

# Stochastic Preconditioning for Neural Field Optimization

SELENA LING, University of Toronto, Canada and NVIDIA, Canada

MERLIN NIMIER-DAVID, NVIDIA, Switzerland

ALEC JACOBSON, University of Toronto and Adobe Research, Canada

NICHOLAS SHARP, NVIDIA, USA

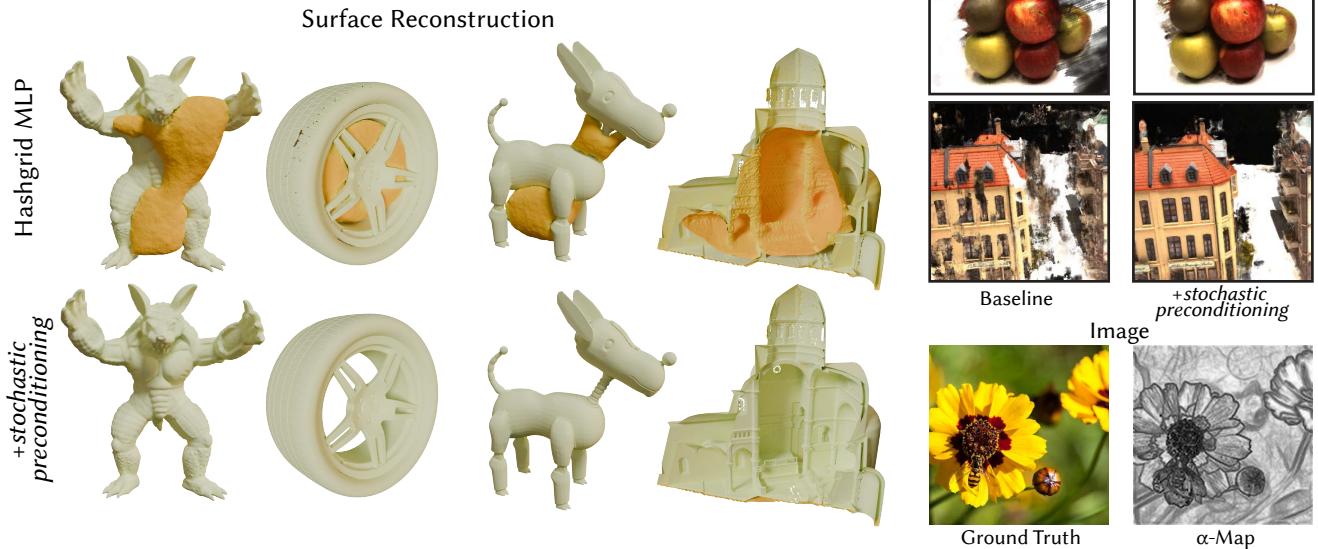


Fig. 1. Stochastic preconditioning adds spatial noise to query locations during neural field optimization; it can be formalized as a stochastic estimate for a blur operator, yet is easily implemented for any field representation in a few lines of code. We find that this simple technique eases optimization and significantly improves quality for neural fields optimization tasks like reconstructing signed distance fields from point clouds (Left), and neural radiance fields (Right), matching or outperforming custom-designed policies and coarse-to-fine schemes. The level of noise can even be treated as an optimization variable, naturally yielding a frequency scale maps for the field (Right, Bottom). **Webpage:** [research.nvidia.com/labs/toronto-ai/stochastic-preconditioning/](http://research.nvidia.com/labs/toronto-ai/stochastic-preconditioning/)

Neural fields are a highly effective representation across visual computing. This work observes that fitting these fields is greatly improved by incorporating spatial stochasticity during training, and that this simple technique can replace or even outperform custom-designed hierarchies and frequency-space constructions. The approach is formalized as implicitly operating on a blurred version of the field, evaluated in-expectation by sampling with Gaussian-distributed offsets. Querying the blurred field during optimization greatly improves convergence and robustness, akin to the role of preconditioners in numerical linear algebra. This implicit, sampling-based perspective fits naturally into the neural field paradigm, comes at no additional cost, and is extremely simple to implement. We describe the basic theory of this technique, including details such as handling boundary conditions, and extending to a spatially-varying blur. Experiments demonstrate this approach on representations including coordinate MLPs, neural hashgrids, triplanes, and more, across tasks including surface reconstruction and radiance fields. In settings where custom-designed hierarchies have already been developed, stochastic preconditioning nearly matches or improves their performance with a simple and unified approach; in settings without existing hierarchies it provides an immediate boost to quality and robustness.

---

Authors' addresses: Selena Ling, University of Toronto, Canada and NVIDIA, Canada, selenaling@mail.utoronto.ca; Merlin Nimier-David, NVIDIA, Switzerland, mnimierdavid@nvidia.com; Alec Jacobson, University of Toronto and Adobe Research, Canada, jacobson@cs.toronto.edu; Nicholas Sharp, NVIDIA, USA, nsharp@nvidia.com.

CCS Concepts: • Computing methodologies → Computer graphics, Machine Learning.

## 1 INTRODUCTION

Neural fields represent signals with a function mapping coordinates to signal values, promising an easy-to-optimize and spatially-adaptive representation for signals such as signed distance functions, densities, and radiance fields in 3D, or even simply the color field of an image in 2D. Many variants have been explored, but the most basic neural field is a function parameterized as a multilayer perceptron (MLP) with weights which are fit to encode the field. Neural fields can allocate their capacity adaptively as-needed across the domain and benefit from the repertoire of techniques common in machine learning, including auto-differentiation, black-box optimizers, and efficient parallel evaluation on GPUs.

However, neural fields are not without challenges. Fitting a neural field amounts to solving an expensive nonlinear optimization problem to train the neural network. This is true even when encoding an already-known signal, and is worsened when optimizing for indirect inverse problems, such as reconstruction from images. Poor local minima in the optimization landscape lead to low-quality solutions, artifacts, and unresolved detail in the desired signal—in

3D scenes, one way these minima manifest is as so-called floater artifacts, as shown in Figures 6 and 9.

The challenge of optimizing neural fields has led to a proliferation of variants in the representation, with alternate input encodings, architectures, hierarchies, and hybrid strategies, including the well-known work of Mildenhall et al. [2021]; Müller et al. [2022]; Tancik et al. [2020], among many others. One perspective on these techniques is that they facilitate easier fitting of the field by improving the optimization landscape with a more suitable parameterization for the neural field, or improvements in computational efficiency and expressive power. However, improved representational power may even worsen the effects of local minima in the optimization process, especially in indirectly supervised and inverse problems with challenging objectives [Härenstam-Nielsen et al. 2024; Li et al. 2023; Yang et al. 2023]. More fundamentally, these many representational variants add significant complexity, and are generally incompatible with one-another. The challenges of neural fields have also been addressed at the application level, with specialized coarse-to-fine optimization schemes, regularization terms and many other task-specific optimization techniques [Härenstam-Nielsen et al. 2024; Hertz et al. 2021; Li et al. 2023; Park et al. 2023; Walker et al. 2024; Yang et al. 2023; Yeom et al. 2024]. Though effective, these solutions are often tightly-coupled to the particular task and representation, and do not easily generalize to other neural field problems.

In this paper, we propose a simple neural field optimization technique that greatly alleviates the challenge of spurious local minima, improves quality, and can be easily integrated into a wide variety of representations and tasks. The basic idea is to stochastically perturb the input coordinates whenever querying the field during optimization. From a principled perspective, this is equivalent in-expectation to sampling from a spatial blurring of the field, yet this relationship is implicit and the approach has only the negligible computational cost of adding a small offset to each sample. We call this technique *stochastic preconditioning* (SP) in reference to the use of preconditioners in linear algebra, to emphasize its benefits in improving the optimization landscape and its applicability to many problems.

We do not suggest any new architectures or tasks, but rather we study how stochastic preconditioning can be applied to a wide range of existing settings, augmenting or even replacing specialized techniques developed for the same purpose (Figure 1). Practical

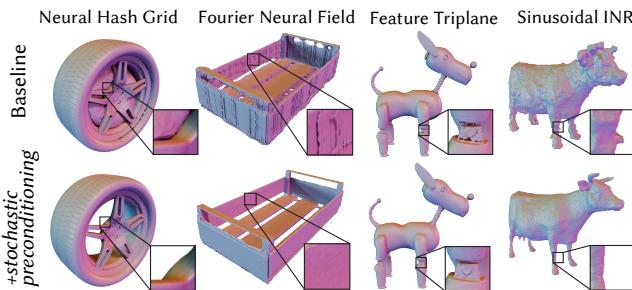


Fig. 2. Stochastic preconditioning can easily be applied to a wide variety of existing neural field representations, consistently reducing artifacts and improving quality, here in the context of fitting signed-distance functions to oriented point clouds.

```

1 def forward():
2     x = generate_samples()
3     # Apply stochastic preconditioning
4     x += α * torch.randn_like(x)
5     return f(x)

```

Fig. 3. Example integration of stochastic preconditioning in an existing implementation of a neural field-based method.

benefits include improved output quality across several problems and datasets (Figures 6, 7 and 11), as well as robustness across a wide range of hyperparameters (Figure 9). Most importantly, this technique is extremely easy to apply, often just a few lines of code, and is broadly applicable to many formulations—the minor code changes required are shown in Figure 3. For problems in which customized hierarchical strategies already exist, much of their benefit could be had from simply using stochastic preconditioning instead (Figure 8). In settings where such techniques have not yet been deployed, there is an immediate and nearly-free boost in quality to be had by enabling stochastic preconditioning (Figure 11).

## 2 RELATED WORK

Since a comprehensive overview of neural fields is out of scope, we recommend the course by Takikawa et al. [2023b]. Here, we focus on neural field architectures and optimization that are directly relevant. See Section 5 for references on downstream task formulations.

*Frequency domain.* Neural field representations quickly moved past naively encoding a field in a MLP. One line of work leverages frequency-domain encodings of the input coordinates, enriching the input space with high-frequency features [Tancik et al. 2020]. The use of this technique for radiance fields [Mildenhall et al. 2021] was key to the broad adoption of neural fields, and continues to be effective in subsequent formulations [Wang et al. 2021]. SIREN [Sitzmann et al. 2020] builds an architecture entirely from periodic activations to exploit their properties, while BACON [Lindell et al. 2022], BANF [Shabanov et al. 2024] and TUNER [Hayes et al. 2023] deepen this perspective with explicit control over the frequency spectrum of the signal. Our stochastic preconditioning likewise leverages frequency-adapted optimization via a blur operation, but in a manner which is compatible with any neural field architecture, and can be applied in conjunction with frequency-space representations for further improvements (Figure 2, Table 2). Another line of work considers signal processing-like operations on neural fields [Nsampi et al. 2023; Xu et al. 2022] including a related stochastic sampling perspective in [Rebain et al. 2024]; our work focuses on blurring as an optimization technique, rather than tasks where a filtered signal arises in the desired input or output.

*Hybrid feature grids.* Another approach is to augment the neural field with trainable input features sampled from some backing store or grid [Fridovich-Keil and Yu et al. 2022]. Instant NGP [Müller et al. 2022] fetches features from a multi-resolution hashgrid encoding, a technique that has since become widely used for large-scale and high-performance neural fields [Barron et al. 2023; Li et al. 2023; Takikawa et al. 2023a]. Though efficient, naively-applied

hash-encoded neural fields seem particularly prone to spurious local minima—adding our stochastic preconditioning offers a significant improvement (Figures 6). Triplane encodings map or learn 3D features to a triplet of planar grids orthogonal to the coordinate axes [Wang et al. 2023], while Karnewar et al. [2022] explore the limit case of a single nonlinearity applied directly to grid features without MLP. In both settings, our approach again has benefits (Figure 2).

*Optimization.* Rather than modifying the objective or architecture of any given problem, our approach modifies the training process. Coarse-to-fine approaches have a long history in optimization, as well as specifically in neural fields [Karnewar et al. 2022; Li et al. 2023; Mujkanovic et al. 2024; Xiangli et al. 2022; Yang et al. 2023], and several important downstream applications [Akgul and Kambhamettu 2003; Yao and Al-Dahle 2019]. Neural fields which leverage hierarchical feature stores [Takikawa et al. 2021] take particular advantage of coarse-to-fine optimization by adaptively unlocking levels of the hierarchy [Hertz et al. 2021; Karnewar et al. 2022; Li et al. 2023; Petit et al. 2025; Walker et al. 2024]. Yang et al. [2023] take a similar approach in the frequency-domain [Tancik et al. 2020], gradually increasing the frequency range of the input encoding. Seo et al. [2024] permute pixel coordinates on images to accelerate optimization. Another work specifically addresses the shortcomings of neural radiance field training from a gradient scaling perspective [Philip and Deschaintre 2023]. These techniques are simple and effective, but specific to the internals of the representation—stochastic preconditioning offers similar coarse-to-fine optimization with a generic approach applicable to any neural field.

*Stochastic perturbations in machine learning.* Perturbation of inputs during neural network training has a rich history in the machine learning literature. Sietsma and Dow [1991] first proposed adding noise to network inputs to improve generalization, though in simpler settings. The theoretical foundations were later established by Bishop [1995], showing that input noise injection is equivalent to Tikhonov regularization for specific loss functions. Grandvalet et al. [1997] further connected Gaussian input noise to heat kernel diffusion, providing additional theoretical guarantees. Adding noise to other quantities during training, such as the activations, was also explored [An 1996; Camuto et al. 2020]. While these classical works focus on perturbations to improve generalization and aim to provide theoretical guarantees in particular settings, our approach instead leverages stochastic smoothing specifically to guide optimization trajectories and avoid local minima in the context of spatial neural fields.

### 3 BACKGROUND

We start by reviewing concepts and establishing notation for neural fields, gradient preconditioning, and Gaussian smoothing.

#### 3.1 Neural Fields (and Other Fields Too)

A neural field encodes a signal on a domain as a function  $f_\theta : \mathbb{R}^m \rightarrow \mathbb{R}^n$ , parameterized by some kind of neural network with parameters  $\theta$  which maps input coordinates to signal values. For example, a

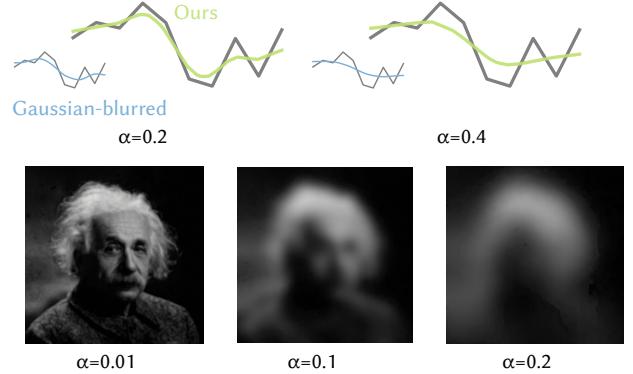


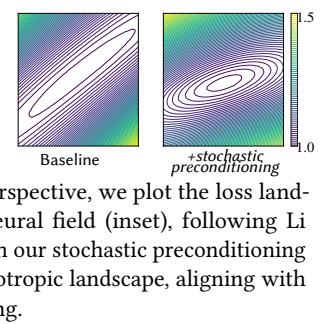
Fig. 4. A simple example of fitting stochastically-blurred neural fields to 1D (top) and 2D (bottom) signals. Sampling from the blurred  $f(\mathbf{x} + \boldsymbol{\delta})$  while fitting yields a field  $f(\mathbf{x})$  which is itself blurred. The level of blur closely approximates the corresponding blur of the true signal (top, inset).

neural field representing an image would have spatial image locations  $(x, y)$  as function inputs with  $m = 2$ , and the RGB color at that location as function outputs with  $n = 3$ ; the image is encoded by the parameters of the neural network.

In fact, some of the field representations discussed in this work are not strictly neural, but more generally any parametric function  $f_\theta$  used to encode a field. For example, Karnewar et al. [2022] sample values from a grid without evaluating any MLP. Stochastic preconditioning applies more broadly to any notion of a queryable field, although we continue to use the term neural field for familiarity.

#### 3.2 Gradient Preconditioning

In numerical linear algebra, preconditioning refers to transforming an ill-conditioned system  $A\mathbf{x} = \mathbf{b}$  into an equivalent system with better numerical properties by carefully choosing a non-singular matrix  $M$  and solving  $AM^{-1}\mathbf{y} = \mathbf{b}$  followed by  $M\mathbf{x} = \mathbf{y}$  instead.



In the context of optimization, the intuition for preconditioning is that it changes the geometry of the parameter space so that the level sets of the loss landscape become more isotropic, improving convergence. In homage to this perspective, we plot the loss landscape around a minimum of a neural field (inset), following Li et al. [2018]. The model trained with our stochastic preconditioning technique has a somewhat more isotropic landscape, aligning with the intuition behind preconditioning.

#### 3.3 Gaussian Smoothing of Signals

Gaussian smoothing, or blurring, is a widely-used operation in signal processing, where it can be viewed as damping of high-frequency content or the action of a heat flow. It can be defined by convolving a signal  $f$  with a Gaussian kernel  $g$ ,

$$\text{Blur}_\alpha[f] = f * g = \int_{\Omega} f(\mathbf{x} - \boldsymbol{\tau}) G_\alpha(\boldsymbol{\tau}) d\boldsymbol{\tau}, \quad (1)$$

where  $\Omega$  is the spatial domain, and  $G_\alpha(\cdot)$  the Gaussian density.  $\alpha$  gives the isotropic blur width, also known as the standard deviation of the normal distribution.

It is straightforward to evaluate Gaussian smoothing for discrete signals on a grid, but nontrivial to do the same for implicit neural fields. Past work has addressed this challenge by designing specific band-limited network architectures and training procedures for low-pass filtered neural field signals, motivated by the benefits of enabling level-of-detail post-processing in downstream applications [Barron et al. 2021; Mujkanovic et al. 2024; Petit et al. 2025; Shabanov et al. 2024; Verbin et al. 2022].

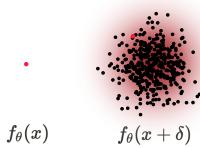
### 3.4 Stochastic Smoothing Evaluation

We can rewrite Equation 1 as the following expectation over a random variable  $\delta$  drawn from the corresponding normal distribution:

$$\text{Blur}[f] = \int_{\Omega} f(x + \tau) G(\tau) = \mathbb{E}[f(x + \delta)], \quad \delta \sim \mathcal{N}(0, \alpha). \quad (2)$$

Thus simply perturbing query locations according to a normal distribution gives a stochastic estimate of the blurred neural field. We will leverage this sampling to access the blurred field  $\text{Blur}_\alpha[f_\theta]$  in an implicit, online manner during optimization.

## 4 STOCHASTIC PRECONDITIONING



We propose to leverage stochastic smoothing as a preconditioner to improve neural field optimization. Concretely: during training, any time the neural field  $f_\theta$  would be queried at position  $x$ , we perturb the sample location according to Gaussian noise to instead query at location  $x + \delta$ , with  $\delta \sim \mathcal{N}(0, \alpha)$  (see inset). Otherwise, sampling, loss evaluation, and auto-differentiation are performed as usual for the representation and task at hand. This can be viewed as a stochastic approximation to optimizing through the blurred field  $\text{Blur}_\alpha[f](x)$ .

Two important details complete the method: (1) we always approximate the blurred field with a single sample, meaning the computation cost is unchanged, and (2) the scale  $\alpha$  is annealed from an initial value down to 0 during training, such that the final iterations of training are performed on the un-blurred field, and we are left with an ordinary neural field—see Section 4.2 for details and Section 5.3 for analysis.

In implementation, this technique requires changing just a few lines of code (Figure 3). Like past techniques, it effectively manipulates the frequency spectrum of the field, but it does so in a simple, purely implicit manner, which is compatible with any neural field representation, making it straightforward to apply in practice and within existing codebases.

*Blurring the field vs. blurring the supervision.* An important subtlety of this approach is that we do not blur the *supervision* (ground-truth data), as has been explored in [Barron et al. 2021, 2023; Karnewar et al. 2022]. Rather, we blur the *field* being optimized, as a preconditioner. The primary reason is that blurring the supervision is only possible in directly supervised problems such as fitting to a known signal, whereas blurring the field is fully general, and improves

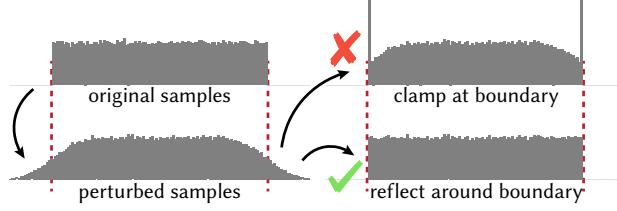


Fig. 5. **Boundary handling.** Stochastic preconditioning will perturb samples outside of bounded domains. Clamping would incorrectly concentrate samples on the boundary, while reflecting around the boundary retains the expected uniform distribution.

optimization even in regions where the field is governed only by a regularizer (Section 5.1). Perhaps surprisingly, this more general approach still causes the field  $f_\theta$  to be optimized at low-frequencies first (Figure 4). We hypothesize that this is because stochastic preconditioning damps high-frequency components, and thus also their gradients, during optimization.

### 4.1 Boundary Handling

Neural fields are often defined only on a bounded domain, such as a normalized  $[0, 1]^3$  box, but the perturbed query points  $x + \delta$  from stochastic preconditioning can fall outside of that domain. Naively clamping samples to the boundary creates artifacts, as it results in vastly more samples being drawn along the boundary during optimization. Rejecting and re-sampling these points avoids this issue, but complicates implementation. Instead, *reflecting* samples across the boundary is easily evaluated in constant time, and preserves a uniform sampling distribution, as shown in Figure 5. This reflection can be implemented with a coordinate-wise modulo operation, handling even rare multiple-reflections. For example, perturbed samples  $x$  can be reflected in the common  $[0, 1]$  domain as

$$x \leftarrow \begin{cases} \text{mod}(x, 2) & \text{if } \text{mod}(x, 2) \leq 1, \\ 2 - \text{mod}(x, 2) & \text{if } \text{mod}(x, 2) > 1. \end{cases} \quad (3)$$

### 4.2 Choosing the Perturbation Scale $\alpha$

Stochastic preconditioning requires choosing the blur scale  $\alpha \in \mathbb{R}^{\geq 0}$ , measured in spatial length units. Large  $\alpha$  corresponds to a large low-frequency blur, while  $\alpha = 0$  leaves the field unchanged. In principle  $\alpha$  could be a more general covariance matrix, but we find a scalar sufficient in our experiments. Motivated by coarse-to-fine optimization, we decrease  $\alpha$  from an initial  $\alpha_0$  to 0 during the optimization process. The details of this policy could be adjusted based on the problem setting, but we recommend exponentially decaying alpha from  $\alpha_0 = 1 - 2\%$  of the diagonal length of the bounding-box, to 0 at 1/3 of training or less, after which stochastic preconditioning is effectively disabled. This policy was found experimentally, see Section 5.3 for a small study. The final result is an ordinary neural field, which can be used in downstream tasks without concern for the stochastic preconditioning used during training.

### 4.3 Spatially Adaptive $\alpha$

We also consider an optional extension of this method to *spatially-varying* scale factors  $\alpha$ , storing  $\alpha(\mathbf{x})$  on a regular grid, which is initialized to  $\alpha_0$  and optimized during training as an additional degree of freedom. Each stochastic-preconditioned query  $f_\theta(\mathbf{x})$  becomes

$$f_\theta(\mathbf{x} + \delta) \quad \delta \sim \mathcal{N}(\mathbf{0}, \alpha(\mathbf{x})), \quad (4)$$

and we use the standard “reparameterization trick” to autodifferentiate through the task objective with respect to  $\alpha(\mathbf{x})$ , by evaluating  $\delta \leftarrow \alpha(\mathbf{x})\mathcal{N}(\mathbf{0}, 1)$ .

Interestingly, we find that even without any additional supervision or guidance,  $\alpha(\mathbf{x})$  naturally converges to encode a level of detail for the target field across the domain (Section 5.5, Figure 10). This procedure applies generally without any need for analysis of a known ground-truth signal. Intuitively, gradients to reduce  $\alpha(\mathbf{x})$  will arise in regions where the desired signal would have content at spatial frequencies smaller than  $\alpha(\mathbf{x})$ , so  $\alpha(\mathbf{x})$  is gradually optimized to that frequency, but no smaller. Furthermore, using this spatially-varying field in place of a constant schedule is a promising alternative in practice, and may even offer small improvements in optimization (see Table 3).

## 5 EXPERIMENTS

We validate and explore stochastic preconditioning on a range of neural field representations and tasks. We make a distinction between *indirectly supervised* and *directly supervised* tasks. In an indirectly supervised task, the goal is to solve a nontrivial inverse problem guided by incomplete supervision; we consider surface reconstruction from images and oriented point clouds, and neural radiance fields (NeRFs). In directly supervised tasks, the goal is to encode an already-known signal into a neural field; we consider signed distance functions and images. In general our method is most beneficial in the indirectly supervised setting, as they are difficult optimization problems with many local minima.

In all experiments, we compare stochastic preconditioning to the baseline neural field optimization at equal training iteration counts. Adding noise is computationally inexpensive, so there is otherwise little change to the runtime or other characteristics of a method, unless e.g., a hierarchy is intentionally removed, as in Section 5.2. For full details of experimental configurations and additional results, please see the **supplemental text and included webpage**.

### 5.1 Neural Signed Distance Functions

Signed distance functions (SDFs) represent closed surfaces via an implicit function  $f : \mathbb{R}^3 \rightarrow \mathbb{R}$  such that  $|f(\mathbf{x})|$  is the distance to the surface with  $f(\mathbf{x}) < 0$  inside and  $f(\mathbf{x}) > 0$  outside; they are widely-used in surface processing and reconstruction with neural fields.

**5.1.1 Neural Surface Reconstruction from Oriented Point Clouds.** Surface reconstruction from point clouds is a classic task in geometry processing; the input is a point cloud with positions and normals  $\mathcal{X} = \{\mathbf{x}_i \in \mathbb{R}^3, \mathbf{n}_i \in \mathbb{R}^3\}$  presumed to be sampled from some underlying surface, and the output is that surface. In the neural fields context, we seek to optimize a scalar field to be the SDF of

Table 1. Surface reconstruction from oriented point clouds, measuring mean Chamfer distance against the true surface (lower is better), on several models. Stochastic preconditioning improves accuracy even without special initialization strategies.

	Nefertiti	Cow	Bunny	Buddha	Armadillo
INGP [Müller et al. 2022]	3.57e-3	3.34e-3	3.70e-3	3.14e-3	2.49e-3
+ geom. init. [Atzmon and Lipman 2020]	2.82e-3	3.69e-3	4.24e-3	1.44e-3	7.25e-4
+ <i>stochastic preconditioning</i>	<b>7.29e-4</b>	<b>9.20e-4</b>	<b>9.01e-4</b>	<b>8.38e-4</b>	7.10e-4
+ geom. init. & <i>stochastic preconditioning</i>	7.89e-4	9.94e-4	9.98e-4	9.27e-4	<b>6.80e-4</b>

Table 2. Surface reconstruction from images, measuring mean PSNR and Chamfer Distance evaluated across 15 scenes in the DTU dataset [Jensen et al. 2014]. Stochastic preconditioning improves quality, especially for hashgrid based representations. See Table 2 and Table 3 for full tables.

	Avg. PSNR	Avg. Chamfer
NeuS	27.29	1.82
+ <i>stochastic preconditioning</i>	27.51	<b>1.45</b>
+ hashgrid	18.60	4.58
+ hashgrid & <i>stochastic preconditioning</i>	<b>28.26</b>	<b>1.45</b>
Neuralangelo	35.87	0.87
+ <i>stochastic preconditioning</i>	<b>36.26</b>	<b>0.76</b>

the reconstructed surface given an input point cloud [Atzmon and Lipman 2020; Gropp et al. 2020; Sitzmann et al. 2020].

As a representative method for this task, we adopt the objective formulation from *SIREN* [Sitzmann et al. 2020]

$$\mathcal{L}_{\text{total}} = \lambda_0 \mathcal{L}_{\text{surface}} + \lambda_1 \mathcal{L}_{\text{normal}} + \lambda_2 \mathcal{L}_{\text{eik}} + \lambda_3 \mathcal{L}_{\text{offsurface}}, \quad (5)$$

where  $\mathcal{L}_{\text{surface}}$  encourages  $f_\theta(\mathbf{p}_i) \approx 0$  and  $\mathcal{L}_{\text{normal}}$  encourages the field gradient at on-surface points to match given normals  $\nabla f_\theta(\mathbf{p}_i) \approx \mathbf{n}_i$ . The regularizer  $\mathcal{L}_{\text{eik}}$  is an eikonal term encouraging  $f_\theta$  to be a distance function  $|\nabla f_\theta(\mathbf{x})| = 1$ , and  $\mathcal{L}_{\text{offsurface}}$  regularizes  $f_\theta$  to have large magnitude away from the surface. The first two terms are evaluated over the input point set, while the regularizers are sampled throughout space.

We consider several field representations, including INGP [Müller et al. 2022], Fourier feature fields [Tancik et al. 2020], neural triplanes [Wang et al. 2023] and sinusoidal INRs [Liu et al. 2024]—we find that stochastic preconditioning improves quality for all of these representations, as shown in Figure 2. The benefit is greatest with the widely-used fast INGP hashgrids, where it resolves spurious floating artifacts (Figure 6).

Atzmon and Lipman [2020] propose a popular *geometric initialization* for fitting SDFs, by carefully initializing the solution to a sphere. This technique is highly effective for objects, but leads to robustness problems for geometry which is far from spherical, such as complex indoor scenes (Figure 1). In Table 1, we measure Chamfer distance against ground truth meshes, and find that stochastic preconditioning alleviates the need for this initialization—optimizing a preconditioned field with no special initialization often outperforms the carefully initialized field.

**5.1.2 Neural Surface Reconstruction from Images.** A more challenging reconstruction task is to recover surfaces from posed images



**Fig. 6. SDFs from oriented point clouds.** Stochastic preconditioning reduces artifacts and improves quality when fitting SDFs to point clouds, here using an INGP Hashgrid MLP [Müller et al. 2022] as the field representation, along with geometric initialization [Atzmon and Lipman 2020]. Here we visualize Chamfer error against known ground-truth. The standard approach exhibits artifacts which are not resolved by training, while stochastic preconditioning successfully converges to a high-fidelity surface.

of a scene via differentiable volumetric rendering of an SDF field. Much work has adapted the NeRF-like [Mildenhall et al. 2021] formulation [Li et al. 2023; Oechsle et al. 2021; Wang et al. 2021; Yariv et al. 2021], defining volumetric density as a function of an SDF field, and rendering that density along with auxiliary field outputs for color or directional radiance. These approaches extract high-fidelity surfaces when successful, but can be challenging to fit due to the difficult inverse nature of the problem.

We consider three baseline methods for this problem: NeuS [Wang et al. 2021], a variant of NeuS replacing the frequency encoding with a hashgrid encoding, and Neuralangelo [Li et al. 2023], which includes a specific coarse-to-fine scheme coupling level unlocking with numerical gradient estimation. For each method, stochastic preconditioning amounts to adding just a few lines of code to the existing implementation. When evaluating spatial gradients with finite differences, we apply the same noise to all samples in the stencil.

We train each method with and without stochastic preconditioning on 15 scenes from the DTU dataset as in prior work [Jensen et al. 2014], and compute the average PSNR scores on all the images of the evaluation set as well as the Chamfer distance to the ground truth mesh (Table 2). Stochastic preconditioning improves quality of most scenes when added to all three methods, again with the most dramatic improvements for hashgrid-based representations. See Table 2 and Table 3 for full tables in the supplemental document.

## 5.2 Neural Radiance Fields

Neural Radiance Fields (NeRF) [Mildenhall et al. 2021] represent the appearance of a scene as a neural field encoding the density and outgoing radiance from each point in space, rendered via volumetric sampling along rays. In Section 5.1.2 we already considered methods which optimize SDFs via NeRF-like rendering; the approaches here

use more general density in place of SDFs, which may yield higher visual fidelity but lack an underlying surface representation.

We evaluate primarily on two cases where overfitting artifacts are particularly likely to occur: sparse supervision and ReLU fields [Karnewar et al. 2022] without hierarchical training.

*Sparse Supervision.* First we consider the sparse-view scenario, fitting NeRFs to a comparatively small number of images, which greatly exacerbates the optimization challenges of local minima and overfitting to floater artifacts. We adopt the setting of FreeNeRF [Yang et al. 2023], a recent approach tackling this challenge via control of frequency-domain encoding and near-camera geometry. Figure 7 shows a comparison against ordinary MipNeRF [Barron et al. 2021] in this setting as a baseline, as well as simply adding our stochastic preconditioning to MipNeRF. As expected, MipNeRF struggles in the sparse view scenario, but adding stochastic preconditioning yields high quality results on-par with FreeNeRF—and while FreeNerf is specific to frequency encodings, stochastic preconditioning is applicable to any underlying neural field representation. See Table 5 in the supplement for additional metrics.

*ReLU Fields.* ReLU fields [Karnewar et al. 2022] are a notably *non*-neural representation for NeRFs, exchanging the neural field for a simple trilinearly-interpolated regular grid with a single ReLU nonlinearity after interpolation, akin to prior work optimizing directly on grids [Fridovich-Keil and Yu et al. 2022]. To successfully optimize NeRFs, ReLU fields leverage a coarse-to-fine approach, resulting in a four-stage pipeline where each stage uses a finer grid representation and higher-resolution target image; optimization fails catastrophically without this hierarchy. We find that omitting the hierarchy and simply using stochastic preconditioning instead yields similar or better results (Figure 8).

In this particular experiment, we remove an existing coarse-to-fine hierarchy and replace it with stochastic preconditioning—this

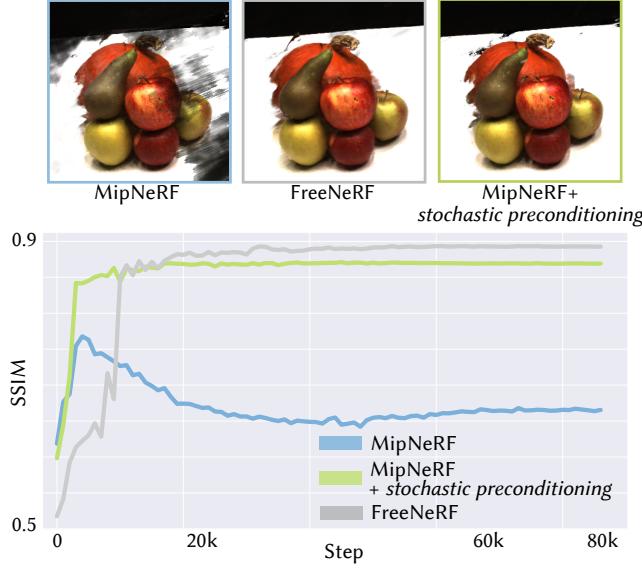


Fig. 7. **Sparse-view NeRF.** We train MipNeRF [Barron et al. 2021], FreeNeRF [Yang et al. 2023] and MipNeRF with our stochastic preconditioning on scene 63 of the DTU dataset [Jensen et al. 2014] with just 6 images as input supervision. Nearly achieves the same benefits as FreeNeRF while being compatible with many representations and tasks. See the supplement for additional results.

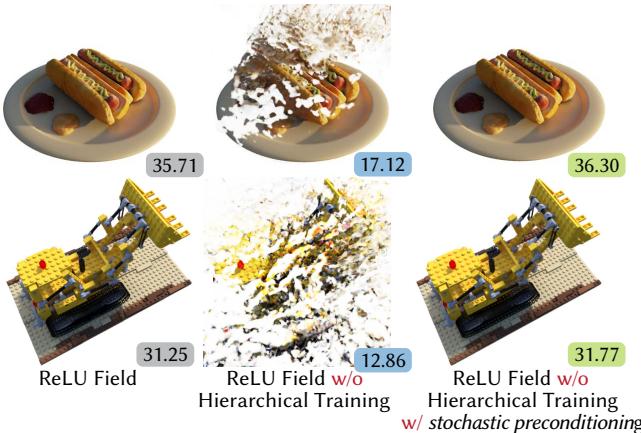


Fig. 8. **ReLU fields made easy.** While the original ReLU fields [Karnewar et al. 2022] method performs well on these synthetic scenes (left), it requires a hierarchical training scheme. Omitting this scheme, training directly at full resolution, results in catastrophic floater artifacts (center). Replacing the coarse-to-fine scheme with our simple stochastic preconditioning method yields on-par or higher-quality results. Each result is labeled with the average PSNR. See Figure 3 of the supplemental document for the convergence curve. simplifies the approach, but comes at some cost of increased training time from 3.1 hours to 5.2 hours, since we no longer benefit from fast iterations at the coarse level.

See Table 8 in the supplement for additional metrics.

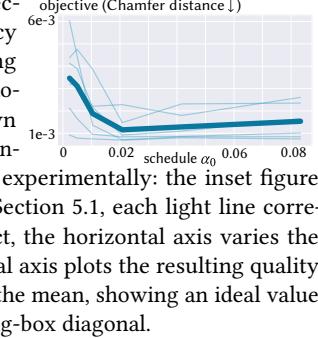
**Large-scale reconstruction.** We also perform a preliminary study applying our preconditioning to large-scale photometric reconstruction with Neuralangelo [Li et al. 2023] on the Tanks and Temples

dataset [Knapitsch et al. 2017]. We disabled the coarse-to-fine optimization in Neuralangelo, and instead use stochastic preconditioning, and evaluate PSNR on the foreground rendered image (the background is rendered by a separate background field). We find that stochastic preconditioning gives a small improvement of 0.16 PSNR on average to 32.52, coming mainly from a significant increase of +1.16 on the Barn scene, where artifacts are resolved by preconditioning. See the Supplement for a breakdown. Future work will be needed to further investigate effectiveness of this technique for large-scale reconstruction.

### 5.3 Analysis

Here we analyze a few important aspects of our method: the initial alpha value, the choice of noise, and the number of stochastic samples used.

**Choice of Initial Alpha.** In Section 4.2 we suggest a default policy of scheduling the preconditioning blur scale  $\alpha$  from  $\alpha_0 = 2\%$  of the domain bounding box diagonal down to  $\alpha = 0$  during the first 1/3 of training time. This policy was found experimentally: the inset figure shows a study in the setting of Section 5.1, each light line corresponds to a different input object, the horizontal axis varies the  $\alpha_0$  schedule value, and the vertical axis plots the resulting quality (lower is better). The dark line is the mean, showing an ideal value of around  $\alpha_0 = 2\%$  of the bounding-box diagonal.



**Choice of Noise.** We default to using Gaussian noise when applying stochastic preconditioning in our experiments for its formal relationship to blur operators, the spectral domain, and heat flow. We additionally experiment with other noise kernels such as uniform and squared Gaussian noise, for surface reconstruction from points as in Section 5.1 and find largely similar results for all kernels, yielding mean Chamfer distance error of 9.1e-4 with a Gaussian kernel, 9.3e-4 with a uniform kernel, and 8.8e-4 with a squared-Gaussian kernel. See the complete set of metrics in Table 7 in Supplement.

**Single Sample v.s. Multiple Samples.** Our method proposes stochastic preconditioning in expectation, via a single perturbed sample. However, we also experiment taking additional samples, to more-accurately estimate approximate the blurred field at each evaluation. In computation, this amounts to averaging multiple samples from  $f(\mathbf{x} + \boldsymbol{\delta})$  with independent  $\boldsymbol{\delta} \sim \mathcal{N}(0, \alpha)$ . Taking additional samples linearly increases cost, but we do not observe any benefit, likely due to the stochasticity already inherent in neural field optimization. In the ReLU field setting as in Section 5.2, we observe an average PSNR of 33.72 with single sample, 33.74 with two samples, and 33.79 with four samples. See Table 6 in the Supplement for full results.

### 5.4 Robustness to Hyperparameters.

Small changes to hyperparameters can easily make the difference between success and getting trapped in a poor local minimum, especially for indirectly supervised tasks. To explore this effect, we take the NeuS with hashgrid configuration on one scene from Section 5.1.2, and sweep through several choices of network learning

rate, hashgrid learning rate, maximum resolution of the hashgrid, and hash table size. For each parameter configuration, we fit the field with and without stochastic preconditioning, and measure the final PSNR. Figure 9 shows a clear distribution shift—although even stochastic preconditioning does not succeed for all choices of hyperparameters, it succeeds much more often across a wider range of hyperparameters. This also provides evidence that the improvement offered by stochastic preconditioning is not an artifact of one particular choice of hyperparameters.

Table 3. PSNR quality for Neuralangelo [Li et al. 2023] reconstruction from images on selected scenes in the DTU dataset, with stochastic preconditioning using spatial-varying scaling vs. a single global scaling.

	DTU 37	DTU 40	DTU 55
Global scheduled $\alpha$	29.80	34.78	31.79
Optimized spatial-varying $\alpha(x)$	<b>30.02</b>	<b>34.80</b>	<b>32.20</b>

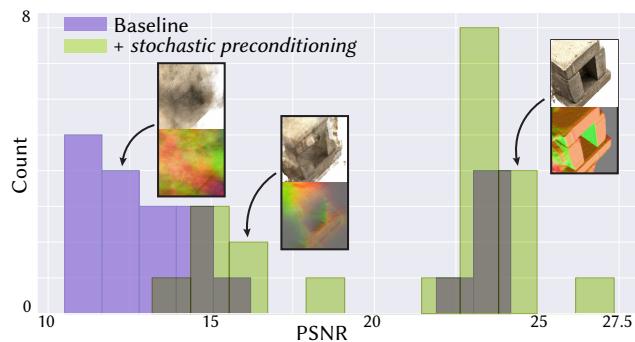


Fig. 9. Stochastic preconditioning increases robustness to hyperparameters, shown here in a histogram of PSNRs from fitting preconditioned and non-preconditioned fields across a range of hyperparameters. See Section 5.4 for details.

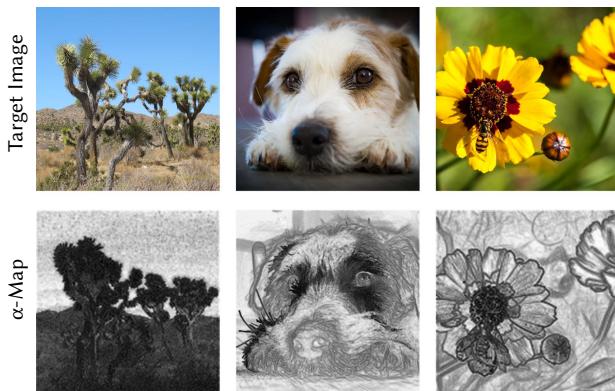


Fig. 10. Visualizations of the automatically-optimized spatially-varying  $\alpha$ -map from fitting target images. Photo credits left to right: ©Luke Jones (CC BY 2.0), ©Jukka Hernetkoski (CC BY-NC 2.0), ©Jez (CC BY-NC-ND 2.0).

## 5.5 Spatially-Varying Blur

Rather than a global blur radius  $\alpha$  decreased with a predetermined schedule, we can instead allow  $\alpha(x)$  to vary over the domain, optimized automatically according to task objectives (Section 4.3). In Figure 10 we apply this technique while fitting neural fields to encode a target image. The resulting  $\alpha(x)$  naturally evolves to a good estimation of the frequency distribution of the image contents. Table 3 shows the result of using the spatially-varying  $\alpha(x)$  for reconstruction on the DTU dataset with Neuralangelo as in Section 5.1.2—the automatic, spatially-varying approach matches or even slightly improves quality.

## 6 CONCLUSION

*Limitations and future work.* Stochastic sampling is an *unbiased* estimator for the blurred field, but sampling a nonlinear loss function with a single sample is a *biased* estimator for the loss on the true blurred field. We do not observe this to cause problems in practice, but there is opportunity for deeper analysis. Although stochastic preconditioning improves quality at no additional cost, it misses out on some accelerations offered by hierarchical schemes—optimizing a low-resolution level of a hierarchy is typically very cheap, but applying stochastic preconditioning with a large blur has the same cost as optimizing the full field. We only briefly investigate spatially-varying noise distributions (Section 5.5); future work may explore additional strategies, including anisotropic distributions, as well as other uses for the optimized  $\alpha$  maps.

*Conclusion.* We believe that stochastic preconditioning will serve as a valuable tool in the neural fields toolbox, easily deployed to improve performance on a wide variety of problems at minimal cost, especially when custom-designed approaches may be cumbersome.

## ACKNOWLEDGMENTS

We thank Zan Gojcic, Thomas Müller, Zian Wang, Sanja Fidler and Alex Keller for their help throughout this work, as well as John Hancock the Dynamic Graphics Project for computing support. We are grateful to the artists of 3D models used for demonstrations in this paper, including Turbosquid users Marc Mons, A\_Akhtar, charbavito, Stasma, sougatadhar16, Poly Forge, Vadim Manoli and Digital Fashionwear BD. Our research is funded in part by NSERC Discovery (RGPIN-2022-04680), the Ontario Early Research Award program, the Canada Research Chairs Program, a Sloan Research Fellowship, the DSI Catalyst Grant program and gifts by Adobe Inc.

## REFERENCES

- Yusuf Sinan Akgul and Chandra Kambhamettu. 2003. A coarse-to-fine deformable contour optimization framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25, 2 (2003), 174–186.
- Guozhong An. 1996. The effects of adding noise during backpropagation training on a generalization performance. *Neural computation* 8, 3 (1996), 643–674.
- Matan Atzmon and Yaron Lipman. 2020. Sal: Sign agnostic learning of shapes from raw data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2565–2574.
- Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. 2021. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*. 5855–5864.
- Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. 2023. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 19697–19705.

- Chris M Bishop. 1995. Training with noise is equivalent to Tikhonov regularization. *Neural computation* 7, 1 (1995), 108–116.
- Alexander Camuto, Matthew Willets, Umut Simsekli, Stephen J Roberts, and Chris C Holmes. 2020. Explicit regularisation in gaussian noise injections. *Advances in Neural Information Processing Systems* 33 (2020), 16603–16614.
- Fridovich-Keil and Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. 2022. Plenoxels: Radiance Fields without Neural Networks. In *CVPR*.
- Yves Grandvalet, Stéphane Canu, and Stéphane Boucheron. 1997. Noise injection: Theoretical prospects. *Neural Computation* 9, 5 (1997), 1093–1108.
- Amos Groppe, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. 2020. Implicit geometric regularization for learning shapes. *arXiv preprint arXiv:2002.10099* (2020).
- Linus Härenstam-Nielsen, Lu Sang, Abhishek Saroha, Nikita Araslanov, and Daniel Cremers. 2024. DiffCD: A Symmetric Differentiable Chamfer Distance for Neural Implicit Surface Fitting. *arXiv preprint arXiv:2407.17058* (2024).
- Ben Hayes, Charalampos Saitis, and György Fazekas. 2023. Sinusoidal frequency estimation by gradient descent. In *ICASSP 2023–2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 1–5.
- Amir Hertz, Or Perel, Raja Giryes, Olga Sorkine-Hornung, and Daniel Cohen-Or. 2021. Sape: Spatially-adaptive progressive encoding for neural optimization. *Advances in Neural Information Processing Systems* 34 (2021), 8820–8832.
- Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. 2014. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 406–413.
- Animesh Karnewar, Tobias Ritschel, Oliver Wang, and Niloy Mitra. 2022. Relu fields: The little non-linearity that could. In *ACM SIGGRAPH 2022 conference proceedings*, 1–9.
- Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. 2017. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–13.
- Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. 2018. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems* 31 (2018).
- Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. 2023. Neuralangelo: High-fidelity neural surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8456–8465.
- David B Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. 2022. Bacon: Band-limited coordinate networks for multiscale scene representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 16252–16262.
- Zhen Liu, Hao Zhu, Qi Zhang, Jingde Fu, Weibing Deng, Zhan Ma, Yanwen Guo, and Kun Cao. 2024. Finer: Flexible spectral-bias tuning in implicit neural representation by variable-periodic activation functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2713–2722.
- Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106.
- Felix Mujkanovic, Ntumba Elie Nsampi, Christian Theobalt, Hans-Peter Seidel, and Thomas Leimkuhler. 2024. Neural Gaussian Scale-Space Fields. *arXiv preprint arXiv:2405.20980* (2024).
- Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)* 41, 4 (2022), 1–15.
- Ntumba Elie Nsampi, Adarsh Djeacoumar, Hans-Peter Seidel, Tobias Ritschel, and Thomas Leimkuhler. 2023. Neural field convolutions by repeated differentiation. *arXiv preprint arXiv:2304.01834* (2023).
- Michael Oechsle, Songyu Peng, and Andreas Geiger. 2021. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 5589–5599.
- Keunhong Park, Philipp Henzler, Ben Mildenhall, Jonathan T Barron, and Ricardo Martin-Brualla. 2023. Camp: Camera preconditioning for neural radiance fields. *ACM Transactions on Graphics (TOG)* 42, 6 (2023), 1–11.
- Doriani Petit, Steve Bourgeois, Dumitru Pavel, Vincent Gay-Bellile, Florian Chabot, and Loïc Barthe. 2025. RING-NeRF: Rethinking Inductive Biases for Versatile and Efficient Neural Fields. In *European Conference on Computer Vision*. Springer, 148–164.
- Julien Philip and Valentin Deschaintre. 2023. Floater no more: Radiance field gradient scaling for improved near-camera training. *arXiv preprint arXiv:2305.02756* (2023).
- Daniel Rebain, Soroosh Yazdani, Kwang Moo Yi, and Andrea Tagliasacchi. 2024. Neural fields as distributions: Signal processing beyond Euclidean space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4274–4283.
- Junwon Seo, Sangyoon Lee, Kwang In Kim, and Jaeho Lee. 2024. In Search of a Data Transformation That Accelerates Neural Field Training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4830–4839.
- Akhmedkhan Shabanov, Shrisudhan Govindarajan, Cody Reading, Lily Goli, Daniel Rebain, Kwang Moo Yi, and Andrea Tagliasacchi. 2024. BANF: Band-limited Neural Fields for Levels of Detail Reconstruction. In *CVPR*. 20571–20580.
- Jocelyn Sietsema and Robert JF Dow. 1991. Creating artificial neural networks that generalize. *Neural networks* 4, 1 (1991), 67–79.
- Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. 2020. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems* 33 (2020), 7462–7473.
- Towaki Takikawa, Joey Litaien, Kangxue Yin, Karsten Kreis, Charles Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. 2021. 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*. 11358–11367.
- Towaki Takikawa, Thomas Müller, Merlin Nimier-David, Alex Evans, Sanja Fidler, Alec Jacobson, and Alexander Keller. 2023a. Compact neural graphics primitives with learned hash probing. In *SIGGRAPH Asia 2023 Conference Papers*. 1–10.
- Towaki Takikawa, Shunsuke Saito, James Tompkin, Vincent Sitzmann, Srinath Sridhar, Or Litany, and Alex Yu. 2023b. Neural Fields for Visual Computing. In *ACM SIGGRAPH 2023 Courses* (Los Angeles, California).
- Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. 2020. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems* 33 (2020), 7537–7547.
- Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. 2022. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 5481–5490.
- Thomas Walker, Octave Mariotti, Amir Vaxman, and Hakan Bilen. 2024. Spatially-Adaptive Hash Encodings For Neural Surface Reconstruction. *arXiv preprint arXiv:2412.05179* (2024).
- Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. 2021. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689* (2021).
- Yiqun Wang, Ivan Skorokhodov, and Peter Wonka. 2023. Pet-neus: Positional encoding tri-planes for neural surfaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12598–12607.
- Yuanbo Xiangli, Lining Xu, Xinggang Pan, Nanxuan Zhao, Anyi Rao, Christian Theobalt, Bo Dai, and Dahua Lin. 2022. Bungeenerf: Progressive neural radiance field for extreme multi-scale scene rendering. In *European conference on computer vision*. Springer, 106–122.
- Dejia Xu, Peihao Wang, Yifan Jiang, Zhiwen Fan, and Zhangyang Wang. 2022. Signal processing for implicit neural representations. *Advances in Neural Information Processing Systems* 35 (2022), 13404–13418.
- Jiawei Yang, Marco Pavone, and Yue Wang. 2023. Freenerf: Improving few-shot neural rendering with free frequency regularization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 8254–8263.
- Jian Yao and Ahmad Al-Dahle. 2019. Coarse-to-fine optimization for speech enhancement. *arXiv preprint arXiv:1908.08044* (2019).
- Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. 2021. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems* 34 (2021), 4805–4815.
- Taesun Yeom, Sangyoon Lee, and Jaeho Lee. 2024. Fast Training of Sinusoidal Neural Fields via Scaling Initialization. *arXiv preprint arXiv:2410.04779* (2024).

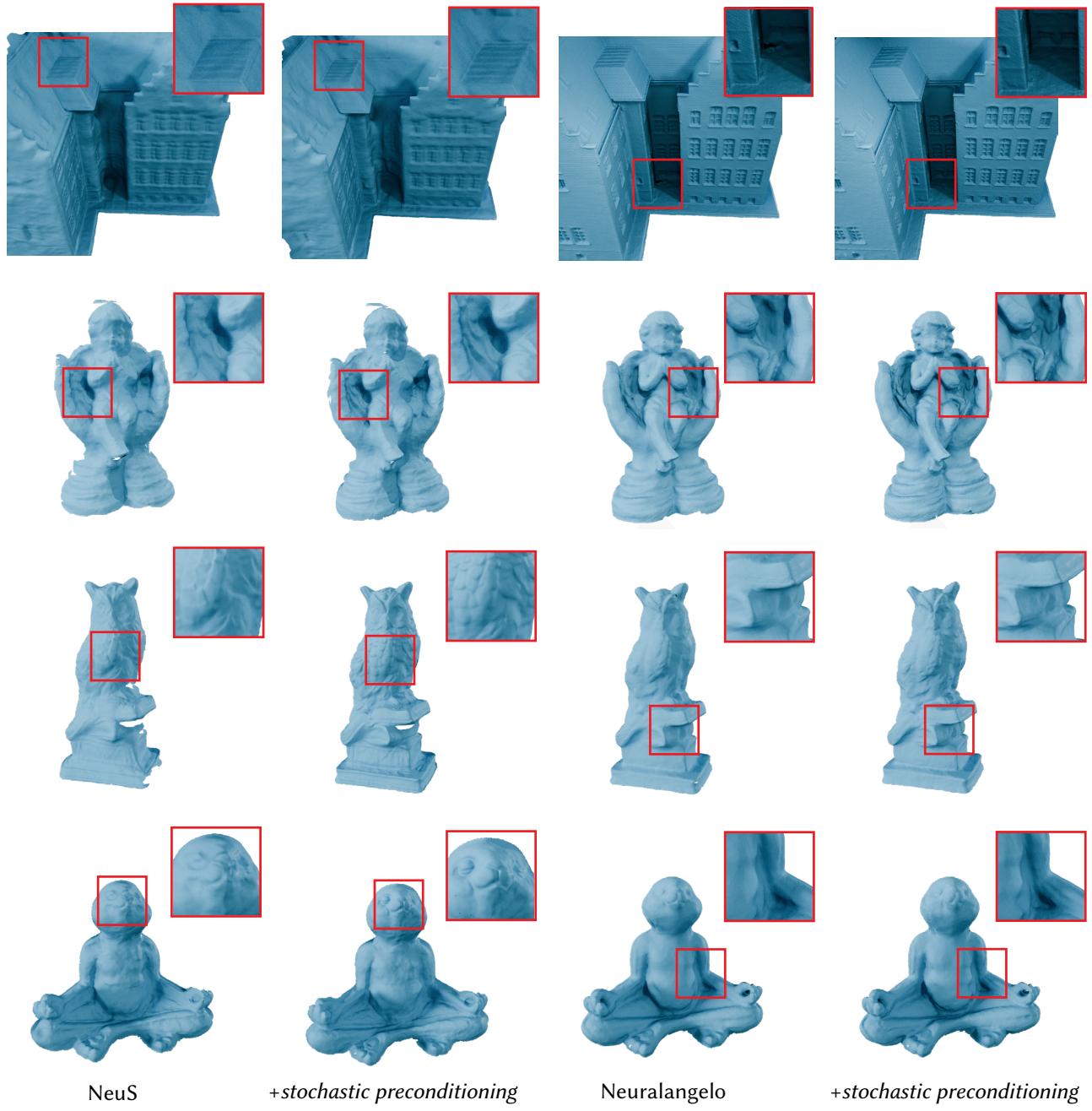


Fig. 11. Here we compare extracted meshes from NeuS [Wang et al. 2021] and Neuralangelo [Li et al. 2023], with and without stochastic preconditioning. We find that stochastic preconditioning offers a small but consistent improvement of high-frequency detail in the fitted result, especially in low-visibility regions.