fields. *arXiv preprint arXiv:2203.09517*, 2022.

[4] A. Chen, Z. Xu, F. Zhao, X. Zhang, F. Xiang, J. Yu, and H. Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14124–14133, 2021.

[5] W. Chen, X. Zhu, R. Sun, J. He, R. Li, X. Shen, and B. Yu. Tensor low-rank reconstruction for semantic segmentation. In *European Conference on Computer Vision*, pages 52–69. Springer, 2020.

[6] Y. Chen, S. Liu, and X. Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8628–8638, June 2021.

[7] R. Fathony, A. K. Sahu, D. Willmott, and J. Z. Kolter. Multiplicative filter networks. In *International Conference on Learning Representations*, 2020.

[8] J. Fessler and B. Sutton. Nonuniform fast fourier transforms using min-max interpolation. *IEEE Transactions on Signal Processing*, 51(2):560–574, 2003.

[9] A. Gropp, L. Yariv, N. Haim, M. Atzmon, and Y. Lipman. Implicit geometric regularization for learning shapes. In *International Conference on Machine Learning*, pages 3789–3799. PMLR, 2020.

[10] P. Hedman, P. P. Srinivasan, B. Mildenhall, J. T. Barron, and P. Debevec. Baking neural radiance fields for real-time view synthesis. *arXiv preprint arXiv:2103.14645*, 2021.

[11] Z. Huang, S. Bai, and J. Z. Kolter. Implicit^2: Implicit layers for implicit representations. *Advances in Neural Information Processing Systems*, 34, 2021.

[12] C. Jensen, R. Reed, R. Marks, M. El-Sharkawi, J.-B. Jung, R. Miyamoto, G. Anderson, and C. Eggen. Inversion of feedforward neural networks: algorithms and applications. *Proceedings of the IEEE*, 87(9):1536–1549, 1999.

[13] M. Ji, J. Gall, H. Zheng, Y. Liu, and L. Fang. SurfaceNet: An end-to-end 3D neural network for multiview stereopsis. In *Proc. ICCV*, 2017.

[14] T. Karras, M. Aittala, S. Laine, E. Härkönen, J. Hellsten, J. Lehtinen, and T. Aila. Alias-free generative adversarial networks. In *Proc. NeurIPS*, 2021.

[15] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[16] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[17] M. Levoy and P. Hanrahan. Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 31–42. ACM, 1996.

[18] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey. Barf: Bundle-adjusting neural radiance fields. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[19] L. Liu, J. Gu, K. Z. Lin, T.-S. Chua, and C. Theobalt. Neural sparse voxel fields. *arXiv preprint arXiv:2007.11571*, 2020.

[20] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Computer Graphics*, 21(4):163–169, 1987.

[21] J. N. Martel, D. B. Lindell, C. Z. Lin, E. R. Chan, M. Monteiro, and G. Wetzstein. Acorn: Adaptive coordinate networks for neural scene representation. *arXiv preprint arXiv:2105.02788*, 2021.

[22] R. Martin-Brualla, N. Radwan, M. S. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7210–7219, 2021.

[23] Q. Meng, A. Chen, H. Luo, M. Wu, H. Su, L. Xu, X. He, and J. Yu. Gnerf: Gan-based neural radiance field without posed camera. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6351–6361, 2021.

[24] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. Occupancy networks: Learning 3d reconstruction in function space. *Proc. CVPR*, 2019.

[25] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019.

[26] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405–421. Springer, 2020.

[27] M. J. Muckley, R. Stern, T. Murrell, and F. Knoll. TorchKbNufft: A high-level, hardware-agnostic non-uniform fast Fourier transform. In *ISMRM Workshop on Data Sampling & Image Reconstruction*, 2020. Source code available at https://github.com/mmuckley/torchkbnufft.

[28] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv preprint arXiv:2201.05989*, 2022.

[29] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[30] M. Oechsle, S. Peng, and A. Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. *arXiv preprint arXiv:2104.10078*, 2021.

[31] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019.

[32] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021.

[33] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger. Convolutional occupancy networks. In *European Conference on Computer Vision*, pages 523–540. Springer, 2020.

[34] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proc. CVPR*, 2016.

[35] C. Reiser, S. Peng, Y. Liao, and A. Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. *arXiv preprint arXiv:2103.13744*, 2021.

[36] A. A. Rusu, D. Rao, J. Sygnowski, O. Vinyals, R. Pascanu, S. Osindero, and R. Hadsell. Meta-learning with latent embedding optimization. *arXiv preprint arXiv:1807.05960*, 2018.

[37] V. Sitzmann, J. N. Martel, A. W. Bergman, D. B. Lindell, and G. Wetzstein. Implicit neural representations with periodic activation functions. In *arXiv*, 2020.

[38] V. Sitzmann, M. Zollhöfer, and G. Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *arXiv preprint arXiv:1906.01618*, 2019.

[39] C. Sun, M. Sun, and H.-T. Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. *arXiv preprint arXiv:2111.11215*, 2021.

[40] T. Takikawa, J. Litalien, K. Yin, K. Kreis, C. Loop, D. Nowrouzezahrai, A. Jacobson, M. McGuire, and S. Fidler. Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11358–11367, 2021.

[41] M. Tancik, P. P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. T. Barron, and R. Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 2020.

[42] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021.

[43] P.-S. Wang, Y. Liu, Y.-Q. Yang, and X. Tong. Spline positional encoding for learning 3d implicit signed distance fields. *arXiv preprint arXiv:2106.01553*, 2021.

[44] D. N. Wood, D. I. Azuma, K. Aldinger, B. Curless, T. Duchamp, D. H. Salesin, and W. Stuetzle. Surface light fields for 3d photography. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 287–296. ACM Press/Addison-Wesley Publishing Co., 2000.

[45] W. Xia, Y. Zhang, Y. Yang, J.-H. Xue, B. Zhou, and M.-H. Yang. Gan inversion: A survey. *arXiv preprint arXiv:2101.05278*, 2021.

[46] L. Yariv, J. Gu, Y. Kasten, and Y. Lipman. Volume rendering of neural implicit surfaces. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[47] L. Yariv, Y. Kasten, D. Moran, M. Galun, M. Atzmon, B. Ronen, and Y. Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. In *Proc. NeurIPS*, 2020.

[48] A. Yu, S. Fridovich-Keil, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa. Plenoxels: Radiance fields without neural networks. *arXiv preprint arXiv:2112.05131*, 2021.

[49] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. *arXiv preprint arXiv:2103.14024*, 2021.

[50] H. Yu, A. Chen, X. Chen, L. Xu, Z. Shao, and J. Yu. Anisotropic fourier features for neural image-based rendering and relighting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

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

[52] E. D. Zhong, T. Bepler, B. Berger, and J. H. Davis. Cryodrgn: reconstruction of heterogeneous cryo-em structures using neural networks. *Nature methods*, 18(2):176–185, 2021.

-Supplementary-

A Phasorial Embedding Fields Implementation Details

Phasor Volume Decomposition. Recall that PREF is a continuous embedding field corresponding to a multi-channel multi-dimensional square Fourier volume. We elaborate on implementation details. Let $\mathbf{P}[u, v, w]$ be a 3D phasor volume representing an embedding field $f(x, y, z)$.

Note that the $\mathbf{P}[u, v, w]$ is Hermitian symmetric when the $f(x, y, z)$ is a real-valued feature embedding, *i.e.*, $\mathbf{P}[u, v, w] = \mathbf{P}^*[-u, -v, -w]$ (i.e., its complex conjugate). Further, base on the observation that natural signals are generally band-limited, we model their corresponding fields with band-limited phasor volumes $\mathbf{P}[u, v, w]$ where we are able to partially mask out some entries and factor the volume along respective dimensions, as shown in Fig 5. Thus we factorize the full spectrum into tri-thine embeddings by reusing the linearity of the Fourier Transform, $f(x, y, z) = \widetilde{\mathcal{T}}(\mathbf{P}_u) + \widetilde{\mathcal{T}}(\mathbf{P}_v) + \widetilde{\mathcal{T}}(\mathbf{P}_w)$.

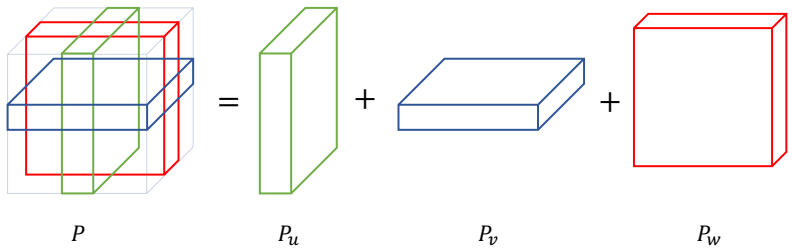


Figure 5: Phasor volume decomposition. Let the phasor volume with zero frequency centered. We selectively mask out some entries of the phasor volume as zero, then approximate the spatial feature embedding of a large phasor volume in terms of the sum of the embedding from three smaller ones, one for each dimension. The complete phasor volume to spatial feature embeddings is shown in the PyTorch pseudo-code in Algorithm 1.

IFT Implementation. Recall that for a 3D phasor volume, we approximate $\widetilde{\mathcal{T}}$ via sub-procedures of applying 2D Fast Fourier Transforms (FFTs) and 1D numerical integration (NI) to achieve high efficiency. Therefore, given a batch of spatial coordinates, our PREF representation transforms them into a batch of feature embeddings in parallel where PREF can serve as a plug-and-play module. Such a module can be applied to many existing implicit neural representations to conduct task-specific neural field reconstructions. We present a sketchy PyTorch pseudo-code in Algorithm 1.

Phasor Volume Initialization. Our PREF approach can be alternatively viewed as a frequency space learning scheme to existing spatial coordinate-based MLPs. In our experiments, we found zero initialization works well for applications ranging from 2D image regression to 5D radiance fields reconstruction while certain applications require more tailored initialization, e.g., geometric initialization in [1, 47]. This is because $\mathbf{P}(\mathrm{k})$ (with k being the frequency coordinate) needs to satisfy the unique constraints of $f(\mathrm{x})$. We thus initialize the phasor volume as follows: Let $f^{\circ}(\mathrm{x})$ be the initialization of $f(\mathrm{x})$. We have $\mathbf{P}^{\circ}(\mathrm{k}) = \mathcal{T}(f^{\circ}(\mathrm{x}))$, with $\mathcal{T}$ as the Fourier transform. We then transform $\mathbf{P}^{\circ}(\mathrm{k})$ via the inverse Fourier transform $\widetilde{\mathcal{T}}$ (due to duality between $f(\mathrm{x})$ and $\mathbf{P}(\mathrm{k})$) as the *approximation* to $f^{\circ}(\mathrm{x})$. We found such a strategy enhances the stability and efficiency.

Computation Time. One of the key benefits of PREF is its efficiency. As discussed in section 4.3, we conduct frequency-based neural field reconstruction by employing IFT, which is computationally low cost and at the same time effective. When the input batch is sufficiently large (e.g., 4096×1024 samples per batch in radiance field reconstruction), the per-sample numerical evaluation will dominate the computational cost. Since such per-sample evaluation can be efficiently implemented using matrix product, it is essentially equivalent to adding a tiny linear layer. The overall implementation makes PREF nearly as fast as the state-of-the-art, e.g., instant-NGP for NeRF. For example, on the Lego example, our PyTorch PREF produces the final result in 16 minutes on a single RTX3090, considerably

Algorithm 1: PyTorch pseudo code for PREF

```
class PREF(nn.Module):
   def _init_(self, res, d, ks):
      # res:  resolution size
      # d:  reduced dim size
      # ks:  output kernel size
      Nx, Ny, Nz = res
      # log sampling freq in reduced dimension
      self.freq = torch.tensor([0]+[2**i for i in torch.arange(d-1)])
      self.Pu = nn.Parameter(torch.zeros(1, ks, d, Ny, Nz).to(torch.complex))
      self.Pv = nn.Parameter(torch.zeros(1, ks, Nx, d, Nz).to(torch.complex))
      self.Pw = nn.Parameter(torch.zeros(1, ks, Nx, Ny, d).to(torch.complex))
   def forward(self, xyz):
      # 2D Fast Fourier Transform
      Pu = torch.fft.ifftn(self.Pu, dim=(3,4))
      Pv = torch.fft.ifftn(self.Pv, dim=(2,4))
      Pw = torch.fft.ifftn(self.Pw, dim=(2,3))
      # 2D Linear interpolation
      xs, ys, zs = xyz.chunk(3, dim=-1)
      Px = grid_sample_cmplx(Pu.transpose(3,3).flatten(1,2), torch.stack([zs,
       ys], dim=-1)[None]).reshape(Pu.shape[1], Pu.shape[2], -1)
      Py = grid_sample_cmplx(Pv.transpose(2,3).flatten(1,2), torch.stack([zs,
       xs], dim=-1)[None]).reshape(Pv.shape[1], Pv.shape[3], -1)
      Pz = grid_sample_cmplx(Pw.transpose(2,4).flatten(1,2), torch.stack([xs,
       ys], dim=-1)[None]).reshape(Pw.shape[1], Pw.shape[4], -1)
      # 1D Numerical Integration
      fx = batch_NI(Px, xs, self.freq)
      fy = batch_NI(Py, ys, self.freq)
      fz = batch_NI(Pz, zs, self.freq)
      # Summation
      return fx+fy+fz
```

faster than the original NeRF and comparable to the PyTorch implementation of i-NGP. We are in the process of implementing PREF on CUDA analogous, and hopefully, it may achieve comparable performance to the CUDA version of i-NGP.

B Application to Image Regression

B.1 Implementation & Reproducibility Details

Pilot experiments. Before proceeding to more sophisticated tasks such as generation and reconstruction, we first explore a toy example that utilizes PREF to continuously parametrize an image, e.g., a 512×512 grayscale image. To better provide insights into our frequency-based learning framework, we use a 100×100 complex-valued grid to regress (upsample) the 512×512 input image via inverse Fourier transform, using PREF vs. bilinear upsampling on the same MLP network. From a signal processing perspective, if the image is band-limited, *e.g.*, Nyquist frequency is the fold of 128, a frequency scheme should perfectly reconstruct the 512×512 image. Yet Bilinear interpolation exhibits aliasing due to the characteristics of the first- and second-order derivatives, as shown in the discussion and Fig. 1 in the paper. We then show the performance using embedding where we expand the grid size to a $8 \times 100 \times 100$ embedding volume followed by a three-layer MLP with a hidden dimension of 256 that maps the embeddings to pixels. Previous studies [37] have shown that improper activation functions can also lead to aliasing in high-order gradients, despite the choice of embedding techniques. Therefore, for comprehensive studies, we further compare various most-seen activation functions, including ReLU, Tanh, and the most recent, Sine [37]. Our experiments show that such a frequency-learning scheme consistently outperforms its spatial counterparts, potential owing to its well-behaved derivatives and continuous nature, as shown in the line plot of Fig. 1.

Image completion. Next, we demonstrate PREF on image completion tasks. We use the commonly adopted setting [11, 7]: given 25% pixels of an image, we set out to predict another 25% pixels. We evaluate PREF vs. SATO on two benchmark datasets - Nature and Text. Specifically, we compare

	Memory (MB)	Armadillo IOU	Armadillo Chamfer$-L_1$	Gargoyle IOU	Gargoyle Chamfer$-L_1$
i-NGP [28]	46.7	99.34	5.54e-6	99.42	1.03e-5
DVGO [39]	128.0	98.81	5.67e-6	97.99	1.19e-5
PE [41]	6.0	96.65	1.21e-5	80.46	1.03e-4
Ours	36.0	99.02	5.57e-6	99.05	1.06e-5

Table 4: SDF regression results. We compare our approach with both coordinate-based models (PE) and embedding-based models (i-NGP, DVGO).

PREF with a dense grid counterpart, and two state-of-the-art coordinate-based MLPs[22, 37]. The dense grid uses a $8 \times 100 \times 100$ resolution whereas PREF uses two 20×9 grids that correspond to the highest frequency of $2^7 = 128$. The two embedding techniques above use the same MLP with three linear layers, 256 hidden dimensions, and ReLU activation. We use Positional Encoding (PE) which consists of a 5-layer MLP with 7 frequencies encoding. We adopt SIREN from [37] that uses a 4-layer MLP and sine activation. Detailed comparisons are listed in Tab 1.

Optimization details. All experiments use the same training configuration. Specifically, we adopt the Adam optimizer [16] with default parameters ($\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1e-8$), a learning rate of $1e^{-4}$. We use L_1 loss with $15k$ iterations to produce the final results.

C Application to Signed Distance Field Reconstruction

C.1 Task description

Next, we conduct the more challenging task of signed distance field (SDF) reconstruction. An SDF describes the shape in terms of a function as:

$$f(\mathrm{x}) = \begin{cases} d(\mathrm{x}, \partial\Omega), \text{if } \mathrm{x} \in \Omega^+ \\ -d(\mathrm{x}, \partial\Omega), \text{if} \mathrm{x} \in \Omega^- \end{cases} \tag{9}$$

where $\partial\Omega$ is a closed surface and Ω^+ and Ω^- correspond to regions outside of and inside the surface respectively. d is the Euclidean distance from a point to the surface. Our goal is to recover a continuous SDF $f(\mathrm{x})$ given a set of discretized samples x^* of value $f(\mathrm{x}^*)$ that commonly refers to samples from a mesh.

C.2 Implementation & Reproducibility Details

Data preparation. We adopt two widely used models: gargoyle (50k vertices) and armadillo (49k vertices). For each training epoch, we scale the model within a bounding box of $[-1, 1]$ and samples $N = 2^{18}$ points for training: $4/8N$ points on the surface, $3/8N$ points around the surface by adding Gaussian noise to the surface point with $scale = 0.01$, and the last $1/8N$ points uniformly sampled within the bounding box.

Metric. We report the IOU of the ground truth mesh and the regressed signed distance field by discretizing them into two 128^3 volumes. We report the Chamfer distance metric by sampling 30k surface points from the extracted mesh using the marching cube technique [20].

Baseline implementation details. For the embedding-based baselines, we use our implementation of the dense volume technique [39]. It contains $16 \times 128 \times 128 \times 128$ learnable parameters that transform the input coordinates to their feature embedding at a length 16 by trilinear-interpolation. We adopted the PyTorch implementation of i-NGP [28] from torch-ngp [3], where they maintain a multi-level hash function to transform the spatial coordinate into feature embedding. We use a 16 num-of-level hash function with dimension 2. Consequently, the output feature embedding is of length 32. Please refer to [28] for more details on the implementation of multi-level hash. For our PREF, we use three $16 \times 128 \times 128 \times 6$ complex-valued volumes to nonlinearly transform the spatial coordinates to a 16d feature embeddings. All the embedding-based baselines and our PREF adopt the

[3] https://github.com/ashawkey/torch-ngp

	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship	Mean	Size (MB)↓
PlenOctrees [49]	34.66	25.37	30.79	36.79	32.95	29.76	33.97	29.62	31.71	1976.3
Plenoxels [48]	33.98	25.35	31.83	36.43	34.10	29.14	33.26	29.62	31.71	778.1
DVGO [39]	34.09	25.44	32.78	36.74	34.46	29.57	33.20	29.12	31.95	612.1
Ours	34.95	25.00	33.08	36.44	35.27	29.33	33.25	29.23	32.08	34.4

Table 5: PSNR results on each scene from the Synthetic-NeRF dataset [26]. We show the comparisons of the dense volume variants with our PREF (frequency-based scheme).

Highest Freq	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship	Mean	Size (MB)↓
256	34.95	25.00	33.08	36.44	35.27	29.33	33.25	29.23	32.08	34.40
128	33.29	24.64	32.70	36.04	33.77	29.37	31.87	27.75	31.18	9.84
64	31.54	23.82	30.43	35.25	30.56	28.82	31.22	27.08	29.83	2.28
32	30.11	22.59	27.77	34.07	27.39	27.65	30.48	25.79	28.23	0.76

Table 6: Ablation study of phasor volume size (related to the highest frequency). We report the PSNR of each scene, the mean PSRN and the corresponding mean model size. We train each scene *less than* 18 minutes using a pure pytorch implementation on a RTX 3090, as discussed in the text.

same MLP structure for fairness that consists of 3 layers that progressively map the input embedding to the 64 dimension features as well as to a scalar, with ReLU as the intermediate activation. Another baseline we compare against the positional encoding (PE) based NeRF that uses a wider and deeper coordinate-based MLP [26] where we encode the input coordinates into six frequencies in PE and use an MLP of 8 linear layers, 512 hidden dimensions, and ReLU activation. Tab 4 lists individual model size and performance of the baseline vs. PREF. Our method manage to be on par with the state-of-the-art i-NGP [28] with a compact model size, and outperform its spatial counterpart [39] and frequency-based proceedings. We owe the improvement of PREF to its globally continuous nature that either allows for preserving details.

Training details. We provide additional details on how we train the baseline. As aforementioned, in each epoch, we sample a batch size of $N = 2^{18}$ to regress the SDF values. The MAPE loss is used for error back-propagation. To optimize the networks, we use the Adam optimizer, with $\beta_1 = 0.9$, $\beta_2 = 0.99$ and $\epsilon = 1e^{-5}$. We use an initial learning rate $1e^{-4}$ and reduce the learning rate to $1e^{-5}$ at the 10th epoch. We adopt a batch size of $N/100$ to optimize all baselines whereas for our method 20 epochs.

D Radiance Fields Reconstruction

D.1 Task description

For radiance field, we focus on rendering novel views from a set of images with known camera poses. Each rgb value in each pixel corresponds to a ray cast from the image plane. We adopt the volume rendering model [22]:

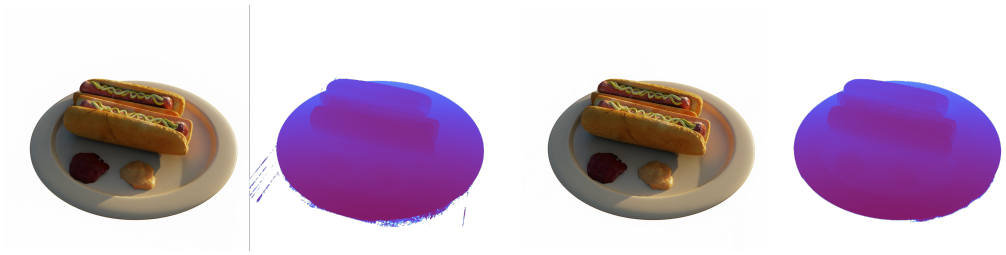
$$\hat{C}(r) = \sum_{i}^{N} T_i(1 - \exp(-\sigma_i \delta_i))c_i, \text{where } T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j) \tag{10}$$

where σ_i and c_i are corresponding density and color at location x_i, δ_i is the interval between adjacent samples. Then we optimize the rendered color with the ground truth color with L_2 loss.

$$L_{rgb} = \frac{1}{M} \sum_{i=0}^{M} \|C(r) - \hat{C}(r)\|^2. \tag{11}$$

D.2 Implementation & Reproducibility Details

PREF model setting. We describe how PREF models the density σ and the radiance c. We use three $16 \times 256 \times 256 \times 1$ phasor volume cascaded with a two-layer MLP with hidden dimension 64 and



(a) w/o L_p (PSNR 35.65) (b) w L_p (PSNR 36.44)

Figure 6: Ablation study of Parsvel regularizer. Different from spatial embedding that easily produces outliers owing to per-location parameterization, PREF globally parameterizes the scene's embedding with sinusoidal waves, often producing fewer spatial outliers (isolated points). Yet, PREF has the risk of frequency outliers (isolated waves), especially when the highest frequency used exceeds the minimal sampling rate of the observation (training samples). As Fig 6 (a) shows, PREF w/o L_p can overfit to spatial frequencies (see the disparity map on the right) and consequently results in a performance drop. We arrive at better results via our proposed Parseval regularizer.

output dimension 1 for computing the density (a scalar). We then use Softplus to map the raw output to the positive-valued density. For the view-dependent radiance branch, we use a relatively large volume of $32 \times 256 \times 256 \times 1$ followed by a linear layer to output $27d$ feature embedding. To render view-dependent radiance, we follow the TensoRF pipeline [5]: we concatenate the result with the positional encoded view directions and feed them into a 2-layer MLP with 128d hidden dimension and a linear layer to map the feature to color with Sigmoid activation. All linear layers except the output layer use ReLU activation.

Rendering. To compare with SOTA [48, 39, 5], we train each scene using $30k$ iterations with a batch size of 4096 rays. We adopt a progressive training scheme: from the highest frequency of 128 to 256. Specifically, we gradually unlock the higher frequencies at the training step $[2000, 3000, 4000, 5500, 7000]$. Accordingly, the number of samples per ray progressively increases from about 384 to about 1024. This allows us to achieve more stable optimization by first covering the lower frequencies and later high-frequency details. During training, we maintain an alpha mask to skip empty space to avoid unnecessary evaluations.

Optimization. As mentioned in the paper, our PREF uses the Parsvel regularizer L_p to avoid overfitting where our objective is set to $L = L_rgb + \lambda L_p$ with $\lambda = 1e^{-2}$. Without regularization, PREF may overfit specific frequencies, as shown in Fig 6. On the NeRF synthetic dataset, PREF converges on average 18 minutes with 30k iterations on a single RTX 3090, with an initial learning rate of 0.002 and gradually decayed by a factor of 10 during the training. The Adam optimizer uses $\beta_1 = 0.9$ and $\beta_2 = 0.99$ by default.

D.3 Additional results

We report the breakdown results of our PREF on the Synthetic-NeRF dataset in Tab 5. To further evaluate the effectiveness of our frequency encoding, we report the performance in Tab 6 under different model sizes (by varying the phasor volume size). Notice that PREF produces reasonable results (with a mean PSNR of 28.23) even when the model size is reduced to ultra-small (0.76 MB), a potential benefit for downstream generative tasks that require training thousands of scenes.

E Application to Shape Editing

E.1 Implementation details

Recall that the continuous embedding field of PREF is synthesized from a phasor volume under various frequencies. Therefore, thanks to Fourier transforms, various tools such as convolution in the continuous embedding fields can be conveniently and efficiently implemented as multiplications. This is therefore a unique advantage of PREF compared with its spatial embedding alternatives [5, 39, 48, 28].

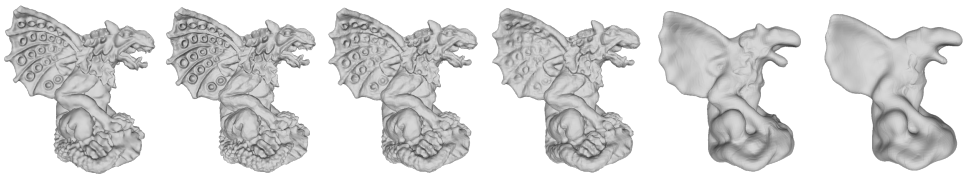


Figure 7: Filtering sign distance fields via Gaussian smoothing using PREF.

Let $\mathcal{M}(;\theta)$ and $\mathbf{P}[u, v, w]$ be the optimized MLP and phasor volume, respectively. $\widetilde{\mathcal{T}}$ represents the inverse Fourier Transform. Recall that we obtain a reconstruction field by $\Phi(\mathrm{x}) = \mathcal{M}(\widetilde{\mathcal{T}}(\mathbf{P}; \mathrm{x}); \theta)$. Modification to the original signal via convolution based filtering can now be derived as:

$$\Phi^*(\mathrm{x}) = \mathcal{M}(\widetilde{\mathcal{T}}(\mathbf{P} \circ \mathrm{G}; \mathrm{x}); \theta) \tag{12}$$

where $\circ$ denotes element-wise multiplication and $\mathrm{G} \in \mathbb{C}^{l \times N^3}$ is a filter.

Now, we explore how to manipulate $\Phi(\mathrm{x})$ via the optimized phasor volume $\mathbf{P}$ and kernel G. For simplicity, we only the Gaussian filter G while more sophisticated filters can also be applied in the same. Assume

$$G(\mathrm{k}) = \exp(-\mathrm{k}^T \mathrm{k} \sigma^2), \tag{13}$$

where $\mathrm{k} = [u/N, v/N, w/N]$ and G covers the complete frequency span of $\mathbf{P}$; that is, we can scale the magnitude of phasor features frequency-wise. For example, by varying the Gaussian kernel size using σ, PREF can denoise the neural representation of the signal at different scales, as shown in Fig 7.

F Limitations

We have demonstrated that PREF enables fast reconstructions of neural signals in the phasor (frequency) space, with smaller model sizes, comparable and sometimes better performances, and more efficient filtering capabilities. Compared with existing spatial embedding techniques, PREF, however, requires additional computational costs for conducting Fourier transforms and therefore is slightly slower than prior art such as i-NGP (PyTorch). Our immediate next step is to implement a CUDA version of PREF. However, certain Autograd libraries do not readily support complex-valued parameters optimization. Therefore additional efforts are required to write customized CUDA modules for PREF.

Similar to PE [22], PREF masks out certain spatial frequencies in the phasor volume to achieve efficiency and compactness. However, this may lead to directional bias, as observed in prior art [41]. However, since PREF uses more frequencies (3D sparse frequencies) than PE (axis-aligned 1D frequencies), PREF effectively reduces these artifacts, as shown in the experiments. For further improvement, one may adopt Non-uniform Fast Fourier Transform (NuFFT) [8, 2, 27] to tackle non-uniform frequency sampling. Overall, by providing a new frequency perspective of neural signal presentation, PREF may stimulate significant future work. To that end, we intend to make our code and data available to the community at GitHub.